ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
НЕФТИ И ГАЗА имени И.М. ГУБКИНА

Кафедра автоматизированных систем управления

Л.В. КУЗНЕЦОВА, Д.Г. ЛЕОНОВ

МЕТОДЫ И СРЕДСТВА ЗАЩИТЫ ИНФОРМАЦИИ

Курс лекций

Издательский центр
Москва — 2009

УДК 004.56 (075)

Кузнецова Л.В., Леонов Д.Г. Методы и средства защиты информации. Курс лекций. М.: Издательский центр РГУ нефти и газа имени И.М. Губкина, 2009. — 216 с.

В данном учебном пособии рассматриваются основные задачи, связанные с защитой информации в современных информационных системах. Рассмотрены общие подходы к организации системы безопасности, вопросы криптографического закрытия информации, безопасность распределенных систем.

Пособие предназначено для использования в рамках курса «Методы и средства защиты информации» студентами 4 курса кафедры АСУ, а также слушателями курсов повышения квалификации.

Рецензенты:
Сарданашвили С.А., д.т.н., профессор кафедры АСУ
РГУ нефти и газа имени И.М. Губкина
Кочуева О.Н., к.т.н., ст. преп. кафедры ПМиКМ
РГУ нефти и газа имени И.М. Губкина

Под общей редакцией профессора Григорьева Л.И.

Содержание

Введение

По мере роста вовлеченности информационных технологий в современную жизнь возрастает и зависимость общества от степени их безопасности. Актуальность задачи обеспечения безопасности информационных систем обусловлена следующими причинами:

- рост вычислительной мощности современных компьютеров в сочетании с упрощением их эксплуатации;
- увеличение объемов информации, накапливаемой, хранимой и обрабатываемой с помощью средств автоматизации;
- сосредоточение в единых базах данных информации различного назначения и различной принадлежности;
- высокие темпы роста парка персональных компьютеров и связанное с этим резкое расширение круга пользователей, имеющих непосредственный доступ к вычислительным ресурсам и массивам данных;
- бурное развитие программных средств, зачастую не удовлетворяющих даже минимальным требованиям безопасности;
- широкое распространение сетевых технологий и развитие глобальной сети Internet, практически не препятствующей нарушениям безопасности информационных систем во всем мире.

В учебном пособии рассматриваются основные задачи, связанные с защитой информации в современных информационных системах.

Первая часть посвящена общим вопросам построения систем защиты.

Во второй части рассмотрены криптографические методы преобразования информации.

В третьей части обсуждаются вопросы безопасности распределенных систем, построение защищенных виртуальных сетей.

I. Организация систем безопасности

1. Информационная безопасность компьютерных систем

1.1. Основные понятия и определения

Рассмотрим основные понятия информационной безопасности компьютерных систем.

Под ***безопасностью АСОИ*** понимают ее защищенность от случайного или преднамеренного вмешательства в нормальный процесс ее функционирования, а также от попыток хищения, изменения или разрушения ее компонентов.

Безопасность АСОИ достигается принятием мер по обеспечению конфиденциальности и целостности обрабатываемой ею информации, а также доступности и целостности компонентов и ресурсов системы.

Под ***доступом к информации*** понимается ознакомление с информацией, ее обработка, в частности копирование, модификация или уничтожение информации.

Объект — пассивный компонент системы, хранящий, принимающий или передающий информацию. Доступ к объекту означает доступ к содержащейся в нем информации.

Субъект — это активный компонент системы, который может инициировать потоки информации от объекта к субъекту или вносить изменения в состояние системы.

Авторизованный субъект — субъект, прошедший проверку допуска к ресурсам системы.

Правила разграничения доступа служат для регламентации права доступа субъектов доступа к объектам доступа.

Различают санкционированный и несанкционированный доступ к информации.

Санкционированный доступ к информации — это доступ к информации, не нарушающий установленные правила разграничения доступа.

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

Конфиденциальность данных — это статус, предоставленный данным и определяющий требуемую степень их защиты. По существу конфиденциальность информации — это свойство информации быть известной только допущенным и прошедшим проверку (авторизированным) субъектам системы (пользователям, процессам, программам). Для остальных субъектов системы эта информация должна быть неизвестной.

Под ***целостностью информации*** понимается отсутствие значимых смысловых отличий между данными в системе и данными в исходных документах, т.е. отсутствие их случайного или преднамеренного искажения или уничтожения.

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

Доступность компонента* или *ресурса системы — это свойство компонента или ресурса быть доступным для авторизованных законных субъектов системы.

Ущерб безопасности подразумевает нарушение состояния защищенности информации, содержащейся и обрабатывающейся в АСОИ. С понятием ущерба безопасности тесно связано понятие уязвимости АСОИ.

Уязвимость АСОИ — это некоторое неудачное свойство системы, которое делает возможным возникновение и реализацию угрозы.

Угроза безопасности — возможное нежелательное воздействие на систему, в результате которого может быть нанесен прямой или косвенный ущерб ее безопасности. Именно противодействие угрозам безопасности является целью защиты информационных систем.

Атака на компьютерную систему — это действие, предпринимаемое злоумышленником, которое заключается в поиске и использовании той или иной уязвимости системы. Таким образом, атака — это реализация угрозы безопасности.

Политика безопасности — это совокупность норм, правил и практических рекомендаций, регламентирующих работу средств защиты АСОИ от заданного множества угроз безопасности.

Инцидент безопасности — любое событие, нарушающее текущую политику безопасности.

Модель безопасности — формальное представление политики безопасности.

Гарантированность — степень уверенности, с которой можно утверждать, что для проведения в жизнь политики безопасности выбран подходящий набор средств, и что каждое из этих средств правильно выполняет свою роль.

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

Безопасная* или *защищенная система — это система со средствами защиты, которые успешно и эффективно противостоят угрозам безопасности.

Необходимо четко понимать, что при построении защищенной системы всегда подразумевается ее разумная достаточность, т.е. эффективное противодействие некоторому заданному подмножеству угроз безопасности, критичных для функционирования организации, использующей информационную систему.

1.2. Основные угрозы безопасности АСОИ

Современная автоматизированная система обработки информации представляет собой сложную систему, состоящую из большого числа компонентов различной степени автономности, которые связаны между собой и обмениваются данными.

Компоненты АСОИ можно разбить на следующие группы:

- ***аппаратные средства*** — ЭВМ и их составные части (процессоры, мониторы, терминалы, периферийные устройства — дисководы, принтеры, контроллеры, кабели, линии связи) и т.д.;
- ***программное обеспечение*** — приобретенные программы, исходные, объектные, загрузочные модули; операционные системы и системные программы (компиляторы, компоновщики и др.), утилиты, диагностические программы и т.д.;

- ***данные*** — хранимые временно и постоянно, на магнитных носителях, печатные, архивы, системные журналы и т.д.;
- ***персонал*** — обслуживающий персонал и пользователи.

Практически каждый компонент может подвергнуться внешнему воздействию или выйти из строя. Существует множество классификаций угроз по различным показателям.

По **цели воздействия** различают три основных типа угроз безопасности АСОИ:

- угрозы нарушения конфиденциальности информации;
- угрозы нарушения целостности информации;
- угрозы нарушения работоспособности системы (отказы в обслуживании).

Угрозы нарушения конфиденциальности направлены на разглашение конфиденциальной или секретной информации. При реализации этих угроз информация становится известной лицам, которые не должны иметь к ней доступ. В терминах компьютерной безопасности угроза нарушения конфиденциальности имеет место всякий раз, когда получен несанкционированный доступ к некоторой закрытой информации, хранящейся в компьютерной системе или передаваемой от одной системы к другой.

Угрозы нарушения целостности информации, хранящейся в компьютерной системе или передаваемой по каналу связи, направлены на ее изменение или искажение, приводящее к нарушению ее качества или полному уничтожению. Целостность информации может быть нарушена умышленно злоумышленником, а также в результате объективных воздействий со стороны среды, окружающей систему. Эта угроза особенно

актуальна для систем передачи информации — компьютерных сетей и систем телекоммуникаций.

Угрозы нарушения работоспособности (отказ в обслуживании) направлены на создание таких ситуаций, когда определенные преднамеренные действия либо снижают работоспособность АСОИ, либо блокируют доступ к некоторым ее ресурсам.

Например, если один пользователь системы запрашивает доступ к некоторой службе, а другой предпринимает действия по блокированию этого доступа, то первый пользователь получает отказ в обслуживании. Блокирование доступа к ресурсу может быть постоянным или временным.

По характеру воздействия угрозы безопасности АСОИ можно подразделить на ***случайные*** и ***преднамеренные***. Анализ опыта проектирования, изготовления и эксплуатации АСОИ показывает, что информация подвергается различным случайным воздействиям на всех этапах цикла жизни и функционирования АСОИ.

Причинами ***случайных воздействий*** при эксплуатации АСОИ могут быть:

- аварийные ситуации из-за стихийных бедствий и отключений электропитания;
- отказы и сбои аппаратуры;
- ошибки в программном обеспечении;
- ошибки в работе обслуживающего персонала и пользователей;
- помехи в линиях связи из-за воздействий внешней среды.

Преднамеренные воздействия связаны с целенаправленными действиями нарушителя. В качестве нарушителя могут выступать служащий, посетитель, конкурент, наемник и т.д. Действия нарушителя могут быть обусловлены разными мотивами: недовольством служащего

своей карьерой, сугубо материальным интересом (взятка), любопытством, конкурентной борьбой, стремлением самоутвердиться любой ценой и т.п. Исходя из возможности возникновения наиболее опасной ситуации, обусловленной действиями нарушителя, можно составить гипотетическую модель потенциального нарушителя:

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

По средствам воздействия:

Вмешательство человека в работу АСОИ. К этому классу относятся:

- организационные средства нарушения безопасности АСОИ (НСД к устройствам хранения и обработки информации, кража носителей информации, порча оборудования и т.д.);
- осуществление нарушителем НСД к программным компонентам.

Меры, противостоящие таким угрозам, носят организационный характер (охрана, режим доступа к устройствам АСОИ), а также включают в себя совершенствование систем разграничения доступа системы обнаружения попыток атак (например, отслеживание попыток подбора паролей.)

Аппаратно-техническое вмешательство в работу АСОИ, например получение информации по электромагнитному излучению устройств АСОИ, электромагнитные воздействия на каналы передачи информации и другие методы. Защита от таких угроз, кроме

организационных мер, предусматривает соответствующие аппаратные (экранирование излучений аппаратуры, защита каналов от прослушивания) и программные меры (шифрование сообщений в каналах связи).

Разрушающее воздействие на программные компоненты АСОИ с помощью программных средств (так называемые разрушающие программные средства — РПС). К ним относятся компьютерные вирусы, троянские кони (закладки), средства проникновения в удаленные системы с использованием локальных и глобальных сетей. Средства борьбы с подобными атаками состоят из программных систем защиты.

1.3. Безопасность распределенных систем

Особо следует остановиться на угрозах, которым могут подвергаться ***компьютерные сети***. Основная особенность любой компьютерной сети состоит в том, что ее компоненты распределены в пространстве. Связь между узлами (объектами) сети осуществляется физически с помощью сетевых линий связи и программно с помощью механизма сообщений. При этом управляющие сообщения и данные, пересылаемые между объектами сети, передаются в виде пакетов обмена. Это добавляет к существующим угрозам возможность **удаленной атаки** на объекты системы, атаки на инфраструктуру и протоколы сети, внедрение в систему ложных объектов и т.п. Под удаленной атакой понимают информационное разрушающее воздействие на распределенную компьютерную сеть, программно осуществленное по каналам связи. Злоумышленник может находиться за тысячи километров от атакуемого объекта, при этом нападению может подвергаться не только конкретный компьютер, но и информация, передающаяся по сетевым каналам связи.

Подобные атаки становятся возможными, в частности, из-за использования широковещательной среды передачи данных (как, например,

при использовании Ethernet, когда фрейм передается на все сетевые интерфейсы, и принимает его тот, MAC-адрес которого совпадает с адресатом фрейма), применения нестойких алгоритмов идентификации удаленных объектов и субъектов и т.п.

Для распределенных систем можно добавить еще несколько классификационных признаков.

По методам воздействия:

Пассивные. Воздействия, не влияющие непосредственно на работу системы, но способные нарушать ее политику безопасности. Примером такого воздействия является прослушивание канала связи. Нарушитель только наблюдает за прохождением информации по каналу связи, не вторгаясь ни в информационный поток, ни в содержание передаваемой информации. Как правило, злоумышленник может определить пункты назначения и идентификаторы либо только факт прохождения сообщения, его длину и частоту обмена, если содержимое сообщения не распознаваемо, т.е. выполнить анализ графика (потока сообщений) в данном канале. Т.к. пассивное воздействие не оставляет никаких явных следов, его трудно обнаружить.

Активные. Приводящие к определенным изменениям в системе. Данные изменения, как правило, дают нам принципиальную возможность обнаружения подобных воздействий. Нарушитель стремится подменить информацию, передаваемую в сообщении. Он может выборочно модифицировать, изменить или добавить правильное или ложное сообщение, удалить, задержать или изменить порядок следования сообщений. Злоумышленник может также аннулировать и задержать все сообщения, передаваемые по каналу. Подобные действия можно квалифицировать как отказ в передаче сообщений.

По наличию обратной связи:

С обратной связью. Дают возможность атакующему реагировать на изменения, происходящие на объекте атаки.

Без обратной связи. Действуют либо в автономном, либо в однонаправленном режиме. Характерны для атак на отказ в обслуживании.

По условию начала осуществления воздействия:

По запросу от атакуемого объекта. Запросом может служить наступление какого-то события, например вход пользователя в систему, либо отправка атакуемым объектом какого-то сообщения (например, DNS-запроса).

Безусловное. Инициатором воздействия является исключительно атакующий субъект.

По относительному расположению субъекта атаки (имеет смысл в современных маршрутизируемых сетях):

Внутрисегментное. Субъект и объект атаки расположены в одном логическом сегменте сети.

Межсегментное. Субъект и объект расположены в разных сегментах. Как правило, межсегментные воздействия гораздо сложнее осуществить, но они и представляют значительно большую опасность.

Причины успеха удаленных атак:

- отсутствие выделенного канала связи;
- недостаточные идентификация и аутентификация;
- взаимодействие объектов без установления виртуального канала;
- использование нестойких алгоритмов идентификации;
- отсутствие возможности контролировать маршрут сообщений;
- отсутствие полной информации об объектах сети;
- отсутствие криптозащиты сообщений.

Подробнее задача защиты распределенных систем будет рассмотрена в третьей части пособия.

В таблице 1.1 показаны основные пути реализации угроз безопасности АСОИ при воздействии на ее компоненты.

Таблица 1. 1
Пути реализации угроз безопасности АСОИ

Объекты воздействия	Нарушение конфиденциальности информации	Нарушение целостности информации	Нарушение работоспособности
Аппаратные средства	НСД-подключение, использование ресурсов, хищение носителей	НСД-подключение, использование ресурсов, модификация, изменение режимов	НСД-изменение режимов, вывод из строя, разрушение
Программное обеспечение	НСД-копирование, хищение, перехват	НСД, внедрение троянского коня, вирусов, червей	НСД- искажение, удаление, подмена
Данные	НСД-копирование, хищение, перехват	НСД- искажение, модификация	НСД- искажение, удаление, подмена
Персонал	разглашение, передача сведений о защите, халатность	"маскарад", вербовка, подкуп персонала	уход с рабочего места, физическое устранение

1.4. Программные средства воздействия на АСОИ

Программные средства воздействия на АСОИ представляют собой наиболее опасный вид угроз, которые используют последние достижения в области информационных технологий.

1.4.1. Типы вредоносного ПО

"Троянский конь" представляет собой программу, которая наряду с действиями, описанными в ее документации, выполняет некоторые другие действия, ведущие к нарушению безопасности системы и деструктивным результатам. Аналогия такой программы с древнегреческим "троянским

конем" вполне оправдана, так как в обоих случаях не вызывающая подозрений оболочка таит серьезную угрозу.

"Троянский конь" использует обман, чтобы побудить пользователя запустить программу со скрытой внутри угрозой. Обычно для этого утверждается, что такая программа выполняет некоторые весьма полезные функции. В частности, такие программы маскируются под какие-нибудь полезные утилиты.

Опасность "троянского коня" заключается в дополнительном блоке команд, вставленном в исходную безвредную программу, которая затем предоставляется пользователям АСОИ. Этот блок команд может срабатывать при наступлении какого-либо условия (даты, состояния системы) либо по команде извне. Пользователь, запустивший такую программу, подвергает опасности как свои файлы, так и всю АСОИ в целом. Приведем для примера некоторые деструктивные функции, реализуемые "троянскими конями".

- Уничтожение информации. Выбор объектов и способов уничтожения определяется фантазией автора вредоносной программы.
- Перехват и передача информации. В частности, известна программа, осуществляющая перехват паролей, набираемых на клавиатуре.
- Целенаправленная модификация текста программы, реализующей функции безопасности и защиты системы.

В общем, "троянские кони" наносят ущерб АСОИ посредством хищения информации и явного разрушения программного обеспечения системы. "Троянский конь" является одной из наиболее опасных угроз безопасности АСОИ. Радикальный способ защиты от этой угрозы заключается в создании замкнутой среды исполнения программ, которые должны храниться и защищаться от несанкционированного доступа.

Компьютерный «вирус» представляет собой своеобразное явление, возникшее в процессе развития компьютерной и информационной техники. Суть этого явления состоит в том, что программы-вирусы обладают рядом свойств, присущих живым организмам, — они рождаются, размножаются и умирают.

Компьютерный вирус — это программа, которая может заражать другие программы, модифицируя их посредством включения в них своей, возможно, измененной копии, причем последняя сохраняет способность к дальнейшему размножению. Ключевыми понятиями в определении компьютерного вируса являются способность вируса к саморазмножению и способность к модификации вычислительного процесса. Вирус обычно разрабатывается злоумышленниками таким образом, чтобы как можно дольше оставаться необнаруженным в компьютерной системе. Начальный период "дремоты" вирусов является механизмом их выживания. Вирус проявляется в конкретный момент времени, когда происходит некоторое событие вызова, например пятница 13-е, конкретная дата и т.п.

Компьютерный вирус пытается тайно записать себя на компьютерные диски. Способ функционирования большинства вирусов заключается в таком изменении системных файлов компьютера, чтобы вирус начинал свою деятельность при каждой загрузке. Например, вирусы, поражающие загрузочный сектор, пытаются инфицировать часть дискеты или жесткого диска, зарезервированную только для операционной системы и хранения файлов запуска. Эти вирусы особенно коварны, так как они загружаются в память при каждом включении компьютера. Такие вирусы обладают наибольшей способностью к размножению и. могут постоянно распространяться на новые диски.

Другая группа вирусов пытается инфицировать исполняемые файлы, чтобы остаться необнаруженными. Обычно вирусы отдают предпочтение EXE- или COM-файлам, применяемым для выполнения кода программы в компьютерной системе. Некоторые вирусы используют для инфицирования компьютерной системы как загрузочный сектор, так и метод заражения файлов. Это затрудняет выявление и идентификацию таких вирусов специальными программами и ведет к их быстрому распространению. Существуют и другие разновидности вирусов. Компьютерные вирусы наносят ущерб системе за счет многообразного размножения и разрушения среды обитания.

Сетевой "червь" представляет собой разновидность программы-вируса, которая распространяется по глобальной сети и не оставляет своей копии на магнитном носителе.

Этот термин используется для именования программ, которые подобно ленточным червям перемещаются по компьютерной сети от одной системы к другой. Первоначально "черви" были разработаны для поиска в сети других компьютеров со свободными ресурсами, чтобы получить возможность выполнить распределенные вычисления. При правильном использовании технология "червей" может быть весьма полезной. Например, "червь" World Wide Web Worm формирует индекс поиска участков Web. Однако "червь" легко превращается во вредоносную программу. "Червь" использует механизмы поддержки сети для определения узла, который может быть поражен. Затем с помощью этих же механизмов передает свое тело в этот узел и либо активизируется, либо ждет подходящих условий для активизации.

Сетевые "черви" являются самым опасным видом вредоносных программ, так как объектом их атаки может стать любой из миллионов

компьютеров, подключенных к глобальной сети Internet. Для защиты от "червя" необходимо принять меры предосторожности против несанкционированного доступа к внутренней сети. Следует отметить, что "троянские кони", компьютерные вирусы и сетевые "черви" относятся к наиболее опасным угрозам АСОИ.

Для защиты АСОИ от всех вышеперечисленных вредоносных программ необходимо применение ряда мер:

- исключение несанкционированного доступа к исполняемым файлам;
- тестирование приобретаемых программных средств;
- контроль целостности исполняемых файлов и системных областей;
- создание замкнутой среды исполнения программ.

1.4.2. Программные воздействия в Internet

Активный рост Internet привел и к росту источников возможных атак на конечных пользователей. В настоящее время World Wide Web– пожалуй, самая популярная служба Internet. Однако, далеко не все пользователи Internet осознают, что, подключившись к сети, они не только получают доступ ко всему информационному богатству, но и открывают свой компьютер для доступа извне, а следовательно, подвергают его угрозам, характерным для хостов Сети.

Вопросы безопасности в World Wide Web разбиваются на три основные группы:

- уязвимость клиентских систем;
- анонимность в сети;
- безопасность серверов.

Можно выделить следующие основные уязвимости клиентских систем:

- уязвимость клиентских приложений;

- уязвимости операционной системы;
- тесная интеграция приложений с ОС;
- разделенные ресурсы;
- вирусы и троянские кони:
- деструктивные;
- открывающие доступ к системе (BackOrifice);
- черви (в т.ч. почтовые);
- мифические (Good Times).

Популярные браузеры в своем развитии уже вышли далеко за рамки простых средств отображения гипертекстовых документов. **HTML** (HyperText Markup Language — язык разметки гипертекстовых документов) изначально был ориентирован исключительно на отображение структурированного текста. Имелась также возможность включить в документ некоторые управляющие элементы для передачи информации на сервер, который после этого мог вернуть клиенту результаты обработки, то есть чисто клиент-серверное решение. Этим и исчерпывались интерактивные возможности Web. По мере развития Сети стало понятно, что ее выразительных средств становится недостаточно для удовлетворения растущих запросов пользователей, и HTML стал включать в себя средства для работы с таблицами, графикой, звуком.

Кроме того, не все устраивало и разработчиков Web-приложений — иногда им хотелось бы иметь дело с более интеллектуальным клиентом, способным на нечто большее, чем простая передача на сервер заполненной формы. Возникла потребность во вспомогательных приложениях клиентской стороны. При разработке этих приложений немедленно всплывают многочисленные проблемы, и безопасность здесь стоит далеко не на последнем месте. Достаточно сказать, что большинство прорех в

системе безопасности браузеров, обнаруженных в последнее время, связано именно с элементами, расширяющими функциональность клиентов.

Наибольшую популярность завоевали следующие подходы к реализации вспомогательных приложений для клиентской стороны:

- подключаемые модули и расширения;
- элементы ActiveX;
- средства подготовки сценариев JavaScript, VBScript, Dynamic HTML;
- приложения на Java;
- Flash.

Использование подключаемых модулей получило в свое время широкое распространение в связи с популярностью браузера Netscape Navigator, предоставляющего такую возможность. С точки зрения безопасности этот подход не выдерживает никакой критики: не обеспечивается ни защита от сбоев, ни защита от злонамеренных действий, предпринимаемых модулем, который имеет полный доступ ко всем ресурсам системы пользователя. Все строится исключительно на доверии к автору модуля.

Управляющие элементы ActiveX — решение компании Microsoft, основанное на вездесущей технологии **COM** (Component Object Model — модель компонентных объектов), перенесенной на этот раз в Internet. Проблема безопасности решается с помощью введения института сертификатов — объекты ActiveX подписываются цифровой подписью автора, заверенной независимой организацией (например, VeriSign Inc.). Таким образом, работа с ActiveX отличается от работы с подключаемыми модулями Netscape только тем, что доверие к автору управляющего элемента может быть подкреплено авторитетом солидной организации. В то

же время эта подпись гарантирует лишь возможность определения авторства объекта, а вовсе не его благонадежности.

При загрузке объекта ActiveX поведение браузера зависит от настроек его системы безопасности — как подписанные, так и неподписанные (либо заверенные неизвестной организацией) объекты могут быть либо автоматически загружены или отвергнуты, либо предъявлены пользователю, с тем чтобы дальнейшее решение принимал он. В Internet Explorer также можно задать разные настройки для различных зон безопасности — для локальной сети, Internet, отдельных подозрительных хостов, и наоборот, достойных особого доверия.

JavaScript, VBScript и т. п. представляют собой упрощенные языки подготовки сценариев, код которых встраивается непосредственно в html-файл и выполняется браузером. Они непригодны для реализации серьезных приложений, в них отсутствуют средства для работы с файлами, сетевого взаимодействия и т. д. Но они широко используются во вспомогательных целях, в качестве средства первоначальной обработки результатов, для оформления, «оживления» html-документов и т. д. Казалось бы, что ограничения, присущие этим языкам, делают их абсолютно безопасными, в действительности же львиная доля ошибок в браузерах связана именно с реализацией этих простейших средств разработки.

Flash-приложения постепенно выросли из простейших анимированных роликов, используемых для украшения web-страниц, в полноценные приложения, которые готовятся с помощью довольно мощного языка программирования.

Java — язык, разработанный Sun Microsystems изначально для приложений бытовой электроники и позднее перенесенный в Internet, что стало для него вторым рождением. Различают обычные Java-приложения и

апплеты, предназначенные для загрузки по сети и выполнения в окне браузера.

Вопросы безопасности Java-апплетов заслуживают того, чтобы о них говорить более подробно, поскольку это первое распространенное средство разработки клиентских приложений, в котором решение проблемы безопасности предлагалось уже на уровне архитектуры, и до сих пор может считаться эталонным.

Основным достоинством Java-приложений является независимость от клиентской платформы. В отличие от традиционных приложений, транслирующихся в исполняемые коды процессора, Java-приложения транслируются в так называемый байт-код, интерпретируемый в дальнейшем виртуальной Java-машиной. При этом байт-код независим от платформы, на которой он в дальнейшем будет выполняться, — достаточно, чтобы для этой платформы существовала Java-машина. Поскольку большинство основных функций реализовано на уровне виртуальной Java-машины, это приводит к существенному уменьшению размеров байт-кода, что является как достоинством, так и недостатком Java-приложений, поскольку байт-код интерпретируются виртуальной машиной, производительность Java-приложений уступает производительности традиционных откомпилированных программ. Частично с этим удается бороться, применяя компиляторы времени исполнения (JIT — just in time compilers), осуществляющие компиляцию приложения при его загрузке в «родной» для данного процессора код. Также возможен вызов функций, реализованных на других языках программирования (C, C++) и откомпилированных для данной платформы, — так называемый **native code** (родной код). Он применяется при реализации наиболее критичных ко времени исполнения фрагментов кода.

Другим достоинством Java-приложений является защищенность. Во-первых, сам язык способствует написанию более надежных и устойчивых к сбоям программ. Синтаксис его восходит к C++, но помимо строгой типизации, управления доступом, работы с исключениями, знакомых программистам и по C++, в Java добавлена автоматическая «сборка мусора» (освобождение неиспользуемой памяти), проверка на выход за границы массива, возможность указать, что данный метод или объект не может быть изменен или переопределен. В языке нет указателей, множественного наследования, шаблонов и переопределенных операторов (о чем можно только сожалеть, но нельзя не признать, что все эти элементы C++ способны вызвать, особенно у новичков, огромное количество проблем).

Все эти нововведения помогают создавать более безопасный код. Рассмотрим теперь особенности Java, *вынуждающие* писать безопасный код.

По мере развития Java развивалась и система безопасности. В JDK 1.0 (Java Development Kit) основу системы безопасности составляли три компонента — **Verifier** (верификатор), **ClassLoader** (загрузчик классов) и **SecurityManager** (менеджер безопасности). Эта модель известна под названием **sandbox** (песочница), в ней выполняются Java-приложения, загруженные из сети (Рис. 1.1).

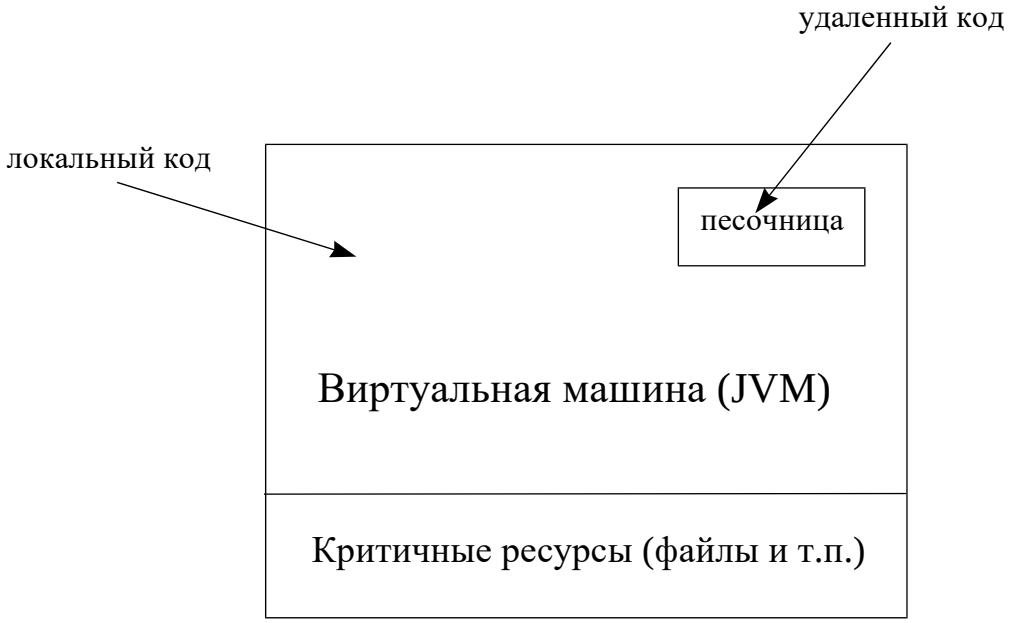


Рис. 1.1. Модель безопасности JDK 1.0

Для полноценного функционирования модели безопасности каждый ее компонент должен работать безошибочно, поскольку только их четкая совместная работа обеспечивает контроль над приложением во время загрузки и исполнения кода.

Первый рубеж обороны — верификатор, проверяющий загружаемый байт-код на корректность, так как у нас нет никакой гарантии, что загружаемый код был получен в результате работы компилятора Java, а не подправлен вручную или не сгенерирован специальным «враждебным» компилятором. После того как код прошел верификацию, гарантируется, что файл класса имеет корректный формат, параметры всех операций имеют правильный тип, в коде не происходит некорректных преобразований типов, например целого числа в указатель), нет нарушений доступа и т. п. Таким образом, проверяется все, что только можно проверить до начала исполнения программы. Верификатор встроен в виртуальную машину и недоступен из Java-программы.

Загрузчики классов определяют, когда и каким образом классы могут быть добавлены в работающую систему. Частью их работы является защита важных составляющих системы, например запрет на загрузку поддельного менеджера безопасности. Они выполняют две основные функции — собственно загрузку байт-кода (с локального диска, по сети, из области памяти и т. д.), определение **namespaces** (пространства имен) для различных классов и способы их взаимодействия (отделяя, к примеру, локальные классы от загруженных по сети).

Существует два вида загрузчиков — **primordial** (первичный) и **Class Loader Object** (реализованный в виде объекта). Первичный существует в единственном экземпляре и является составной частью виртуальной машины. Он занимается загрузкой доверенных классов (обычно находящихся на локальном диске). Загрузчик второго типа представляет собой экземпляр обычного Java-класса, унаследованного от абстрактного класса java.lang.ClassLoader. С его помощью можно осуществить загрузку класса по сети либо динамическое конструирование приложением.

Загрузчик работает совместно с менеджером безопасности, определяющим, можно ли загружать данный класс. Весь алгоритм действий загрузчика обычно выглядит следующим образом:

1. Определить, не был ли загружен этот класс раньше, и, если да, вернуть его.
2. Проконсультироваться с первичным загрузчиком на предмет существования внутреннего класса с этим именем.
3. Запросить у менеджера безопасности разрешение на загрузку данного класса.
4. Считать файл класса в виде массива байтов — по сети, с диска и т. п.
5. Создать экземпляр класса Class.

6. Загрузить используемые классы.
7. Передать класс верификатору на проверку.

Загрузчик является существенным элементом модели безопасности, поэтому писать загрузчики следует особо осторожно — малейшая ошибка может привести к полному краху всей системы безопасности. В нормальной ситуации апплеты не имеют возможности устанавливать свои загрузчики.

Класс SecurityManager отвечает за политику безопасности приложения. Он позволяет приложению перед выполнением потенциально опасной операции выяснить, выполняется ли она классом, загруженным первичным загрузчиком, либо с помощью некоторого ClassLoader'a (к последним, особенно при загрузке из сети, доверия должно быть гораздо меньше). Далее менеджер безопасности может определить, разрешить ли эту операцию или наложить на нее вето. Класс SecurityManager определяет ряд методов, начинающихся со слова «check» (checkDelete, checkExec, checkConnect и т. п.), которые вызываются всеми методами стандартной библиотеки, выполняющими потенциально опасные действия (работа с файлами, сетевыми соединениями и т. п.).

В JDK 1.1 система безопасности получила дальнейшее развитие (Рис. 1.2). Принципиально ничего не изменилось, но была добавлена возможность цифровой подписи классов — аналог сертификатов в ActiveX. Теперь можно решить, заслуживает ли подписанный удаленный код полного доверия, и не накладывать на него стандартные ограничения.

Побочным эффектом введения этого механизма стало появление в составе стандартной библиотеки Java криптографических функций — Crypto API.

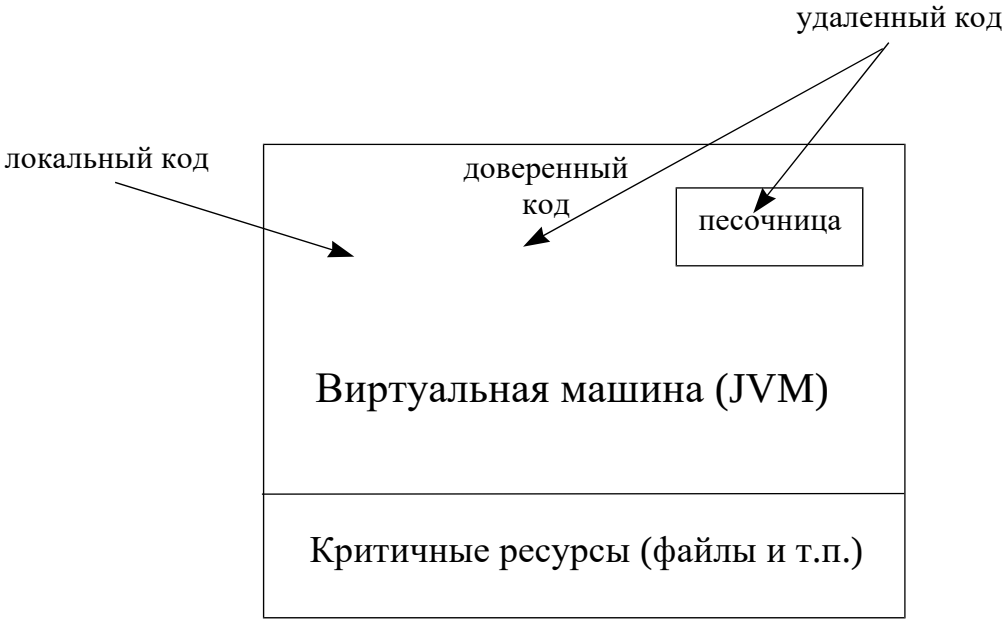


Рис. 1.2. Модель безопасности JDK 1.1

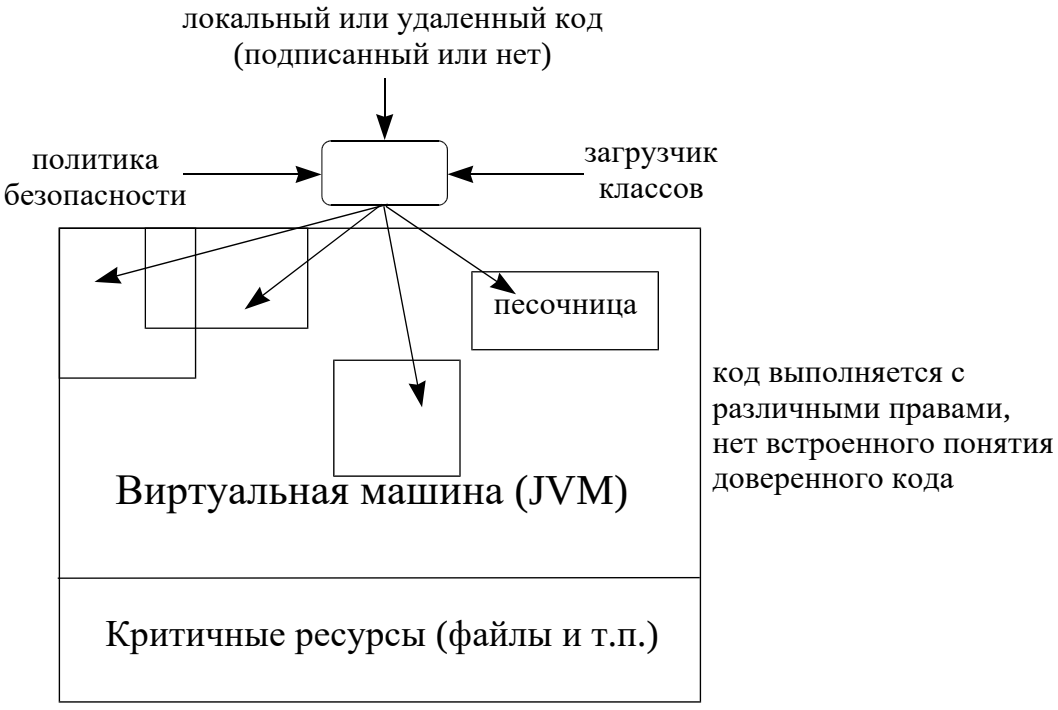


Рис. 1.3. Модель безопасности JDK 1.3

Модель безопасности JDK 1.1 отличалась черно-белым взглядом на мир — мы либо полностью доверяем загруженному коду, либо нет. В Java 2 (JDK 1.2), вышедшей в декабре 1998, была представлена новая модель безопасности, основанная на привилегиях и правах доступа (Рис. 1.3). Теперь появилась возможность гибко настраивать права доступа в зависимости от места расположения кода и его подписи.

Несмотря на чувство полной анонимности своих действий, которое возникает у многих пользователей сети, на самом деле клиентские приложения, с помощью которых они получают информацию, оставляют довольно много информации о системах, на которых они запущены.

- Информация, которую оставляет браузер:
- IP-адрес (клиента или proxy-сервера);
- тип браузера, версия операционной системы;
- адрес предыдущей страницы;
- cookie.
- Информация из почтовых программ:
- IP-адрес отправителя и путь письма в заголовке сообщения.
- Средства интерактивного общения:
- IP-адрес собеседника (в явном виде, или с использованием дополнительных средств).

Обеспечить полную анонимность в современых условиях довольно сложно, но существует ряд средств, помогающих полностью или частично решить эту задачу:

- использование proxy-серверов;
- анонимные службы;
- web-анонимайзеры;
- почтовые ремейлеры;

- распределенные системы — Freedom, Tor.
- зоны безопасности Internet Explorer;
- программы-фильтры, обеспечивающие блокировку cookie, фильтрацию http-заголовков и т.п.
- локальные proxy;
- контролирующие весь сетевой трафик;
- SOCKS-proxy.

Для серверных систем можно выделить следующие основные уязвимости:

- уязвимости операционной системы;
- уязвимости сетевых служб, особенно web-серверов;
- уязвимость серверных приложений:
- расчет на «хорошее» поведение пользователя;
- переполнение буфера;
- вызов внешних программ;
- рассчитывающие на определенные значения внешних переменных;
- хранение критичной информации в открытых для доступа файлов;
- возможность использования для атаки на других пользователей (XSS).

Устанавливая последнюю версию Web-сервера, мы можем быть уверены хотя бы в том, что она не содержит очевидных ошибок, опубликованных по всей Сети пару лет тому назад. На появление новых ошибок производители реагируют достаточно быстро, а задача администратора сводится к тому, чтобы быть в курсе происходящего. С серверными приложениями ситуация несколько иная — на сей раз в роли разработчика зачастую выступают владельцы сайтов, которые должны сами заботиться о безопасности приложений. Не стоит особо доверять и готовым скриптам — огромное количество ошибок обнаруживается именно в

примерах, поставляемых вместе с Web-серверами, а также во многих популярных скриптах.

Ошибки в серверных web-приложениях коварны, от них нельзя защититься установкой межсетевого экрана, поскольку атака с их использованием производится на уровне разрешенного протокола HTTP, и даже использование дорогостоящей системы обнаружения атак на прикладном уровне мало чем поможет в предотвращении атаки на самостоятельно написанное корпоративное приложение.

Успешно же проведенная подобная атака вполне может оказаться первым шагом на пути ко взлому всей корпоративной сети.

Практически все беды начинающих программистов проистекают из надежды на то, что пользователь будет себя вести «хорошо» и обращаться с программой именно так, как задумано автором. Это справедливо не только для CGI-приложений, но и для любого программного обеспечения, однако когда автор многочисленных утилит «для себя» решает попробовать свои силы в программировании для серверов, он немедленно попадает в другую весовую категорию. Ему может казаться, что он продолжает писать «для себя», в действительности же его потенциальными пользователями становятся все обитатели сети, а уж от них ждать снисхождения не приходится. И не стоит успокаивать себя тем, что CGI-приложения выполняются в контексте пользователя с минимальными правами — даже в хорошо сконфигурированной системе этих прав зачастую достаточно для выдачи информации, которой можно воспользоваться при взломе системы.

1.5. Обеспечение безопасности АСОИ

Основным назначением АСОИ является переработка (сбор, хранение, обработка и выдача) информации, поэтому проблема обеспечения информационной безопасности является для АСОИ центральной.

Обеспечение безопасности АСОИ предполагает организацию противодействия любому несанкционированному вторжению в процесс функционирования АСОИ, а также попыткам модификации, хищения, выведения из строя или разрушения ее компонентов, т.е. защиту всех компонентов АСОИ — аппаратных средств, программного обеспечения, данных и персонала.

Существуют два подхода к проблеме обеспечения безопасности АСОИ: "фрагментарный" и комплексный.

"Фрагментарный" подход направлен на противодействие четко определенным угрозам в заданных условиях. В качестве примеров реализации такого подхода можно указать отдельные средства управления доступом, автономные средства шифрования, специализированные антивирусные программы и т.п.

Достоинством такого подхода является высокая избирательность к конкретной угрозе. Существенным недостатком данного подхода является отсутствие единой защищенной среды обработки информации. Фрагментарные меры защиты информации обеспечивают защиту конкретных объектов АСОИ только от конкретной угрозы. Даже небольшое видоизменение угрозы ведет к потере эффективности защиты.

Комплексный подход ориентирован на создание защищенной среды обработки информации в АСОИ, объединяющей в единый комплекс разнородные меры противодействия угрозам. Организация защищенной среды обработки информации позволяет гарантировать определенный уровень безопасности АСОИ, что является несомненным достоинством комплексного подхода. К недостаткам этого подхода относятся: ограничения на свободу действий пользователей АСОИ, большая

чувствительность к ошибкам установки и настройки средств защиты, сложность управления.

Комплексный подход применяют для защиты АСОИ крупных организаций или небольших АСОИ, выполняющих ответственные задачи или обрабатывающих особо важную информацию. Нарушение безопасности информации в АСОИ крупных организаций может нанести огромный материальный ущерб как самим организациям, так и их клиентам. Поэтому такие организации вынуждены уделять особое внимание гарантиям безопасности и реализовывать комплексную защиту. Комплексного подхода придерживаются большинство государственных и крупных коммерческих предприятий и учреждений. Этот подход нашел свое отражение в различных стандартах.

Комплексный подход к проблеме обеспечения безопасности основан на разработанной для конкретной АСОИ политике безопасности.

Политика безопасности представляет собой набор норм, правил и практических рекомендаций, на которых строится управление, защита и распределение информации в АСОИ. Политика безопасности регламентирует эффективную работу средств защиты АСОИ. Она охватывает все особенности процесса обработки информации, определяя поведение системы в различных ситуациях.

Политика безопасности реализуется посредством административно-организационных мер, физических и программно-технических средств и определяет архитектуру системы защиты. Для конкретной организации политика безопасности должна носить индивидуальный характер и зависеть от конкретной технологии обработки информации и используемых программных и технических средств.

1.6. Этапы построения системы защиты АСОИ

Процесс построения системы защиты включает следующие этапы:

- анализ возможных угроз АСОИ;
- планирование системы защиты;
- реализация системы защиты;
- сопровождение системы защиты.

Этап анализа возможных угроз АСОИ необходим для фиксации состояния АСОИ (конфигурации аппаратных и программных средств, технологии обработки информации) и определения учитываемых воздействий на компоненты системы. Практически невозможно обеспечить защиту АСОИ от всех воздействий, поскольку невозможно полностью установить все угрозы и способы их реализации. Поэтому из всего множества вероятных воздействий выбирают только такие воздействия, которые могут реально произойти и нанести серьезный ущерб.

На ***этапе планирования*** формулируется система защиты как единая совокупность мер противодействия угрозам различной природы.

Результатом этапа планирования является развернутый план защиты АСОИ, содержащий перечень защищаемых компонентов АСОИ и возможных воздействий на них, цель защиты информации в АСОИ, правила обработки информации в АСОИ, обеспечивающие ее защиту от различных воздействий, а также описание планируемой системы защиты информации.

Сущность ***этапа реализации*** системы защиты заключается в установке и настройке средств защиты, необходимых для реализации запланированных правил обработки информации.

Заключительный ***этап сопровождения*** заключается в контроле работы системы, регистрации происходящих в ней событий, их анализе с целью обнаружения нарушений безопасности, коррекции системы защиты.

1.7. Меры обеспечения безопасности

Меры обеспечения безопасности компьютерных систем подразделяют на:

- правовые (законодательные);
- морально-этические;
- административные;
- физические;
- аппаратно-программные.

Перечисленные меры безопасности АСОИ можно рассматривать как последовательность барьеров или рубежей защиты информации. Для того чтобы добраться до защищаемой информации, нужно последовательно преодолеть, несколько рубежей защиты. Рассмотрим их подробнее.

Первый рубеж защиты, встающий на пути человека, пытающегося осуществить НСД к информации, является чисто правовым. Этот аспект защиты информации связан с необходимостью соблюдения юридических норм при передаче и обработке информации. К ***правовым мерам*** защиты информации относятся действующие в стране законы, указы и другие нормативные акты, регламентирующие правила обращения с информацией ограниченного использования и ответственности за их нарушения. Этим они препятствуют несанкционированному использованию информации и являются сдерживающим фактором для потенциальных нарушителей. Законодательная база в России отстает от потребностей практики. Имеющиеся законы и указы имеют в основном запретительный характер:

- Федеральный Закон "Об информации, информатизации и защите информации";
- Указ президента РФ от 3 апреля 1995 г. N 334 "О мерах по соблюдению законности в области разработки, производства,

реализации и эксплуатации шифровальных средств, а также предоставления услуг в области шифрования информации";

С 1997 года начали действовать новые статьи УК РФ, где указывается возможная уголовная ответственность, которую могут нести граждане РФ за преступления в сфере компьютерной информации.

Глава 28 УК РФ:

Статья 272. Неправомерный доступ к компьютерной информации.

Статья 273. Создание, использование и распространение вредоносных программ для ЭВМ.

Статья 274. Нарушение правил эксплуатации ЭВМ, системы ЭВМ или их сети.

Второй рубеж защиты образуют ***морально-этические меры***. Этический момент в соблюдении требований защиты имеет весьма большое значение. К морально-этическим мерам противодействия относятся всевозможные нормы поведения, которые традиционно сложились или складываются в обществе по мере распространения компьютеров в стране. Эти нормы большей частью не являются обязательными, как законодательно утвержденные, но их несоблюдение обычно ведет к падению престижа человека, группы лиц или организации.

Третьим рубежом, препятствующим неправомочному использованию информации, являются ***административные меры***. Администраторы всех рангов с учетом правовых норм и социальных аспектов определяют административные меры защиты информации.

Административные меры защиты относятся к мерам организационного характера. Они регламентируют:

- процессы функционирования АСОИ;
- использование ресурсов АСОИ;

- деятельность ее персонала;
- порядок взаимодействия пользователей с системой, с тем чтобы в наибольшей степени затруднить или исключить возможность реализации угроз безопасности.

Административные меры включают:

- разработку правил обработки информации в АСОИ;
- совокупность действий при проектировании и оборудовании вычислительных центров и других объектов АСОИ (учет влияния стихии, пожаров, охрана помещений и т.п.);
- совокупность действий при подборе и подготовке персонала (проверка новых сотрудников, ознакомление их с порядком работы с конфиденциальной информацией, с мерами ответственности за нарушение правил ее обработки; создание условий, при которых персоналу было бы невыгодно допускать злоупотребления и т.д.)
- организацию надежного пропускного режима;
- организацию учета, хранения, использования и уничтожения документов и носителей с конфиденциальной информацией;
- распределение реквизитов разграничения доступа (паролей, полномочий и т.п.);
- организацию скрытого контроля за работой пользователей и персонала АСОИ;
- совокупность действий при проектировании, разработке, ремонте и модификации оборудования и программного обеспечения (сертификация используемых технических и программных средств, строгое санкционирование, рассмотрение и утверждение всех изменений, проверка на удовлетворение требованиям защиты, документальная фиксация изменений и т.п.).

Важно отметить, что, пока не будут реализованы действенные меры административной защиты ЭВМ, прочие меры будут, несомненно, неэффективны.

Четвертым рубежом являются ***физические меры защиты***. К физическим мерам защиты относятся разного рода механические, электро- и электронно-механические устройства или сооружения, специально предназначенные для создания физических препятствий на возможных путях проникновения и доступа потенциальных нарушителей к компонентам системы и защищаемой информации.

Пятым рубежом являются ***аппаратно-программные средства защиты***. К ним относятся различные электронные устройства и специальные программы, которые реализуют самостоятельно или в комплексе с другими средствами следующие способы защиты:

- идентификацию и аутентификацию субъектов (пользователей, процессов) АСОИ;
- управление доступом к ресурсам АСОИ;
- протоколирование и аудит;
- криптография;
- экранирование.

1.8. Принципы проектирования системы защиты.

Основным требованием при разработке системы защиты является параллельное проектирование системы защиты с проектированием информационной системы. Невыполнение этого принципа, т.е. встраивание средств защиты в уже готовую систему, может привести к низкой эффективности защиты, снижению производительности вычислительных средств, а также к большим стоимостным затратам.

Техническое задание на проектируемую информационную систему должно содержать перечень сведений и характеристик, подлежащих защите, возможные пути циркуляции и места их сосредоточения, а также специальные требования к системе защиты информации.

Решение вопросов создания СЗИ должно поручаться лицам одного уровня с лицами, занимающимися вопросами функционирования АСОД. Разработка СЗИ требует привлечения специалистов широкого профиля, знающих, кроме системных вопросов, вопросов программного обеспечения, разработки комплексов и отдельных технических средств, специальные вопросы защиты информации.

Важную роль играет простота СЗИ. Она должна быть настолько простой, насколько позволяют требования по ее эффективности. Простота защиты повышает ее надежность, экономичность, уменьшает ее влияние на вероятностные характеристики информационной системы. При неудобных средствах защиты пользователь будет стараться найти пути обхода ее, отключать ее механизм, что сделает защиту бесполезной.

Можно кратко сформулировать ряд принципов, отображающих основные положения по безопасности информации, которые целесообразно учитывать при проектирование системы СЗИ:

- Экономическая эффективность. Стоимость средств защиты должна быть меньше, чем размеры возможного ущерба.
- Минимум привилегий. Каждый пользователь должен иметь минимальный набор привилегий, необходимый для работы.
- Принцип враждебного окружения. Система защиты должна проектироваться в расчете на враждебное окружение. Разработчики должны исходить из предположения, что пользователи имеют наихудшие намерения, что они будут совершать серьезные ошибки и искать пути обхода механизмов защиты.
- Отключаемость защиты. При нормальном функционировании защита не должна отключаться. Только в особых случаях сотрудник со специальными полномочиями может отключить систему защиты.
- Привлечение человека. Наиболее важные и критические решения должны приниматься человеком.
- Простота. Защита тем более эффективна, чем легче пользователю с ней работать.
- Открытость проектирования и функционирования механизмов защиты. Специалисты, имеющие отношение к системе защиты, должны полностью представлять себе принципы ее функционирования и адекватно реагировать на затруднительные ситуации.
- Всеобщий контроль. Любые исключения из множества контролируемых субъектов и объектов защиты снижают защищенность автоматизированного комплекса обработки информации.
- Независимость системы защиты от субъектов защиты. Лица, занимавшиеся разработкой системы защиты, не должны быть в числе тех, кого эта система будет контролировать.

- Отчетность и подконтрольность. Система защиты должна предоставлять доказательства корректности своей работы.
- Ответственность. Подразумевается личная ответственность лиц, занимающихся обеспечением безопасности информации.
- Отсутствие излишней информации о существовании механизмов защиты. Существование механизмов защиты должно быть по возможности скрыто от контролируемых пользователей.
- Изоляция и разделение. Объекты защиты целесообразно разделять на группы таким образом, чтобы нарушение защиты в одной из групп не влияло на безопасность других групп.
- Полнота и согласованность. Надежная система защиты должна быть полностью специфицирована, протестирована и согласована.
- Параметризация. Защита становится более эффективной и гибкой, если она допускает изменение своих параметров со стороны администратора.

2. Идентификация и аутентификация

2.1. Понятия идентификации и аутентификации

С каждым зарегистрированным в АСОИ субъектом (пользователем или процессом, действующим от имени пользователя) связана некоторая информация, однозначно идентифицирующая его. Это может быть число или строка символов, именующие данный субъект. Такую информацию называют идентификатором субъекта. Имея идентификатор, зарегистрированный в сети, пользователь считается легальным (законным).

Прежде чем получить доступ к ресурсам АСОИ, пользователь должен пройти процесс первичного взаимодействия с системой, который включает две стадии: идентификацию и аутентификацию.

Идентификация — это процедура распознавания пользователя по его идентификатору (имени). Эта функция выполняется в первую очередь, когда пользователь делает попытку войти в АСОИ. Он сообщает системе по ее запросу свой идентификатор и система проверяет в своей базе данных его наличие.

Аутентификация — процедура проверки подлинности, позволяющая достоверно убедиться, что пользователь является именно тем, кем он себя объявляет. Обычно пользователь подтверждает свою идентификацию, вводя в систему уникальную информацию о себе. Для подтверждения своей подлинности субъект может предъявлять системе разные сущности. В зависимости от предъявляемых сущностей процедуры аутентификации могут быть разделены на следующие категории:

- на основе знания чего-либо (пароль, персональный идентификационный номер PIN);
- на основе обладания чем-либо (магнитные карты, смарт-карты, сертификаты, устройства touch-memory и персональные генераторы, которые используются для создания одноразовых паролей);
- на основе каких-либо неотъемлемых характеристик, т. е. проверка биометрических характеристик человека (голос, отпечатки пальцев, сетчатка глаза).

Идентификация и аутентификация — взаимосвязанные процессы распознавания и проверки подлинности субъектов (пользователей). От них зависит решение системы, можно ли разрешить доступ к ресурсам системы конкретному пользователю или процессу. Обычно подлинность устанавливается один раз, но в системах с высокой степенью безопасности может проводиться периодическая перепроверка или проверка в определенных условиях, например после системного сбоя.

2.2. Пароли

2.2.1. Схемы использования паролей

Одним из распространенных методов аутентификации является применение пароля и хранение его значения в ВС. Существует несколько типов паролей.

1. Метод **простого пароля** требует, чтобы пользователь ввел строку символов для проверки их в ЭВМ. В схеме с простым паролем пользователю обычно разрешается самому выбрать себе пароль, чтобы его было легко запомнить.

Чем больше длина пароля, тем большую безопасность будет обеспечивать система, так как нарушителю потребуются большие усилия для отгадывания пароля.

Недостатком простого пароля является то, что им может пользоваться нарушитель без ведома зарегистрированного пользователя. Одним из путей решения этой проблемы является выдача системой на терминал пользователя очередного номера, за которым зарегистрировано его обращение к системе на данный день.

Схема простого пароля имеет модификации:

1.1. Выборка символов — при обращении пользователя к системе его могут попросить ввести отдельные символы из пароля. Каждый раз это будут другие буквы. Такой прием не позволит нарушителю при пассивном перехвате в сети узнать пароль.

1.2. Пароль однократного использования — пользователю заранее выдается список из N паролей. Такой же список в зашифрованном виде храниться в ЭВМ. После использования пароля пользователь вычеркивает его из списка, в следующий раз введет очередной пароль из списка.

Схема паролей однократного использования имеет серьезный недостаток: пользователь должен помнить или иметь при себе весь список паролей и следить за текущим.

Пароли однократного использования могут применяться также для установления подлинности окончания сеанса как со стороны пользователя, так и ЭВМ. Всякий раз, когда пользователь завершает работу, ЭВМ передает ему свой пароль однократного использования и затем прерывает связь. Если пользователь был отключен, но не получил истинного пароля от ЭВМ, ему следует принять меры предосторожности. Эта ситуация может означать следующее: нарушитель прослушивает незащищенную линию связи, посылает пользователю ложное сообщение об отключении, а затем использует линию связи по своему усмотрению.

Пароль однократного использования можно применять при относительно редких обращениях к системе или при передаче специальной информации.

2. При использовании метода **"ЗАПРОС-ОТВЕТ"** применяют набор ответов на M стандартных и N ориентированных на пользователя вопросов. Система задает пользователю некоторые (или все) вопросы в произвольном порядке.

Ст Год рождения вашего отца?

Ст Номер вашей школы?

По В каком городе вы жили в 1993 году?

3. В некоторых случаях система защиты может потребовать, чтобы пользователь доказал свою подлинность с помощью корректной обработки алгоритмов. Это процедура носит название "рукопожатие" (*handshake*), она может быть выполнена как между пользователем и ЭВМ, так и между двумя ЭВМ.

2.2.2. Стойкость паролей

При выборе пароля следует определить, каким должна быть его длина и стойкость к несанкционированному подбору. Естественно, чем больше длина пароля, тем большую безопасность будет обеспечивать система. В системах защиты информации определяют ожидаемое время раскрытия пароля или ожидаемое безопасное время **Тб.**

Ожидаемое безопасное время **Тб** — это полупроизведение числа возможных паролей и времени, необходимое чтобы проверить каждый пароль из последовательности запросов.

$T_б = 0.5 * (A^s * E / R)$

R — скорость передачи символов в линии связи (символы/мин);

E — число символов в передаваемом сообщении;

S — длина пароля;

A — число символов в алфавите, из которых составляется пароль (26 — английский, 32 русский).

Пример: при R=600 (сим/мин), E=6, S=6, A=26

$T_б = 0.5 * (26^6 * 6 / 600) = 107$ дней

Если после каждой неудачной попытки подбора предусматривается задержка в 10 секунд, безопасное время резко увеличивается.

Рассмотрим формулу Андерсона, в которую также входит вероятность P того, что данный пароль может быть раскрыт посторонним лицом за время M, в течении которого могут быть предприняты попытки (в месяцах при работе по 24 часа в сутки).

$4{,}32 * 10^4 * (R * M / E * P) <= A^s$

Если значения R,E,M,A фиксированы, то каждая длина пароля будет давать различную вероятность P отгадывания пароля.

Таблица 2. 1
Количество возможных ключей и время тотального перебора (перебор ведется со скоростью 1 млн. вариантов/сек)

	6 байт	7 байт	8 байт
Строчные буквы (26)	$3.2*10^{8}$ 6 мин	$8.\ 11*10^{9}$ 2.3 час	$2.2*10^{11}$ 2.5 дн
Строчные буквы и цифры (36)	$2.3*10^{9}$ 37 мин	$7.9*10^{10}$ 23 час	$2.9*10^{12}$ 34 дн
Буквы и цифры (62)	$5.8*10^{10}$ 17 час	$3.6*10^{11}$ 42 дн	$2.3*10^{14}$ 7 лет
Печатаемые символы (95)	$7.5*10^{11}$ 8.6 дн	$7.1*10^{13}$ 2.3 лет	$6.7*10^{15}$ 211 лет
Все ASCII символы	$2.9*10^{14}$ 9 лет	$7.3*10^{16}$ 2400 лет	$1.9*10^{19}$ 590000 лет

Пример: требуется определить длину пароля, чтобы вероятность P его отгадывания была не более 0,001 после трехмесячного систематического тестирования (M=3) для английского алфавита. Скорость передачи R=600 сим/мин, за одну попытку посылается E=20 символов:

$4.32 * 10^{4} * 3 * 10^{3} * 600 / 20 <= 26^{s}$ или $3.888 *10^{9} <= 26^{s}$

Для S=6 $26^{s} = 3.08 *10^{8}$, т.е. $< 3.888*10^{9}$

Для S=7 $26^{s} = 8.08 *10^{9}$, т.е. $> 3\ .888*10^{9}$

Следовательно, выбираем длину пароля 7 символов.

В настоящее время широко используются 8-10-ти символьные пароля, которые часто разбиваются на две части (пользователь-карточка).

2.2.3. Повышение надежности парольной защиты

- Обучение и убеждение пользователей в необходимости соблюдения мер повышенной безопасности.
- Наложение технических ограничений: автоматическая проверка длины пароля, т.е. он не должен быть слишком коротким.
- Использование программных генераторов паролей, которые на основе простых правил, формируют благозвучные, но может быть бессмысленные пароли, которые легко запоминаются.

- Управление сроком действия паролей, их периодическая смена, т.к. за большой период времени увеличивается вероятность их перехвата по сети, путем прямого хищения носителя, снятия его копии и пр.
- Ограничение доступа к хранилищу паролей (несмотря на то, что в современных системах, как правило, хранятся не пароли, а их хэши, по которым невозможно непосредственное восстановление пароля, прямой доступ к ним существенно облегчает задачу взломщика).
- Ограничение числа неудачных попыток входа в систему, что затруднит подбор пароля перебором.

Что **не надо** делать:

- В качестве пароля не следует использовать ваше имя, фамилию или имена членов семьи, детские прозвища, клички домашних животных, т.к. нарушитель может знать вашу личную жизнь.
- Следует избегать повторений символов (4488), т.к. они легче подбираются программно.
- Не используйте для пароля профессиональный жаргон и вообще любые "реальные" слова. Программы подбора паролей могут проверить сотни тысяч слов за считаные минуты.
- Не надеяться на "хитрые" трюки наподобие ввода реального слова в другой раскладке клавиатуры — соответствующая модификация словарей для программ взлома тривиальна и неоднократно проделана.

Что **надо** делать:

- Выбирайте длинные пароли, по 6-8 символов.
- Меняйте регистры и используйте знаки препинания.
- Выбирайте легко запоминающийся пароль, чтобы избежать его записывания.

- Выбирайте пароль, который легко вводить, чтобы нельзя было угадать по движению рук, ведь если пароль сложный, его вводят медленнее.
- "Два слова в два приема" — в крайнем случае, для упрощения запоминания используйте в пароле два простых слова, разделенных знаком препинания (моЙ+Дом, кОт=сЫр), или формируйте целые предложения.

Если в результате проверки система выдала отказ в доступе, то необходимо вести регистрацию всей относящейся к делу информации и дополнительно к этому следует ввести временную задержку, в течении которой будут игнорироваться новые попытки доступа к системе от того же источника. Это необходимо сделать для уменьшения вероятности раскрытия паролей с помощью программного подбора. Если количество попыток ввести пароль превышает заданное число, то происходит отключение пользователя от системы.

3. Управление доступом

3.1. Средства управления доступом

Средства управления доступом позволяют определять и контролировать действия, которые субъекты (пользователи и процессы) могут выполнять над объектами (информацией или другими компьютерными ресурсами). Логическое управление доступом — это основной механизм многопользовательских систем, призванный обеспечить конфиденциальность и целостность объектов.

Если дается разрешение на выполнение затребованного действия, говорят, что объект, осуществляющий запрос, имеет полномочия по

отношению к указанному элементу данных. Элементом данных может быть файл, запись, поле, отношение или некоторая другая структура.

При принятии решения о предоставлении доступа обычно анализируется следующая информация:

- Идентификатор субъекта (идентификатор субъекта, сетевой адрес компьютера и т.д.). Подобные идентификаторы являются основой произвольного управления доступом.
- Атрибуты субъекта (метка безопасности, группа пользователя). Метки безопасности являются основой принудительного управления доступом.
- Место действия (системная консоль, узел в сети и т.п.).
- Время действия (большинство действий целесообразно разрешать только в рабочее время).
- Внутренние ограничения сервиса (число одновременно работающих пользователей, сумма денег, которую разрешено выдавать наличными и пр.)

3.2. Произвольное управление доступом

Произвольное (добровольное, дискреционное) управление доступом — это метод ограничения доступа к объектам, основанный на учете личности субъекта или группы, в которую субъект входит. Добровольность управления состоит в том, что некоторое лицо (администратор) по своему усмотрению может давать другим субъектам или отбирать у них права доступа к объекту. Администратор задает множество разрешенных отношений доступа, например в виде троек <объект, субъект, тип доступа>. Обычно для описания свойств избирательного управления доступом применяют математическую модель на основе матрицы доступа. ***Матрица доступа (полномочий)*** представляет

собой матрицу, в которой столбец соответствует объекту системы, а строка — субъекту. На пересечении столбца и строки матрицы указывается тип разрешенного доступа субъекта к объекту. Обычно выделяют такие типы доступа субъекта к объекту, как "доступ на чтение", "доступ на запись", "доступ на исполнение" и т.п.

Текущее состояние прав доступа (профиль полномочий) описывается матрицей полномочий: по строкам перечислены субъекты, в столбцах — объекты, на пересечении записываются способы доступа, допустимые для субъекта по отношению к объекту. Каждый элемент A(i,j) в матрице установления полномочий определяет права доступа i-го пользователя к j-му ресурсу.

00 — в i-строке и j-м столбце означает, что терминалу из i-й строки запрещены все виды доступа к элементу данных, описанному в j-м столбце.

01 — означает право читать;

10 — означает право писать;

11 — означает, что с терминала можно как читать, так и записывать элемент данных.

Таблица 3. 1
Пример матрицы полномочий

Место расположения терминала	Имя служащего	Адрес	Таб. №	Квалификация	Данные об окладе	Прогноз объема продаж	Цена закупки
Отдел кадров	11	11	11	11	11	00	00
Касса	01	00	01	00	11	00	00
Отдел сбыта	00	00	00	01	00	11	01
Снабжение	00	00	00	00	00	00	11
Отдел маркетинга	00	00	01	01	00	00	01

Дополнительно к правам ***чтение***, ***запись*** существуют другие общие типы прав: ***исполнение*** (исполнить процедуру, когда элементом данных является процедура), ***удаление*** (удалить элемент данных из базы данных) и

присоединение (добавить что-либо к концу элемента данных без изменения его первоначального содержания).

Элементы матрицы установления полномочий обычно содержат биты, соответствующие действиям, которые могут быть выполнены с терминала при обращении к элементу данных. При необходимости элементы матрицы могут содержать указатели на процедуры проверки. Процедуры могут принимать решения о доступе с учетом многих факторов, например время суток, текущее состояние базы данных.

МУП является центральным звеном системы обеспечения безопасности. За счет включения большего или меньшего количества информации в МУП можно изменять сложность контрольных проверок, зависимых от содержания данных. Обычно МУП хранится как отдельный зашифрованный файл и его строки помещаются в оперативную память только в случае необходимости.

Произвольное управление доступом широко применяется в АСОИ коммерческого сектора, так как ее реализация соответствует требованиям коммерческих организаций по разграничению доступа и подотчетности, а также имеет приемлемую стоимость.

3.3. Принудительное управление доступом

Принудительное (мандатное, нормативное) управление доступом — управление, основанное на совокупности правил предоставления доступа, определенных на множестве атрибутов безопасности субъектов и объектов, т.е. с объектами и субъектами ассоциируются метки безопасности.

Принудительное управление доступом подразумевает, что:

- все субъекты и объекты системы однозначно идентифицированы;

- каждому объекту системы присвоена метка конфиденциальности информации, определяющая ценность содержащейся в нем информации;
- каждому субъекту системы присвоен определенный уровень допуска, определяющий максимальное значение метки конфиденциальности информации объектов, к которым субъект имеет доступ.

Чем важнее объект, тем выше его метка конфиденциальности. Поэтому наиболее защищенными оказываются объекты с наиболее высокими значениями метки конфиденциальности.

Основным назначением полномочной политики безопасности является регулирование доступа субъектов системы к объектам с различными уровнями конфиденциальности, предотвращение утечки информации с верхних уровней должностной иерархии на нижние, а также блокирование возможных проникновении с нижних уровней на верхние.

Метка безопасности субъекта описывает его благонадежность, метка безопасности объекта — степень закрытости содержащейся в ней информации. Метки безопасности состоят из двух частей:

- уровень секретности;
- список категорий. Категории позволяют описать предметную область, к которой относятся данные. В военной сфере, категория может представлять собой виды вооружения (танки, самолеты), в коммерческой организации направления деятельности (финансы, кадры, производство).

Запросы на доступ отвергаются во всех случаях, когда уровень субъекта, запрашивающего разрешения на доступ, ниже уровня полномочий операции и (или) запрашиваемых данных. Типовые уровни секретности (безопасности) информации приведены в таблице 3.2.

Таблица 3. 2
Уровни секретности информации

Правительственные организации	Коммерческие организации
совершенно секретно	максимальная безопасность (финансы)
секретно	ограниченное распространение (кадры)
конфиденциально	конфиденциально (исключительно внутри фирмы)
неклассифицируемая	общедоступная

Можно комбинировать подходы по уровню полномочий и по категориям. Рассмотрим пример. Допустим, что существует 16 различных по категориям групп данных: C1, C2, ..., C6. Если правом пользователя (то, что позволено делать пользователю) является полномочие СОВЕРШЕННО СЕКРЕТНО и разрешен доступ к группам C2,C3,C4, а правом используемого терминала является полномочие СЕКРЕТНО и группы C1, C4, то пользователю с данного терминала будет разрешен доступ с грифом КОНФИДЕНЦИАЛЬНО и СЕКРЕТНО в группе C4. Если же терминалу разрешен доступ только к группам C1 и C5, то пользователь не будет иметь доступа ни к какой из 6 групп, так как пересечение прав пользователя и терминала будет пустым.

3.4. Протоколирование и аудит

Под протоколированием понимается сбор и накопление информации о событиях, происходящих в информационной системе предприятия. У каждого информационного сервиса свой набор возможных событий, которые можно разделить на:

- внешние, вызванные действиями других сервисов;
- внутренние, вызванные действиями самого сервиса;
- клиентские, вызванные действиями пользователей и администраторов.

Аудит — это анализ накопленной информации, проводимый оперативно или периодически.

Протоколирование и аудит являются одним из эффективных методов увеличения безопасности вычислительных систем. Протоколирование и аудит преследуют следующие цели:

- обеспечение подотчетности пользователей и администраторов;
- обеспечение возможности реконструкции последовательности событий;
- обнаружение попыток нарушений информационной безопасности;
- предоставление информации для выявления и анализа проблем.

Информация заносится в регистрационный журнал. Современные вычислительные системы способны записывать все осуществленные или неосуществленные попытки доступа к данным или программам.

Журнал должен содержать информацию, которая позволит сотруднику, ответственному за безопасность, определить для любого заданного периода времени следующие показатели:

- профили полномочий, связанные с любым защищаемым ресурсом;
- учет всех изменений в профилях полномочий;
- сами изменения, сделанные в профилях;
- все доступы, осуществляемые к любому защищаемому ресурсу, отсортированные по пользователю, программе, терминалу, заданию, элементу данных или любому другому требуемому ключу;
- все отказы в доступе;
- любые случаи, когда доступ к защищаемым данным был системой разрешен, но уполномоченный пользователь его не использовал.

Регистрационный журнал можно анализировать (проводить аудит) как периодически, так и непрерывно. Система безопасности может

периодически выводить на печать информацию, отсортированную по пользователям, терминалам, датам, программам и элементам данных. Копию журнала получает должностное лицо, ответственное за безопасность системы. Соответствующие выдержки из журнала предоставляются пользователям для проверки любых злоупотреблений их файлами.

Кроме функций обеспечения безопасности, журнал регистрации можно использовать для настройки вычислительного процесса. Например, можно выявить наиболее часто используемые программы или определить, что некоторые пользователи никогда не обращаются к определенным элементам данных и их профили безопасности можно скорректировать. Если в журнале отмечать дату изменения пароля, то можно заранее предупреждать пользователя об истечении времени безопасного использования пароля. Журналы часто используются для возврата системы в исходное состояние и восстановления массивов данных, которые были искажены из-за сбоев ВС.

4. Стандарты информационной безопасности

Главная задача стандартов информационной безопасности — создать основу для взаимодействия между производителями, потребителями и экспертами по сертификации продуктов информационных технологий. Знание критериев оценки информационной безопасности может помочь при выборе и комплектовании аппаратно-программной конфигурации.

Во всех существующих стандартах используют ряд одинаковых понятий информационной безопасности и исходят из того, что защитные мероприятия должны обеспечить:

- конфиденциальность, т.е. защиту от НСД;
- целостность, т.е. защиту от несанкционированного изменения;

- доступность, т.е. защиту от несанкционированного удержания информации и ресурсов.

4.1. Этапы развития систем защиты информации

Решение проблемы защиты информации в системах обработки данных в настоящее время осуществляется на промышленной основе с вложением значительных средств.

Подходы к организации системы защиты информации менялись по мере развития вычислительной техники и усложнения автоматизированных систем обработки данных (АСОД). Выделяют следующие этапы:

1. В 60-х годах XX века было принято считать, что основными средствами защиты являются программные, причем считалось, что программы защиты информации будут работать эффективнее, если они будут включены в состав общесистемных компонентов программного обеспечения. Поэтому первоначально программы защиты включались в состав операционных систем (IBM OS/360) или СУБД. Практика показала, что надежность подобной защиты недостаточна.

2. В 70-е годы основной идеей стала организация дифференцированного разграничения доступа пользователей к данным, находящимся в АСОД. Для этого идентифицировались все пользователи и все элементы защищаемых данных, определенным образом устанавливалось соответствие между идентификаторами, затем строилась алгоритмическая процедура проверки лояльности каждого запроса пользователя. В это же время интенсивно развивались технические и криптографические средства защиты.

3. 80-е годы характеризуются повышением системности подхода к проблеме защиты информации. При этом требуется обеспечить регулярность процесса, осуществляемого на всех этапах жизненного цикла

АСОД при комплексном использовании всех имеющихся средств защиты. Кроме того, требуется чтобы все средства, методы и мероприятия, используемые для защиты информации, рациональным образом были объединены в единый целостный механизм — систему защиты. В этой системе должно быть по крайней мере 4 защитных пояса:

- пояс, охватывающий всю территорию, на которой расположены сооружения АСОД;
- пояс сооружений, помещений или устройств АСОД;
- пояс компонентов системы (технические средства, программное обеспечение, элементы баз данных);
- пояс технологических процессов обработки данных (ввод/вывод, внутренняя обработка).

4. С 90-х годов и по настоящее время возросла актуальность распределенных систем, на первый план стали выходить обеспечение безопасности удаленного доступа и использование незащищенных каналов связи. Резко возросла актуальность криптографических средств защиты. Взрывной рост вычислительной мощности потребовал пересмотра ряда ранее принятых решений.

4.2. «Критерии оценки надежных компьютерных систем»

Информационной безопасностью первоначально занимались государственные организации, имеющие дело с секретной информацией.

В 1983 году министерство обороны США выпустило книгу "Критерии оценки надежных компьютерных систем". Предложенные в этом документе концепции защиты послужили основой для формирования других стандартов безопасности, появившихся впоследствии. Во второй половине 80-х годов аналогичные документы были изданы в ряде европейских стран, в Канаде. В 1992 году в России Гостехкомиссия при Президенте РФ издала

серию книг, посвященных проблеме защиты от несанкционированного доступа.

Критерии оценки были разработаны министерством обороны США в 1983 году с целью:

- определение требований безопасности, предъявляемых к аппаратному, программному и специальному обеспечению компьютерных систем;
- выработки соответствующей методологии анализа политики безопасности.

"Оранжевая книга" ориентирована на многопользовательские операционные системы, используемые в системах военного применения, поэтому главными в ней оказались требования, направленные на обеспечение секретности в физически защищенных военных вычислительных системах. Дальнейшее развитие тема безопасности получила в стандарте США "Федеральные критерии безопасности информационных технологий" (1992 год).

4.2.1. Группы требований безопасности

В "Оранжевой книге" предложены три группы требований безопасности: политика безопасности, подотчетность и гарантированность (корректность).

1. Основные элементы политики безопасности:

- добровольное управление доступом;
- метки безопасности;
- принудительное управление доступом;
- безопасность повторного использования объектов.

Безопасность повторного использования объектов — предохраняет от случайного или преднамеренного извлечения секретной информации из "мусора".

Безопасность повторного использования должна гарантироваться для областей оперативной памяти, магнитных носителей всех видов, для буферной памяти принтеров.

Если сотрудник (пользователь) уходит из организации, следует не только лишить его возможности входа в систему, но и запретить доступ от его имени ко всем объектам. В противном случае, новый сотрудник может получить ранее использованный идентификатор, а с ним и все права своего предшественника.

2. Подотчетность включает:

- идентификация и аутентификация объектов и субъектов;
- анализ регистрационной информации;
- предоставление надежного пути.

Анализ (аудит) имеет дело с действиями (событиями), связанными с безопасностью системы. К числу таких событий относятся:

- вход в систему (успешный или нет);
- выход из системы;
- обращение к удаленной системе;
- операции с файлами (открыть, закрыть, переименовать, удалить);
- изменение привилегий или иных атрибутов безопасности (режим доступа, уровня благонадежности пользователя).

Перечень событий, которые следует протоколировать, зависит от избранной политики безопасности и специфики системы, т.к. если фиксировать все события, объем регистрационной информации будет расти очень быстро и ее эффективный анализ будет невозможным.

"Оранжевая книга" предусматривает наличие средств выборочного протоколирования действий как отдельных пользователей, так и определенных событий.

Надежный путь связывает пользователя непосредственно с надежной вычислительной базой, минуя потенциально опасные компоненты. Цель предоставления надежного пути — дать пользователю уверенность в подлинности обслуживающей его системы.

3. Гарантированность (корректность) содержит:

- контроль корректности функционирования средств защиты;
- непрерывность защиты.

Средства защиты должны содержать независимые аппаратные и/или программные компоненты, обеспечивающие работоспособность функций защиты. Это означает, что все средства защиты должны находится под контролем средств, проверяющих правильность их работы. Кроме того, все средства защиты должны быть защищены от несанкционированного вмешательства и/или отключения. Эта защита должна быть постоянной и непрерывной в любом режиме функционирования как компьютера, так и самой системы защиты.

4.2.2. Классы безопасности информационных систем

"Оранжевая книга" определяет четыре уровня безопасности (надежности) систем — D,C,B,A, которые соответствуют различной степени защищенности информационных систем: от минимальной (группа D) до максимальной, формально доказанной (группа A). Каждая группа включает класс или несколько классов: A1, B1,B2,B3, C1,C2, D1.

Уровень безопасности возрастает от группы D к A, а внутри группы с возрастанием номера класса.

Группа D включает все системы, которые не удовлетворяют требованиям других классов.

Группа C характеризуется наличием произвольного управления доступом и регистрацией действий субъектов.

Группа B — основные требования этой группы нормативное (мандатное) управление доступом с использованием меток безопасности, поддержка модели и политики безопасности.

Группа A характеризуется применением формальных методов проверки корректности работы механизмов управления доступом (произвольного и нормативного).

4.3. Руководящие документы Гостехкомиссии России

В 1992 году Гостехкомиссия России при Президенте РФ опубликовала пять Руководящих документов, посвященных вопросам защиты от несанкционированного доступа (НСД) к информации. В руководящих документах различают понятия средств вычислительной техники (СВТ) и автоматизированной системы (АС) и вводят два направления в проблеме защиты информации от НСД. Формулируются следующие основные принципы защиты информации от НСД:

1. Защита СВТ обеспечивается комплексом программно-технических средств.

2. Защита АС обеспечивается комплексом программно-технических средств и поддерживающих их организационных мер. Защита АС должна обеспечиваться на всех технологических этапах обработки информации и во всех режимах функционирования, в том числе при проведении ремонтных и регламентных работ. Программно-технические средства защиты не должны существенно ухудшать основные функциональные

характеристики АС, такие как надежность, быстродействие, возможность изменения конфигурации.

Должен быть предусмотрен контроль эффективности средств защиты от НСД. Этот контроль может быть либо периодическим, либо проводиться по мере необходимости пользователями АС или контролирующими органами.

В руководящих документах все положения о защите информации ориентируются на физически защищенную среду, проникновение в которую посторонних лиц считается невозможным, поэтому нарушитель определяется как субъект, имеющий доступ к работе со штатными средствами АС и СВТ.

Нарушители классифицируются по уровню возможностей, которые предоставляют им штатные средства АС и СВТ. Классификация является иерархической, т.е. каждый следующий уровень включает в себя функциональные возможности предыдущего. Выделяют 4 уровня:

1-й уровень определяет самый низкий уровень возможностей — это запуск программ из фиксированного набора, реализующих заранее предусмотренные функции по обработке информации.

2-й уровень определяется возможностью создания и запуска собственных программ с новыми функциями по обработке информации.

3-й уровень определяется возможностью управления функционированием АС, т.е. воздействием на базовое программное обеспечение системы и на состав и конфигурацию ее оборудования.

4-й уровень определяется всем объемом возможностей лиц, осуществляющих проектирование, реализацию и ремонт технических средств АС, вплоть до включения в состав СВТ собственных технических средств с новыми функциями по обработке информации. На этом уровне

нарушитель является специалистом высшей категории, знает все об АС и, в частности, о системе и средствах защиты.

В качестве главного средства защиты от НСД рассматривается система разграничения доступа (СРД) субъектов к объектам доступа.

Руководящие документы предлагают две группы критериев безопасности:

- показатели защищенности средств вычислительной техники от НСД;
- критерии защищенности автоматизированных систем обработки данных.

4.3.1. Показатели защищенности СВТ от НСД

Данные показатели содержат требования защищенности и применяются к общесистемным программным средствам и операционным системам. Установлено семь классов защищенности СВТ от НСД: самые низкие требования предъявляются к системам, соответствующим седьмому классу, самые высокие — к первому. Классы делятся на 4 группы, отличающиеся качественным уровнем защиты.

1-я группа содержит только один седьмой класс. 2-я группа характеризуется произвольной защитой и содержит 6-й и 5-й классы. 3-я группа характеризуется мандатной защитой и содержит классы 4,3,2. 4-я группа характеризуется мандатной защитой и содержит только первый класс. Седьмой класс присваивают СВТ, к которым предъявлялись требования по защите от НСД, однако при оценке защищенность СВТ оказалась ниже уровня требований шестого класса.

Таблица 4. 1

Распределение показателей защищенности по классам СВТ

Наименование показателя	Класс защищенности					
	6	5	4	3	2	1
Дискреционный принцип контроля доступа	+	+	+	=	+	=

Мандатный принцип контроля доступа	-	-	+	=	=	=
Очистка памяти	-	+	+	+	=	=
Изоляция модулей	-	-	+	=	+	=
Маркировка документов	-	-	+	=	=	=
Защита ввода и вывода на отчужденный физический носитель	-	-	+	=	=	=
Сопоставление пользователя с устройством	-	-	+	=	=	=
Идентификация и аутентификация	+	=	+	=	=	=
Гарантии проектирования	-	+	+	+	+	+
Регистрация	-	+	+	+	=	=
Взаимодействие пользователя с комплексом средств защиты (КСЗ)	-	-	-	+	=	=
Надежное восстановление	-	-	-	+	=	=
Целостность КСЗ	-	+	+	+	=	=
Контроль модификации	-	-	-	-	+	=
Контроль дистрибуции	-	-	-	-	+	=
Гарантии архитектуры	-	-	-	-	-	+
Тестирование	+	+	+	+	+	=
Руководство пользователя	+	=	=	=	=	=
Руководство по КСЗ	+	+	=	+	+	=
Текстовая документация	+	+	+	+	+	=
Конструкторская(проектная)документация	+	+	+	+	+	+

Обозначения:
«-» — нет требований к данному классу
«+» — новые или дополнительные требования
«=» — требования совпадают с требованиями к СВТ предыдущего класса

4.3.2. Классификация АС по уровню защищенности от НСД

Установлено девять классов защищенности АС, каждый класс характеризуется определенной минимальной совокупностью требований по защите. Классы подразделяются на три группы, отличающиеся особенностями обработки информации в АС.

Третья группа классифицирует АС, в которых работает один пользователь, допущенный ко всей информации АС, размещенной на носителях одного уровня конфиденциальности. Группа содержит 2 класса — 3Б и 3А.

Вторая группа классифицирует АС, в которых пользователи имеют одинаковые полномочия ко всей информации, обрабатываемой и (или)

хранимой на носителях различного уровня конфиденциальности. Группа содержит 2 класса — 2Б и 2А.

Таблица 4. 2
Требования к средствам защиты АС от НСД

Подсистема Управления доступом	Подсистема Регистрации и учета	Криптографическая система	Подсистема Обеспечения Целостности
Идентификация, проверка подлинности, контроль доступа	Регистрация и учет	Шифрование конфиденциальной информации	Обеспечение целостности программных средств и обрабатываемой информации
Управление потоками информации	Учет носителей информации	Шифрование информации, принадлежащей различным субъектам доступа на разных ключах	Физическая охрана СВТ и носителей информации
	Очистка освобождаемых областей памяти	Использование сертифицированных криптографических средств	Наличие администратора защиты информации
	Сигнализация попыток нарушения защиты		Периодическое тестирование СЗИ НСД
			Наличие средств восстановления СЗИ НСД
			Использование сертифицированных средств защиты

Таблица 4. 3
Требования к классам защищенности АС

Подсистемы и требования	Классы								
	3Б	3А	2Б	2А	1Д	1Г	1В	1Б	1А
1. Подсистема управления доступом 1.1. Идентификация. Проверка подлинности и контроль доступа субъектов в систему,	+	+	+	+	+	+	+	+	+
к терминалам, ЭВМ, узлам сети ЭВМ, каналом связи, внешним устройством ЭВМ,				+		+	+	+	+
к программам,				+		+	+	+	+
к томам, каталогам, файлам, записям, полям записей.				+		+	+	+	+
1.2. Управление потоками информации.				+			+	+	+
2. Подсистема регистрации и учета 2.1.Регистрация и учет: входа/выхода субъектов доступа в/из системы (узла сети,	+	+						+	+
выдачи печатных (графических) выходных документов,		+		+		+	+	+	+
запуска/завершения программ и процессов заданий, задач,				+		+	+	+	+
доступа программ субъектов к защищаемым файлом, включая их создание и удаление, передачу по линиям и каналам связи,				+		+	+	+	+

доступа программ субъектов, доступа к терминалам, ЭВМ. узлам сети ЭВМ, каналам связи, внешним устройствам ЭВМ, программам, каталогам, файлом, записям, полям записей,				+		+	+	+	+
изменения полномочий субъектов доступа,							+	+	+
создаваемых защищаемых объектов доступа.				+			+	+	+
2.2. Учет носителей информации	+	+	+	+	+	+	+	+	+
2.3. Очистка (обнуление, обезличивание) освобождаемых областей оперативной памяти ЭВМ и внешних накопителей		+		+		+	+	+	+
2.4. Сигнализация попыток нарушения защиты							+	+	+
3. Криптографическая подсистема 3.1. Шифрование конфиденциальной информации				+				+	+
3.2. Шифрование информации, принадлежащей различным субъектам доступа (группам субъектов) на разных ключах									+
3.3. Использование аттестованных сертифицированных) криптографических средств				+				+	+
4. Подсистема обеспечения целостности 4.1. Обеспечение целостности программных средств и обрабатываемой информации	+	+	+	+	+	+	+	+	+
4.2. Физическая охрана средств вычислительной техники и носителей информации	+	+	+	+	+	+	+	+-	+
4.3. Наличие администратора (службы) защиты информации в АС				+			+	+	+
4-4. Периодическое тестирование СЗИ НСД	+	+	+	+	+	+	+	+	+
4.5. Наличие средств восстановления СЗИ НСД	+	+	+	+	+	+	+	+	+
4.6. Использование сертифицированных средств защиты		+		+			+	+	+

Обозначения:
«+» — требование к данному классу присутствует
СЗИ НСД — система защиты информации от несанкционированного доступа

Первая группа классифицирует многопользовательские АС, в которых одновременно обрабатывается и (или) хранится информация разных уровней конфиденциальности и не все пользователи имеют право доступа ко всей информации АС. Группа содержит 5 классов — 1Д, 1Г, 1В, 1Б И 1А.

4.4. Другие стандарты безопасности

В 1991 году Франция, Германия, Нидерланды и Англия выпустили "Согласованные Европейские Критерии оценки безопасности информационных технологий". Особенностью Европейских Критериев является отсутствие заранее определенных условий работы информационной системы (В "Оранжевой книге" это были закрытые военные системы).

Организация, запрашивающая сертификационные услуги, формулирует цель оценки, т.е. описывает условия в которых должна функционировать система, возможные угрозы ее безопасности и предоставляемые ею защитные функции. Задача организации, выполняющей сертификацию, оценить насколько полно достигаются поставленные цели в соответствии со стандартом.

"Канадские критерии безопасности компьютерных систем" были выпущены в 1993. Критерии разрабатывались как основа для оценки эффективности средств обеспечения безопасности компьютерных систем, при этом преследовались следующие цели:

- предложить единую шкалу критериев оценки безопасности, позволяющую сравнивать системы обработки конфиденциальной информации;
- разработать спецификации безопасных систем, которые могли бы использоваться разработчиками при проектировании подобных

систем в качестве руководства для определения состава функций и средств защиты;

- предложить унифицированный подход и стандартные средства для описания характеристик безопасности компьютерных систем.

Предыдущие стандарты создавались в расчете на централизованные конфигурации, основу которых составляют большие машины. Распределенная организация современных информационных систем требует внесения существенных изменений и дополнений как в политику безопасности, так и в способы проведения ее в жизнь. Основополагающим документом в области защиты распределенных систем стали рекомендации X.800 (США, 1991 г).

В 1996 появились «Единые критерии безопасности информационных технологий». При создании единых критериев учитывали ранее созданные критерии безопасности ("Федеральные", "Европейские" и "Канадские"). "Единые критерии" согласованы с существующими стандартами и развивают их путем введения новых концепций, соответствующих современному уровню развития информационных технологий. Это первая работа, которая содержит разделы, адресованные всем участникам: потребителям, производителям и экспертам по сертификации продуктов информационных технологий.

В середине 90-х годов Британский институт стандартов (BSI) при участии коммерческих организаций, таких как Shell, National Westminster Bank, Midland Bank, Unilever, British Telecommunications, Marks & Spencer, Logica и др., занялся разработкой стандарта управления информационной безопасностью. И в 1995 г. был принят национальный британский стандарт BS 7799 управления информационной безопасностью организации вне зависимости от сферы ее деятельности. Первая часть стандарта носила

рекомендательный характер. Вторая часть была предназначена для сертификации и содержала часть обязательных требований, не входивших в первую часть, основные требования первой части стандарта приведены в приложении.

В конце 1999 эксперты международной организации по стандартизации ISO пришли к выводу, что в рамках существующих стандартов ISO отсутствует специализированный стандарт управления информационной безопасностью. Соответственно, ISO было принято решение не заниматься разработкой нового стандарта, а по согласованию с британским институтом стандартов, взяв за базу BS 7799:1, принять стандарт ISO 17799.

ISO 17799 охватывает следующие вопросы:

- политика безопасности;
- организационные методы обеспечения информационной безопасности;
- управление ресурсами;
- пользователи информационной системы;
- физическая безопасность;
- управление коммуникациями и процессами;
- контроль доступа;
- приобретение, разработка и сопровождение информационных систем;
- управление инцидентами информационной безопасности;
- управление непрерывностью ведения бизнеса;
- соответствие требованиям.

В 2005 году был принят стандарт ISO 27001, определяющий основные требования к разработке и функционированию системы управления

информационной безопасностью. В 2006 и 2008 году были приняты соответствующие российские стандарты: ГОСТ 17799 и ГОСТ 27001.

5. Практические подходы к обеспечению информационной безопасности

Информационные стандарты и рекомендации, рассмотренные ранее, образуют базис, на котором строятся все работы по обеспечению информационной безопасности. Однако, стандарты статичны, они не учитывают постоянной перестройки защищаемых систем и их окружения. Во-вторых, стандарты не содержат практических рекомендаций по формированию режима безопасности. В-третьих, стандарты ориентированы на производителей и организации, занимающиеся сертификацией систем.

Проблема обеспечения безопасности носит комплексный характер, для ее решения необходимо сочетание законодательных, организационных и программно-технических мер.

Программно-технические меры включают следующие механизмы безопасности: идентификация и аутентификация; управление доступом; протоколирование и аудит; криптография; экранирование.

Создание системы безопасности организация должна начать с выработки политики безопасности. Под политикой безопасности понимается совокупность документированных управленческих решений, направленных на защиту информации и связанных с ней ресурсов.

С практической точки зрения политику безопасности делят на три уровня:

1. К верхнему уровню относят решения, затрагивающие организацию в целом. Они носят общий характер и исходят от руководства организации.

2. К среднему уровню относят вопросы, касающиеся отдельных аспектов информационной безопасности, но важные для различных систем, эксплуатируемых организацией.
3. Политика безопасности нижнего уровня относится к конкретным информационным сервисам.

После завершения формирования политики безопасности, переходят к составлению программы ее реализации. Проведение политики безопасности в жизнь требует использования трех видов регуляторов — управленческих, операционных (организационных) и программно-технических.

5.1. Управленческие мероприятия

Для реализации программы безопасности, ее целесообразно разделить на уровни, обычно в соответствии со структурой организации. В простом и самом распространенном случае достаточно двух уровней: верхнего и нижнего (сервисного), который относится к отдельным сервисам или группе однородных сервисов.

Программу **верхнего уровня** возглавляет лицо, отвечающее за информационную безопасность организации. Программа должна занимать четко определенное место в деятельности организации, она должны официально приниматься и поддерживаться руководством, у нее должны быть определенные штаты и бюджет. У программы следующие главные цели:

- управление рисками. Деятельность любой организации подвержена множеству рисков. Суть работы по управлению рисками состоит в том, чтобы оценить их размер, выработать эффективные меры по их уменьшению и убедиться, что риски заключены в приемлемые рамки;

- координация деятельности. Управление должно быть организовано так, чтобы исключить дублирование в деятельности сотрудников и в максимальной степени использовать их знания. Однако, отсутствие дублирования противоречит надежности. Если в организации есть специалист, знания которого уникальны, его временное отсутствие или увольнение могут привести к затруднительной ситуации. Поэтому целесообразно документировать накопленные знания и освоенные процедуры.
- стратегическое планирование. На верхнем уровне принимаются стратегические решения по безопасности, оцениваются технологические новинки. Информационные технологии развиваются очень быстро, и необходимо иметь четкую политику отслеживания и внедрения новых средств.
- контроль деятельности.

  Контроль деятельности имеет два направления:

1. Надо гарантировать, что действия организации не противоречат законам. Обязательны при этом контакты с внешними контролирующими организациями;
2. Нужно постоянно отслеживать состояние безопасности внутри организации, реагировать на случаи нарушений, дорабатывать защитные меры с учетом изменения обстановки.

Цель программы **нижнего уровня** — обеспечить надежную и экономичную защиту конкретного сервиса или группы однородных сервисов. На этом уровне решается, какие механизмы защиты использовать, закупаются и устанавливаются технические средства, выполняется повседневное администрирование, отслеживается состояние слабых мест. Обычно за программу нижнего уровня отвечают администраторы сервисов.

5.2. Организационные мероприятия

При выработке политики безопасности большое значение имеют меры безопасности, ориентированные на людей. Именно люди формируют режим информационной безопасности и они же оказываются главной угрозой для нее. К вопросам, связанными с людьми, относится:

- управление персоналом;
- организация физической защиты;
- поддержание работоспособности;
- планирование восстановительных работ.
- реагирование на нарушение режима безопасности;

5.2.1. Управление персоналом

Управление персоналом начинается с составления должностной инструкции. На этом этапе желательно привлечение специалиста по информационной безопасности для определения компьютерных полномочий, связанных с должностью. Существует два общих принципа, которые следует иметь в виду:

- разделение обязанностей;
- минимизация привилегий.

Принцип распределения обязанностей предписывает так распределять роли и ответственность, чтобы один человек не смог нарушить критически важный для организации процесс.

Принцип минимизации привилегий предписывает выделять пользователям только те права доступа, которые необходимы им для выполнения служебных обязанностей. Это позволит уменьшить ущерб от случайных или умышленных некорректных действий пользователя.

С момента заведения системного счета нового сотрудника начинается его администрирование, а также протоколирование и анализ действий.

Ликвидация системного счета пользователя, особенно в случае конфликта между сотрудником и организацией, должна проводится максимально оперативно. Возможно и физическое ограничение доступа к рабочему месту.

Важную роль в обеспечении безопасности имеет обучение. Если сотрудник не знаком с политикой безопасности своей организации, он не сможет стремиться к достижению сформулированных в ней целей, если он не знает мер безопасности, он не будет их соблюдать. С другой стороны, если сотрудник знает, что его действия протоколируются, он, вероятно, воздержится от нарушений. Обучение должно проводиться регулярно и в то же время каждый раз по-иному, иначе оно превратиться в формальность и потеряет эффективность.

5.2.2. Физическая защита

Безопасность компьютерной системы зависит от окружения, в котором она работает, следовательно, необходимо принять меры для защиты зданий и прилегающей территории, поддерживающей инфраструктуры и самих компьютеров. К основным направлениям физической защиты относятся следующие направления:

- физическое управление доступом;
- противопожарные меры;
- защита поддерживающей инфраструктуры;
- защита от перехвата данных;
- защита мобильных систем.

Меры физического управления доступом позволяют контролировать и при необходимости ограничивать вход и выход сотрудников и посетителей. Контролировать можно все здание организации и, кроме того, отдельные помещения, например те, где расположены серверы,

коммуникационная аппаратура. Необходимо, чтобы посетители отличались от штатных сотрудников, например у сотрудников могут быть нагрудные идентификационные карточки.

В организации желательно устанавливать противопожарную сигнализацию и автоматические средства пожаротушения.

К поддерживающей инфраструктуре можно отнести системы электро-, водо- и теплоснабжения, кондиционеры, средства коммуникаций. В принципе к ним применимы те же требования целостности и доступности, что и к информационным системам. Для обеспечения целостности нужно защищать оборудование от краж и повреждений.

Для поддержания доступности целесообразно выбирать оборудование с максимальным временем наработки на отказ, дублировать ответственные узлы, иметь запчасти. Отдельную проблему составляют аварии водопровода: при размещении компьютеров следует учитывать расположение водопроводных и канализационных труб.

Мобильные и портативные компьютеры можно легко украсть, т.к. их часто оставляют без присмотра в автомобиле, на работе. Для снижения ущерба от несанкционированного доступа к информации сотрудникам рекомендуется шифрование данных на жестких дисках ПК подобного типа.

5.2.3. Поддержание работоспособности.

Рассмотрим повседневные действия, направленные на поддержание работоспособности компьютерных систем и имеющие отношение к информационной безопасности.

Можно выделить следующие направления:

- поддержка пользователей;
- поддержка программного обеспечения;
- конфигурационное управление;

- резервное копирование;
- управление носителями;
- документирование;
- регламентные работы.

Поддержка пользователей состоит в консультировании и оказании помощи, т.к. нечаянные ошибки пользователей грозят повреждением аппаратуры, разрушением программ и данных. Поддержка программного обеспечения — одно из важнейших средств обеспечения целостности информации. Необходимо контролировать, какое ПО выполняется на компьютерах. Если пользователи могут устанавливать программы самостоятельно, это может привести к заражению вирусами, а также появлению утилит, действующих в обход защитных средств.

Конфигурационное управление позволяет контролировать и фиксировать изменения, вносимые в программы. Необходимо избегать случайных или непродуманных модификаций, иметь возможность возвращаться к предыдущей, работающей версии. Технологию конфигурационного управления необходимо применять и к изменениям в аппаратуре, т.е. фиксировать появление новых компьютеров, их перемещения и пр.

Резервное копирование необходимо для восстановления программ и данных после аварии. Целесообразно хранить копии в безопасном месте, защищенном от пожаров и других угроз. Иногда в тестовых целях следует проверять возможность восстановления информации с копий.

Управление носителями служит для обеспечения физической защиты и учета дискет, лент, распечаток и т.п. Под физической защитой понимается не только предотвращение НСД, но и предохранение от вредных влияний окружающей среды (жара, холод, магнетизм).

Документирование — важная часть информационной безопасности. В виде документов оформляется почти все — от политики безопасности до журнала учета дискет. Важно, чтобы документация была актуальной, отражала текущее состояние дел. К хранению некоторых документов, например содержащих анализ слабых системных мест и угроз, применимы требования обеспечения конфиденциальности, к другим, например к плану восстановления после аварии — требования целостности и доступности, т.е. план необходимо взять и прочитать.

Регламентные работы — серьезная угроза безопасности. Лицо, осуществляющее регламентные работы, получает исключительный доступ к системе, и на практике трудно проконтролировать, какие именно действия совершаются. Здесь на первый план выходит степень доверия к тем, кто выполняет работы.

5.2.4. Планирование восстановительных работ

Ни одна организация не застрахована от серьезных аварий, вызванных естественными причинами, чьим-то злым умыслом, халатностью или некомпетентностью. В то же время у каждой организации есть функции, которые она считает критически важными, выполнение которых необходимо в любом случае. Планирование восстановительных работ позволяет подготовиться к авариям, уменьшить ущерб от них и сохранить способность к функционированию хотя бы в минимальном объеме.

Лучший способ минимизировать ущерб от инцидентов безопасности — готовиться к ним. Следующие типовые шаги позволят значительно облегчить работу по восстановлению:

- Протоколирование и аудит.
- Регулярное резервное копирование. Использование систем контроля целостности информации. Это позволяет эффективно обнаружить

факт атаки, а также определить, какие данные необходимо восстановить.

- Подготовка документов, регламентирующих действия сотрудников в ситуациях нарушения режима безопасности и спланировать восстановительные работы.
- Подготовка инструментария, необходимого для восстановительных работ, и хранение его в доступном месте.

Процесс планирования восстановительных работ можно разделить на следующие этапы:

- выявление критически важных функций, установление приоритетов:
- идентификация ресурсов, необходимых для выполнения критически важных функций;
- определение перечня возможных аварий;
- разработка стратегии восстановительных работ;
- подготовка к реализации выбранной стратегии;
- проверка стратегии.

Планируя восстановительные работы, следует учитывать, что полностью сохранить функционирование организации не всегда возможно. Значит, необходимо выявить критически важные функции и среди них также расставить приоритеты.

Идентифицируя ресурсы, необходимые для выполнения критически важных функций, следует помнить, что многие из них имеют некомпьютерный характер. Кроме того, целесообразно учесть временной фактор их использования: большинство ресурсов нужны постоянно, но некоторые из них работают только в определенные периоды, например в конце месяца или квартала при составлении отчетов.

Критичные ресурсы обычно относятся к одной из следующих групп:

- персонал;
- информационная инфраструктура;
- физическая инфраструктура.

Составляя списки критически важных специалистов, следует учитывать, что некоторые из них могут непосредственно пострадать от аварии, или часть сотрудников не сможет добраться до работы и пр. В связи с этим, желательно иметь некоторый резерв специалистов или заранее определить каналы, по которым можно будет на время привлечь дополнительный персонал (филиалы организации).

Информационная инфраструктура включает в себя следующие элементы:

- компьютеры;
- программы и данные;
- информационные сервисы внешних организаций;
- документация.

В случае серьезной аварии, организация может быть эвакуирована в другое помещение. При этом аппаратная платформа может отличаться от исходной. Следовательно, надо продумать меры поддержания совместимости по программам и данным.

Среди внешних информационных сервисов для коммерческих организаций, вероятно, будет получение оперативной рыночной информации и связь с государственными службами, курирующими данный сектор экономики.

Документация важна потому, что не вся информация, с которой работает организация, представлена в электронной форме.

К физической инфраструктуре относятся здания, инженерные коммуникации, средства связи, оргтехника и пр. Компьютерная техника не

может работать в плохих условиях, без нормального электропитания, охлаждения и пр.

При составлении перечня возможных аварий нужно попытаться разработать их возможные сценарии и ответить на вопросы типа: как будут развиваться события, каковы могут оказаться масштабы бедствия и т.д. Стратегия восстановительных работ должна базироваться на наличных ресурсах и быть не слишком дорогостоящей для организации. При разработке стратегии целесообразно провести анализ рисков, которым подвержены критичные функции, и попытаться выбрать наиболее экономичное решение. Стратегия должна предусматривать не только работу по временной схеме, но и возвращение к нормальному функционированию.

Подготовка к реализации выбранной стратегии состоит в проработке детального плана действий в экстренных ситуациях и по их окончании, а также в обеспечении некоторой избыточности критичных ресурсов.

Проверка стратегии производится путем анализа подготовленного плана, принятых и намеченных мер с привлечением специалистов разного профиля.

5.2.5. Реакция на нарушение режима безопасности

Программа безопасности, принятая организацией, должна предусматривать набор оперативных мероприятий, направленных на обнаружение и нейтрализацию вторжений хакеров и программных вирусов. Подобные действия целесообразно проводить по подготовленному плану. Идеальным вариантом было бы проводить регулярные проверки готовности персонала к работе в критических ситуациях. На практике это, конечно, далеко не всегда осуществимо, но по крайней мере план действий в подобных ситуациях должен быть разработан и находиться в доступном месте.

Реакция на нарушение режима безопасности преследует три цели:

- блокирование нарушителя
- уменьшение наносимого ущерба;
- недопущение подобных нарушений в дальнейшем.

В организации должен быть человек, отвечающий за реакцию на нарушения и доступный в любое время. Для недопущения повторных нарушений необходимо анализировать каждый инцидент, выявлять причины, накапливать статистику.

Реакция на инцидент включает:

- **Оценку ситуации.** Удалось ли атакующему проникнуть в систему, находится ли атака в процессе развития или завершена, какие узлы ей затронуты и т.п.
- **Изоляцию.** Атакованные узлы должны быть изолированы от остальной системы, а, возможно, и вся система изолирована от внешнего мира. Изоляция подразумевает, по крайней мере, отключение/блокирование сетевых соединений, или, возможно, выключение части атакованных машин, хотя обычно последнее является не самым лучшим решением, поскольку может затруднить последующий анализ.
- **Извещение лиц, которым необходимо знать о случившемся инциденте** — как в своей организации, так и в смежных, если их работа зависит от вашей, или, наоборот, есть подозрение в проникновении с их стороны.
- **Фиксацию состояния системы** — например, с помощью резервного копирования.
- **Переход к восстановительным работам.** Это наименее детерминированный этап. Незаменимую помощь здесь могут оказать

те действия по минимизации ущерба, которые предпринимались на подготовительном этапе.

В зависимости от серьезности атаки, восстановительные работы могут включать те или иные шаги из приведенного ниже списка:

- Выявление способа проникновения — утечка пароля, уязвимость приложений, системных служб, программная «закладка» и т.п.
- Выявление нарушителя и анализ его действий.
- Выявление возможных «закладок», оставленных нарушителем.
- Документирование инцидента.
- Восстановление системы с резервной копии.
- Обновление/переписывание уязвимых приложений/служб.
- Смену паролей пользователей.
- Усиление режима протоколирования и аудита.

II. Криптографическое преобразование информации

6. Основные понятия криптографии

6.1. Алгоритмы шифрования

Криптографическое преобразование информации используется человечеством в течение нескольких тысячелетий. Криптография всегда активно использовалась для защиты государственных и военных секретов, а в последние десятилетия интерес к ней значительно возрос, и не в последнюю очередь благодаря развитию телекоммуникационных технологий. Электронный банкинг, передача критичной информации по открытым сетям, цифровая подпись — одни из многих современных применений криптографии. В общем случае, криптография призвана решить две задачи — обеспечить возможно более секретную передачу информации, и гарантировать ее аутентичность. Между кодированием и шифрованием не существует отчетливой разницы. В настоящее время слово "кодирование" применяют в целях цифрового представления информации при ее обработке на технических средствах, а "шифрование" — для преобразования информации в целях защиты от НСД.

При шифровании обычно используется некоторый алгоритм, который может быть известен широкому кругу лиц, и некоторый сменный элемент шифрования, обычно называемый ключом. Ключ управляет процессом шифрования таким образом, что из одних и тех же входных данных при использовании одного и того же алгоритма и разных ключей получаются различные зашифрованные данные. При использовании криптографических решений обычно придерживаются принципа Керкгоффса, согласно которому стойкость шифра определяется только секретностью ключа, т.е. предполагается, что потенциальному криптоаналитику будет известны все

детали работы алгоритма и, соответственно, одной из основных проблем становится обеспечение безопасного канала передачи ключей между абонентами. Среди других требования к алгоритмам шифрования можно выделить следующие:

- Число операций, необходимых для атаки шифра по известному тексту (когда известен фрагмент шифрованного и соответствующего ему открытого текста), должно быть не меньше общего числа возможных ключей.
- Незначительные изменения ключа шифрования или шифруемого текста должны приводить к существенному изменению вида зашифрованного текста.
- Любой ключ из множества возможных должен обеспечивать надежную защиту информации.

За время своего существования криптография накопила массу методов шифрования, и современные шифры, как правило, используют комбинации нескольких методов.

- замена (подстановка);
  - простая (одноалфавитная);
  - многоалфавитная простая;
  - многоалфавитная одноконтурная;
  - многоалфавитная многоконтурная.
- перестановка;
  - простая;
  - усложненная по таблицам;
  - усложненная по маршрутам.
- аналитическое преобразование;
- гаммирование;

- с конечной гаммой;
- с бесконечной гаммой.

- смысловое кодирование.

6.2. Шифрование подстановкой (заменой)

6.2.1. Моноалфавитная подстановка

В прямых подстановках каждый знак исходного текста заменяется одним или несколькими знаками, но порядок следования символов в сообщении не изменяется.

Один из простейших шифров замены представляет собой исходный алфавит, смещенный на определенное число позиций. Например, если каждая буква была смещена на три позиции, то алфавит

АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ

превращается в

ГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯАБВ

где буквы АБВ выдвинулись и добавились в конец. Для того, чтобы закодировать сообщение, используя данный метод, вы просто подставляете символы сдвинутого алфавита вместо реальных символов.

Например, сообщение

СЕГОДНЯ ПРЕКРАСНАЯ ПОГОДА

превращается в

фзесжрв тузнугфргв тсесжг

Все методы моноалфавитной подстановки можно представить как числовые преобразования букв исходного текста, рассматриваемых как числа. Каждая буква в тексте умножается на некоторое число (десятичный коэффициент) и прибавляется к некоторому другому числу (коэффициент сдвига).

C = (a*P + s) mod K,

где a — десятичный коэффициент;

P — позиция символа в алфавите;

s — коэффициент сдвига.

Полученное число уменьшается по правилу вычитания модуля К, где К — размер алфавита, зашифрованный текст формируется из соответствующих ему алфавитных эквивалентов. А и К должны быть взаимно простыми числами.

Для нашего первого примера a=1, s=3, K=33.

В моноалфавитных подстановках устанавливается взаимооднозначное соответствие между каждым знаком a(i) алфавита сообщений А и соответствующим знаком h(i) зашифрованного текста.

Однако шифр замены, основанный на постоянном смещении, можно легко раскрыть. Так для русского языка существует только 33 возможных смещений и легко, за довольно короткое время, перебрать их все. Вторая слабость шифра простой замены состоит в том, что он сохраняет пробелы между словами, что еще упрощает его дешифрацию.

Можно улучшить шифр замены, используя перемешанный алфавит вместо просто смещенного. Кроме того, следует кодировать пробел и другие знаки препинания.

АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯпробел

ЙФЯЦЫЧУВСК#АМЕПИНРТГОЬШЛБЩДЮЗЖЭХЪЁ

СЕГОДНЯ ПРЕКРАСНАЯ ПОГОДА

тчциыпъёнрчарйтпйъёниципый

Стойкость шифрования при использовании перемешанного варианта алфавита по сравнению с вариантом простого смещения значительно

улучшается. Это объясняется тем, что существует 33! вариантов перестановок алфавита, а с пробелом число вариантов достигает 34!.

Но даже такой улучшенный код замены может быть сравнительно легко дешифрован если использовать частотные таблиц русского или любого другого языка, в которых дана статистическая информация по использованию каждой буквы алфавита. Чем больше шифрованное сообщение, тем легче расшифровать его с помощью частотных таблиц.

6.2.2. Многоалфавитные подстановки

В подобных подстановках одна и та же буква в исходном сообщении преобразуется в разные буквы в шифрованном тексте. Простейший метод множественной замены состоит в добавлении второго перемешанного алфавита и переключении с одного алфавита на другой при встрече в тексте пробела. Пусть вторым перемешанным алфавитом будет следующий:

АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯпробел
ЙФЯЦЫЧУВСК#АМЕПИНРТГОЪШЛБЩДЮЗЖЭХЬЁ
УВСКАМЕПИНРТГОЬШЛБЩДЮЗЖЁХЭЪ#ЙФЯЦЫЧ

В начале работы программы используется первый алфавит. Когда встречается первый пробел, происходит переключение на второй алфавит. Следующий пробел заставляет использовать снова первый алфавит. Этот процесс продолжается до тех пор, пока не будет закодировано все сообщение. Тогда, используя данный подход, сообщение будет зашифровано в виде:

СЕГОДНЯ ПРЕКРАСНАЯ ПОГОДА
тчциыпъёлбмтбущьуыёницийый

Более стойкой к раскрытию является схема шифрования, основанная на использовании таблицы Вижинера. Таблица представляет собой квадратную матрицу с числом элементов в строке К, где К — количество

символов в алфавите. Первая строка содержит алфавит в обычном порядке, вторая — алфавит, смещенный на одну позицию, третья строка — алфавит, смещенный на две позиции и так далее.

```
АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ
БВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯА
ВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯАБ
. . . . . .
ЯАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮ
```

Для шифрования текста устанавливается ключ — некоторое слово или набор букв. Далее из полной матрицы выбирается подматрица шифрования, включающая первую строку и строки, которые начинаются на очередную букву из ключа.

Пример: ключ ***МОРЕ***, подматрица шифрования содержит строки

```
АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ
МНОПРСТУФХЦЧШЩЪЫЬЭЮЯАБВГДЕЁЖЗИЙКЛ
ОПРСТУФХЦЧШЩЪЫЬЭЮЯАБВГДЕЁЖЗИЙКЛМН
РСТУФХЦЧШЩЪЫЬЭЮЯАБВГДЕЁЖЗИЙКЛМНОП
ЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯАБВГД
```

Под словами исходного текста много раз записывается ключ. Шифруемый текст по подматрице заменяется буквами, расположенными на пересечении линий, соединяющих буквы текста из первой строки и строки подматрицы, начинающейся с буквы ключа, которая располагается под исходным текстом.

СЕГОДНЯ ПРЕКРАСНАЯ ПОГОДА

МОРЕМОР ЕМОРЕМОРЕМ ОРЕМОРЕ

текст после замены:

ЮУУУРЬП ФЭУЫХМАЮЕЛ ЮЯЗЫТР

Затем выходной текст делится на группы, например по 4 знака.

ЮУУУ РЪПФ ЭУЫХ МАЪЕЛ ЮЯЗЫ ТР

Использование шифров множественной замены существенно затрудняет дешифрацию кода с применением частотных таблиц, из-за того, что одна и та же буква преобразуется в разные символы. К недостаткам относят ненадежность шифрования при небольшой длине ключа и сложность формирования длинных ключей, т.к. ключ не должен содержать повторяющихся символов.

Развитие этого подхода — многоалфавитная одноконтурная монофоническая подстановка и многоалфавитная многоконтурная подстановка.

В **монофонической подстановке** количество и состав алфавитов выбирается таким образом, чтобы частоты появления всех символов в зашифрованном тексте были одинаковыми. При таком положении затрудняется криптоанализ зашифрованного текста с помощью его статистической обработки. Выравнивание частот появления символов достигается за счет того, что для часто встречающихся символов исходного текста предусматривается большее число заменяющих символов, чем для редко встречающихся.

Шифрование проводится так же, как и при простой подстановке, с той лишь разницей, что после шифрования каждого символа соответствующий ему столбец алфавитов циклически сдвигается вверх на одну позицию. Таким образом, столбцы алфавитов как бы образуют независимые друг от друга кольца, поворачиваемые вверх на один знак каждый раз после шифрования соответствующего знака исходного текста.

Многоконтурная подстановка заключается в том, что для шифрования используются несколько наборов (контуров) алфавитов, используемых циклически, причем каждый контур в общем случае имеет свой индивидуальный период применения. Частным случаем многоконтурной полиалфавитной подстановки является замена по таблице Вижинера, если для шифрования используется несколько ключей, каждый из которых имеет свой период применения.

Усложнение многоалфавитной подстановки существенно повышает ее стойкость. Монофоническая подстановка может быть весьма стойкой (и даже теоретически нераскрываемой), однако строго монофоническую подстановку реализовать на практике трудно, а любые отклонения от монофоничности снижают реальную стойкость шифра.

6.2.3. Частотный анализ

Большинство естественных языков имеют характерное частотное распределение букв и других знаков. Распределение вероятностей букв:

Английский алфавит	Русский алфавит
E — 0,12 N — 0,072	O — 0,09 Н,Т — 0,053
T — 0,096 W- 0.02	Е — 0,072 Щ,Э — 0,03
A — 0.081 Z — 0.001	А,И — 0,062 Ф — 0,02
O — 0,079	

Многие сообщения, зашифрованные методом одноалфавитной подстановки, сохраняют характерное частотное распределение и, таким образом, дают криптоаналитику путь к раскрытию шифра. Криптоаналитики часто используют индекс соответствия (ИС) для определения того, каким методом пользовались при шифровании исходного текста. ИС представляет собой оценку суммы квадратов вероятностей

каждого символа. Теоретически ожидаемые значения ИС вычисляются по формуле:

$$ИС_{теор} = A\frac{N-m}{m(n-1)} + \frac{N(m-1)}{Lm(N-1)},$$

$$A = \sum_{i=1}^{L} p_i^2$$

где N — число символов в криптограмме, m — длина ключа, p_i — вероятность встречаемости i-ой буквы алфавита, L — мощность алфавита.

$$ИС_{прак} = \frac{\sum_{i=1}^{L} f_i(f_i-1)}{N(N-1)},$$

где N — число символов в криптограмме, f_i — сколько раз i-ая буква встретилась в криптограмме.

Существуют таблицы, содержащие теоретически ожидаемые значения ИС для разных длин ключа. Для английского языка в шифрованных сообщениях, у которых ИС большие чем 0.066, вероятно, применялась одноалфавитная подстановка; если 0.052< ИС <0.066, то, вероятно, был использован двухалфавитный шифр подстановки, 0,047 < ИС < 0,052 был использован 3-х алфавитный шифр. Для многоалфавитной подстановки ИС=0.038, что говорит о равномерном распределении символов и существенно затрудняет раскрытие шифра. Для русского языка соответствующие пороговые значения: 0.055, 0.0395, 0.0355.

6.3. Шифрование перестановкой

Суть методов перестановки состоит в том, исходный текст делится на блоки, в каждом из которых выполняется перестановка в соответствии с заданным правилом.

Например, переставить символы в группе из четырех букв, находящихся в порядке 1-2-3-4 в порядок 3-1-4-2. Первоначальный текст разбивается на группы, затем преобразуется в зашифрованный:

1. ЖДИ У МОРЯ ПОГОДЫ

2. ЖДИ#У#МОРЯ#ПОГОДЫ

Если длина шифруемого текста не кратна числу элементов, то при последней перестановке в свободные элементы заносится произвольный символ. В примере *- пустой знак, используемый для дополнения исходного текста до группы.

3. ЖДИ# У#МО РЯ#П ОГОД Ы***

4. ИЖ#Д МУО# #РПЯ ООДГ *Ы**

Еще один из способов перестановки заключается в следующем: исходное текстовое сообщение в размещается в матрице одним способом, а выводится другим способом. Допустим, фраза "ЖДИ У МОРЯ ПОГОДЫ" представляющая собой строку длиной 25 знаков (байт), записывается как матрица 5x5 (или другой размерности). Так как сообщение помещается в матрицу фиксированного размера, то существует вероятность того, что не все элементы матрицы будут использованы.

Ж	Д	И	#	У
#	М	О	Р	Я
#	П	О	Г	О
Д	Ы	#	#	#
#	#	#	#	#

Если затем осуществить запись матрицы по столбцам, то сообщение будет выглядеть следующим образом:

Ж##Д#ДМПЫ#ИОО###РГ##УЯО##

Для декодирования сообщения заполняются столбцы матрицы. Затем матрица может быть отображена в нормальном порядке.

Можно усложнить кодировку, добавив в таблицу строку с ключом, задающим последовательность, в которой отбираются столбца.

Для дальнейшего усложнения перестановки по таблицам для повышения стойкости шифра в таблицу перестановки вводятся неиспользуемые клетки таблицы. Количество и расположение неиспользуемых элементов является дополнительным ключом шифрования.

При шифровании текста в неиспользуемые элементы не заносятся символы текста и в зашифрованный текст из них не записываются никакие символы — они просто пропускаются. При расшифровке символы зашифрованного текста также не заносятся в неиспользуемые элементы.

Для дальнейшего увеличения криптостойкости шифра можно в процессе шифрования менять ключи, размеры таблицы перестановки, количество и расположение неиспользуемых элементов по некоторому алгоритму, причем этот алгоритм становится дополнительным ключом шифра.

Еще более высокую стойкость шифрования можно обеспечить усложнением перестановок по маршрутам типа гамильтоновских. При этом для записи символов шифруемого текста используются вершины некоторого гиперкуба, а знаки зашифрованного текста считываются по маршрутам Гамильтона, причем используются несколько различных маршрутов. Для примера рассмотрим шифрование по маршрутам Гамильтона при n=3.

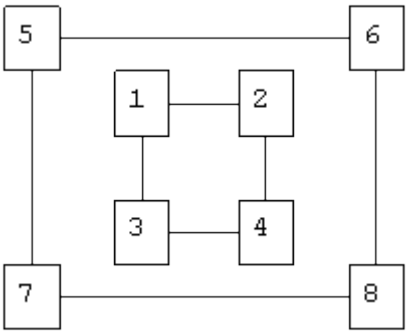


Рис. 6.1. Структура трехмерного гиперкуба

Номера вершин куба определяют последовательность его заполнения символами шифруемого текста при формировании блока. В общем случае n-мерный гиперкуб имеет 2^n вершин.

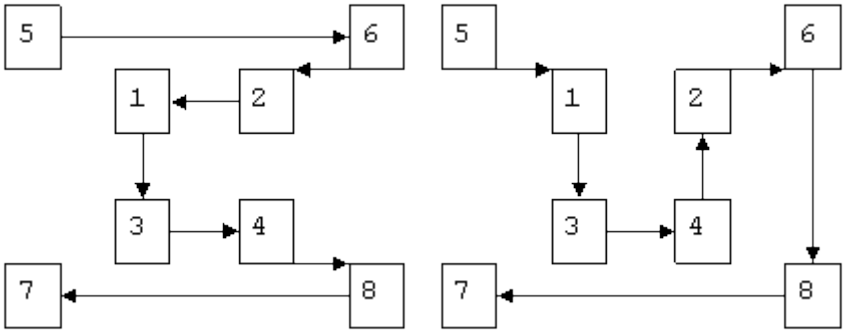


Рис. 6.2. Маршруты Гамильтона

Последовательность перестановок символов в шифруемом блоке для первой схемы 5-6-2-1-3-4-8-7, а для второй 5-1-3-4-2-6-8-7. Аналогично можно получить последовательность перестановок для других маршрутов: 5-7-3-1-2-6-8-4, 5-6-8-7-3-1-2-4, 5-1-2-4-3-7-8-6 и т.д.

Размерность гиперкуба, количество и вид выбираемых маршрутов Гамильтона составляют секретный ключ метода.

Стойкость простой перестановки однозначно определяется размерами используемой матрицы перестановки. Например, при использовании

матрицы 16*16 число возможных перестановок достигает 1.4E26. Такое число вариантов уже невозможно перебрать даже с использованием ЭВМ. Стойкость усложненных перестановок еще выше. Однако следует иметь в виду, что при шифровании перестановкой полностью сохраняются вероятностные характеристики исходного текста, что облегчает криптоанализ.

6.4. Методы шифрования, ориентированные на ЭВМ

6.4.1. Генераторы случайных чисел

Принципиальное значение для надежности шифрования имеет длина кода ключа, т.е. отношение его длины к длине закрываемого им текста. Чем больше оно приближается к единице, тем надежнее шифрование.

При этом следует иметь в виду, что это отношение должно распространятся не только на одиночное сообщение, переданное однократно по назначению, но и на все остальные сообщения, закрытые этим же ключом и передаваемые постоянно в течение времени его существования до замены новым значением. Это объясняется тем, что не известен момент возможного подключения нарушителя к линии связи и поэтому заранее предполагается наихудший вариант, когда нарушитель подключен постоянно. В этом случае при многократном повторении кода ключа по всей длине текста существует опасность его раскрытия статистическим методом.

Для повышения эффективности шифрования используют различные приемы: "длинный" ключ; "двойной" ключ, где два ключа добавляются по модулю два к исходному тексту и другие методы.

Эффективным способом является применение генератора псевдослучайных чисел. Важно, чтобы ПСЧ были воспроизводимыми.

Хорошими показателями обладает линейный генератор ПСЧ, который вырабатывает последовательность ПСЧ T1,T2,T3,...Tm,..., используя соотношение

Ti+1=(aT+c) mod m,

где a, c — константы;

Ti — исходная величина, выбранная в качестве порождающего числа.

Указанное уравнение генерирует ПСЧ с определенным периодом повторения, зависящим от a и c. Значение **m** обычно устанавливают равным $\mathbf{2}^{\mathbf{b-1}}$ или $\mathbf{2}^{\mathbf{b}}$, где **b** длина машинного слова в битах.

ПСЧ, получаемые по данной формуле, можно использовать как последовательность, из которой выбираются ключи. Каждый ключ затем объединяется обратимым образом с соответствующим объемом текста.

Полученный зашифрованный текст достаточно труден для раскрытия, т.к. ключ является переменным. Считается, что если периодичность ключа превышает длину посланного текста и никакой исходный текст не был похищен, то шифр теоретически не раскрываем.

В генераторе ПСЧ может использоваться пароль пользователя или некоторая его трансформацию в качестве инициирующего порождающего числа. Таким образом, каждый пользователь может иметь свой собственный ключ закрытия секретной информации. Для повышения стойкости шифра последовательность ключа может начинаться не ранее, чем с некоторого фиксированного номера j от начала циклов. Например, если j=8, тогда T0, основанное на пароле, и T1,...,T7 будут генерироваться и отбрасываться, а первым элементом последовательности ключа, реально используемым для шифрования будет T8.

6.4.2. Шифрование файлов произвольного доступа

Генераторы ПСЧ широко используются для шифрования последовательных файлов и сообщений, передаваемых по линиям связи, а также их применяют для шифрования файлов прямого доступа ограниченных размеров (базы данных).

В методе Андерсона генерируется две таблицы, каждая равна по длине корню квадратному из количества записей в файле. В этих таблицах содержаться возможные значения **T0** и **C**. Номер записи преобразуется в два числа, по которым из этих таблиц выбираются **T0** и **C** для соответствующей записи, обеспечивая уникальную ключевую последовательность для каждой записи. Даже если информация в отдельной записи будет раскрыта, другая запись не будет расшифрована автоматически, т.к. они зашифрованы различными ключевыми последовательностями. Информация внутри самой записи шифруется/дешифруется аналогично последовательным файлам.

6.4.3. Методы побитовой шифрации. Гаммирование.

ЭВМ положили начало новому методу шифрования, основанному на обработке бит, составляющих символы исходного текста. Хотя побитовая обработка является вариацией шифра замены, но концепции, методы и возможности отличаются довольно значительно, поэтому он рассматривается как отдельный вид шифра.

Шифры побитовой обработки преобразуют исходную информацию в шифрограмму, изменяя исходную битовую комбинацию каждого символа с помощью одного или нескольких следующих логических операторов:

AND OR NOT XOR

Простейший и наименее стойкий шифр использует только оператор NOT (0->1, 1->0). В улучшенном методе кодирования с помощью побитовой обработки используется оператор XOR.

Оператор XOR имеет следующую таблицу истинности:

XOR	0	1
0	0	1
1	1	0

Результат операции **XOR** равен **TRUE** (истина) тогда и только тогда, когда один оператор равен **TRUE**, а второй — **FALSE**(ложь). Это дает **XOR** уникальные свойства: если выполнить операцию XOR над двумя байтами, один из которых называется ключом, а затем взять результат и ключ и снова применить к ним операцию XOR, то в результате получим исходный байт, как показано ниже:

```
          1101 1001 /исходный текст/
XOR       0101 0011 /ключ/
          ____________________________

          1000 1010 /шифр/


          1000 1010 /шифр/
XOR       0101 0011 /ключ/
          ____________________________

          1101 1001 /исходный текст/
```

К методам побитовой обработки информации можно отнести и аддитивные методы. В качестве ключа в аддитивных методах используется некоторая последовательность букв того же алфавита и такой же длины, что и в исходном тексте. Шифрование выполняется путем сложения символов исходного текста и ключа по модулю, равному числу букв в алфавите

(например, если используется двоичный алфавит, то производится сложение по модулю два). Примером такого метода является гаммирование — наложение на исходный текст некоторой последовательности кодов, называемой гаммой.

Процесс наложения осуществляется следующим образом:

1. символы исходного текста и гаммы представляются в двоичном виде и располагаются друг под другом;
2. выполняется сложение по модулю два;
3. полученная последовательность шифрованного текста заменяется символами алфавита.

Если ключ шифрования выбран случайным образом, то раскрыть информацию практически невозможно, однако на практике длина ключа ограничивается возможностями вычислительной техники и аппаратурой обмена данными.

Пример: закодировать числа 012345, используя гамму 49

0000 0001 0010 0011 0100 0101 /исходный текст/

0100 1001 0100 1001 0100 1001 /гамма/

0100 1000 0110 1010 0000 1100 /результат/

При малой длине ключа метод малоэффективен, поэтому чтобы обеспечить большую эффективность шифрования, применяют "длинный" ключ, "двойной" ключ, ключ изменяется с каждым словом.

6.4.4. Методы сжатия

Методы сжатия данных позволяют заданное количество информации упаковать в меньший объем. Они часто используются в вычислительных системах для увеличения ресурсов памяти за счет снижения необходимых объемов, для снижения времени передачи информации, для повышения уровня секретности.

Методы сжатия обычно основаны на замене общих строк или последовательно встречающихся одинаковых знаков другими зарезервированными для этого знаками.

Кодирование по Хаффману.

Часто используемым знакам присваивают короткие коды и более длинные коды редко используемым. Затем сжатые сообщения можно зашифровать с использованием какого-либо криптографического метода. Хаффман представил все буквы английского алфавита в виде двоичного дерева, с учетом частотного распределения букв.

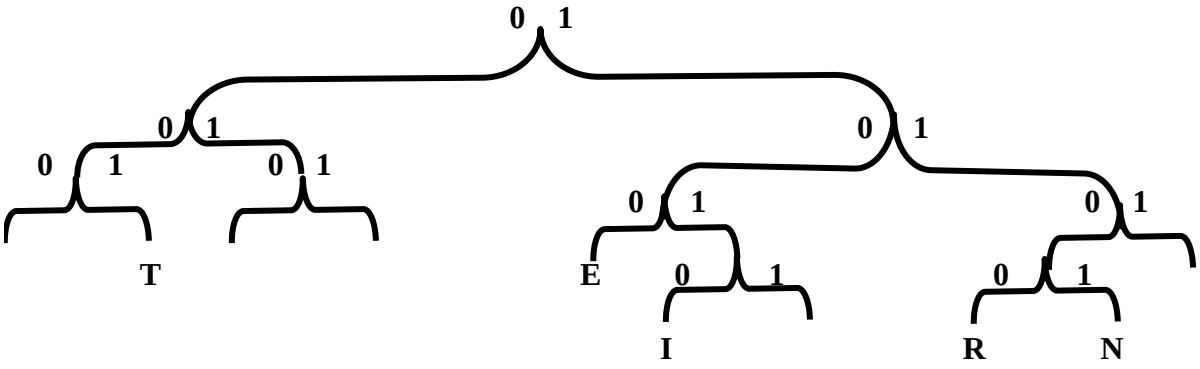


Рис. 6.3. Кодирование по Хаффману

100(e) 1100(n) 011101(w) 11011(d) 00011(m)

1011(r) 0110(s) 00010(u) 1110(o)

Английская фраза WE NEED MORE SUN преобразуется при использовании кодирования Хоффмана в

W E N E E D M O R E S U N

011101 100 1100 100 100 11011 00011 1110 1011 100 0110 00010 1100

Затем строка передается в виде непрерывной строки символов:

011101100110010010011011000111110101110001100000101100

или разбивается на группы по 8 бит:

01110110 01100100 10011011 0001111 10101110 00110000 101100

Для восстановления исходного текста цепочку битов следует просканировать слева направо, что позволит единственным образом расчленить текст на буквы.

Кодирование «восемь в семь».

Большинство компьютеров для кодирования символов использует 8 бит (один байт). Для английского языка заглавные, строчные буквы и знаки пунктуации занимают первую половину таблицы ASCII-кодов, т.е. для их кодирования достаточно 7 бит (2570=128). Поэтому старший бит каждого из семи байт можно использовать для запоминания восьмого символа (упаковка 8 символов в 7 байт).

Данный метод экономит 12,5% памяти. Использование данного типа упаковки данных возможно только с файлами типа ASCII, которые не используют восьмого бита.

Пример: Даны 8 символов, представленные как 8-битовые байты:

байт 1 **0111 0101**

байт 2 0111 1101

байт 3 0010 0011

байт 4 0101 0110

байт 5 0001 0000

байт 6 0110 1101

байт 7 0010 1010

байт 8 0111 1001

Самый простой способ сжатия 8 символов в 7 байт состоит в том, чтобы распределить 7 значащих бит первого байта в семь неиспользуемых старших битовых позиций байтов 2-8. Семь оставшихся байт будут выглядеть следующим образом:

байт 2 **1**111 1101

байт 3 **1**010 0011

байт 4 **1**101 0110

байт 5 **0**001 0000

байт 6 **1**110 1101

байт 7 **0**010 1010

байт 8 **1**111 1001

байт 1 — читать вниз

Для восстановления первого байта вместе собирают старшие биты каждого из 7 байт.

7. Современные криптографические алгоритмы

7.1. Симметричное шифрование

В вычислительных системах широко применяется шифрование с закрытым ключом (симметричное шифрование). При этом один и тот же ключ используется как для кодирования, так и декодирования. При этом обе стороны должны знать секретный ключ, который формируется в центре (ВЦ головной организации, штаб военной организации и т.п.). В такой системе центр не только снабжает пользователей ключами, но и несет ответственность за сохранение их в секрете при изготовлении и доставке.

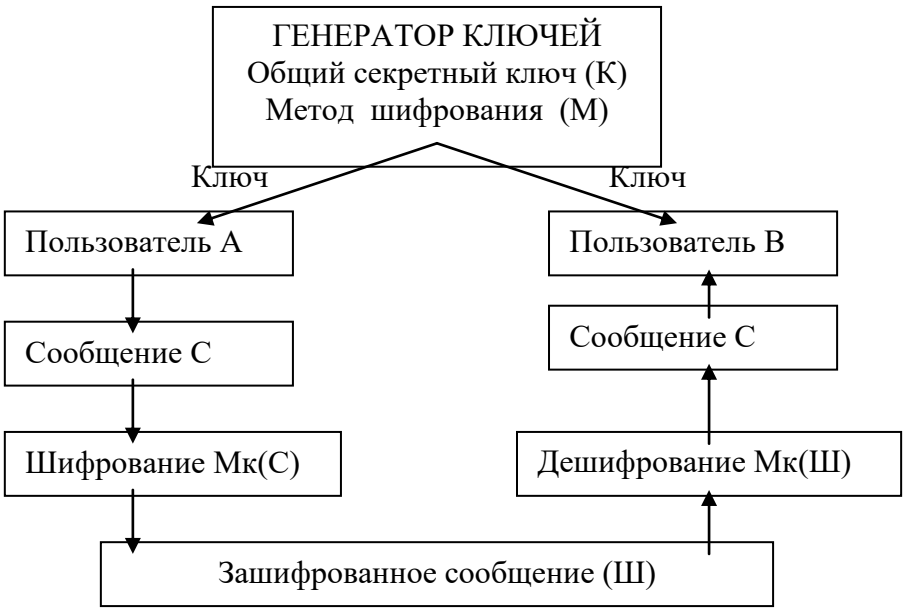


Рис. 7.1. Схема симметричного метода шифрования

7.1.1. Работа с ключами

При использовании симметричного шифрования встает вопрос, как организовать надежное распределение ключей между участниками. Существует две разновидности ключей:

- ключи шифрования данных используются для шифрования сообщений, отправляемых по каналам связи;
- ключи шифрования ключей применяются для шифровки ключей.

Применяя ключи шифрования ключей, зашифрованный ключ передается по тем же каналам, что и сами сообщения. Другой способ распределения ключей состоит в разбиении ключа на части и в передаче их по различным каналам связи.

Сами ключи шифрования ключей передаются по каналам, исключающим подслушивание, например с помощью курьеров, т.к. передача ключей шифрования ключей выполняется сравнительно редко.

Однако компрометация этого ключа даст возможность противнику прочесть всю секретную переписку.

Любой ключ должен использоваться в течение ограниченного периода времени, т.к.

- чем дольше ключ находится в действии, тем больше вероятность, что он будет скомпрометирован;
- длительное пользование ключом увеличивает потенциальный ущерб, который может быть нанесен в случае его компрометации;
- у противника появляется стимул потратить на его вскрытие значительные ресурсы, поскольку полученная выгода позволит оправдать понесенные расходы;
- криптоаналитическую атаку на шифр вести тем легче, чем больше перехваченного шифротекста для него накоплено.

Продолжительность использования ключа зависит от криптосистемы:

- для шифрования речевых сообщений, передаваемых по телефону, имеет смысл менять ключ после каждого разговора;
- в выделенных (закрытых) каналах связи продолжительность использования ключа определяется ценностью шифруемой информации и скорость ее передачи: чем больше скорость, тем чаще надо менять ключи. Если условия позволяют, такие ключи необходимо менять ежедневно.

Ключи шифрования ключей не требуют частой смены, это может быть раз в месяц или даже раз в год.

Ключи, применяемые для шифрования файлов на дисках, слишком часто менять не стоит. Регулярное повторное шифрование файлов на новых ключах только даст больше полезной информации криптоаналитику, который будет пытаться их вскрыть. Лучше применить подход, при котором

каждый файл шифруется при помощи своего ключа. Сами ключи, в свою очередь, зашифровываются на ключе шифрования ключей, который затем прячется в надежное место.

По истечении срока действия ключи должны быть безвозвратно уничтожены.

7.1.2. Алгоритм DES

В начале 70-х годов фирма IBM разработала алгоритм DES — шифровальный стандарт с закрытыми ключами. DES утвержден в качестве национального стандарта США.

Метод DES основан на комбинированном использовании перестановки, замены и гаммированиии. Основные достоинства алгоритма:

- используется только один ключ длиной 56 бит;
- зашифровав сообщение с помощью одного пакета программ, для расшифровки можно использовать любой другой пакет программ, соответствующий стандарту DES;
- относительная простота алгоритма обеспечивает высокую скорость обработки;
- достаточно высокая стойкость алгоритма.

Исходный текст разбивается на блоки длиной 64 бита, которые перемешиваются, затем каждый блок 16 раз шифруется, выполняется конечная перестановка битов. В качестве ключа, который служит для генерирования последовательности знаков случайной гаммы, используется последовательность в 64 бит, из которых значащими являются 56 бит, остальные 8 бит — проверочные для контроля на четность. Такой ключ дает 10^16 различных комбинаций гаммы. Расшифрование в DES является операцией обратной шифрованию и выполняется путем повторения операций шифрования в обратном порядке.

Алгоритм DES вполне подходит как для шифрования, так и для аутентификации данных. Он позволяет непосредственно преобразовывать 64-битовый входной открытый текст в 64-битовый выходной шифрованный текст. Чтобы воспользоваться алгоритмом DES для решения разнообразных криптографических задач, разработаны четыре рабочих режима:

- Электронная кодовая книга ECB (Electronic Code Book).
- Сцепление блоков шифра CBC (Cipher Block Chaining)
- Обратная связь по шифротексту CFB (Cipher Feed Back)
- Обратная связь по выходу OFB (Output Feed Back).

Режим ECB хорошо подходит для шифрования ключей; CFB, как правило, предназначается для шифрования отдельных символов; OFB нередко применяется для шифрования в спутниковых системах связи. Режимы CBC и CFB пригодны для аутентификации данных. Эти режимы позволяют использовать алгоритм DES для:

- интерактивного шифрования при обмене данными между терминалом и главной ЭВМ;
- шифрования криптографического ключа в практике автоматизированного распространения ключей;
- шифрования файлов, почтовых отправлений, данных спутников и других практических задач.

DES применяется очень широко для хранения и передачи данных между различными ВС, в почтовых системах, в электронных системах чертежей. Одним из наиболее важных применений алгоритма DES является защита сообщений электронной системы платежей при операциях с широкой клиентурой и между банками. DES реализован в банковских автоматах, терминалах в торговых точках.

Существуют программные и аппаратные варианты его исполнения, удовлетворяющие пользователей по стоимостным и скоростным характеристикам (20,000- 100,000 бит/с). Это позволяет выполнять кодирование/декодирование в реальном масштабе времени, что особенно важно при работе в быстродействующих вычислительных сетях.

7.1.3. Другие симметричные криптоалгоритмы

В России аналогом DES является стандарт ГОСТ 28147-89. Он представляет собой 64-битовый блочный алгоритм с 256-битовым ключом. Стандарт обязателен для организаций, применяющих криптографическую защиту данных, хранимых и передаваемых в сетях ЭВМ, в отдельных вычислительных комплексах и ЭВМ. Алгоритм предназначен для программной и аппаратной реализации.

Алгоритм IDEA (International Data Encryption Algorithm) , разработан в Швейцарии и принят в качестве европейского стандарта в 1990 году. Это алгоритм блочного шифрования, работающий с блоками открытого текста длиной 64 бита и преобразующий их в блоки шифротекста такой же длины. IDEA значительно безопаснее DES, т.к. имеет ключа длиной 128 бит. IDEA обеспечивает лучшую устойчивость к криптоанализу. Программные реализации IDEA примерно вдвое быстрее реализаций DES. IBM PC 486 шифрует со скоростью 2,4 (Мбит/сек), реализация IDEA на СБИС шифрует со скоростью 177 (Мбит/сек) при частоте 25 МГц.

Алгоритм RC2 представляет собой 64-битовый блочный шифр с ключом переменной длины. Этот алгоритм примерно в два раза быстрее, чем DES. Может использоваться в тех же режимах, что и DES, включая тройное шифрование.

Алгоритм RC5 представляет собой быстрый битовый блочный шифр, который имеет размер блока 32, 64 или 128 бит, ключом длиной от 0 до 2048 бит. Выполняет от 0 до 255 проходов.

Алгоритм CAST представляет собой 64-битовый блочный шифр, использует ключи длиной от 40 до 64 бит, выполняет от 8 проходов. Вскрыть этот шифр можно только путем прямого перебора, другие способы вскрытия не известны.

Алгоритм Blowfish — 64-битовый блочный шифр с ключом переменной длины до 448 бит, выполняет 16 проходов, на каждом их которых осуществляются перестановки, зависящие от ключа, и подстановки, зависящие от ключа и данных. Этот алгоритм быстрее DES.

AES — новый стандарт шифрования США. Этот алгоритм Rijndael разработали два специалиста по криптографии Дж. Деймен и В. Риджмен из Бельгии (2000 год). Алгоритм представляет каждый блок кодируемых данных в виде двухмерного массива байтов размером 4x4, 4x6 или 4x8 в зависимости от установленной длины блока. Далее на соответствующих этапах производятся преобразования над независимыми столбцами, или над независимыми строками, либо вообще над отдельными байтами в таблице. Все преобразования в шифре AES имеют строгое математическое обоснование. Сама структура и последовательность операций позволяют выполнять данный алгоритм эффективно как на 8-битовых, так и 32-битовых процессорах. В структуре алгоритма заложена возможность параллельного исполнения некоторых операций, что может поднять скорость шифрования на многопроцессорных рабочих станциях в четыре раза.

7.1.4. Комбинирование блочных алгоритмов.

До настоящего времени никто не указал какую-либо фундаментальную слабость алгоритма DES. Тем не менее, к недостатку DES относят малую длину ключа в 56 бит.

Современная микропроцессорная техника позволяет уже сегодня за достаточно приемлемое время взламывать симметричные блочные шифры с длиной ключа 40 бит. Для такого взлома используется метод простого перебора — тотального опробования всех возможных значений ключа.

В настоящее время на рынок поступили FPGA-чипы, обладающие возможность перебирать до 30 млн. значений ключа в секунду. Еще большие возможности имеют ASIC-чипы — реализуют скорость перебора до 200 млн. операций в секунду. Стоимость этих чипов составляет всего несколько десятков долларов.

Появление подобных средств ставит под угрозу надежность алгоритмов подобных DES, у которого ключ шифра имеет $\mathbf{2^{56}}$ возможных значений.

Таблица 7. 1
Сравнительный анализ трудоемкости взлома криптоалгоритма DES с длиной ключа 56 бит

Тип атакующего	Бюджет атакующего	Средство атаки	Затраты времени на успешную атаку
Хакер	До 500 $	ПК	Несколько десятков лет
Небольшие фирмы	До 10 тыс. $	FPGA	18 месяцев
Корпоративные департаменты	До 300 тыс. $	FPGA ASIC	19 дней 3 дня
Большие корпорации	До 10 млн. $	FPGA ASIC супер-ЭВМ	13 часов 3 дня 6 минут

Для повышения криптостойкости алгоритмов используют комбинирование блочных алгоритмов для получения новых алгоритмов.

Одним из способов является многократное шифрование, то есть использование блочного алгоритма несколько раз с разными ключами для шифрования оного и того же блока открытого текста. Так, более надежным является тройной DES. Метод кодирует данные, используя алгоритм DES трижды, на каждом этапе используется свой ключ, таким образом, текст шифруется 168-битовым ключом (3x56).

7.2. Асимметричное шифрование

В 70-х годах были разработаны методы шифрования с открытым ключом: для шифрования используется один ключ, а для расшифровки — другой. Такая пара ключей называется открытый /закрытый ключ.

Открытый ключ спокойно передается партнеру, который передает сообщение, зашифрованное открытым ключом. При этом сообщение, зашифрованное с помощью открытого ключа, нельзя расшифровать этим же открытым ключом. Расшифровать подобное сообщение можно только с помощью закрытого ключа владельца.

Чтобы упростить процесс обмена шифрованными сообщениями, открытые ключи всех абонентов единой сети связи часто помещают в справочную базу данных, находящуюся в общем пользовании этих абонентов.

7.2.1. Алгоритм RSA

В криптографии с открытым ключом используют необратимые или односторонние функции, которые обладают следующими свойствами: при заданном значении X относительно просто вычислить значение f(X), однако если У=f(X), то нет простого пути для вычисления X, т.е. очень трудно вычислить f^{-1} (У). Другими словами, невозможно вычислить один ключ из другого.

В 1976 Диффи и Хелман исследовали функцию дискретного возведения в степень и разработали один из методов асимметричного шифрования.

В настоящее время широко распространена система шифрования и аутентификации документов RSA (1977 год авторы Райвест, Шамир, Адлеман). В России аналогом RSA является ГОСТ 34.10.

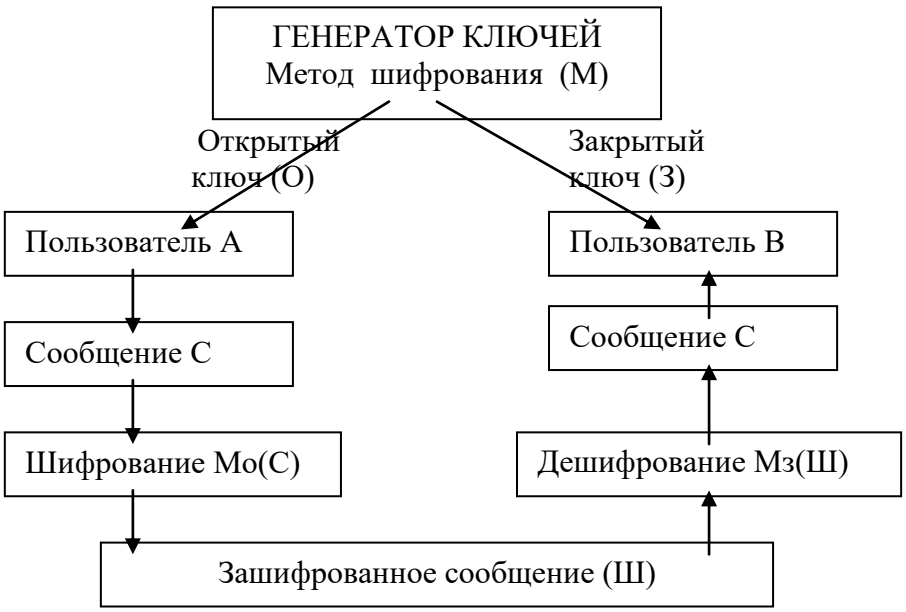


Рис. 7.2. Схема асимметричного метода шифрования

Чтобы использовать алгоритм RSA, сначала надо сгенерировать открытый и закрытый ключи по следующему сценарию.

1. Выбрать два больших простых числа **p** и **q**.
2. Определить **n=p*q.**
3. Выбрать большое случайное число d, которое должно быть взаимно простым с результатом умножения **(p-1)*(q-1)**
4. Определить такое число e, для которого выполняется соотношение (e*d) mod ((p-1)*(q-1)) = 1
5. Назовем открытым ключом числа e и n, закрытым ключом — **d** и **n**.

Чтобы зашифровать данные по известному ключу **{e,n}** выполняют следующие шаги:

1. Разбить исходный текст на блоки, в котором каждый символ можно обозначить целым числом от 0 до (n-1).
2. Зашифровать текст, рассматриваемый как последовательность чисел M(i), над каждым из которых выполняется операция **$C(i)=(M(i)^e) \bmod n$.**

Чтобы раскрыть данные, используя секретный ключ **{d,n}** необходимо выполнить вычисления **$M(i)=(C(i)^d) \bmod n$.**

В результате будет получено исходное множество чисел M(i).

Пример 1.

1. Берем p=3 и q=11.
2. Определяем n=3*11=33
3. Найдем (p-1)*(q-1)=2*10=20. В качестве d выбираем любое число, которое является взаимно простым с 20, например d=3.
4. Выберем число e, для которого выполняется отношение (e*3)mod 20=1, например 7.
5. Представим шифруемое сообщение {A,B,C} как последовательность целых чисел в диапазоне 0-32 {1,2,3} и выполним операцию **$C(i)=(M(i)^e) \bmod n$.**
   C1=(1^7) mod 33 =1 mod 33=1;
   C2=(2^7) mod 33 =128 mod 33=29;
   C3=(3^7) mod 33 =2187 mod 33=9;
   Получили зашифрованное сообщение {1,29,9}
6. Расшифровать {1,29,9} на основе секретного ключа {3,33}
   $M(i)=(C(i)^d) \bmod n$.
   M1=(1^3) mod 33 =1 mod 33=1;
   M2=(29^3) mod 33 =24389 mod 33=2;
   M3=(9^3) mod 33 =729 mod 33=3;

Пример 2.

Исходный текст GAS {7,1 19}
p=5
q= 11
n=55
(p-1)*(q-1)=4*10=40

d=7
e=23
C1=(7^23) mod 55=27368747340080916343 mod55=13
C2=(1^23) mod 55=1
C3=(19^23) mod 55=2578296279453077272482260672 59 mod 55=39
Получим зашифрованное сообщение {13,1,39}
M1=13^7 mod 55= 62748517 mod 55 = 7
M2=1^7 mod 55= 1
M1=39^7 mod 55= 137231006679 mod 55 = 19

Стойкость системы RSA есть функция сложности разложения произведения p*q на простые множители p и q. При достаточной длине этих простых чисел (тысяча двоичных разрядов) такое разложение вычислительно невозможно (т.е. требует ресурсов, недоступных в настоящее время).

В платежных системах Internet система RSA использует ключи длиной 512 и 1024 бит. Системы аутентификации документов, основанные на асимметричном шифровании: RSA, DSS, PGP.

7.2.2. Совместное использование симметричных и асимметричных алгоритмов

Таблица 7. 2
Характеристики криптографических алгоритмов

Характеристика	DES	RSA
Тип ключа	симметричный	асимметричный
Длина ключа	56 бит	256-600 бит
Время генерации ключа	миллисекунды	десятки секунд
Используемая функция	Перестановка и подстановка	Возведение в степень
Скорость работы	быстрая	Медленная
Время на раскрытие шифра	столетия	зависит от длины ключа

Методы открытого ключа целесообразно применять для шифрования небольших информационных массивов, т.к. основной операцией в них является возведение в степень 500-1000 битовых чисел, поэтому при программной реализации шифрование /дешифрование идет на 3-4 порядка медленнее, чем при использовании симметричных методов шифрования. В

связи с этим для обработки больших потоков информации используют специализированные процессоры, выполняющие эту операцию.

На практике криптосистемы с открытым ключом используют для шифрования сеансовых ключей , а не для шифрования сообщений. При этом нет необходимости хранить ключи, после окончания сеанса ключ уничтожается. Последовательность шагов в подобной гибридной системе следующая:

1. Генерируют сеансовый ключ;
2. Сообщение шифруют симметричным методом, используя сеансовый ключ;
3. Затем сеансовый ключ шифруют асимметричным методом, используя открытый ключ получателя;
4. Сообщение и ключ отправляют по сети.

Продолжительность использования открытых ключей зависит от области их применения. Если открытый ключ используется для целей аутентификации или для цифровой подписи, рекомендуют менять его каждые два-три года, чтобы в распоряжении криптоаналитика накапливалось меньше шифротекста, необходимого для организации атаки. При этом старый ключ надо продолжать хранить в секрете: он может понадобиться, например, для подтверждения подлинность подписи, поставленной в течение периода, когда этот ключ был действующим.

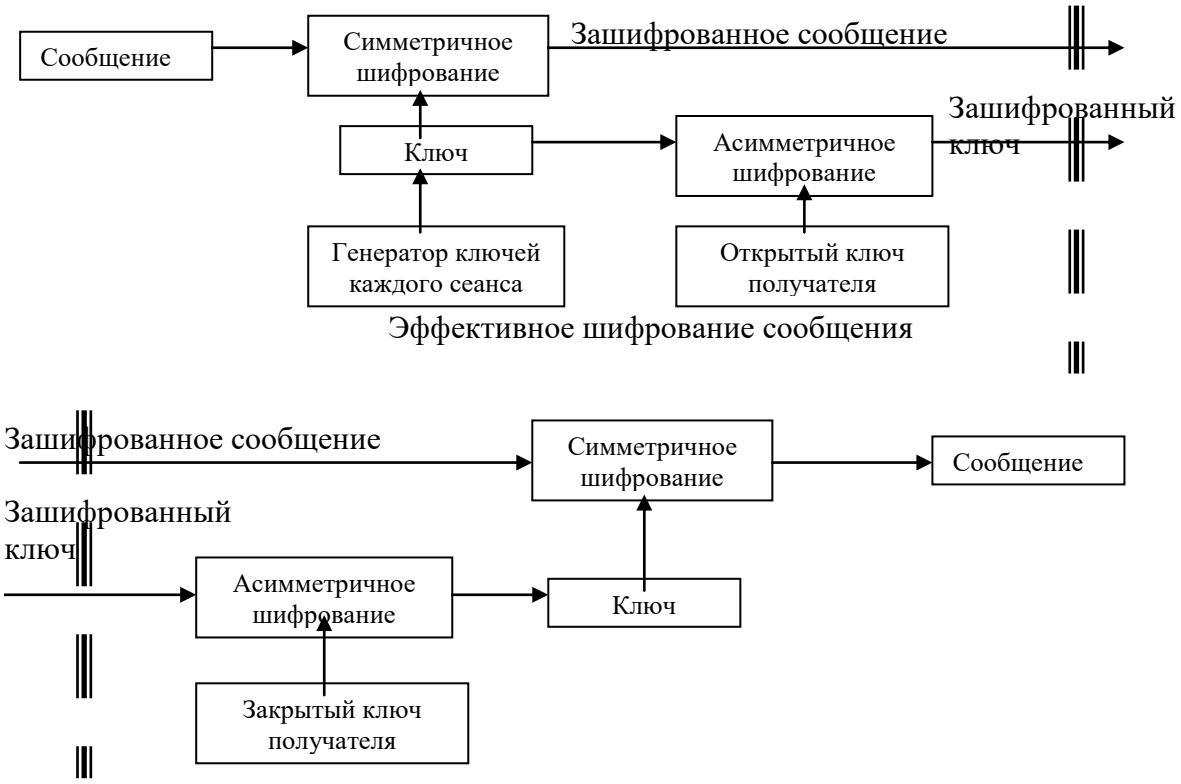


Рис. 7.3. Совместное использование синхронного и асинхронного шифрования

7.3. Электронная цифровая подпись

7.3.1. Задача аутентификации электронных документов

При передаче электронных документов по сетям связи возникает проблема аутентификации автора документа и самого текста, т.е. установления подлинности автора и отсутствия изменений в полученном документе. Целью аутентификации электронных документов является их защита от возможных видов злоумышленных действий, которым относятся:

- активный перехват — нарушитель, подключившийся к сети, перехватывает документы (файлы) и изменяет их;

- маскарад — абонент С посылает документ абоненту В от имени абонента А;
- ренегатство — абонент А заявляет, что не посылал сообщения абоненту В, хотя на самом деле послал;
- подмена — абонент В изменяет или формирует новый документ и заявляет, что получил его т абонента А;
- повтор — абонент С повторяет ранее переданный документ, который абонент А посылал абоненту В.

Эти виды злонамеренных действий могут нанести существенный ущерб банковским и коммерческим структурам, государственным предприятиям, организациям, частным лицам, применяющим в своей деятельности компьютерные информационные технологии.

Решением данной задачи является формирование цифровой электронной подписи (ЭЦП), которая служит для удостоверения подлинности сообщений при пересылке по сети, например для оплаты счетов при покупке товаров.

7.3.2. Подписание документов при помощи симметричных методов

Предположим, что пользователь А должен послать подписанное им цифровое сообщение пользователю Б. Для этого можно использовать симметричную криптосистему и арбитра, с которым имеют шифрованную связь обе стороны со своими собственными ключами Ка и Кб. Последовательность действий (или протокол) следующая:

1. Участник А шифрует свое сообщение на ключе Ка и посылает его арбитру;
2. Арбитр расшифровывает полученное сообщение, присоединяет к нему собственное заявление о принадлежности этого сообщения участнику А, шифрует все на ключе Кб и отправляет участнику Б;

3. Второй участник расшифровывает сообщение, пришедшее от арбитра и содержащее сообщение от А.

Данный протокол предполагает, что:

- оба участника А и Б полностью доверяют арбитру;
- арбитр уверен в авторстве участника А, т.к. только А владеет секретным ключом А.

7.3.3. Подписание документов при помощи асимметричных методов

Протокол подписания документа такой:

1. Участник А шифрует документ с использование своего тайного ключа, тем самым ставя свою подпись;
2. Участник Б расшифровывает документ с использованием открытого ключа участника А, тем самым проверяя подлинность документа.

7.3.4. Отметка о времени подписания документа

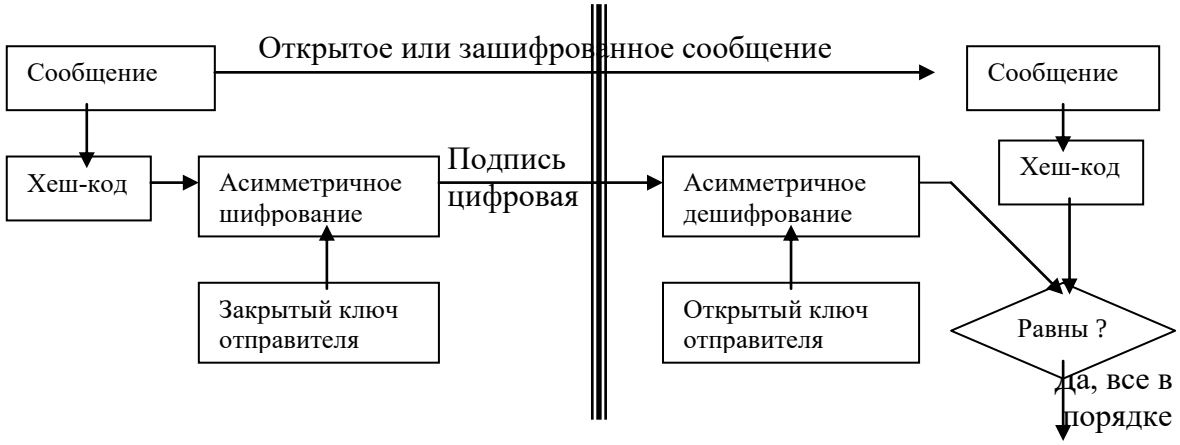
Чтобы избежать подделки денежных чеков, на них обязательно ставится дата и время их подписания, которые в свою очередь, также снабжаются цифровой подписью наравне с самим документом. В результате банк сможет разоблачить попытку дважды обналичить один и тот же чек.

7.3.5. Использование однонаправленных хэш-функций

На практике криптосистемы с открытым ключом не всегда эффективны при подписании больших документов. В целях экономии времени можно подписывать не сам документ, а хэш-значение, вычисленное для этого документа с помощью однонаправленной хэш-функции. Участники протокола должны заранее договорится какой алгоритм шифрования и какую хэш-функцию они будут использовать.

Цифровая подпись формируется следующим образом:

1. Берут файл, который требуется переслать (текстовый или двоичный). Чтобы сформировать подпись, над файлом выполняют операцию хеширования (перемешивания). В результате получают файл, содержащий некоторую строку битов (смесь — hash code) . Хеш-код зависит от информации, заключенной в документе.
2. Полученный хеш-код шифруется асимметричным методом с помощью закрытого ключа отправителя — это и будет цифровая подпись.
3. Получателю передается основное сообщение (зашифрованное или нет) и зашифрованная цифровая подпись.



Эффективная генерация цифровой подписи

Проверка эффективно сгенерированной цифровой подписи

Рис. 7.4. Электронная цифровая подпись

Когда сообщение пришло к получателю, он проделывает следующие действия:

1. С помощью открытого ключа отправителя извлекает из сообщения хеш-код.

2. Прогоняет основной документ через тот же алгоритм хеширования, получает у себя хеш-код, а затем сравнивает оба хеш-кода. Если они совпали, означает, что сообщение пришло в неизмененном виде и от того, кто именно посылал сообщение.

При формировании цифровой подписи применяется шифрование с открытым ключом "наоборот". Сначала отправитель использует свой закрытый ключ для шифрования или подписи документа, и никто кроме него этого сделать не может. Второй (открытый) ключ для расшифровки находится у партнера, который может проверить подпись, раскрыв ее открытым ключом.

ЭЦП обычно содержит дополнительную информацию, однозначно идентифицирующую автора подписанного документа. Эта информация добавляется к документу до вычисления ЭЦП, что обеспечивает и ее целостность. Каждая подпись содержит следующую информацию:

- Дату подписи;
- Срок окончания действия ключа данной подписи;
- ФИО и должность отправителя, название его фирмы;
- Открытый ключ;
- Собственно цифровую подпись.

Важное условие — алгоритм хеширования должен устойчиво получать один и тот же хеш-код из одной и той же информации и различные смеси из различной информации. Так, если добавить в исходный документ один новый символ, хеш-код должен получиться уже другим. Кроме того, чтобы исключить подделку цифровой подписи, функция хеширования должна быть односторонней, это значит, что из исходного файла получить хеш-код можно, а восстановить текст по хеш-коду нельзя.

Простейший способ хеширования заключается в разбивке документа на блоки определенной длины с последующим их сложением в двоичном виде. В настоящее время в криптографии используют следующие программные реализации хеш-функций: MD2,MD4,MD5,SHS.

7.3.6. Несколько подписей под одним документом

В протоколе подписания используется асимметричный метод кодирования и однонаправленная хэш-функция.

1. Участник А подписывает своим закрытым ключом хэш-код документа, отправляет его участнику Б;
2. Участник Б проверяет подлинность подписи участника А, подписывает хэш-код того же самого документа и отправляет свою подпись участнику А;
3. Участник А отправляет документ уже с двумя подписями третьему участнику В, который проверяет подлинность подписей;

8. Работа с Microsoft CryptoAPI

8.1. Архитектура CryptoAPI

Начиная с Windows 95 OSR2 и Windows NT SP3 Microsoft включает в свои операционные системы встроенные средства шифрования, доступные прикладному программисту через библиотеку Microsoft CryptoAPI. Кроме того, для упрощенного доступа к криптографическим функциям из приложений на Visual Basic, VBScript и т.п. предназначен интерфейс CAPICOM, основанный на технологии COM.

Начиная с Windows Vista и Windows Server 2008 поддерживается библиотека Cryptography API: Next Generation (CNG), призванная в будущем заменить CryptoAPI.

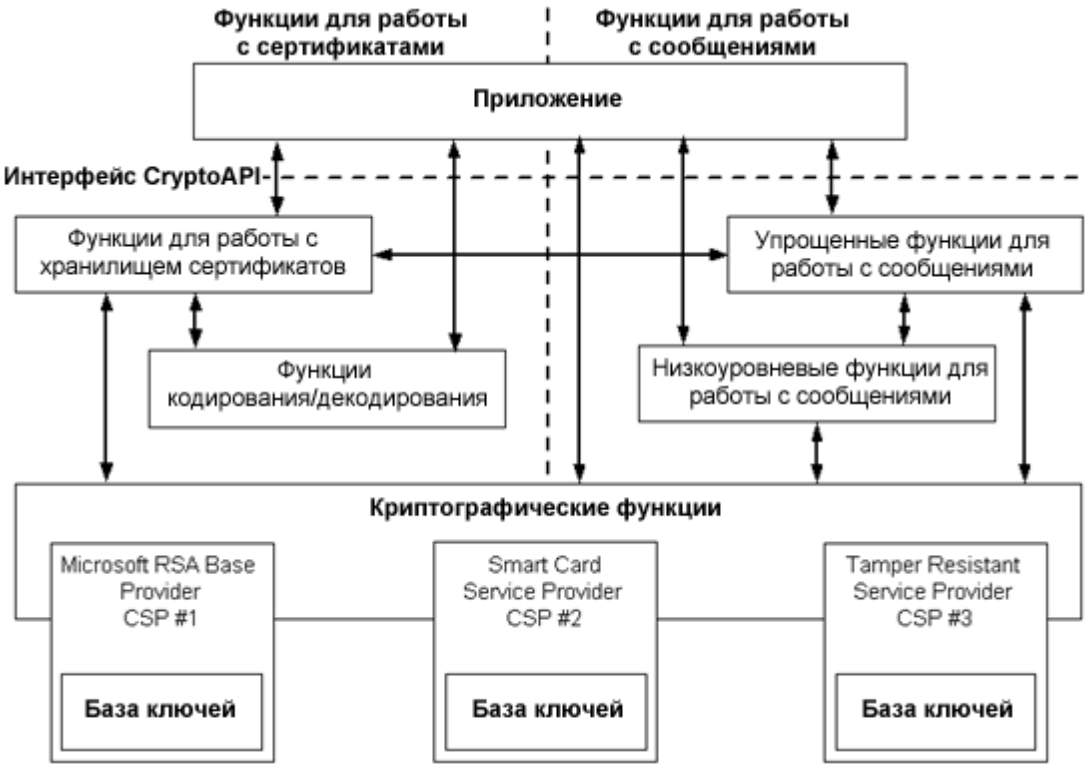


Рис. 8.1. Архитектура CryptoAPI

Концепция CryptoAPI подразумевает сокрытие от программиста всех тонкостей процесса шифрования данных. Одним из ключевых понятий CryptoAPI является криптопровайдер (CSP, Cryptographic Service Provider), реализующий базовый набор криптографических функций, соответствующих тому или иному криптографическому алгоритму (или семейству алгоритмов). Криптопровайдеры реализованы в виде отдельных DLL, так что сторонние разработчики могут самостоятельно включать в состав CryptoAPI реализации своих алгоритмов (хотя для распространения криптопровайдера потребуется подписать его цифровой подписью в Microsoft).

В CryptoAPI можно выделить пять основных функциональных областей (Рис. 8.1). Функции, входящие в каждую из областей, как правило, содержат в имени соответствующие ключевые слова.

- Базовые криптографические функции (***Crypt***). В эту группу входят функции для подключения к криптопровайдеру, генерации и хранения ключей, обмена ключами, управления параметрами ключа.
- Функции кодирования/декодирования (***Crypt***) обеспечивают доступ к алгоритмам шифрования/дешифрования и хэширования.
- Функции работы с хранилищем сертификатов (***Store***) предназначены для управления наборами цифровых сертификатов, используемых при цифровой подписи и шифровании с открытым ключом.
- Упрощенные функции для работы с сообщениями (***Message***) позволяют шифровать/дешифровать сообщения и блоки данных, подписывать их и проверять цифровую подпись.
- Низкоуровневые функции для работы с сообщениями (***Msg***) позволяют более гибко выполнить ту же работу, что и упрощенные функции, но требуют для этого больших усилий со стороны программиста.

Для использования функций CryptoAPI в приложениях, написанных на C/C++, необходимо использовать заголовочный файл ***WinCrypt.h*** и подключить к приложению библиотеку импорта ***Crypt32.lib***.

8.2. Криптопровайдеры

Как было сказано выше, за непосредственную реализацию криптографических алгоритмов в CryptoAPI отвечают криптопровайдеры. При работе с криптопровайдерами приложения имеют ряд ограничений:

- Приложение не может напрямую воспользоваться реализациями функций из криптопровайдера, и все взаимодействие проходит через базовые криптографические функции. В идеальном случае возможна смена одного криптопровайдера на другого без модификации использующего его приложения, хотя на практике некоторые

приложения могут потребовать функциональности определенного криптопровайдера.

- Приложение не имеет прямого доступа к ключам, используемым функциями криптопровайдера.
- Приложения не могут влиять на выполнение криптографических операций иначе как заданием алгоритма и выбором ключа.
- Приложения не отвечают за аутентификацию пользователей, все эту работу выполняет криптопровайдер.

Для работы с криптопровайдером приложение использует его хэндл, играющий такую же роль, как и контекст устройства в GDI. Для получения доступа к криптопровайдеру используется функция ***CryptAcquireContext***, по завершении работы необходимо вызвать функцию ***CryptReleaseContext***. Функции ***CryptSetProvParam*** и ***CryptGetProvParam*** позволяют соответственно задать и узнать ряд характеристик криптопровайдера.

CryptoAPI предоставляет следующие стандартные криптопровайдеры:

- Microsoft Base Cryptographic Provider.
- Microsoft Strong Cryptographic Provider (поставляется начиная с Windows 2000, является расширением базового).
- Microsoft Enhanced Cryptographic Provider (по сравнению с базовым обеспечивает работу с ключами большей длины).
- Microsoft AES Cryptographic Provider (поддержка AES).
- Microsoft DSS Cryptographic Provider (реализует алгоритмы SHA и DSS).
- Microsoft Base DSS and Diffie-Hellman Cryptographic Provider (добавлена поддержка алгоритма обмена ключей Диффи-Хеллмана).

- Microsoft DSS and Diffie-Hellman/Schannel Cryptographic Provider (добавлена аутентификация клиентов по защищенному протоколу взаимодействия).
- Microsoft RSA/Schannel Cryptographic Provider (реализация алгоритмов RSA).

Все эти CSP отличаются друг от друга своими типама, которые определяются набором параметров, включающим:

- алгоритм обмена сессионным (симметричным) ключом.
- алгоритм вычисление цифровой подписи
- формат цифровой подписи
- схема генерирования сессионного ключа по хешу
- длина ключа

8.3. Ключи шифрования

CryptoAPI предоставляет методы для работы с сессионными (симметричными) ключами и с открытыми ключами. Все ключи управляются и используются при помощи неких идентификаторов, и приложение не получает открытого доступа к ним.

При подключении к криптопровайдеру он может предоставить доступ к стандартному хранилищу ключей, либо воспользоваться хранилищем, созданным по запросу приложения.

Создать ключи можно с помощью функций ***CryptGenKey*** (создается случайный ключ) и ***CryptDeriveKey*** (ключ создается на основе некоторых данных, например пользовательского пароля), по завершении работы с ключом он уничтожается функцией ***CryptDestroyKey***. Функции ***CryptSetKeyParam*** и ***CryptGetKeyParam*** позволяют соответственно задать и узнать различные характеристики ключа. Само шифрование данных

производится с помощью функции ***CryptEncrypt***, а дешифрование — ***CryptDecrypt***.

При возникновении необходимости передать ключ за пределы приложения (например, сохранить сессионный ключ для дальнейшего использования, передать публичный или приватный ключ на другой компьютер и т.п.) можно воспользоваться функций экспорта ключей ***CryptExportKey***. Для последующего использования этого ключа он должен быть импортирован с помощью функции ***CryptImportKey***. При экспорте ключа возможно его одновременное шифрование (как правило, с помощью публичного ключа).

8.4. Хэши и цифровая подпись

CryptoAPI позволяет использовать ряд алгоритмов хэширования (в первую очередь, для работы с цифровыми подписями). Доступны следующие алгоритмы:

- MD2, MD4, MD5 — хорошо известные и популярные алгоритмы хэширования, разработанные RSA. Генерируют 128-битные хэши, к использованию не рекомендуются.
- Secure Hash Algorithm (SHA) — алгоритм, разработанный Национальным институтом стандартов и технологий (National Institute of Standards and Technology, NIST) и Агентством национальной безопасности (National Security Agency, NSA) США. Генерирует 160-битный хэш, используется в сочетании с алгоритмами DSA (Digital Signature Algorithm) и DSS (Digital Signature Standard).
- SSL3 Client Authorization Algorithm. Используется для аутентификации клиентов, реализован как конкатенация хэшей MD5 и SHA, подписанная приватным RSA-ключом.

- Message Authentication Code (MAC). Алгоритмы MAC схожи с обычными алгоритмами хэширования, но используют в своей работе шифрование с симметричным ключом.
- Cipher Block Chaining (CBC) MAC. Использует алгоритм блочного шифрования и берет его последний блок в качестве значения хэша.
- HMAC. Более сложная модификация CBC.

Для хэширования данных сначала необходимо создать объект хэширования с помощью функции ***CryptCreateHash***, после чего наполнить его хэшируемыми данными с помощью функции ***CryptHashData***. После передачи последней порции данных, получить значение хэша можно функцией ***CryptGetHashParam***. По окончании работы объект хэширования удаляется функцией ***CryptDestroyHash***. Для проверки цифровой подписи схема работы аналогична, она проводится с помощью вызова функции ***CryptVerifySignature*** после заполнения объекта хэширования данными.

8.5. Сертификаты

Алгоритмы шифрования с открытым ключом и цифровые подписи тесно связаны с понятием сертификата. Цифровой сертификат выдается некой авторизованной организацией (Certification Authority, CA) и помимо пары публичный/закрытый ключ содержит информацию, позволяющую идентифицировать его владельца.

CryptoAPI поддерживает сертификаты, соответствующие спецификации X.509 и содержащие следующие поля:

- Version. Номер версии сертификата.
- Serial Number. Серийный номер сертификата.
- Algorithm Identifier. Алгоритм цифровой подписи, использованный при подписи сертификата.
- Issuer Name. Название организации, выдавшей сертификат.

- Not Before. Дата начала действия сертификата.
- Not After. Дата конца действия сертификата.
- Subject Name. Имя владельца сертификата.
- Algorithm. Алгоритм, для которого будет использован публичный ключ.
- Subject Public Key. Строка, содержащая публичный ключ.
- Issuer Unique ID. Необязательное поле, содержащее уникальный идентификатор организации, выдавшей сертификат. При его наличии поле Version должно быть равно 2.
- Subject Unique ID. Необязательное поле, содержащее уникальный идентификатор владельца сертификата. При его наличии поле Version должно быть равно 2.
- Extensions. Необязательное поле, содержащее возможную дополнительную информацию. При его наличии поле Version должно быть равно 3.

Для получения доступа к хранилищу сертификатов используется функция ***CertOpenSystemStore***, для завершения работы — ***CertCloseStore***. ***CertFindCertificateInStore*** позволяет найти сертификат в хранилище, ***CertEnumCertificatesInStore*** перебрать все сертификаты в хранилище. Обе эти функции возвращают указатель на структуру ***CERT_CONTEXT***, поле ***pCertInfo*** которой содержит указатель на структуру ***CERT_INFO***, заполненную перечисленными выше полями сертификата.

8.6. Сообщения

Под сообщениями в CryptoAPI понимаются данные в формате PKCS #7, разработанном RSA. Существует два класса данных, которые способно содержать сообщение PKCS #7 — базовый (содержит простые данные без

криптографической обработки) и расширенный. Этим двум классам соответствует шесть типов содержимого:

- Data. Данные, представленные строкой байтов.
- Digested Data. Данные вместе с хэшем.
- Encrypted Data. Зашифрованные данные любого вида.
- Signed Data. Данные вместе с зашифрованным хэшем.
- Enveloped Data. Зашифрованные данные вместе с зашифрованным ключом шифрования данных, предназначенным для одного или нескольких получателей.
- Signed-and-Enveloped Data. Зашифрованные данные, зашифрованный ключ шифрования данных, дважды зашифрованный (приватным ключом отправителя и ключом шифрования данных) хэш.

В CryptoAPI предусмотрено два набора функций для работы с сообщениями — низкоуровневые и упрощенные. При использовании низкоуровневых функций вся ответственность за последовательность действий по шифрованию данных, формированию подписей, экспорту, шифрованию и добавлению в сообщение сессионных ключей и т.п. ложится на плечи программиста. В случае упрощенных функций большая часть работы выполняется однократным вызовом одной из перечисленных ниже функций:

- CryptEncryptMessage
- CryptHashMessage
- CryptSignMessage
- CryptDecodeMessage
- CryptDecryptMessage
- CryptDecryptAndVerifyMessageSignature
- CryptGetMessageCertificates
- CryptSignAndEncryptMessage
- CryptVerifyMessageHash
- CryptVerifyMessageSignature
- CryptVerifyDetachedMessageHash
- CryptVerifyDetachedMessageSignature

III. Безопасность распределенных вычислительных систем

9. Основы сетевого взаимодействия

9.1. Сети и протоколы

Под компьютерной сетью понимается объединение двух и более компьютеров, организованное в целях совместного использования ресурсов, передачи информации, осуществления интерактивной связи и т.п. Для подключения к сети компьютер, как минимум, должен иметь плату сетевого интерфейса, ее драйверы и установленную сетевую операционную систему.

Компьютерные сети стали логическим продолжением спирали развития вычислительной техники: от больших ЭВМ с терминалами, обладающими минимальными самостоятельными возможностями, и отдельных персональных компьютеров, обладающих минимальными средствами взаимодействия, к распределенным вычислительным системам, узлы которых способны как решать самостоятельные задачи, так и выделять всей системе те или иные свои ресурсы.

Преимущества вычислительных сетей по сравнению с разрозненными системами достаточно очевидны — это совместное использование различных ресурсов (данных, периферийных устройств), упрощение поддержки пользователей (например, упрощение установки и обновления программного обеспечения), возможность использования средств электронного документооборота, электронной почты и т.п.

В зависимости от масштаба сети выделяют локальные вычислительные сети (ЛВС, LAN), сети масштаба предприятия и глобальные вычислительные сети (ГВС, WAN). Под ЛВС понимается относительно небольшая сеть, состоящая из нескольких компьютеров и

периферийных устройств, соединенных кабелем в пределах ограниченной территории. Сети масштаба предприятия возникают при объединении нескольких локальных сетей, возможно, использующих различную архитектуру и топологию. WAN подразумевают большую географическую протяженность. Граница между сетями масштаба предприятия и WAN становится достаточно условной с появлением мобильных пользователей, подключающихся к сети предприятия через Internet.

В зависимости от способа управления ресурсами вычислительные сети делятся на одноранговые (peer-to-peer) и использующие выделенный сервер (server based). В одноранговых сетях все компьютеры равноправны, каждый функционирует и как клиент, и как сервер, и нет отдельного компьютера, ответственного за администрирование всей сети. Каждый пользователь решает, какие ресурсы своей системы сделать общедоступными. Такой подход целесообразно применять в сетях, включающих небольшое число компактно расположенных компьютеров и пользователей, в которых не слишком критичны вопросы защиты. По мере роста такой сети администрирование ей все усложняется и, как правило, встает вопрос о переходе на схему с выделенным сервером.

Схема с выделенным специализированным сервером (или серверами) предполагает использование отдельных компьютеров, функционирующих исключительно в качестве серверов, оптимизированных для быстрой обработки запросов от сетевых клиентов, для управления защитой файлов и каталогов, ориентированных не централизованное администрирование.

Наибольшую популярность в настоящее время получили комбинированные сети, в которых централизованные серверы отвечают за совместное использование основных приложений, данных, разделенных устройств, авторизацию пользователей и т.п., в то время как программное

обеспечение клиентских рабочих станций дает возможность при необходимости разделять и их ресурсы. Это может осуществляться как пользователями отдельных рабочих станций (при наличии соответствующих прав доступа), так и администраторами сети с помощью централизованной службы каталогов.

Любое взаимодействие в вычислительных системах (а, по сути говоря, и во всем окружающем нас мире), осуществляется в соответствии с некоторым набором правил и процедур, определяющих порядок осуществления данного взаимодействия. Эти правила и технические процедуры, позволяющие двум абонентам общаться друг с другом, называются *коммуникационными протоколами*. Простейший пример подобного протокола — телефонный разговор, при подготовке которого необходимо выполнить ряд действий: поднять трубку, дождаться гудка (в случае обычного телефона, конечно), набрать номер, дождаться ответа и т.д. В протоколах с установкой соединения перед обменом данными отправитель и получатель должны предварительно установить логическое соединение. В процессе установки соединения, часто называемом рукопожатием (handshake) они договариваются о параметрах процедуры обмена, действующих только в рамках данного соединения. После завершения диалога происходит разрыв соединения. В протоколах без предварительной установки соединения, или дейтаграммных, отправитель просто пересылает сообщение при его готовности. Иерархически организованный набор работающих совместно протоколов различных уровней называется *стеком протоколов*. Коммуникационные протоколы могут быть реализованы как программно, так и аппаратно. Протоколы нижних уровней часто реализуются комбинацией программных и аппаратных средств, а протоколы верхних — программными средствами.

В начале 80-х годов International Standards Organization (ISO) выпустила так называемую эталонную модель взаимодействия открытых систем (Open System Interconnection reference model, OSI), основным назначением которой было описание взаимодействия программного и аппаратного обеспечения в разнородных системах при осуществлении сеанса связи. В идеале в эту модель должны были полностью уложиться все существующие и создаваемые впоследствии технологии сетевого взаимодействия. На практике же полное соответствие никогда не достигалось, так что эта модель носит достаточно отвлеченный характер, хотя и позволяет с некоторым уровнем погрешности представить себе общую картину.

В рамках модели OSI все сетевые функции распределены между семью уровнями. Каждый уровень предоставляет некоторый набор услуг, уровни отделяются друг от друга границами — интерфейсами, через которые передаются все запросы от одного уровня к другому. Каждый уровень предоставляет услуги вышележащему уровню, маскируя от него детали реализации. Кроме того, на каждом уровне осуществляется установка виртуальной связи с тем же уровнем на стороне второго абонента.

При передаче информации по сетям она разбивается на относительно небольшие порции данных, называемые пакетами или кадрами (фреймами). Эта разбивка осуществляется для большей управляемости — теоретически несколько пакетов одного сообщения могут пройти от источника адресату по разным маршрутам; при сбое в передаче достаточно будет повторно переслать только отдельный пакет; передача большого сообщения, разбитого на пакеты, не блокирует всю систему. Пакеты могут содержать как собственно передаваемую информацию, так и коды управления сеансом (например, запрос на повторную передачу). При разбиении данных на

пакеты к данным добавляется некоторая управляющая информация. Как правило, это заголовок, включающий в себя, как минимум, адреса источника (source) и назначения (destination), инструкции о маршруте, информацию о том, как в дальнейшем собрать все пакеты, входящие в сообщение, а также, возможно, завершающий пакет трейлер, обычно содержащий информацию для контроля целостности переданной информации — как правило, контрольную сумму.

При прохождении сообщения через протоколы различных уровней, оно постепенно «обрастает» заголовками различных уровней, при необходимости разбиваясь на пакеты меньшего размера. Пакеты нижнего уровня включают, или инкапсулируют в себя пакеты верхних уровней.

Самый низкий, *физический* уровень модели OSI описывает физические свойства среды и сигналов, переносящих неструктурированный, «сырой» поток битов. Здесь реализуются физические характеристики кабеля, электрический, оптический и др. интерфейсы с кабелем, способы соединения, количество контактов в разъемах, способ перевода битов в соответствующие электрические или оптические импульсы и т.п. Содержание битов на данном уровне не имеет никакого значения.

На *канальном* уровне обеспечивается перенос данных по физической среде. Здесь происходит передача кадров данных от вышележащих уровней физическому, упаковка/распаковка кадров данных в «сырой» поток битов и обратно. Этот уровень поделен на два подуровня: управления логической связью (Logical Link Control, LLC) и управления доступом к среде (Media Access Control, MAC). LLC отвечает за контроль ошибок и управление потоком данных и может использовать различные реализации уровня MAC.

MAC работает с применяемыми, например, в Ethernet физическими адресами, привязанными к платам сетевых адаптеров (MAC-адреса).

Сетевой уровень отвечает за адресацию сообщений и преобразование логических адресов в физические. Основные функции этого уровня — маршрутизация, решение задач и проблем, связанных с сетевым трафиком.

Транспортный уровень предоставляет услуги, аналогичные услугам сетевого. Гарантированную доставку обеспечивают далеко не все реализации сетевых уровней, поэтому ее относят к числу функций транспортного уровня. Он гарантирует доставку пакетов без ошибок, в той же последовательности и без дублирования. На нем может происходить переупаковка пакетов — длинные разбиваются, короткие объединяются в один.

Сеансовый уровень обеспечивает установление, управление и разрыв сеансов. Под сеансом здесь понимается логическое соединение между двумя конечными пунктами. Установка сеанса не всегда является необходимым этапом. Так, при телефонном разговоре установка сеанса необходима, при передаче же письма по почте в нем нет необходимости.

Представительский уровень позволяет стекам протоколов, работающих на разных концах соединения, договориться о синтаксисе передаваемых данных. На этом этапе происходит преобразование данных в общепринятый формат, может происходить шифрование, смена кодовой таблицы, сжатие данных и т.п.

Прикладной уровень обеспечивает доступ прикладным процессам к сетевым услугам. Он реализует услуги, напрямую поддерживающие пользовательские приложения — работающие с передачей файлов, электронной почтой и т.п.

9.2. Компоненты вычислительных сетей

Карта сетевого интерфейса, или сетевой адаптер — устройство, взаимодействующее со средой передачи данных. Работает под непосредственным управлением драйвера операционной системы, и разделение функций между адаптером и драйвером может быть различным в разных реализациях. Функционирует на физическом и канальном уровне модели OSI. На физическом уровне работает часть сетевого адаптера, называемая трансивером, которая отвечает за прием и передачу сигналов по линии связи, на канальном — функции получения доступа к разделяемой среде передачи, распознавание MAC-адреса компьютера.

Хост (host) — как правило, компьютер, на котором функционирует некоторый стек сетевых протоколов. В общем случае однозначного соответствия нет: на одном компьютере может поддерживаться несколько хостов, и, наоборот, один хост может представлять несколько компьютеров.

Повторитель (репитер, repeater) — устройство, применяемое для увеличения длины сегментов за счет усиления сигнала. Работает не физическом уровне модели OSI.

Концентратор (хаб, hub) — устройство, позволяющее объединить физические сегменты сети в один логический сегмент. Функционирует на физическом уровне модели OSI. Обычно включает в себя функции повторителя, иногда — дополнительные функции, такие как объединение сегментов с различными физическими средами.

Мост (bridge) — устройство, позволяющее разбить общую среду передачи данных на логические сегменты. Мосты включают в себя несколько портов, связанных с разными сегментами. В процессе работы они выделяют MAC-адреса из принимаемых кадров данных и избирательно их пересылают: в случае, если адресат и отправитель находятся по одну

сторону моста, кадр игнорируется, если по разные — переправляются на соответствующий порт. Некоторое время после начала работы мосту приходится работать в режиме повторителя, пересылая принятые кадры на все свои порты, постепенно формируя таблицу соответствия между номерами портом и адресами компьютеров. Также возможен вариант предварительного исследования топологии сети с помощью тестовых кадров. Важное преимущество мостов — сокращение трафика в отдельных сегментах сети. Мост способен работать как накопительно-передающее устройство, принимающее кадр целиком перед тем как передать его на нужный порт. При этом может производиться дополнительная проверка кадров на целостность. Также мосты могут объединять сегменты с различными физическими средами. Работают на канальном уровне модели OSI.

Коммутатор (switch, switching hub) — устройство, работающее аналогично мосту. Основное отличие классического коммутатора от моста заключается в том, что он не является накопительным устройством и, следовательно, может отправлять кадр в соответствующий порт сразу после получения. С одной стороны, это ускоряет работу, с другой — пересылаются все кадры, включая поврежденные. Современные коммутаторы также могут включать в себя функции сетевого уровня. Такой коммутатор при получении первого пакета данных декодирует его, определяя адрес сетевого уровня, который некоторым образом (например, с помощью ARP-протокола) сопоставляется с нужным портом. Последующие пакеты от данного отправителя к данному адресату коммутируются уже на канальном уровне.

Маршрутизатор (router) — наиболее сложное сетевое устройство, занимающееся пересылкой пакетов, основываясь на таблице логических

адресов. В отличие от моста или коммутатора, он может обладать топологической информацией о всей сети, а не только о примыкающих сегментах, что позволяет ему выбрать оптимальный маршрут в зависимости от заданных критериев. Для составления карты сети маршрутизаторы обмениваются между собой служебными сообщениями, содержащих информацию о тех подсетях, которые им известны. Они обеспечивают более надежную защиту сети — в отличие от коммутатора, обязанного повторить на всех своих портах кадр с неправильным адресом, маршрутизатор его проигнорирует. Функционирование осуществляется на сетевом уровне модели OSI.

Шлюз (gateway) — аппаратное и программное обеспечение, осуществляющее трансляцию протоколов, размещенное между взаимодействующими сетями. Может работать практически на любом уровне модели OSI.

Прокси (proxy, посредник) — практически синоним шлюза, работающий, как правило, на прикладном уровне. Может осуществлять дополнительную обработку передаваемых данных — например, их кэширование.

Узел (node) — обобщенное название устройств, входящих в сеть.

9.3. Стек протоколов TCP/IP

В настоящее время стек протоколов TCP/IP является практически самым распространенным средством организации распределенных систем. Он включает в себя четыре уровня, достаточно приблизительно соответствующим уровням модели OSI. Разбивка данных по пакетам/фреймам, их «обрастание» заголовками при перемещении с уровня на уровень происходит здесь так же, как это было описано в главе 9.1.

Нижний уровень стека TCP/IP в целом соответствует физическому и канальному уровням модели OSI. Здесь располагается аппаратно-зависимое программное обеспечение, реализующее распространение информации на том или ином отрезке среды передачи данных, а также определяется сама среда передачи данных. Непосредственно в протоколах TCP/IP этот уровень не регламентируется, но включает все популярные стандарты, такие как Ethernet, Token Ring, PPP, X.25 и т.п. Включение этих и будущих стандартов в стек TCP/IP осуществляется заданием правил инкапсуляции пакетов верхних уровней в их кадры.

Уровень *межсетевого взаимодействия* соответствует сетевому уровню модели OSI. Он реализует концепцию коммутации пакетов без установления соединений. Основным протоколом здесь является межсетевой протокол IP (Internet Protocol), изначально спроектированный как протокол передачи пакетов в сетях со сложной топологией. Протокол IP является дейтаграммным, то есть не гарантирующим доставку. На этом же уровне работает и несколько других протоколов, например отвечающие за составление и модификацию таблиц маршрутизации — RIP (Routing Internet Protocol), OSPF (Open Shortest Path First), отвечающий за поиск физического адрес устройства при известном IP-адресе протокол ARP (Address Resulution Protocol), протокол межсетевых управляющих сообщений ICMP (Internet Control Message Protocol), предназначенный для обмена информацией об ошибках между маршрутизаторами и источниками пакетов.

Далее следует *транспортный*, или основной уровень, примерно соответствующий транспортному и сеансовому уровню модели OSI. Здесь работают два основных протокола — **UDP (User Datagram Protocol)**, представляющий собой простейшую надстройку над IP и также не гарантирующий доставку, и протокол управления передачей **TCP**

(Transmission Control Protocol), обеспечивающий надежный, защищенный от ошибок канал связи. TCP самостоятельно обрабатывает потерянные или повторяющиеся IP-дейтаграммы, и при необходимости переупорядочивает дейтаграммы, принятые с нарушением порядка.

Наконец, на вершине стека расположен *прикладной* уровень, соответствующий представительскому и прикладному уровню модели OSI. Здесь работает масса протоколов, используемых прикладными программами и интернет-сервисами — такие как ftp, http, smtp, pop3 и многие другие.

Рассмотрим более подробно основные протоколы сетевого и транспортного уровней, являющиеся, по сути, основой функционирования большинства современных сетей.

Для передачи данных между двумя компьютерами, включенными в сеть, состоящую из множества машин, необходимо выбрать способ отличать один от другого, т.е. ввести систему идентификации, или адресации. В современных IP-сетях используется трехуровневая адресация, включающая в себя физический, или MAC-адрес, логический IP-адрес и символьное, или доменное имя.

MAC-адрес — это физический адрес, привязанный, например, к плате сетевого адаптера. Для Ethernet-адаптеров он представляет собой целое число длиной 6 байт, из которых старшие три байта являются идентификатором фирмы производителя, младшие назначаются самим производителем уникальным образом. В современных сетевых адаптерах возможна и самостоятельная смена MAC-адреса пользователем.

В протоколе IP версии 4 (IPv4) логический IP-адрес представляет собой целое четырехбайтное число, обычно записываемое тетрадами вида w.x.y.z, где w, x, y, z — десятичные числа от 0 до 255. Для облегчения

управления весь массив адресов разделен на сети, принадлежащие к одному из пяти классов — A, B, C, D и E. На практике используются сети первых трех классов. Класс D зарезервирован для группового вещания (multicast), класс E зарезервирован для будущих применений. С каждым IP-адресом связана 32-разрядная маска подсети, которая при побитном умножении на IP-адрес делит его на две части: уникальный идентификатор сети и уникальный идентификатор хоста в пределах данной сети. Различия между классами приведены в таблице 9.1.

Таблица 9. 1
Классы сетей в IP-адресации

Класс	Старшие биты адреса	w	ID сети	ID хоста	Маска подсети	Число сетей	Число хостов
A	0	1-126	W	x.y.z	255.0.0.0	126	16777214
B	10	128-191	w.x	y.z	255.255.0.0	16384	65534
C	110	192-223	w.x.y	z	255.255.255.0	2097151	254
D	1110	224-239	–	–	–	–	–
E	11110	240-247	–	–	–	–	–

Возможно переопределение маски подсети путем увеличения в ней количества битов, равных единице. Этот процесс называется делением на подсети (subnetting).

Часть адресов имеет предопределенные значения:

- 0.0.0.0 — адрес узла, сгенерировавшего пакет;
- <0[.0[.0]]>.<id узла> — узел, принадлежащий той же сети, что и пакет;
- 255.255.255.255 — широковещание в сети, совпадающей с источником пакета (local broadcast);
- <id сети>.255[.255[.255]] — широковещание в сети класса A, B или C;
- 127.<любой> –адрес локальной заглушки (loopback).

Поиск физического адреса устройства при известном IP-адресе осуществляется с помощью **ARP-протокола**, и этот процесс называется

разрешением адреса (address resolution). В локальных сетях протокол ARP использует широковещательные кадры протокола канального уровня для поиска в сети узла с заданным IP-адресом, рассылая ARP-запрос с вложенным в него IP-адресом, по всем узлам сети. Все узлы сравнивают IP-адрес со своим и в случае совпадения формируют ARP-ответ, включая в него свой физический адрес. Для обратной процедуры — получения IP-адреса по физическому — используется RARP-протокол. Формат ARP/RARP-сообщения выглядит следующим образом:

<table>
<tr><td colspan="2">Тип оборудования (16 бит)</td><td>ID протокола (16 бит)</td></tr>
<tr><td>Длина физического адреса (8 бит)</td><td>Длина IP-адреса (8 бит)</td><td>Операция (16 бит)</td></tr>
<tr><td colspan="3">IP-адрес отправителя (32 бита)</td></tr>
<tr><td colspan="3">Аппаратный адрес цели (переменная длина, для Ethernet — 48 бит)</td></tr>
<tr><td colspan="3">IP-адрес цели</td></tr>
</table>

В дальнейшем сетевые операционные системы кэшируют результаты ARP-запросов в локальных таблицах, обновляемых с некоторым таймаутом.

Как уже упоминалось, основным протоколом сетевого уровня в ip-Сетях является IP-протокол, передающий информацию в дейтаграммах следующего формата:

<table>
<tr><td>Версия (4 бита)</td><td>Длина заголовка (4 бита)</td><td>Тип обслуживания (8 бит)</td><td colspan="2">Длина пакета (16 бит)</td></tr>
<tr><td colspan="3">Идентификатор (6 бит)</td><td>Флаги (3 бита)</td><td>Смещение фрагмента (13 бит)</td></tr>
<tr><td colspan="2">Время жизни (8 бит)</td><td>ID протокола (8 бит)</td><td colspan="2">Контрольная сумма (16 бит)</td></tr>
<tr><td colspan="5">IP-адрес отправителя</td></tr>
<tr><td colspan="5">IP-адрес получателя</td></tr>
<tr><td colspan="4">Параметры</td><td>Выравнивание</td></tr>
<tr><td colspan="5">Данные</td></tr>
</table>

- Версия — 4 для IPv4.

- Длина заголовка — длина IP-заголовка в 32-разрядных словах. Минимальное значение — 5 (20 байт).
- Тип обслуживания — указывает необходимость специальной обработки, в настоящее время обычно игнорируется.
- Длина — общая длина пакета в байтах.
- Идентификатор — предназначен для объединения данных, разбитых на фрагменты, имеет одинаковое значение во всех фрагмента.
- Флаги — указывают на фрагментирование пакетов.
- Время жизни (Time To Live, TTL) — в настоящее время используется как счетчик количества узлов, через которые прошел пакет. По умолчанию стартовое значение равно 32, каждый узел уменьшает его на 1, при достижении нуля пакет удаляется.
- Идентификатор протокола — показывает, пакет какого протокол верхнего уровня вложен в поле данных.
- Контрольная сумма — используется при выявлении ошибок передачи пакета, рассчитывается только для заголовка. Пересчитывается каждым узлом из-за смены значения TTL.
- Параметры — поле переменной длины, может отсутствовать, обычно используется при отладке сети.
- Выравнивание — дополнительное поле, добавленное для того, чтобы IP-заголовок всегда имел длину, кратную 32 байтам.
- Данные — собственно передаваемая информация.

Поскольку длина IP-пакета описывается 16-битным числом, его максимальный размер может быть длиннее 65535 байт, а с учетом того, что длина заголовка — минимум 20 байт, на долю данных остается не более 65515 байт. В то же время сети, по которым передаются IP-пакеты, могут иметь различные максимальные размеры кадров (фреймов). Эта величина

называется максимальной единицей транспортировки (Maximim Transfer Unit, MTU), для сетей Ethernet, например, она равна 1500 байт. Именно в функции IP-протокола входит разбиение полученных данных на пакеты достаточной длины для инкапсуляции в кадры нижнего уровня и последующая их сборка.

Задачей протоколов транспортного уровня является передача информации между прикладными процессами. Соответственно, в отличие от IP-протокола, которому достаточно доставить пакет хосту-получателю, и которому для решения задачи идентификации отдельного хоста достаточно иметь его IP-адрес, здесь возникает вопрос, каким образом можно различать потоки данных, направленные различным сетевыми приложениям, одновременно работающим на одном хосте. Для этого в стеке протоколов TCP/IP начиная с транспортного уровня возникает понятие *портов*. Порты представляют собой очереди, организованные операционной системой, в которых накапливаются пакеты транспортного уровня, и которые однозначно связаны с некоторыми прикладными процессами. Для их идентификации используются номера портов — целые числа от 1 до 65535. Таким образом, поток данных однозначно идентифицируется набором из четырех значений: IP адресами отправителя и получателя и номерами портов отправителя и получателя. Ряд номеров портов закреплен за наиболее популярными сетевыми сервисами, хотя в принципе любое сетевое приложение способно занять любой свободный порт. Исходящие, или клиентские порты, распределяются динамически. Один серверный порт способен обслуживать несколько потоков данных от разных подключенных клиентов.

Простейшим протоколом транспортного уровня является UDP. Он занимается уплотнением и разуплотнением канала (в другой

терминологии — мультиплексированием и демультиплексированием) на основе портов, позволяя одновременно взаимодействовать нескольким приложениям с помощью потоков дейтаграмм. Также он занимается вычислением контрольной суммы и проверкой целостности принятых дейтаграмм (в отличие от IP, который контролирует лишь целостность заголовка).

UDP работает напрямую с IP, вкладывая в его данные свои пакеты. UPD-пакет выглядит следующим образом:

Порт отправителя (16 бит)	Порт получателя (16 бит)
Длина пакета (16 бит)	Контрольная сумма (16 бит)
Данные	

Контрольная сумма представляет собой 16-битное дополнение суммы байт псевдозаголовка, UDP-заголовка и данных. Псевдозаголовок UDP имеет следующий формат:

IP-адрес отправителя (32 бита)		
IP-адрес получателя (32 бита)		
8 нулевых бит	IP протокола = 17	UDP-длина (16 бит)

Т.е. в начало UDP пакета добавляется псевдозаголовок, после чего вычисляется общая контрольная сумма.

Несмотря на всю простоту UDP, большинство прикладных приложений предпочитает использовать в работе второй протокол транспортного уровня — TCP, обеспечивающий надежную транспортировку данных между прикладными процессами за счет установления логического соединения. Как и UDP, он использует порты. Как и IP, он разбивает поток данных на пакеты (в терминах TCP — сегменты), на этот раз имеющие достаточный размер для размещения внутри IP-дейтаграмм. Сегменты могут иметь разный размер, но перед

соединением его участники договариваются о максимальном размере сегмента.

В процессе передачи данных получатель отсылает подтверждение каждой принятой IP-дейтаграммы. Если отправитель в течение некоторого времени не получает подтверждения, он повторно отправляет дейтаграмму. Это подтверждение называется квитанцией, а весь процесс — квитированием. В TCP используется механизм «скользящего окна», позволяющий минимизировать простои, связанные с ожиданием подтверждения. Окно в данном случае определено как максимальное число одновременно отправляемых пакетов, которые источник может послать без подтверждения приема. Т.е. при размере окна в N пакетов при отправке первого пакета источнику разрешается отправить еще N-1 пакет, не дожидаясь подтверждения. При регулярном поступлении потока квитанций, скорость передачи данных приближается к максимальной для данного протокола в данном канале. Заметим, что при N=1 мы получаем вырожденный случай, сводящийся к простой старт-стопной полудуплексной передаче с ожиданием подтверждения. Регулируя размер окна, можно влиять на загрузку сети

Передача данных в TCP происходит в рамках установленного логического соединения, или сессии. Установка его происходит с следующей последовательности:

1. Инициатор соединения отсылает TCP запрос на открытие порта для передачи.
2. Далее TCP на стороне инициатора отсылает запрос приложению, с которым должно быть установлено соединение. Запрос включает в себя целочисленное значение номера стартовой последовательности (Initial Sequence Number, ISNa).

3. Приемная сторона открывает порт для приема данных, порт для передачи данных и передает инициатору соединения подтверждающее сообщение, включающее в себя ISN приемника ISNb и значение ISNa, увеличенное на 1.
4. Инициатор соединения открывает порт для приема и отсылает приемнику ISNa+1 и ISNb+1.

Посылкой этого пакета завершается процедура рукопожатия (handshake), и TCP-соединение между хостами считается установленным. Далее по этому каналу передается информация, сопровождаемая увеличивающимися значениями SNa и SNb, которые фактически являются идентификаторами соединения. Пакеты с некорректными значениями SN игнорируются. Принципиально возможна атака с подменой одного из участников, если атакующему удалось подобрать «правильные» стартовые значения ISN.

TCP-сегмент имеет следующий формат:

<table>
<tr><td colspan="3">Порт отправителя (16 бит)</td><td>Порт получателя (16 бит)</td></tr>
<tr><td colspan="4">Порядковый номер</td></tr>
<tr><td colspan="4">Номер подтверждения</td></tr>
<tr><td>Смещение данных (4 бита)</td><td>Резерв (6 нулевых бит)</td><td>Флаги (6 бит)</td><td>Размер окна (16 бит)</td></tr>
<tr><td colspan="3">Контрольная сумма (16 бит)</td><td>Указатель на срочные данные (16 бит)</td></tr>
<tr><td colspan="3">Параметры</td><td>Выравнивание (до величины, кратной 32)</td></tr>
<tr><td colspan="4">Данные</td></tr>
</table>

Любопытно, что в заголовке отсутствует информация о размере блока данных сегмента — он определяется как разность между размером полезной нагрузки IP-пакета, в который заключен TCP-сегмент, и размером TCP-заголовка. TCP-заголовки имеют переменную длину, но не менее 20 байт.

Поле «Флаги» описывает содержимое сегмента. Оно может содержать комбинацию следующих значений:

- URG (32): есть срочные данные;
- ACK (16): подтверждение приема;
- PSH (8): как можно быстрее передать данные приложению;
- RST (4): переустановить соединение;
- SYN (2): синхронизация соединения;
- FIN (1): конец передачи.

Флаг SYN передается при установке соединения, значение, переданное при этом в поле «Порядковый номер», и является упомянутым ранее ISNa. Если установлен флаг ACK, то в поле «Номер подтверждения» находится значение, которое отправитель ожидает увидеть в поле «Порядковый номер» следующего сегмента. Подтверждения могут поглощаться — приемник может сделать паузу, принять несколько сегментов, идущих подряд, и отослать подтверждение для последнего сегмента.

Поле «Смещение данных» содержит размер заголовка в 32-битных словах.

«Контрольная сумма» представляет собой 16-битное дополнение суммы 16-разрядных слов TCP-заголовка, данных и псевдозаголовка, имеющего такой же формат, как и у псевдозаголовка UPD.

«Указатель на неотложные данные» позволяет сигнализировать о наличии в потоке срочных данных, его значение в сумме с текущим порядковым номером дает порядковый номер данных, следующих непосредственно за срочными.

В протоколе предусмотрен механизм «зондирования пустым окном» — когда передающий узел не дожидается подтверждения приема, он отправляет сообщение с одним байтом данных, на которое должен немедленно ответить принимающий узел. Это делается для выявления различия между обрывом связи и ситуацией, когда принимающий узел не может принять очередные данные из-за того, что их не успевает обработать приложение верхнего уровня. Такое зондирование позволяет сохранить TCP-соединение активным до тех пор, пока приемник не окажется готовым продолжить работу. Первое «пустое окно» отправляется после истечения стандартного интервала ожидания, в случае отсутствия ответа интервалы между последующими «пустыми окнами» экспоненциально увеличиваются.

Хосты и маршрутизаторы обмениваются между собой управляющей информацией с помощью протокола ICMP (Internet Control Message Protocol). Его пакеты также вкладываются в IP-пакеты. Маршрутизаторы используют ICMP, отправляя сообщения об ошибках, хосты обычно отправляют ICMP-сообщения для проверки качества связи и оценки времени отклика (ping).

9.4. Основы маршрутизации

Маршрутизаторы являются основным средством, позволяющим объединить несколько сетей. В процессе своей работы они используют таблицы маршрутизации, которые позволяют определить дальнейший маршрут пакета. Таблица маршрутизации должна периодически обновляться, чтобы соответствовать топологии сети, и для этого используют алгоритмы маршрутизации. Алгоритмы маршрутизации работают поверх маршрутизируемых протоколов, например IP. Существует несколько классификаций алгоритмов маршрутизации:

- **Статическая и динамическая маршрутизации.** Под статической маршрутизацией понимается ручная корректировка таблицы администратором. Алгоритмы динамической маршрутизации способны самостоятельно адаптироваться к изменениям конфигурации сети.
- **Централизованная и распределенная маршрутизации.** В случае централизованной маршрутизации таблицы составляются в одном центре управления маршрутизацией, после чего рассылается по маршрутизаторам. Основной плюс этой схемы — единое представление о сети у всех маршрутизаторов. Минус — ее уязвимость в случае выхода из строя центра маршрутизации. В случае распределенной маршрутизации формированием таблиц занимаются различные узлы, после чего таблицы сводятся воедино. Этот вариант более надежный, но требует большего времени на синхронизацию таблиц по всей сети, что может привести к ошибкам маршрутизации.
- **Канальная (или по состоянию канала) и векторная маршрутизация.** В случае канальной маршрутизации маршрутизатор широковещательно рассылает список подключенных к нему узлов,

который называется состоянием канала. В случае векторной маршрутизации маршрутизаторы рассылают всю известную им таблицу маршрутизации всем смежным маршрутизаторам. Векторная маршрутизация проще и требует меньше памяти, но синхронизация таблиц маршрутизации всей сети в ее случае занимает большее время. Как правило, она применяется в относительно небольших сетях.

- **Внутридоменная и междоменная маршрутизации.** В случае, когда сеть предприятия состоит из нескольких подсетей, некоторые маршрутизаторы могут играть роль внутридоменных, не знающих ничего о маршрутизации вне данной сети. Маршрутизатор стоящий между сетью и внешним миром, является междоменным.
- **Одноуровневая и иерархическая маршрутизации.** В первом случае все маршрутизаторы равноправны, в последнем — доступ к некоторым узлам осуществляется только через маршрутизаторы верхнего уровня.
- **Однопутевая и многопутевая маршрутизации.** В случае однопутевой маршрутизации каждому адресату соответствует один путь, что сокращает размеры таблицы маршрутизации, в случае многопутевой — несколько, что повышает надежность сети.
- **Маршрутизация источником и маршрутизатором.** В первом случае передающий хост указывает в каждом пакете его полный маршрут, и на долю маршрутизатора остается только следовать этим указаниям. Предварительно хост отправляет по всем известным ему маршрутам пробный пакет, в который маршрутизаторы дописывают информацию о себе и пересылают дальше. Эта схема используется крайне редко. Гораздо чаще хост просто отправляет пакет на адрес получателя, а все решения принимают маршрутизаторы.

Перечислим наиболее популярные алгоритмы динамической маршрутизации.

- **RIP (Routing Information Protocol).** Динамический, распределенный, векторный, внутридоменный, одноуровневый, однопутевой. Один из старейших протоколов динамической маршрутизации, используемый до сих пор. Предназначен для использования в малых и средних сетях, максимальное длина пути — 15 хопов. Не способен оперативно учитывать изменение свойств сети, например загруженность каналов.
- **OSPF (Open Shortest Path First).** Динамический, распределенный, канальный, внутридоменный, иерархический, многопутевой. При изменении таблицы состояния каналов маршрутизатор извещает о нем другие узлы, для выявления сбоев рассылает сообщения «еще жив», поддерживает запросы QoS (quality of service), позволяющие как можно быстрее отправить срочные данные. Применяется в более крупных сетях.
- **EGP (Exterior Gateway Protocol).** Динамический, распределенный, векторный, междоменный, одноуровневый, однопутевой. Регулярно рассылает по смежным узлам обновления, содержащие информацию о том, какие маршрутизаторы и хосты достижимы из каких маршрутизаторов. Используется для межсетевой маршрутизации, например в Internet.
- **BGP (Border Gateway Protocol).** Динамический, распределенный, векторный, междоменный, одноуровневый, многопутевой. Разрабатывался в качестве преемника EGP. Предусматривает выявление кольцевых маршрутов и использование метрик при определении оптимального маршрута. Изначально маршрутизаторы обмениваются полными таблицами маршрутизации, после чего —

только обновлениями, причем, хотя таблица маршрутизации может содержать несколько возможных маршрутов к приемнику, в обновлении рассылаются только оптимальные маршруты. Поддерживаются сообщения «еще жив». Интересно, что этот протокол работает поверх TCP, что обеспечивает ему дополнительную надежность.

- **IGRP (Interior Gateway Routing Protocol).** Динамический, распределенный, векторный, внутридоменный, одноуровневый, многопутевой. Разработан для применения в сложных сетях. При выборе маршрута учитывает различные факторы — такие как размеры сообщений, пропускная способность сети, ее надежность, загруженность. Вес каждого фактора может быть задан администратором.

Таблица маршрутизации содержит следующие основные поля:

- Назначение (network destination).
- Маска (netmask).
- Шлюз (Gateway).
- Интерфейс (Interface).
- Метрика (Metric).

Таблица применяется для выбора интерфейса, на который должен быть переправлен пришедший пакет. Выбор происходит следующим образом: происходит последовательное логическое умножение адреса назначения пакета на маски подсетей, выбирается строчка таблицы, в которой результат логического умножения совпадет со значением поля «назначение», и пакет переправляется на соответствующий интерфейс для передачи на соответствующий шлюз. Алгоритмы динамической маршрутизации могут использовать метрики для выбора одного маршрута

из нескольких подходящих. При статической маршрутизации единственный случай использования метрик — задание нулевого значения для задания локальных интерфейсов, не использующих шлюзы. Порядок хранения и просмотра записей в таблице может быть различным, но он должен обеспечивать приоритет записей с меньшим адресным пространством (в масках которых больше значимых битов) перед записями с большим адресным пространством. Другими словами, таблица маршрутизации строится по древовидному принципу — задаются общие правила для сетей большего размера, которые могут уточняться для их подсетей, подсетей подсетей и т.п.

Своя таблица маршрутизации находится на каждом узле, подключенном к сети. В простейшем случае отдельного хоста стартовая таблица формируется прозрачно для пользователя на основе сетевых настроек. Вывести информацию о ее текущем состоянии можно с помощью команды *netstat –r* (в Windows — также *route print*). В ОС семейства Unix в поле «интерфейс» при этом обычно выводится имя соответствующего интерфейса, в ОС семейства Windows — его IP-адрес.

Минимальная таблица маршрутизации состоит из двух записей. Для хоста с ip-адресом 192.168.0.55 сети класса C в формате *<назначение>/<маска>/<шлюз>/<интерфейс>* это выглядит так:

- 127.0.0.1/255.0.0.0/127.0.0.1/127.0.0.1 — задание интерфейса локальной заглушки или петли (loopback).
- 0.0.0.0/0.0.0.0/192.168.0.55/192.168.0.55 — задание шлюза по умолчанию, которым в данным случае является локальная плата сетевого интерфейса.

Для более реалистичного примера сети с одним маршрутизатором, имеющим внутренний интерфейс с адресом 192.168.0.1 (опуская запись, описывающую loopback, которая всегда настраивается одинаково):

- 0.0.0.0/0.0.0.0/192.168.0.1/192.168.0.55 — все пакеты во внешний мир теперь должны идти через маршрутизатор.
- 192.168.0.0/255.255.255.0/192.168.0.55/192.168.0.55 — пакеты в локальную сеть проходят через локальную сетевую плату.

Пакету с адресом назначения 192.168.0.66 теперь в принципе соответствуют обе записи, но использоваться будет вторая, как более частная.

Если маршрутизатор объединяет данную сеть с сетью 192.168.1.x, в которой имеет адрес 192.168.1.1, то его таблица включает следующие записи:

- 192.168.0.0/255.255.255.0/192.168.0.1/192.168.0.1
- 192.168.1.0/255.255.255.0/192.168.1.1/192.168.1.1

По мере наращивания сложности сети — например, если наша первая сеть помимо маршрутизатора, соединяющего ее со второй сетью, содержит второй маршрутизатор, объединяющий ее с третью сетью и т.п., перед администратором встает выбор — настраивать ли каждую рабочую станцию так, чтобы она сама могла выбирать, в каком случае на какой маршрутизатор отсылать пакеты, или оставить конфигурацию с одним шлюзом по умолчанию, доверив окончательный выбор этому шлюзу. Единственный минус второго подхода — увеличение числа переходов в том случае, если пакет все-таки должен был быть направлен на второй маршрутизатор — как правило, с избытком компенсируется упрощением конфигурирования сети.

Рассмотрим подробнее конфигурацию сети, описанную выше:

- сети 192.168.0.x, 192.168.1.x и 192.168.2.x
- маршрутизатор, установленный между сетями 192.168.0.x и 192.168.1.x, имеющий адреса 192.168.0.1 и 192.168.1.1.
- маршрутизатор, установленный между сетями 192.168.1.x и 192.168.2.x, имеющий адреса 192.168.1.2 и 192.168.2.1.

В большинстве случаев, конечно, гораздо эффективнее было бы объединить все три сети с помощью одного маршрутизатора, но тем не менее ситуация вполне допустимая. В таком случае все хосты в каждом сегмента настраиваются на использование одного шлюза по умолчанию (допустим для простоты, что это хосты 192.168.x.1). Тогда второй маршрутизатор настраивается аналогично предыдущему случаю:

- 192.168.1.0/255.255.255.0/192.168.1.2/192.168.1.2
- 192.168.2.0/255.255.255.0/192.168.2.1/192.168.2.1

В конфигурацию же первого добавляется правило для перехвата пакетов, предназначенных для третьей сети:

- 192.168.0.0/255.255.255.0/192.168.0.1/192.168.0.1
- 192.168.1.0/255.255.255.0/192.168.1.1/192.168.1.1
- 192.168.2.0/255.255.255.0/192.168.1.2/192.168.1.1

Таким образом, при отправке с хоста 192.168.1.55 пакета на адрес 192.168.2.55 пакет попадет на первый маршрутизатор, после чего будет переправлен на второй.

9.5. Протоколы прикладного уровня

Существует огромное количество прикладных программ, использующих в своей работе прикладные протоколы, работающие поверх стека TCP/IP. В принципе, никто не мешает при разработке нового приложения разработать и новый протокол прикладного уровня. Часто именно так и происходит. С другой стороны, использование стандартных

протоколов обеспечивает достаточно высокую степень совместимости между самыми различными приложениями.

Ряд протоколов, обеспечивающих базовые средства обмена информацией в Internet (передачу файлов, электронной почты, гипертекстовой информации и т.п.) стандартизирован достаточно давно. Рассмотрим некоторые из них.

Telnet. Обычно под telnet понимают триаду, состоящую из telnet-демона (сервера), telnet-клиента и telnet-протокола.

Протокол описывает двунаправленное взаимодействие между терминальным устройством и сервером. В его основе лежит концепция сетевого виртуального терминала (Network Virtual Terminal, NVT). NVT — воображаемое устройство, находящееся на обоих концах соединения. При установке соединения клиент и сервер договариваются о параметрах диалога на основе спецификации NVT, в которую клиентское и серверное приложение преобразовывают характеристики физических устройств. Таким образом обеспечивается совместимость устройств с разными возможностями — характеристики диалога определяются устройством с меньшими возможностями. Взаимодействие может происходить в двух режима — в командном (command mode) и в режиме удаленного терминала (input mode). В режиме удаленного возможно использование режима с буферизацией (line-by-line), в котором символы накапливаются в локальном буфере и отправляются на удаленную машину одним пакетом, и без буферизации (character-at-a-time), в котором каждый введенный символ передается на удаленную машину, после чего оттуда приходит «эхо».

Telnet-сервер использует стандартный порт 23 и обеспечивает создание для каждого удаленного клиента псевдотерминала. Telnet-клиент обеспечивает пользовательский интерфейс и взаимодействует с сервером.

Наиболее часто этот протокол используется для эмуляции терминала удаленного компьютера. Часто telnet-клиенты используются для тестирования других протоколов.

FTP (File Transfer Protocol) — протокол, предназначенный для перемещения файлов между двумя системами. Он предусматривает первичную аутентификацию пользователя при его подключении к серверу, который далее может просматривать каталоги сервера и передавать файлы в обоих направлениях. В процессе работы устанавливается два TCP-соединения: по одному передаются данные, по другому — команды. За выполнение команд отвечает модель интерпретатора протокола (Protocol Interpreter, PI), за передачу файлов — модуль передачи данных (Data Transfer, DT). В качестве PI используется протокол Telnet — или независимая реализация, или реализация средствами ftp-сервера. FTP-команды и FTP-ответы передаются открытым текстов. FTP-команды представляют собой текстовые строки с необязательными параметрами, они позволяют пройти процесс аутентификации, управлять каталогами и файлами, задавать режимы передачи и т.п. FTP-ответ — некое трехзначное число, которое интерпретируется клиентской программой, после которого следует некоторый текст, предназначенный для пользователя.

Данные могут пересылаться в нескольких форматах, из которых обычно используются текстовый (ascii) и бинарный (binary). Возможно три режима передачи данных — поточный (файл представляется как последовательность байт), блочный (файл передается как набор блоков, в заголовках которых указан их размер, признак последнего блока и бит четности) и сжатый (строка из нескольких повторений одного байта заменяется двумя байтами).

Сервер использует в работе два TCP-соединения — управляющее и для передачи данных. Команды передаются клиентами на 20-й порт, а с 21 порта сервер инициализирует соединение с тем портом, с которого был отправлен клиентский запрос.

Также существует упрощенный аналог FTP — TFTP (Trivial File Transfer Protocol). В отличие от FTP, он не поддерживает механизм идентификации пользователя, в качестве транспорта использует протокол UDP и не позволяет просматривать каталоги, осуществлять по ним навигацию и т.п. Его единственное назначение — передача файлов. Данные передаются блоками по 512 байт, требуется подтверждение каждого блока. Используемый порт — 69-й UDP.

Самые популярные протоколы, используемые для поддержки электронной почты — **SMTP, POP3, IMAP**. SMTP служит для отправки почты клиентом и доставке ее на сервер получателя, где она и хранится, как в обычном почтовом ящике. POP3 и IMAP предназначены для окончательной доставки получателю.

Сообщение электронной почты состоит из двух частей. Телу письма предшествует ряд заголовков, отделенных от него пустой строкой, содержащей только символы возврата строки и перевода строки. Каждый заголовок представляет собой текстовую строку, содержащую ключевое слово (From, To, Reply to и т.п.), далее следует символ «:», за ним — значение заголовка, и заканчивается все символами возврата каретки и перевода строки. Как минимум, здесь указывается адрес получателя, сюда же добавляется информация о почтовых серверах, через которые прошло письмо.

Изначально протоколы электронной почты были приспособлены только под передачу текстовых сообщений. В дальнейшем была предложена

технология MIME (Multipurpose Internet Mail Extension), определяющая способ передачи и более сложных сообщений, включающих текстовую и бинарную информацию. При использовании MIME возможна передача составных сообщений, включающих различные виды данных, передача одной информации в нескольких форматах (позволяя сделать выбор используемого формата принимающей программе), передача нескольких частей сообщения, которые должны воспроизводиться одновременно (например, звук и видео) и т.п. То, как должны интерпретироваться данные, задается в MIME-заголовках. При использовании MIME данные кодируются одним из способов, позволяющих преобразовать их из вида, в котором значащими являются все 8 бит, в 7-разрядный ASCII-код. Чаще всего используются кодировки Quoted-Printable и Base64.

Протокол SMTP (Simple Mail Transfer Protocol) предназначен для передачи почты от клиента серверу и передачи от одного сервера другому. Обычно серверы ожидают соединения на 25-м порту. В качестве транспорта используется протокол TCP.

Как и в случае FTP, стороны обмениваются командами и ответами. Команды представляют собой текстовые строки, заканчивающиеся символами возврата каретки и перевода строки. Ответы также содержат трехзначный код возврата. Обычно диалог начинается с отправки команды HELO, после чего отдается команда MAIL, начинающая передачу почты. Если сервер отвечает OK, отправитель шлет команду RCPT, содержащую адрес получателя. Если сервер готов принять почту на этот адрес, отдается команда DATA, за которой следует само сообщение. В протоколе определен и ряд других вспомогательных команд.

Существует рад расширений SMTP, поддерживающих обратную совместимость. Основной недостаток классического SMTP — его полная

анонимность, позволяющая легко подключаться к произвольным серверам, отправляя сообщения от чужого имени.

POP3 (Post Office Protocol версии 3) предназначен для получения почты конечным адресатом с почтового сервера. Используя его, почтовый клиент подключается к серверу и получает все накопившиеся для него сообщения. Он использует все ту же схему команда/ответ, командами опять-таки служат текстовые строки, начинающийся с ключевого слова. Диалог, как правило, начинается с команд USER и PASS, предназначенные для авторизации пользователя. Далее с помощью команды LIST можно получить информацию об отдельном или всех сообщениях, командой RETR получить текст сообщения с указанным номеров, командой DELE удалить указанное сообщений, командой QUIT прекратить сеанс связи. Порт по умолчанию — 110.

Наряду с POP3 используется протокол IMAP (Internet Message Access Protocol), ориентированный на выборочную работу с сообщениями. В отличие от POP3, при использовании которого почта, как правило, немедленно удаляется клиентом с сервера, IMAP-клиенты обычно хранят ее на сервере, загружая на компьютер пользователя только выбранные сообщения. IMAP более сложен в реализации, и предъявляет повышенные требования к устройствам хранения данных сервера. С другой стороны, он обладает рядом преимуществ — удобством использования одного почтового ящика из нескольких точек доступа, обработку почты сервером, коллективные рабочие ящики и т.п. Порт по умолчанию — 143.

Протокол **HTTP (HyperText Transfer Protocol)** — протокол передачи гипертекстовой информации, лежащий в основе World Wide Web.

Для адресации документов в WWW используется понятие унифицированного указателя ресурсов (Uniform Resource Locator, URL). В самом общем виде URL записывается следующим образом:

[протокол]://[имя[:пароль]@][адрес[:порт]/][путь][документ][?дополнительная информация]

Содержимое квадратных скобок является необязательным, любая часть URL может быть опущена. Здесь

- *протокол* — символьное обозначение протокола, используемого для доступа к ресурсу (помимо http, это может быть, например, ftp);
- *имя* — имя пользователя;
- *пароль* — в сочетании с именем пользователя используется при работе с ресурсами, доступ к которым ограничен;
- *адрес* — адрес узла в доменной или цифровой форме;
- *порт* — номер порта, если он отсутствует, используется порт по умолчанию для данного протокола;
- *путь* — путь на сервере от его корневого каталога, либо относительно текущего каталога;
- *документ* — имя документа;
- *дополнительная информация* — используется при работе с серверными приложениями.

Гипертекстовые документы, представленные в WWW, имеют одно принципиальное отличие от традиционных гипертекстовых документов — связи, в них использующиеся, не ограничены одним документом, и более того, не ограничены одним компьютером. Для подготовки гипертекстовых документов используется язык HTML (HyperText Markup Language — язык разметки гипертекстовых документов), предоставляющий широкие возможности по форматированию и структурной разметке документов,

организации связей между различными документами, средства включения графической и мультимедийной информации. HTML-документы просматриваются с помощью специальной программы — браузера. HTML-документ состоит из текста, представляющего собой содержание документа, и *тегов*, определяющих его структуру и внешний вид при отображении браузером.

Тег представляет собой ключевое слово, заключенное в угловые скобки. Различают одинарные теги, как например <*p*>, и парные, как <*body*> <*/body*>, в последнем случае действие тега распространяется только на текст между его открывающей и закрывающей скобкой. Теги также могут иметь параметры — например, при описании страницы можно задать цвет фона, цвет шрифта и т.д.: <*body bgcolor=”white” text=”black”*>.

Текст всего документа заключается в теги <*html*>, сам документ разбивается на две части — заголовок и тело. Заголовок описывается тегами <*head*>, в которые могут быть включены название документа (с помощью тегов <*title*>) и другие параметры, использующиеся браузером при отображении документа. Тело документа заключено в теги <*body*> и содержит собственно информацию, которую видит пользователь. При отсутствии тегов форматирования весь текст выводится в окно браузера сплошным потоком, переводы строк, пробелы и табуляции рассматриваются как пробельные символы, несколько пробельных символов, идущих подряд, заменяются на один. Для форматирования используются следующие основные теги:

<p> – начало нового абзаца, может иметь параметр, определяющий выравнивание: <p align=right>;

 – перевод строки в пределах текущего абзаца;

 – выделение текста полужирным шрифтом;

 – выделение текста курсивом;

<u></u> – выделение текста подчеркиванием

Ссылка на другой документ устанавливается с помощью тега *…*, где URL — полный или относительный адрес документа. При этом текст, заключенный в тег <a>, обычно выделяется подчеркиванием и цветом, и после щелчка мышью по этой ссылке браузер открывает документ, адрес которого указан в параметре href. Графические изображения вставляются в документ с помощью тега *<img src=”URL”>*.

HTTP — компактный, быстрый и простой протокол, использующий небольшое количество команд. Инкапсуляция данных в нем осуществляется с помощью расширений MIME, а структура пересылаемых данных напоминает электронную почту. Пользовательская программа обращается к HTTP-серверу, передавая ему команду, URL и некоторую дополнительную информацию. Используемый по умолчанию порт — 80.

Запросы HTTP-клиента содержат командную строку, включающую команду и URL запрашиваемого документа, а также версию протокола. Далее могут следовать HTTP-заголовки, каждый из которых размещается на отдельной строке. Список заголовков заканчивается пустой строкой, после которой может следовать собственно тело запроса. Наиболее часто используются команды GET и POST. GET предназначена для поиска ресурса на сервере и его передачи клиенту. POST позволяет передать информацию на сервер в теле запроса.

В зависимости от результата обработки запроса, сервер возвращает клиенту строку с кодом состояния (который может означать успешную обработку запроса, информацию об ошибке, приглашение перейти на другой адрес и т.п.), набор заголовков, формирующихся по правилам, аналогичных клиентским, и тело ответа.

Основным недостатком HTTP-протокола с точки зрения написания серверных приложений является то, что в отличие от перечисленных выше протоколов, он не имеет состояний. Запрос каждой страницы с сервера идет индивидуально, независимо от других, что является довольно большой проблемой при написании серверных приложений. Одним из решений, позволяющих обойти это ограничение протокола, стало использование так называемых *cookie*. При необходимости сохранить, например, идентификатор сессии, сервер формирует специальный заголовок *Set-Cookie:*. Клиент, получив этот заголовок, может извлечь из него информацию и сохранить на диске пользователя. При дальнейшем обращении к данному серверу клиент всякий раз будет передавать эту информацию в заголовке *Cookie:*. Предельный размер каждого cookie определен в 4К, и для каждого сервера допускается наличие не более 20 cookie. Информация, записанная сервером в cookie, может быть считана только этим же сервером.

Для взаимодействия с серверными приложениями используется механизм **CGI (Common Gateway Interface, общий шлюзовой интерфейс)**. В ответ на действия пользователя, используя CGI, web-сервер вызывает внешнюю программу (CGI-приложение) и передает ей информацию, полученную от клиента (например, переданную Web-браузером). Далее CGI-приложение обрабатывает полученную информацию, и результаты ее работы передаются клиенту.

Схема взаимодействии выглядит следующим образом:

1. Пользователь заполняет экранную форму, описанную в html-файле с помощью тега <form>, и нажимает на кнопку «Submit». Возможен также запрос при непосредственном использовании адреса

CGI-приложения — указывая его в строке Location браузера, в тэге <img> с помощью средств включения сервера (SSI) и т. д.

2. На основе информации из формы браузер формирует HTTP-запрос и отправляет его серверу. Информация приводится к виду $param_1=value_1\&param_2=value_2\ldots\&param_N=value_N$, где $param_i$ — имя соответствующего поля ввода, $value_i$ — введенное в него значение. Все символы, за исключением букв латинского алфавита, цифр, символа подчеркивания, дефиса и точки при этом передаются в закодированном виде %XX, где XX — шестнадцатеричное представление данного символа. Символ пробела может быть заменен символом «+». Если указано, что при передаче должен использоваться метод GET, эта строка передается непосредственно в URL (например, http://www.somehost.com/cgi-bin/script.cgi?param1=value1&param2=value2). При использовании метода POST через заголовок передается информация о типе содержимого запроса (для форм это, как правило, application/x-www-form-urlencoded), а также длина строки. Сама строка в этом случае передается непосредственно в теле запроса (примеры приведены чуть ниже). В заголовках запроса также передается значительное количество вспомогательной информации: тип браузера, адрес страницы, с которой был произведен запрос, и т. д. Вся эта информация передается в HTTP-заголовках, имеющих вид «Имя: значение». Отделяются друг от друга заголовки с помощью символа новой строки, завершается их список еще одним символов новой строки.
3. Сервер вызывает CGI-приложение, и в зависимости от метода запроса передает информацию из формы через переменную окружения QUERY_STRING (в случае GET) либо через стандартный ввод (в

случае POST). Также формируются другие переменные окружения, такие как HTTP_USER_AGENT, REMOTE_HOST и др. Информация для окружения берется из HTTP-заголовков.

4. CGI-приложение считывает строку с переданной информацией со стандартного ввода (stdin) или из переменной окружения QUERY_STRING. Обработав информацию, программа, как правило, либо переадресует браузер на некоторый существующий документ с помощью http-заголовка Location, либо формирует виртуальный документ, посылая информацию на стандартный вывод (stdout). Телу документа предшествуют HTTP-заголовки, описывающие тип возвращаемых данных, управляющие кэшированием, работой с cookies и т. д. Все это передается серверу.
5. Сервер пересылает ответ CGI-приложения браузеру, дополняя их при необходимости кодом возврата и вспомогательными заголовками. При этом используется один из двух способов — перенаправление браузера на новый адрес с помощью http-заголовка Location, либо формирование виртуального документа. В последнем случае значение, переданное в заголовке Content-type, используется браузером для интерпретации идущей следом информации — например, text/html для виртуальных html-документов.
6. Браузер, основываясь на заголовках HTTP, интерпретирует ответ CGI-приложения и выводит его для просмотра пользователем.
7. Реализовать CGI-приложение можно на любом языке, способном генерировать код для серверной платформы или для которого доступен интерпретатор.

9.6. Управление сетями

Помимо протоколов и служб, непосредственно отвечающих за передачу пользовательской информации, в современных сетях используется ряд технологий, имеющих к этой задаче косвенное отношение, но без которых, тем не менее, нормальное функционирование сетей было бы невозможным. Некоторые из них, такие как протокол ARP и протоколы динамической маршрутизации, были уже описаны ранее. В данном разделе мы рассмотрим некоторые технологии, применяемые для управления сетями.

В разделе, посвященном IP-адресации, мы остановились на том, что каждый компьютер подключенный к сети, использующей TCP/IP, должен иметь уникальный IP-адрес. IP-адреса могут быть назначены отдельным сетевым интерфейсам явным образом, путем указания соответствующих настроек. По мере роста сети усложняется и ее администрирование, появляется необходимость вести список выделенных адресов, устранять возможные конфликты и т.п. Для упрощения этой задачи обычно используется протокол **DHCP (Dynamic Host Configuration Protocol)**, позволяющий хостам получать настроечную информацию от DHCP-сервера. Эта информация включает IP-адрес, маску подсети, адрес шлюза, время жизни пакета, адрес DNS-сервера и т.п. Причем клиенты могут воспользоваться ей как полностью, так и частично — например, компьютеры, у которых явно указан IP-адрес, могут получить с DHCP-сервера информацию о шлюзе.

DHCP-сервер хранит базу IP-адресов, доступных для выделения клиентам. Клиент отправляет широковещательный запрос, получив который, сервер возвращает ему ответ с предложением своего ресурса. Возможна ситуация, когда в одной локальной сети действуют несколько

DHCP-серверов. Клиент, таким образом, может получить сразу несколько предложений. Выбрав одно из них, он опять рассылает широковещательное сообщение, об этом информирующее. Для всех прочих DHCP-серверов это означает, что клиент не намерен использовать их ресурсы. DHCP-сервер способен резервировать адреса. Обычно это делается для явно назначенных IP-адресов, например адресов маршрутизаторов.

IP-адреса успешно справляются с задачей идентификации хостов, но имеют небольшой недостаток — их неудобно использовать людям. Уже на ранних этапах существования Internet наряду с числовыми адресами использовались и символьные имена хостов. Задача соответствия решалась с помощью специального файла hosts, который представлял собой обычный текстовый файл, каждая строчка которого содержала две записи — IP-адрес и имя хоста, разделенные пробельными символами. Этот файл до сих пор можно найти практически в любой системе, использующей IP-адресацию. Когда-то в файлах hosts были перечислены все хосты, подключенные к Internet, и при добавлении нового узла их приходилось синхронизировать. Разумеется, из-за быстрого роста сети это не могло долго продолжаться.

В настоящее время задача трансляции символьных имен и IP-адресов возложена на службы **DNS (Domain Name Service)**. Под доменом понимается множество машин, которые администрируются и поддерживаются как единое целое. DNS поддерживает иерархически организованное пространство имен, традиционно представляемое в виде дерева. На вершине дерева находится корневой узел (root). Доменное имя строится по следующему принципу: к имени хоста добавляются имена родительского узла и всех его прародителей, вплоть до корневого узла. Имена разделяются точками, корневой узел тоже обозначается точкой, но обычно опускается: my_host.my_domain.my_parent_domain.ru. Каждый

компонент имени домена может включать до 63 байт, общая длина — не более 256 байт.

Под корневым доменом расположен ряд доменов первого уровня — .com, .edu, .net, .gov, .mil и т.п. Кроме того, к доменам верхнего уровня относятся национальные домены с двухбуквенными именами (например, **.ru**, **.de**), администрированием которых занимаются национальные институты.

Поддерево пространства имен DNS называется зоной. Каждая зона DNS имеет свой сервер имен, который для надежности дублируется. Один сервер назначается главным, остальные — подчиненными. Операции обновления проводятся только на главном сервере, а подчиненные периодически опрашивают главный для получения обновлений. Обновление обычно происходит с использованием TCP.

DNS содержит записи различных типов, позволяющие задать ip-адреса для имен, адрес используемого обработчика почты (используемого почтовыми серверами при пересылке почты), адрес ответственного сервера имен и т.п. Записи включают поле «время жизни» (TTL), определяющее время в секундах, в течение которого описатель ресурса должен храниться в кэше.

На клиенте обычно находится модуль анализатора (resolver), который посылает DNS-серверу запросы и кэширует ответы. Запросы анализатора обычно используют UDP и могут быть двух типов –рекурсивные и итеративные. В первом случае DNS-сервер, не обнаружив в своей базе запрашиваемого имени, обращается к следующему серверу, тот при необходимости к третьему и т.п. В случае итеративных запросов анализатор последовательно опрашивает все известные ему серверы.

DNS использует в работе 53-й порт. Адрес DNS-сервера также входит в информацию, которая может быть передана DHCP-сервером сетевому клиенту.

Протокол **SNMP (Simple Network Management Protocol)** разрабатывался для удаленного контроля и управления маршрутизаторами. С ростом популярности его стали применять и для управления любым коммуникационным оборудованием — концентраторами, мостами, сетевыми адаптерами и т.д. Он включает минимальный набор команд, позволяющий тем не менее выполнять практически весь спектр задач управления сетевыми устройствами — от получения информации о местонахождении конкретного устройства до возможности производить его тестирование.

Проблема управления в протоколе SNMP разделяется на две задачи.

Первая задача связана с контролируемыми переменными, характеризующими состояние управляемого устройства. Вся необходимая для управления устройством информация хранится на самом устройстве в так называемой базе данных MIB (Management Information Base). MIB может хранить количество пакетов, обработанных устройством, состояние его интерфейсов, время функционирования и т.п. Каждый производитель сетевого оборудования, помимо стандартных переменных, включает в MIB какие-либо параметры, специфичные для данного устройства. Каждый управляемый объект, информация о котором находится в MIB, имеет уникальный идентификатор (OID, object identifier). Все существующие OID объединены в единую древовидную структуру.

Протокол предоставляет только набор команд для работы с переменными MIB. Таким образом, чтобы проконтролировать работу некоторого сетевого устройства, достаточно получить доступ к его MIB и

проанализировать значения некоторых переменных. Кроме того, определены переменные MIB, играющие роль переключателей — изменение значения которых воспринимается устройством как указание выполнить некоторую команду.

Вторая задача связана с передачей информации. Протоколы передачи управляющей информации определяют процедуру взаимодействия SNMP-агента, работающего в управляемом оборудовании, и SNMP-монитора, работающего на компьютере администратора, который часто называют также консолью управления. Протоколы передачи определяют форматы сообщений, которыми обмениваются агенты и монитор. Агент принимает SNMP пакеты и выполняет команды — возвращает/изменяет значение переменной, выполняет периодическое обновление информации MIB, выполняет какие-то операции в ответ на установку соответствующих переменных. SNMP-агенты объединяются в сообщества, в рамках которых устанавливаются единые правила доступа. SNMP-запрос включает в себя идентификаторы сообщества, к которому относится запрашиваемый агент. Кроме того, каждому SNMP-запросу присваивается свой уникальный идентификатор, который задается монитором и включается агентом в ответ, что позволяет монитору связать между собой пару ответ/запрос.

При работе используются два порта для двустороннего общения: порт для обычных сообщений (по умолчанию 161) и порт для информирования о происходящих событиях сетевых объектов (smtp-trap, порт 162). Первая версия протокола определяет пять используемых команд:

- GetRequest — получить значение переменной либо информацию о состоянии элемента;
- GetNextRequest — получить информацию о следующем по очереди объекте MIB;

- SetRequest — присвоить значение переменной;
- GetResponce — отклик на предыдущие три команды;
- Trap — информация сетевого объекта о происходящем событии.

Во второй версии протокола добавились команды:

- InformRequest — позволяет послать сообщение типа Trap от одного монитора к другому и запросить ответ;
- GetBulkRequest — применяется для более эффективной работы с большими объемами данных;
- Report — команда, формат которой определяется конкретной реализацией;
- Trap2 и Response — синонимы Trap и GetResponce из первой версии.

10. Средства обеспечения сетевой безопасности

10.1. Классификация

Средства сетевой защиты можно разбить на три основные группы:

- Превентивные — так называемые системы анализа защищенности. Предназначены для поиска возможных уязвимостей, которые могут быть использованы при атаке на систему. Включают средства анализа конфигурационных файлов («проверка изнутри»), средства контроля стойкости паролей, (так называемые сканеры уязвимостей способные имитировать действия взломщика, воспроизводя сценарии атак, использующих наиболее распространенные уязвимости, и анализируя реакцию системы («проверка извне»). Наиболее известные представители — SATAN, Nessus, Internet Scanner. Необходимо понимать, что подобные автоматизированные системы поиска уязвимостей реализуют лишь первичную проверку системы, отнюдь

не гарантируя ее стопроцентную безопасность. Тем не менее, их применение целесообразно хотя бы потому, что именно они используются в первую очередь и при реальных взломах. Другими словами, если они не нашли проблем, то это еще не повод для самоуспокоения, если же нашли — будьте уверены, их обнаружат и реальные взломщики.

- Средства активного противодействия. Наиболее большая группа, включающая в себя межсетевые экраны различных уровней, системы обнаружения атак, обманные системы, средства контроля содержания. В дальнейшем этот класс средств защиты будет рассмотрен подробнее.
- Средства анализа и восстановления после совершенных атак. Системы протоколирования и аудита, а также контроля целостности. Позволяют эффективно отследить атаки, которым сопутствует какое-либо изменение в файловой системе, в том числе и неизвестные ранее. Особенно полезны при восстановительных работах. К алгоритмам формирования контрольной суммы предъявляются все требования, традиционные для кэшей. Системы контроля целостности работают по замкнутому циклу, обрабатывая файлы, системные объекты и атрибуты системных объектов с целью получения контрольных сумм; затем они сравнивают их с контрольными суммами, полученными на предыдущем цикле, отыскивая изменения. Когда изменение обнаружено, продукт посылает сообщение администратору безопасности, при этом фиксируя время, соответствующее времени вероятного изменения. Среди недостатков можно назвать малую пригодность для функционирования в реальном времени и возможное

влияние на производительность системы при наличии большого числа контролируемых объектов.

10.2. Межсетевые экраны

10.2.1. Задачи

Межсетевый экран (МЭ, файрволл, брандмауэр) в самом общем случае — программная или программно-аппаратная система, контролирующая потоки данных, проходящие между сетями.

МЭ могут функционировать на различных уровнях и решать следующие задачи:

- По определению, задачу разделения сетей. Традиционное размещение МЭ — между доверенной сетью и «враждебным окружением», в роли которого выступают внешние сети, например Internet. Хотя зачастую вполне оправданным становится и размещение МЭ между сегментами корпоративной сети.
- Защита от попыток сканирования, DoS-атак, подслушивания и т.п.
- Фильтрация IP-адресов и портов. Позволяет заблокировать пакеты, приходящие с/на определенных адресов и/или портов.
- Фильтрация содержимого. Реализуется обычно посредниками прикладного уровня, ориентированными на работу с несколькими основными протоколами прикладного уровня — почтовыми, http, nntp. Могут интегрироваться с антивирусными системами, использоваться в качестве средства борьбы со спамом, обнаружения нежелательной активности сотрудников и т.п.
- Перенаправление пакетов и трансляция адресов (NAT, Network Address Translation). Позволяет в целях экономии использовать в локальной сети блок фиктивных IP-адресов (например, вида

192.168.*.*) и осуществлять их трансляцию при взаимодействии с хостами из внешней сети, а также, например, перенаправлять пакеты, пришедшие на некоторый порт МЭ, на определенный порт одного из хостов локальной сети.

- Дополнительная аутентификация и шифрование.
- Протоколирование.

Существует две стратегии определения правил доступа, реализуемых МЭ:

- Разрешать все, что не запрещено (стратегия «черного списка»).
- Запрещать все, что не разрешено (стратегия «белого списка»).

Первая стратегия требует меньших усилий со стороны администратора, более снисходительна по отношению к пользователям, но и менее надежна. Вторая доставляет больше неудобств пользователям и администраторам, но и более надежна. Как правило, предпочтение отдается именно ей.

В зависимости от уровня, на котором они работают, и от набора реализуемых функций, выделяют следующие типа межсетевых экранов:

10.2.2. Пакетные фильтры

Фактически, это один из самых старых и известных типов МЭ. Фильтрация осуществляется на основе информации, заключенной в заголовках TP, TCP и UDP пакетов. При этом могут анализироваться адреса отправителя и получателя, порты, протокола, флаги и т.п. Фактически, МЭ такого типа являются интеллектуальными маршрутизаторами (и на практике просто реализованы в качестве составной части большинства маршрутизаторов).

Можно выделить следующие их недостатки:

- доступность внутренней сети извне;

- относительную легкость обмана за счет подделки IP-адресов в пакетах;
- сложность конфигурирования;
- отсутствие аутентификации на пользовательском уровне.

Достоинства:

- относительно невысокая стоимость;
- гибкость в определении правил фильтрации;
- высокая скорость работы.

10.2.3. Шлюзы сеансового уровня

Также могут называться системами трансляции сетевых адресов (Network Address Translation, NAT). Основная идея здесь заключается в блокировке непосредственного взаимодействия узлов. Шлюз сеансового уровня принимает запрос доверенного клиента на конкретные услуги, и после проверки допустимости запрошенного сеанса устанавливает соединение с внешним хостом. После установки соединения пакеты с одной стороны шлюза просто копируются на другую сторону, без дополнительных проверок. В процессе сеанса связи шлюз хранит во внутренней таблице сведения об адресах получателя и отправителя, состоянии соединения и т.п. После завершения передачи данных соединение разрывается и информация удаляется.

Помимо собственно фильтрации, такие шлюзы могут использоваться в качестве сервера-посредника, выполняющего процедуру трансляции адресов. При которой происходит преобразование внутренних IP-адресов локальный сети в единственный IP-адрес сеансового шлюза, который становится единственным IP-адресом, доступным извне. Поскольку соединение устанавливается только после первичной проверки, это также предотвращает атаки, связанные с подделкой IP-адресов.

Из недостатков данного подхода можно выделить отсутствие проверки информации прикладного уровня и аутентификации пользователей.

Достоинства:

- высокая скорость работы.
- изоляция взаимодействующих узлов;
- контроль состояния соединения.

10.2.4. Шлюзы прикладного уровня

Работают по тому же принципу, что и шлюзы сеансового уровня, но умеют также учитывать в работе информацию из пакетов прикладного уровня. Они включают в свой состав ряд отдельных посредников приложений (application proxy), которые знают о правилах функционирования протоколов прикладного уровня и могут вмешаться в их работу. Например, такой посредник, работающий с HTTP-протоколом, оказывается в состоянии отследить опасные GET-запросы, приходящие к web-серверу, и блокировать их при необходимости. При этом может осуществляться дополнительная аутентификация пользователей, протоколирование действий, контроль содержимого и т.п.

Проблема здесь лишь в том, что для каждого протокола прикладного уровня необходимо реализовать свой посредник, а в случае его отсутствия приложение, его использующее, будет либо полностью блокировано, либо будет работать без нашего контроля. Возможно как явное использование посредников прикладного уровня с помощью настроек в прикладных программах, так и прозрачное использование — например, все пакеты, направляющиеся на 80-й порт, могут быть принудительно направлены пакетным фильтром посреднику прикладного уровня. Недостаток

последнего подхода — при использовании нестандартных портов запросы могут избежать фильтрации.

Недостатки шлюзов прикладного уровня — их более высокая стоимость и низкое быстродействие.

Достоинства:

- невидимость внутренней структуры сети;
- надежная регистрация и аутентификация;
- простые правила фильтрации;
- изоляция взаимодействующих узлов;
- контроль состояния соединения.

10.2.5. Фильтры с проверкой состояния (stateful inspection), или инспекторы состояний

Их разрабатывали с оглядкой на вышеперечисленные типы МЭ, стремясь объединить их лучшие черты. Основная идея здесь — отслеживание сетевых сессий. Т.е., получив один раз пакет с флагом SYN, такой фильтр будет в состоянии отследить принадлежность последующих TCP-пакетов данной сессии. Более того, подобные фильтры в состоянии организовывать нечто вроде псевдосессий для таких протоколов как UDP. Возможна реализация данного подхода от сетевого до прикладного уровня. Подобные фильтры значительно лучше защищены от подделки пакетов — например, от передачи TCP-пакетов с узла, с которым не установлена сессия. Хорошим примером их применения является протокол FTP, работающих в разных направлениях по разным портам. Если МЭ обнаруживает входящее FTP-соединение с данными при отсутствии принадлежащего этой же сессии исходящего управляющего соединения, он просто его блокирует.

10.2.6. Схемы подключения МЭ

При подключении корпоративной сети к глобальным сетям решаются следующие задачи:

- защита сети от несанкционированного доступа извне;
- сокрытие информации о структуре внутренней сети и ее компонентов;
- разграничение доступа в защищаемую сеть извне и из внутренней сети во внешнюю.

Кроме того, часто возникает потребность иметь в составе корпоративной сети сегменты с несколькими уровнями защищенности:

- свободно доступные сегменты;
- сегмент с ограниченным доступом (например, для работы с филиалами);
- закрытые сегменты (например, локальная сеть для отделов, работающих с финансами).

Рассмотрим некоторые популярные схемы подключения МЭ.

1. Межсетевой экран — пакетный фильтр, расположен между защищаемой сетью и внешним миром, блокируя исходящие/входящие пакеты на основе информации из заголовков. Большая часть доступа извне блокируется, доступ изнутри в основном разрешен. Простейший и самый распространенный случай.
2. МЭ с пакетным фильтром и прикладным шлюзом. Первичная защита может осуществляется пакетным фильтром, например, открывающим некоторые соединения с внешним миром напрямую, и заставляющим некоторые из них проходить через посредники прикладного уровня. Здесь возможны различные комбинации:
   - Полное закрытие сети. Вся связь с внешним миром осуществляется только через прикладные посредники.

- Частичное закрытие сети. Основная часть сети полностью закрыта пакетным фильтром и прикладным шлюзом. За пределами прикладного шлюза размещен так называемый бастионный хост, к которому по какой-либо причине должен быть обеспечен доступ извне. Им может быть, например информационный web-сервер. Запросы к нему идут напрямую, в остальную сеть — через прикладной шлюз.
- Создание так называемой демилитаризованной зоны (ДМЗ). Информационных серверы размещены за МЭ, который отделяет их как от внешнего мира, так и от внутренней сети. Возможна реализация демилитаризованной зоны как с одним МЭ, так и в двумя (один — отделяющий ДМЗ от внешнего мира, другой — от внутренней сети). Введение ДМЗ позволяет отсечь внутреннюю защищенную сеть от проблем, связанных с возможным взломом внешнего менее защищенного сегмента, к которому должен быть обеспечен контролируемый доступ извне.

10.3. Другие средства активного противодействия

10.3.1. Системы обнаружения атак

Этот класс средств защиты стал набирать популярность в последнее время. Обычно здесь выделяют системы обнаружения атак на уровне узла (host based) и на уровне сети (network based).

Как следует из названия, системы первого типа устанавливаются на защищаемых узлах, после чего приступают к анализу поступающих запросов. Могут функционировать как на уровне операционной системы, так и на уровне отдельных приложений. Для функционирования в реальном времени, как правило, требуется их встраивание в защищаемые

приложения — с помощью модификации исходного кода, с помощью перехвата системных функций, встраивания в виде локального прокси-сервера и т.п. Также возможны варианты реализации, использующие в своей работе анализ протоколируемых данных, что, естественно, приводит к задержке их реакции на атаку.

Достоинства:

- подтверждение факта атаки;
- контроль действия конкретного узла;
- обнаружение атак, не обнаруживаемых другими средствами;
- работа в коммутируемых сетях и в сетях с канальным шифрованием;
- высокая скорость реакции
- низкая цена отдельной системы

Недостатки:

- чувствительность к уязвимостям ОС
- игнорирование атак сетевого уровня и уровня приложений
- возможное требование дополнительных ресурсов
- зависимость от конкретной платформу
- высокая стоимость эксплуатации и управления

Системы, работающие на уровне сети, могут устанавливаться на один из узлов, входящих в сеть, после чего анализируют проходящие по сети потоки информации, пытаясь выявить в них сигнатуры известных атак. Могут рассматривать как весь сетевой трафик в случае широковещательной среды передачи данных (например, Ethernet), так и трафик, проходящий через сетевой шлюз или маршрутизатор. Наиболее известные представители — RealSecure, Snort.

Достоинства:

- обнаружение сетевых атак, в том числе на инфраструктуру
- невозможность «заметания следов»
- обнаружение и реагирование в реальном масштабе времени
- низкая стоимость эксплуатации
- обнаружение неудавшихся атак
- независимость от ОС и приложений

  Недостатки:
- неэффективная работа в коммутируемых сетях
- невозможность работы в сетях с канальным и прикладным шифрованием данных
- зависимость от сетевых протоколов
- игнорирование атак уровня ОС и приложений
- трудности функционирования на высоких скоростях

10.3.2. Системы контроля содержимого

Могут входить как составной элемент в МЭ, но могут быть реализованы и как частный случай систем обнаружения атак, работающих на уровне сети. Основное назначение — отслеживание нежелательной активности пользователелей, борьба с утечкой информации.

10.3.3. Обманные системы

Еще один класс защитных программ, набирающих популярность в последнее время. Основные механизмы использования обмана в защитных целях

- Сокрытие (межсетевые экраны)
- Камуфляж («ручной», обманные системы)
- Дезинформация (обманные системы)

Позволяют решить две задачи: во-первых, маскировки используемой операционной системы и/или сервисов, что вынуждает взломщика тратить силы на взлом «не той» системы. В качестве простых примеров можно привести выкладывание на веб-сайт фиктивного файла /etc/passwd, модификация стандартных приглашений сетевых сервисов и т.п. Во-вторых, существует ряд систем, позволяющий имитировать работу на данном хосте сетевых служб с известными уязвимостями (или даже имитировать работу целых сетевых сегментов). Основная задача подобных систем — заманить взломщика, занять его работой по взлому несуществующей уязвимости, отведя его усилия от реально работающих служб, а также давая возможность «вычислить» атакующего. Типичный представитель — The Deception Toolkit.

11. Построение защищенных виртуальных сетей

11.1. Общие сведения

Безопасность информационного взаимодействия локальных сетей и отдельных компьютеров через открытые сети требует качественного решения двух базовых задач:

- защиты подключенных к открытым каналам связи локальных сетей и отдельных компьютеров от несанкционированных действий со стороны внешней среды;
- защиты информации в процессе передачи по открытым каналам связи.

Решение первой задачи основано на использовании межсетевых экранов и других средств защиты, рассмотренных в предыдущей главе.

Защита информации в процессе передачи по открытым каналам связи основана на выполнении следующих функций:

- аутентификации взаимодействующих сторон;

- криптографическом закрытии передаваемых данных;
- подтверждении подлинности и целостности доставленной информации;
- защите от повтора, задержки и удаления сообщений;
- защите от отрицания фактов отправления и приема сообщений.

Перечисленные функции во многом связаны друг с другом, и их реализация основана на криптографической защите передаваемых данных. Высокая эффективность такой защиты обеспечивается за счет совместного использования симметричных и асимметричных криптографических систем.

Объединение локальных сетей и отдельных компьютеров через открытую внешнюю среду передачи информации в единую виртуальную сеть, обеспечивающую безопасность циркулирующих данных, называют ***защищенной виртуальной сетью*** (Virtual Private Network — VPN). Виртуальная сеть формируется на основе каналов связи открытой сети. Сам термин "виртуальная" подчеркивает, что каналы связи виртуальной сети моделируются с помощью каналов связи реальной сети.

Защита информации в процессе ее передачи по открытым каналам основана на построении защищенных виртуальных каналов связи, называемых криптозащищенными туннелями и туннелями VPN. Каждый такой туннель представляет собой соединение, проведенное через открытую сеть, по которому передаются криптозащищенные пакеты сообщений виртуальной сети.

Создание защищенного туннеля выполняют компоненты виртуальной сети, функционирующие на узлах, между которыми формируется туннель. Эти компоненты принято называть инициатором и терминатором туннеля. Инициатор туннеля инкапсулирует (встраивает) пакеты в новый пакет,

содержащий наряду с исходными данными новый заголовок с информацией об отправителе и получателе. Хотя все передаваемые по туннелю пакеты являются пакетами IP, инкапсулируемые пакеты могут принадлежать к протоколу любого типа, включая пакеты немаршрутизируемых протоколов, таких как NetBEUI. Маршрут между инициатором и терминатором туннеля определяет обычная маршрутизируемая сеть IP, которая может быть сетью отличной от Internet. Терминатор туннеля выполняет процесс обратный инкапсуляции — он удаляет новые заголовки и направляет каждый исходный пакет в локальный стек протоколов или адресату в локальной сети.

Сама по себе инкапсуляция никак не влияет на защищенность пакетов сообщений, передаваемых по туннелю VPN. Но благодаря инкапсуляции появляется возможность полной криптографической защиты инкапсулируемых пакетов. Конфиденциальность инкапсулируемых пакетов обеспечивается путем их **криптографического закрытия**, а целостность и подлинность — путем **формирования цифровой подписи.** Поскольку существует большое множество методов криптозащиты данных, очень важно, чтобы инициатор и терминатор туннеля использовали одни и те же методы и могли согласовывать друг с другом эту информацию. Кроме того, для возможности расшифровывания данных и проверки цифровой подписи при приеме инициатор и терминатор туннеля должны поддерживать функции **безопасного обмена ключами.** Чтобы туннели VPN создавались только между уполномоченными пользователями, конечные стороны взаимодействия требуется **аутентифицировать.**

Возможны следующие варианты построения VPN:

- Объединение с помощью открытых каналов (например, через Internet) нескольких сегментов корпоративной сети.

- Установка защищенного соединения между двумя узлами корпоративной сети — создание «сети внутри сети».
- Подключение к сети внешних пользователей или сетей, доверие к которым меньше чем к своим сотрудникам.
- Подключение к защищенному сегменту сети удаленных или мобильных пользователей. Основное отличие от предыдущих случаев заключается в отсутствии у такого пользователя своего адреса, совместимого с адресацией, принятой в данной сети.

11.2. Обзор протоколов

Технически реализация защищенных виртуальных сетей стала возможной уже достаточно давно. Инкапсуляция использовалась раньше и применяется сейчас для передачи немаршрутизируемого трафика через маршрутизируемые сети, а также для ограничения многопротокольного трафика одним протоколом. Технологии шифрования также появились задолго до широкого внедрения глобальных сетевых технологий. Однако общепринятые протоколы для создания защищенных виртуальных сетей разработаны недавно и сейчас продолжается работа над их совершенствованием и расширением. Они являются открытыми, т. е. свободными для распространения и реализации.

Для независимости от прикладных протоколов и приложений защищенные виртуальные сети формируются на одном из более низких уровней модели OSI — канальном, сетевом или сеансовом. Канальному (второму) уровню соответствуют такие протоколы реализации VPN, как PPTP, L2F и L2TP, сетевому (третьему) уровню — IPSec, SKIP, а сеансовому (пятому) уровню — SSL/TLS и SOCKS. Чем ниже уровень эталонной модели, на котором реализуется защита, тем она прозрачнее для приложений и незаметнее для пользователей. Однако при снижении этого

уровня уменьшается набор реализуемых услуг безопасности и становится сложнее организация управления. Чем выше защитный уровень в соответствии с моделью OSI, тем шире набор услуг безопасности, надежнее контроль доступа и проще конфигурирование системы зашиты. Тем не менее в этом случае усиливается зависимость от используемых протоколов обмена и приложений. В виртуальной сети криптозащита может одновременно выполняться на нескольких уровнях эталонной модели. При этом увеличивается криптостойкость, но по причине снижения общей скорости криптографических преобразований уменьшается пропускная способность виртуальной сети. Поэтому на практике защищенные виртуальные сети формируются на одном уровне модели OSI (канальном, сетевом, транспортном или сеансовом).

11.2.1. Канальный уровень модели OSI

Одним из самых популярных протоколов канального уровня, используемых в VPN-сетях, является протокол PPTP (Point-to-Point Tunneling Protocol), разработанный совместно Microsoft, 3Com, Ascend и другими компаниями. Он реализован как расширение протокола PPP (Point-to-Point Protocol), играющего на последовательных линиях связи практически такую же роль, как Ethernet в локальных сетях. Протокол может инкапсулировать пакеты таких протоколов как IP, IPX или NetBEUI, для обмена информацией по обслуживанию соединения используется TCP, а для переноса данных — PPP. Он может применяться и в тех случаях, когда его поддерживают только взаимодействующие клиент и конечный сервер.

Канальному уровню модели OSI соответствует также протокол туннелирования L2F (Layer-2 Forwarding), разработанный компанией Cisco Systems при поддержке компаний Shiva и Northern Telecom. В данном протоколе также не специфицируются конкретные методы аутентификации

и шифрования. В отличие от протокола PPTP протокол L2F позволяет использовать для удаленного доступа к провайдеру Internet не только протокол PPP, но и другие протоколы, например SLIP. Для переноса данных через защищенный туннель могут использоваться различные протоколы сетевого уровня, а не только IP, как в протоколе PPTP. Протокол L2F стал компонентом операционной системы IOS (Internetwork Operating System) компании Cisco и поддерживается во всех выпускаемых ею устройствах межсетевого взаимодействия и удаленного доступа. Существенным моментом является обязательная поддержка этого протокола всеми маршрутизаторами и серверами, через которые проходит VPN-соединение.

Протоколы PPTP и L2F были представлены в организацию Internet Engineering Task Force (IETF) и в 1996 году соответствующие комитеты решили их объединить. Получившийся в результате протокол, включивший все лучшее из PPTP и L2F, был назван протоколом туннелирования второго уровня (Layer-2 Tunneling Protocol — L2TP). Его поддерживают компании Cisco, Microsoft, 3Com, Ascend и многие другие производители. Как и предшествующие протоколы канального уровня, спецификация L2TP не описывает методы аутентификации и шифрования. Протокол L2TP является расширением PPP на канальном уровне и может поддерживать любые высокоуровневые протоколы.

Протоколы формирования защищенного туннеля на канальном уровне независимы от протоколов сетевого уровня модели OSI, по которым функционируют локальные сети, входящие в состав виртуальных сетей. Они позволяют создавать защищенные каналы для обмена данными между удаленными компьютерами и локальными сетями, функционирующими по различным протоколам сетевого уровня — IP, IPX или NetBEUI. Пакеты этих протоколов криптографически защищаются и инкапсулируются в IP-

пакеты сети Internet, которые и переносятся к месту назначения, образуя защищенные виртуальные каналы. Многопротокольность — основное преимущество инкапсулирующих протоколов канального уровня.

Вместе с тем формирование криптозащищенных туннелей между объединяемыми локальными сетями на основе протоколов канального уровня приводит к сложности конфигурирования и поддержки виртуальных каналов связи. Туннели на основе PPP требуют, чтобы конечные точки поддерживали информацию о состоянии каждого канала (например такую, как тайм-ауты), и, следовательно, не обладают хорошей масштабируемостью при необходимости иметь несколько туннелей с общими конечными точками. Кроме того, протоколы формирования защищенных туннелей на канальном уровне не специфицируют конкретные методы шифрования, аутентификации, проверки целостности каждого передаваемого пакета, а также средств управления ключами.

Исходя из вышесказанного, можно сделать вывод, что протоколы создания защищенных виртуальных каналов на канальном уровне лучше всего подходят для защиты информационного взаимодействия при удаленном доступе к локальной сети.

11.2.2. Сетевой уровень модели OSI

Спецификацией, где описаны стандартные методы для всех компонентов и функций защищенных виртуальных сетей, является протокол Internet Protocol Security (IPSec), соответствующий сетевому уровню модели OSI и входящий в состав новой версии протокола IP — IPv6. Протокол IPSec иногда еще называют протоколом туннелирования третьего уровня (Layer-3 Tunneling). IPSec предусматривает стандартные методы аутентификации пользователей или компьютеров при инициации туннеля, стандартные способы шифрования конечными точками туннеля,

формирования и проверки цифровой подписи, а также стандартные методы обмена и управления криптографическими ключами между конечными точками. Этот гибкий стандарт предлагает несколько способов для выполнения каждой задачи. Выбранные методы для одной задачи обычно не зависят от методов реализации других задач. Для функций аутентификации IPSec поддерживает цифровые сертификаты популярного стандарта X.509.

Туннель IPSec между двумя локальными сетями может поддерживать множество индивидуальных каналов передачи данных, в результате чего приложения данного типа получают преимущества с точки зрения масштабирования по сравнению с технологией второго уровня. Протокол IPSec может использоваться вместе с протоколом L2TP. Совместно эти два протокола обеспечивают наиболее высокий уровень гибкости при защите виртуальных каналов. Дело в том, что спецификация IPSec ориентирована на протокол IP и, таким образом, бесполезна для трафика любых других протоколов сетевого уровня. Протокол L2TP отличается независимостью от транспортного уровня, что может быть полезно в гетерогенных сетях, состоящих из IP, IPX и AppleTalk-сегментов.

К IP-данным, готовым к передаче по VPN, IPSec добавляет заголовок для идентификации защищенных пакетов. Перед передачей по Internet эти пакеты инкапсулируются в другие IP-пакеты.

Аутентифицирующий заголовок (Authentication header, AH), как правило, располагается между основным заголовком пакета IP и полем данных. Он не имеет отношение к шифрованию информации, его назначение — обеспечение целостности пакета. Протоколы более высокого уровня должны быть модифицированы в целях осуществления проверки аутентичности полученных данных.

Формат AH достаточно прост и состоит из 96-битового заголовка и данных переменной длины, состоящих из 32-битовых слов. Основные поля:

- Next Header — указывает на следующий заголовок;
- Payload Len — представляет длину пакета;
- SPI — указатель на контекст безопасности;
- Sequence Number Field — последовательный номер пакета. Значение этого поля формируется отправителем и служит для защиты от атак, связанных с повторным использованием данных процесса аутентификации. Получатель должен хранить информацию о максимальном последовательном номере пакета, прошедшего успешную аутентификацию, и о получении некоторого числа пакетов, содержащих предыдущие последовательные номера.

В процессе формирования AH последовательно вычисляется хэш-функция от объединения самого пакета и некоторого предварительно согласованного ключа, а затем от объединения полученного результата и преобразованного ключа.

Вторая составляющая IPSec — протокол ESP (Encapsulating Security Protocol), отвечающие за обеспечение конфиденциальности данных. Формат его заголовка может меняться в зависимости от используемых алгоритмов шифрования. Тем не менее, можно выделить следующие обязательные поля:

- SPI, указывает на контекст безопасности;
- Sequence Number Field, содержит последовательный номер пакета
- контрольная сумма, предназначенная для защиты от атак на целостность зашифрованных данных.

Кроме этого, как правило, в теле ESP присутствуют параметры и данные применяемого алгоритма шифрования. Часть ESP заголовка может

быть зашифрована с помощью открытого ключа получателя, или с помощью секретного сессионного ключа. Получатель пакета ESP расшифровывает заголовок и использует параметры и данные применяемого алгоритма шифрования для декодирования информации транспортного уровня.

ESP может использоваться в двух режимах — транспортном и туннельном.

Транспортный режим используется для шифрования поля данных IP-пакета, не затрагивая информацию из заголовков. Все промежуточные узлы на маршруте пакета от отправителя к получателю используют только открытую информацию сетевого уровня. Недостатком транспортного режима является отсутствие механизмов скрытия конкретных отправителя и получателя пакета, а также возможность проведения анализа трафика.

Туннельный режим предполагает шифрование всего пакета, включая заголовок сетевого уровня. При этом адресные поля заголовка сетевого уровня пакета, использующего туннельный режим, заполняются инициатором на стороне отправителя и не содержат информации о конкретном отправителе пакета. После расшифровки терминатором начального заголовка сетевого уровня пакет направляется получателю.

Третье ключевое понятие IPSec — Security Association (SA). SA связывается с каждым соединением, которое один клиент устанавливает с другим через IPSec. SA сохраняет:

- информацию об используемом алгоритме шифрования;
- сессионный ключ;
- информацию об алгоритме хэширования.

IPSec поддерживает две схемы управления ключами, с помощью которых участники могут согласовать параметры сеанса: Internet Secure

Association Key Management Protocol (ISAKMP), и Simple Key Management for Internet Protocol (SKIP).

Протокол IPsec в настоящее время стал практически стандартным средством реализации VPN. В настоящее время на рынке представлены как программные, так и программно-аппаратные решения (он, например, встроен в Windows'2000 и Windows XP, используется в решениях Cisco, Nokia и т.п.). Несмотря на большое число реализаций, они достаточно хорошо совместимы друг с другом.

11.2.3. Сеансовый уровень модели OSI

Защищенные виртуальные каналы могут быть сформированы и на сеансовом уровне модели OSI. Для этого применяются так называемые "посредники каналов" (circuit proxy). Посредник функционирует над транспортным уровнем, шифрует и ретранслирует трафик из защищенной сети в общедоступную сеть Internet для каждого сокета в отдельности. При приеме выполняется обратная процедура. Сокет IP идентифицируется комбинацией TCP-соединения и конкретного порта или заданным портом UDP.

Для шифрования информации на сеансовом уровне наибольшую популярность получил протокол SSL/TLS (Secure Sockets Layer/ Transport Layer Security), разработанный компанией Netscape Communications. Этот протокол создает защищенный туннель между конечными точками виртуальной сети, обеспечивая взаимную аутентификацию абонентов, а также конфиденциальность, подлинность и целостность циркулирующих по туннелю данных. Ядром протокола SSL/TLS является технология комплексного использования асимметричных и симметричных криптосистем компании RSA Data Security. Для аутентификации взаимодействующих сторон и криптозащиты ключа симметричного

шифрования используются цифровые сертификаты открытых ключей пользователей (клиента и сервера), заверенные цифровыми подписями специальных Сертификационных Центров. Поддерживаются цифровые сертификаты, соответствующие общепринятому стандарту X.509.

С целью стандартизации процедуры взаимодействия клиент-серверных приложений TCP/IP через сервер-посредник (брандмауэр) комитет IETF утвердил протокол под названием SOCKS, и в настоящее время пятая версия данного протокола (SOCKS 5) применяется для стандартизованной реализации посредников каналов. SOCKS поддерживает приложения, требующие контроля над направлениями информационных потоков и настройки условий доступа в зависимости от атрибутов пользователя и/или информации.

В соответствии с SOCKS 5 клиентский компьютер устанавливает аутентифицированный сеанс с сервером, исполняющим роль посредника (proxy). Использование этого посредника является единственным способом связи через брандмауэр. Посредник, в свою очередь, проводит любые операции, запрашиваемые клиентом. Поскольку посреднику известно о трафике на уровне сеанса (сокета), он может осуществлять тщательный контроль, например блокировать конкретные приложения пользователей, если они не имеют необходимых полномочий.

В отличие от виртуальных сетей, защищенных на сеансовом уровне модели OSI, виртуальные сети, защищенные на канальном или сетевом уровне, обычно просто открывают или закрывают канал для всего трафика по аутентифицированному туннелю. Это может представлять проблему, если локальная сеть на другом конце туннеля является неблагонадежной. Кроме того, созданные туннели канального и сетевого уровня функционируют одинаково в обоих направлениях, а виртуальные сети,

защищенные на сеансовом уровне, допускают независимое управление передачей в каждом направлении.

Виртуальные сети с посредником канала типа IPSec ориентированы на протокол IP. Если IPSec, по существу, разграничивает защищенные виртуальные каналы между разными парами взаимодействующих сторон, то протокол SOCKS 5 обеспечивает создание защищенных туннелей для каждого приложения и сеанса в отдельности. Аналогично протоколу IPSec и протоколам туннелирования канального уровня, виртуальные сети сеансового уровня можно использовать с другими типами виртуальных частных сетей, поскольку данные технологии не являются взаимоисключающими.

Следует отметить, что имеются протоколы для реализации защищенного взаимодействия и на прикладном уровне модели OSI. Эти протоколы, как правило, являются дополнениями к различным протоколам прикладного уровня. Например, протокол Secure HTTP (SHTTP) является дополнением по функциям защиты к протоколу передачи гипертекста HTTP, а протокол Secure MIME (S/MIME) — дополнением по защитным функциям к протоколу электронной почты MIME. Прикладные протоколы защиты информационного взаимодействия не относят к протоколам построения защищенных виртуальных сетей, так как они полностью зависят от используемых сервисов и приложений. Протоколы формирования защищенных виртуальных каналов прозрачны для прикладных протоколов защиты. Соответственно применение приложений, реализующих, например, SHTTP или S/MIME, наряду с криптографической защитой на более низком уровне, нисколько не уменьшает, а только увеличивает уровень безопасности.

11.3. Распределение ключей и согласование параметров туннелей

При построении защищенных виртуальных сетей одними из важнейших функций являются функции распределения криптографических ключей и согласования параметров защищенных туннелей. Данные функции выполняются при формировании каждого криптозащищенного канала.

В защищенных виртуальных сетях по длительности использования различают следующие типы криптографических ключей:

- основные ключи, которые применяются в течение относительно долгого периода времени, например недели, месяца или нескольких месяцев;
- временные ключи, каждый из которых генерируется для криптозащиты информации в рамках одного защищенного канала.

Основные ключи закрепляются за пользователями и серверами компьютерной сети и обеспечивают аутентификацию сторон, а также криптозащиту распределяемых временных ключей. После аутентификации сторон и безопасного распределения временных ключей, а также согласования параметров защищенного туннеля криптозащита трафика в рамках этого туннеля осуществляется на основе распределенных временных ключей.

Основные ключи шифрования, применяемые в течение относительно долгого периода времени, должны распределяться заблаговременно до непосредственного формирования защищенных виртуальных соединений. При этом способ распределения этих ключей зависит от вида криптосистемы, которой они соответствуют.

При формировании противоположными сторонами сеансового ключа, если вычисление этого ключа осуществляется не на основе ранее

распределенных, а на основе передаваемых друг другу открытых ключей, то либо эти ключи должны быть заверены центром сертификации, либо перед процедурой непосредственного формирования временного ключа необходимо выполнить аутентификацию сторон. Для повышения безопасности защищенного туннеля временный ключ целесообразно генерировать не только на основе открытых ключей, но и в зависимости от согласованного случайного числа. Одним из наиболее популярных алгоритмов формирования сеансового ключа на основе распределенных или передаваемых друг другу открытых ключей является алгоритм Диффи-Хеллмана.

11.3.1. Формирование ключей в протоколах канального уровня

В протоколах формирования защищенных туннелей на канальном уровне модели OSI (PPTP, L2F, L2TP) временные ключи чаще всего генерируются на основе паролей пользователей. Генерация выполняется каждой стороной информационного взаимодействия после взаимной аутентификации.

В протоколе PAP (Password Authentication Protocol) используется простой двухэтапный метод подтверждения личности объектов. Инициатор соединения посылает запрос серверу (аутентификатору), включая в него свой идентификатор и пароль. Сервер проверяет их и отправляет обратно ответ, где сообщает, удостоверена личность, или нет. Основной недостаток этого протокола — незашифрованная передача идентификатора и пароля.

Протокол CHAP (Challenge Handshake Authentication Protocol) более надежен. Он разбивается на три этапа. На первом этапе клиент получает от аутентификатора запрос, в ответ на который отправляет ответ, сформированный с помощью однонаправленной хэш-функции на основе

идентификатора пользователя, его пароля и содержимого запроса. Далее сервер вычисляет предполагаемый ответ и сравнивает с ответом клиента.

Аналогично работает протокол MS-CHAP, основное отличие которого от CHAP заключается в том, что сервер не использует идентификатор и пароль клиента в открытом виде даже при вычислении предполагаемого ответа.

Учитывая, что пароли являются аналогами основных ключей симметричного шифрования, более эффективным способом является распределение временных ключей на канальном уровне является их централизованное распределение, например на основе протокола Kerberos.

11.3.2. Управление ключей в IPSec

Какуже упоминалось, IPSec поддерживает две схемы управления ключами, с помощью которых участники могут согласовать параметры сеанса: Internet Secure Association Key Management Protocol (ISAKMP), и Simple Key Management for Internet Protocol (SKIP). SKIP проще в реализации, но он не поддерживает переговоров по поводу алгоритмов шифрования. Если получатель, использующий SKIP, не в состоянии расшифровать пакет, он уже не сможет согласовать метод шифрования с противоположной стороной. ISAKMP поддерживает такие переговоры и выбран в качестве обязательного протокола для управления ключами в IPSec для IPv6.

В процессе переговоров абоненты договариваются о механизмах, используемых для создания в дальнейшем защищенного соединения. Согласовываются:

- Алгоритм шифрования (DES либо 3DES);
- Алгоритм хэширования (MD5 или SHA1);

- Способ аутентификации (заранее распределенные секретные ключи, публичные ключи, ключи, распределенные с помощью протокола Kerberos);
- Общий единственный ключ или стартовое значение для генерации сессионных ключей, а также частота обновления ключей — с помощью протокола определения ключей Oakley, использующего в свою очередь алгоритм замены ключа Диффи-Хеллмана.

11.3.3. Протокол SKIP

Протокол SKIP (Simple Key management for Internet Protocol — простой протокол управления ключами для Internet) разработан компанией Sun Microsystems в 1994 году и предложен в качестве стандарта Internet. Протокол SKIP, являясь IP-совместимым протоколом, обеспечивает не только управление ключами шифрования, но и прозрачную для приложений криптозащиту IP-пакетов на сетевом уровне модели OSI. Реализация SKIP, устанавливаемая непосредственно над IP-драйвером, обрабатывает весь трафик, не накладывая никаких ограничений ни на вышележащее программное обеспечение, ни на физические каналы, используемые для передачи информации.

SKIP обеспечивает два режима работы:

- шифрование данных IP- пакета с сохранением в открытом виде исходного заголовка пакета;
- инкапсуляция (туннелирование) исходного IP-пакета в SKIP-пакет с заменой исходного заголовка.

В протоколе SKIP используется система открытых ключей Диффи-Хеллмана. Алгоритм открытого распределения ключей Диффи-Хеллмана был предложен в 1976г. Его безопасность обсусловлена трудностью

вычисления дискретных логарифмов в конечном поле, в отличие от легкости дискретного возведения в степень в том же конечном поле.

Предположим, что два пользователя хотят организовать защищенный коммуникационный канал.

1. Обе стороны договариваются о модуле N (N — простое число) и примитивном элементе **g (1<= g <= N-1),** который образует все ненулевые элементы множества $\mathbf{Z_N\{g,g^2, \ldots\ldots, g^{n-1} = 1\} = Z_N\text{-}\{0\}}$. Эти два числа **N, g** могут не храниться в секрете. Как правило, эти значения являются общими для всех пользователей системы**.**
2. Затем пользователи А и В независимо друг от друга выбирают собственные секретные ключи Ка и Кв, которые являются случайными большими числами и хранятся пользователями А и В в секрете.
3. Далее пользователь А вычисляет открытый ключ $\mathbf{Уа=g^{Ka} (mod\ N),}$ а пользователь В свой открытый ключ $\mathbf{Ув=g^{Кв} (mod\ N).}$

Затем стороны А и В обмениваются вычисленными значениями открытых ключей **Уа** и **Ув** по незащищенному каналу.

Далее пользователи А и В вычисляют общий секретный ключ, используя следующие сравнения:

Пользователь А $K= (Yb)^{Ka} = (g^{Kb})^{Ka} (\text{mod } N)$;

Пользователь В $K'= (Ya)^{Kb} = (g^{Ka})^{Kb} (\text{mod } N)$.

При этом К=К’, т.к. $(g^{Kb})^{Ka} = (g^{Ka})^{Kb} (\text{mod } N)$.

Ключ **К** может использоваться в качестве общего секретного ключа (ключа шифрования ключей) в симметричной критпосистеме.

Кроме того, обе стороны А и В могут шифровать сообщения, используя следующее преобразование шифрования (типа RSA) $\mathbf{C = E_K (M) = M^K (mod\ N).}$

Для выполнения расшифрования получатель сначала находит ключ расшифрования **К"** с помощью сравнения **К* К" ≡ 1 (mod N-1),** а затем восстанавливает сообщение $\mathbf{M = D_K (C) = C^{K"} (mod\ N).}$

Злоумышленник, перехватив значения **N,g,Ya,Yb** хотел бы определить значение ключа **К.** Путь для решения этой задачи состоит в вычислении такого значения **Ка** по **N,g,Ya,** что $\mathbf{g^{Ka} (mod\ N) = Уа}$, поскольку в этом случае, вычислив **Ка**, можно найти $\mathbf{К=(Уb)^{Ka}\ (mod\ N).}$ Однако нахождение **Ка** по **N,g,Ya** — задача нахождения дискретного логарифма в конечном поле, которая считается неразрешимой.

Пример: Допустим, модуль N = 47, примитивный элемент g=23. Предположим, что пользователи А и В выбрали свои секретные ключи Ка=12(mod 47) и Кв=33(mod 47).

Вычисляем значения открытых ключей

$Уа^{Кв} = g^{Ka} = 23^{12} = 27\ (mod\ 47)$

$Ув^{Ka} = g^{Кв} = 23^{33} = 33\ (mod\ 47)$

После того, как обменялись открытыми ключами, вычисляет общий секретный ключ:

$К= (Уb)^{Ka} = (Уа^{Kb}) =33^{12} = 27^{33} = 23^{12*33} = 25\ (mod\ 47)$

Вычисляем секретный ключ расшифрования, используя следующее сравнение

К* К" ≡ 1 (mod N-1) , откуда К"=35(mod 46).

Если исходное сообщение М=16, то зашифрованное сообщение

$С=М^{К} =16^{25} =21\ (mod\ 47)$

Расшифровать можно так $М=С^{К"'} =21^{35} =16\ (mod\ 47)$.

В технологии SKIP каждый узел сети снабжается секретным и открытым ключами. Узел А, адресующий свои сообщения узлу В,

вырабатывает общую секретную часть Кав, так называемый "секрет". Однако этот, требующий высокой степени защиты общий секрет, не используется напрямую для шифрования данных. Для шифрования конкретного пакета или их небольшой партии узел А вырабатывает специальный пакетный ключ Кп, шифрует этим ключом данные, укладывает их блок данных SKIP-пакета. Далее, пакетный ключ шифруется на основе ключа, вырабатываемого из общей секретной части, и тоже записывается в передаваемый пакет. Пакет снабжается SKIP-заголовком, по синтаксису совпадающим с заголовком IP-пакета и отправляется в сеть.

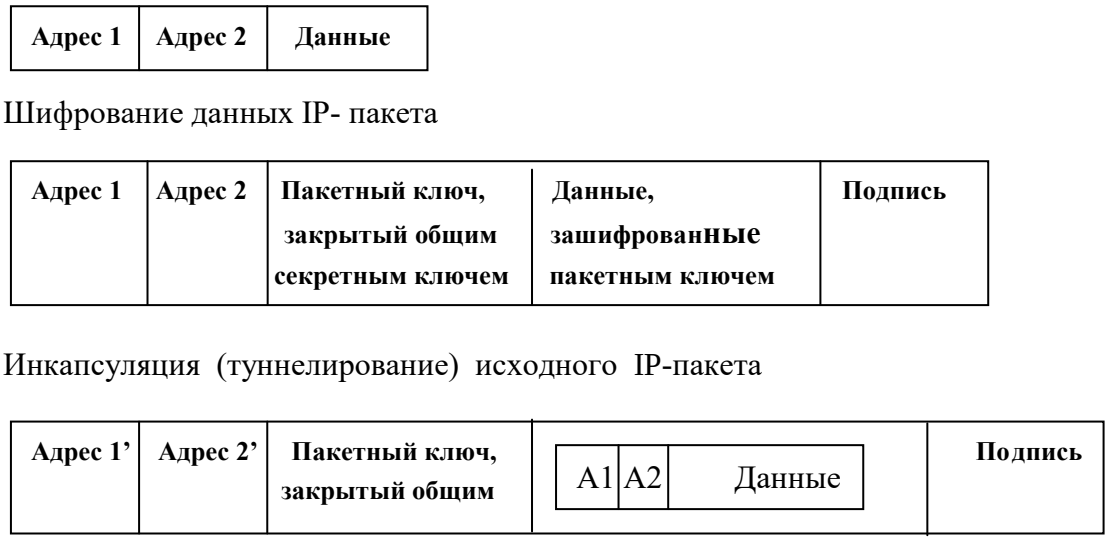


Рис. 11.1. Схемы работы SKIP

Узел В, получив пакет и вычислив разделяемую секретную часть Kab, дешифрует пакетный ключ Кп и с его помощью дешифрует весь пакет. Протокол SKIP независим от системы шифрования, т.е. различные системы шифрования могут подключаться как внешние динамические модули. Например, для шифрования данных можно использовать алгоритмы симметричного шифрования DES, RC2, RC4.

Пакет может быть подписан цифровой подписью отправителя (по алгоритму RSA), что обеспечивает контроль целостности и аутентичности. За счет того, что истинные адреса инкапсулируются в SKIP-пакете, это дает возможность скрыть топологию сегментов корпоративной сети.

11.4. Аутентификация удаленных пользователей

11.4.1. Общие сведения

Доступ удаленных пользователей к ресурсам локальной сети должен контролироваться в соответствии с политикой безопасности, проводимой в организации, которой принадлежит локальная сеть. Надежность разграничения доступа к компьютерным ресурсам может быть обеспечена только в случае надежной аутентификации пользователей. По отношению к удаленным пользователям требования по надежности проверки их подлинности существенно возрастают. Это связано с тем, что удаленным пользователям, в отличие от пользователей локальных, для доступа к ресурсам локальной сети не нужно проходить процедуру физического контроля полномочий по допуску на территорию организации. При работе с "невидимыми" удаленными пользователями становится значительно труднее гарантировать, что доступ к ресурсам локальной сети смогут получить только лица, имеющие на это соответствующие полномочия.

В случае удаленного доступа к локальной сети для надежной проверки подлинности взаимодействующих сторон должны поддерживаться следующие функциональные возможности:

- согласование используемых протоколов аутентификации и отсутствие жесткой привязки к конкретным протоколам проверки подлинности;

- блокирование любых попыток обхода фазы аутентификации после установки удаленного соединения;
- аутентификация каждой из взаимодействующих сторон — как удаленного пользователя, так и сервера удаленного доступа, что исключает возможность маскировки под одного из участников взаимодействия;
- проведение не только начальной аутентификации перед допуском к ресурсам локальной сети, но и динамической аутентификации взаимодействующих сторон в процессе работы удаленного соединения; данная функция устраняет риск перехвата соединения и маскировки под одного из участников взаимодействия после окончания начальной аутентификации;
- использование одноразовых паролей либо криптозащита передаваемых секретных паролей, исключающая возможность повторного использования перехваченной информации для подложной аутентификации.

Сравнивая существующие протоколы удаленного доступа к локальным сетям, можно сделать вывод, что выше перечисленные возможности по аутентификации могут быть реализованы только на основе протокола PPP.

Протокол PPP предусматривает использование в процессе установления соединения уже рассмотренных выше протоколов PAP или CHAP.

Учитывая, что в протоколе PAP пароли передаются по линии связи в открытом виде, данный протокол необходимо применять только совместно с протоколом, ориентированным на аутентификацию по одноразовым

паролям. Наиболее распространенным протоколом аутентификации по одноразовым паролям является протокол S/Key.

11.4.2. Протокол S/Key

Одним из наиболее популярных протоколов аутентификации на основе одноразовых паролей является стандартизованный в Internet протокол S/Key (RFC 1760). Перехват одноразового пароля, передаваемого по сети в процессе аутентификации, не предоставляет возможности воспользоваться этим паролем повторно, так как при следующей проверке подлинности будет использоваться уже другой пароль. Поэтому схема аутентификации на основе одноразовых паролей, в частности S/Key, позволяет передавать по сети одноразовый пароль в открытом виде и, таким образом, компенсирует основной недостаток протокола аутентификации PAP.

Протокол S/Key не исключает необходимость задания для каждого пользователя секретного пароля. Однако данный пароль используется только для генерации одноразовых паролей. Для того, чтобы злоумышленник не смог по перехваченному одноразовому паролю вычислить секретный исходный пароль, генерация одноразовых паролей выполняется с помощью односторонней, или, как ее еще называют, необратимой функции. В качестве такой односторонней функции в спецификации S/Key определен алгоритм хэширования MD4 (Message Digest Algorithm 4). Некоторые реализации S/Key в качестве односторонней функции используют криптографический алгоритм хэширования MD5 (Message Digest Algorithm 5). Для восстановления за приемлемое время значения аргумента по значению функции, задаваемой криптографическим алгоритмом хэширования, требуются практически недоступные вычислительные ресурсы.

Основная идея протокола S/Key состоит в следующем. Проверяемой стороне заранее назначается генерируемый случайный ключ ***K,*** выступающий в качестве ее секретного постоянного пароля. Далее проверяющей стороной выполняется процедура инициализации списка одноразовых **M** паролей. В процессе данной процедуры с помощью односторонней функции ***F*** вычисляется по ключу ***K*** проверочное значение Y_{M+1} для первого одноразового пароля. Для вычисления этого значения ключ ***K*** подставляется в качестве аргумента функции ***F*** и данная функция рекурсивно выполняется **M+1** раз: $Y_{M+1}=F(F(F(...(F(K))...)))=F^{M+1}(K)$. Идентификатор пользователя и соответствующий этому пользователю секретный ключ ***K,*** а также несекретные числа **M** и Y_{M+1} сохраняются в базе данных проверяющей стороны. Число **M** считается номером одноразового пароля для очередной аутентификации из списка одноразовых паролей.

В процессе очередной после инициализации аутентификации проверяемая сторона предоставляет проверяющей стороне свой идентификатор, а та возвращает соответствующее идентификатору число **M**. Далее проверяемая сторона вычисляет по своему секретному ключу ***K*** одноразовый пароль

$Y'_M=F(F(F(...(F(K'))...)))=F^M(K')$

и посылает его проверяющей стороне. Получив это значение, проверяющая сторона выполняет над ним один раз одностороннюю функцию **F**: $Y'_{M+1}=F(Y'_M)$. Далее полученное значение Y'_{M+1} сравнивается со значением Y_{M+1} из базы данных. Если они совпадают, то это значит, что и $Y_M = Y_M$ и, следовательно, аутентификация является успешной. В случае успешной аутентификации проверяющая сторона заменяет в базе данных для проверяемой стороны число Y'_{M+1} на полученное от нее число Y'_M, а число **M** на **M=M-1.** С учетом того, что в случае успешного опознания

номер одноразового пароля **M** для очередной аутентификации уменьшился на единицу, в базе данных проверяющей стороны совместно с идентификатором и секретным ключом ***K*** проверяемой стороны будут храниться числа **M** и $\mathbf{Y_{M+1}}$. Здесь под $\mathbf{Y_{M+1}}$ понимается полученный от проверяемой стороны при успешной аутентификации последний одноразовый пароль. После использования списка одноразовых паролей процедура инициализации должна выполняться снова.

11.4.3. Централизованный контроль удаленного доступа

Даже при наличии в локальной сети одного сервера удаленного доступа, целесообразно применять централизованную систему аутентификации. Поддержание отдельной базы данных с учетными записями на сервере удаленного доступа приводит к избыточности функций администрирования и может стать причиной несогласованности в правилах контроля доступа к ресурсам сети. Эффективность администрирования и надежность защиты увеличивается, если сервер удаленного доступа запрашивает необходимую для аутентификации информацию непосредственно у сервера, на котором хранится общая база данных системы защиты компьютерной сети, например у контроллера домена Windows NT или сервера intranetWare.

Однако в большинстве случаев серверы удаленного доступа требуют посредника для взаимодействия с центральной базой данных системы защиты, например со службой каталогов. Дело в том, что стандартами удаленной аутентификации, поддерживаемыми серверами удаленного доступа, являются протоколы CHAP и PAP, которые без дополнений не подходят для аутентификации с использованием дерева Novell Directory Services (NDS) или доменной службы Windows NT. Для проверки ответа на вызов сервера, поступившего от проходящего аутентификацию

пользователя, реализация протокола CHAP применяет незашифрованную копию пароля в простой текстовой форме. При аутентификации на основе PAP пароль также используется в открытом виде. Большинство же сетевых операционных систем и служб каталогов сохраняют эталонные пароли пользователей с односторонним хэш-кодированием, что не позволяет серверам удаленного доступа, стандартно реализующим протоколы PAP и CHAP, извлечь открытый эталонный пароль для проверки ответа.

Серверы удаленного доступа, позволяющие взаимодействовать с общей базой данных системы защиты, существуют. Но они, как правило, ориентированы на один тип такой базы и часто требуют наличия этой базы непосредственно на сервере удаленного доступа. Например, служба RAS (Remote Access Service) ОС Windows NT позволяет взаимодействовать с реестром контроллера домена Windows NT, но лишь до тех пор, пока сервер RAS и контроллер домена исполняются на одном и том же сервере.

Роль посредника во взаимодействии между серверами удаленного доступа и центральной базой данных системы защиты может быть возложена на сервер аутентификации. Централизованный контроль удаленного доступа к компьютерным ресурсам с помощью сервера аутентификации выполняют на основе специализированных протоколов. Данные протоколы позволяют объединять используемые серверы удаленного доступа и сервер аутентификации в одну подсистему, выполняющую все функции контроля.

Система Kerberos обеспечивает защиту сети от несанкционированного доступа. Она имеет структуру типа клиент/сервер и состоит из Kerberos-сервера (или серверов), располагающегося на каком-либо (не обязательно выделенном) компьютере и клиентских частей, установленных на все машины сети (рабочие станции пользователей и

серверы). Kerberos-сервер содержит базу с именами и паролями всех пользователей. Кроме того, все целевые серверы в сети обладают общим кодовым ключом с Kerberos.

Kerberos-сервер делится на две равноправные части: сервер идентификации и сервер выдачи разрешений. Модель работы Kerberos описывается следующим образом.

Клиент от своего имени направляет запрос идентификационному серверу на выдачу ему "разрешение на получение разрешения", которое даст возможность обратиться к серверу выдачи разрешений. Идентификационный сервер обращается к базе данных, хранящей информацию о всех пользователях, и определяет пароль пользователя. Затем клиенту отсылается "разрешение на получение разрешения" и специальный код сеанса, которые шифруются с помощью пароля пользователя в качестве ключа.

При получении этой информации пользователь на своей рабочей станции должен ввести свой пароль, и если он совпадает с паролем, хранящимся в базе Kerberos-сервера, то "разрешение на получение разрешения" и код сеанса будут успешно расшифрованы. Таким образом, решается проблема с защитой пароля, т.к. он не передается по сети.

Затем пользователь в зашифрованном виде отправляет запрос серверу выдачи разрешений на получение доступа к требуемым ресурсам сети.

Сервер выдачи разрешений расшифровывает полученное от клиента "разрешение", проверяет, не истек ли срок его действия, после чего сервер выдачи разрешений отсылает пользователю разрешение на доступ к ресурсам сети. Клиент обращается с запросами к целевому серверу, используя полученное разрешение, которое может использоваться многократно в течение некоторого периода времени.

Для обеспечения более высокого уровня защиты, клиент, в свою очередь может потребовать идентификации целевого сервера. Теперь клиент и сервер готовы к передаче информации с должной степенью защиты.

Литература

1. Зегжда Д.П., Ивашко А.М. Основы безопасности информационных систем. Горячая Линия — Телеком, 2000.
2. Зима В., Молдовян А., Молдовян Н. Безопасность глобальных сетевых технологий. С-П.: BHV. 2003.
3. Леонов Д.Г., Лукацкий А.В., Медведовский И.Д., Семьянов П.В. Атака из Internet. М.: Солон, 2002.
4. Лукацкий А.В. Обнаружение атак. СПб: «БХВ-Петербург», 2003.
5. Молдовян А.А., Молдовян Н.А., Советов Б.Я. Криптография. Лань, 2005.
6. Снытиков А.А. Лицензирование и сертификация в области защиты информации. Гелиос АРВ, 2003.
7. Чмора А. Современная прикладная криптография. М.: Гелиос АРВ,2001.

Кузнецова Лариса Валентиновна

Леонов Дмитрий Геннадьевич

МЕТОДЫ И СРЕДСТВА ЗАЩИТЫ ИНФОРМАЦИИ

Подписано в печать . Формат 60x90/16. Усл. п. л. 11.

Гарнитура Таймс. Бумага офсетная. Печать офсетная

Тираж 100 экз. Заказ №

Издательский центр

РГУ нефти и газа имени И.М. Губкина

119991, Москва, Ленинский проспект, 65

Тел./факс: (499) 233-95-44