# Часть VIII

# Безопасность компьютерных сетей

- Глава 26. Основные понятия и принципы информационной безопасности
- Глава 27. Технологии аутентификации, авторизации и управления доступом
- Глава 28. Технологии безопасности на основе анализа трафика
- Глава 29. Атаки на транспортную инфраструктуру сети
- Глава 30. Безопасность программного кода и сетевых служб

Эта часть книги открывается главой, в которой рассматриваются *фундаментальные* понятия и технологии, лежащие в основе любой системы информационной защиты: конфиденциальность, доступность, целостность, уязвимость, угроза, атака, ущерб, аутентификация, авторизация, контроль доступа. Эти понятия поясняются примерами некоторых распространенных пассивных и активных атак на транспортную систему и программное обеспечение сети: отказ в обслуживании, ответвление трафика, спуфинг, шпионы, кража личности. В качестве важнейшей концепции представлен системный подход к обеспечению безопасности — привлечение средств самой различной природы: законодательства, административных мер, процедур управления персоналом, средств физической защиты и программно-технического оборудования. Такой подход включает также следование универсальным принципам построения системы защиты: непрерывности и разумной достаточности, эшелонированного характера и сбалансированности. Завершает главу обзор криптографических методов, являющихся краеугольным камнем практически всех методов информационной безопасности.

Глава 27 посвящена *конкретным* технологиям аутентификации, авторизации и контроля доступа. Рассматриваются процедуры, основанные на одноразовых и многоразовых паролях, цифровых сертификатах и цифровой подписи, описываются мандатный, дискреционный и ролевой способы управления доступом. Приводятся примеры как локальных, так и централизованных систем аутентификации и авторизации, в частности, подробно обсуждается система Kerberos.

В следующей главе показано, как различные способы фильтрации могут быть использованы для экранирования сегментов сети от вредительского трафика. Рассматриваются системы мониторинга трафика на основе анализаторов протоколов и агентов протокола NetFlow, которые позволяют распознать атаку путем выявления отклонений образцов трафика от стандартного поведения. Изучаются различные типы файерволов: без запоминания состояния, с запоминанием состояния, с трансляцией адресов (NAT), с функцией прокси-сервера. Рассматриваются особенности анализа трафика и событий системами обнаружения вторжения (IDS). Дается типовая схема разбиения крупной корпоративной сети на зоны безопасности.

В главе 29 изучаются уязвимости и методы защиты транспортной инфраструктуры сети. Особое внимание уделяется безопасности службы DNS и протоколу маршрутизации BGP — двум очень важным элементам архитектуры Интернета, обеспечивающим связность составляющих сетей и узлов в глобальном масштабе. В главе также подробно рассматривается технология защищенного канала IPSec.

В заключительной главе изучаются типичные уязвимости программного обеспечения компьютерной сети, а также приводятся рекомендации по их устранению. Освещены различные методы обнаружения и обезвреживания вредоносного кода: троянских программ, червей, вирусов и программных закладок. Значительная часть этой главы посвящена вопросам безопасности сетевых служб — веб-службы, почты, облачных вычислений.

# ГЛАВА 26 Основные понятия и принципы информационной безопасности

## Идентификация, аутентификация и авторизация

Для пояснения таких базовых понятий информационной безопасности, как «идентификация», «аутентификация» и «авторизация», представим информационную систему (далее также — ИС) в виде упрощенной модели контролируемого доступа, когда несколько пользователей совместно работают с ресурсами информационной системы. Родившаяся полвека назад и направленная на повышение эффективности применения компьютера концепция разделения ресурсов выдвинула проблемы безопасности вычислительных систем на первый план — необходимо было *контролировать доступ* пользователей к компьютеру, защищая системные и пользовательские данные от ошибочных или злонамеренных действий. Контролируемый доступ является важным направлением обеспечения безопасности наряду с другими средствами безопасности: криптографической защитой, аудитом, сегментацией сети и др.

В модели контролируемого доступа определены объекты, субъекты, операции и система контроля доступа (рис. 26.1).

**Объекты** представляют физические и логические информационные ресурсы ИС. К физическим ресурсам относятся как отдельные устройства целиком (процессор, внешние устройства, маршрутизаторы, коммутаторы, физические каналы связи и др.), так и физические разделяемые ресурсы устройств (разделы и секторы диска, процессорное время, физические соединения канала связи). Логическими ресурсами являются файлы, вычислительные процессы, сетевые сервисы, приложения, пропускная способность каналов связи и т. п.

**Субъекты** представляют сущности, между которыми разделяются информационные ресурсы. Это могут быть легальные пользователи ИС: персонал, поддерживающий работу ИС, внешние и внутренние клиенты; группы легальных пользователей, объединенных по различным признакам. Пользователь осуществляет доступ к объектам ИС не непосредственно, а с помощью прикладных процессов, которые запускаются от его имени. Поэтому в качестве субъектов выступают также прикладные вычислительные процессы. Иногда оказывается полезным представление в качестве субъектов и системных вычислительных процессов.

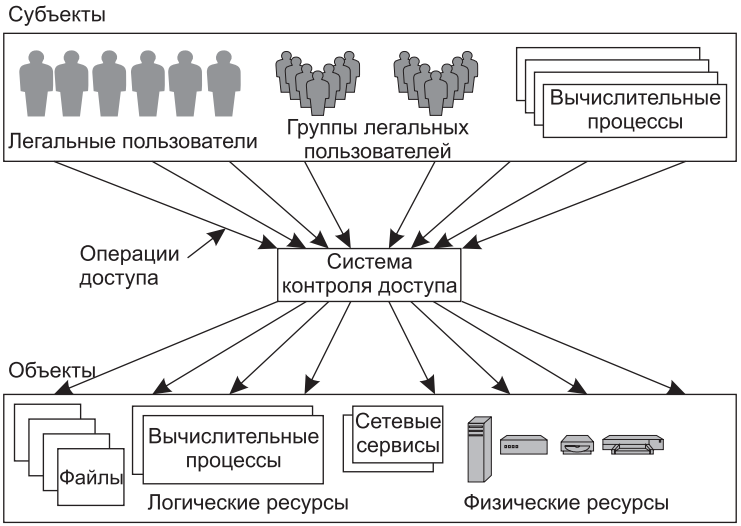



**Рис. 26.1.** Модель контролируемого доступа

**Операции** выполняются субъектами над объектами. Для каждого типа объектов существует собственный набор операций, которые с ними может выполнять субъект. Например, для файлов это операции чтения, записи, удаления, выполнения; для принтера — печать, перезапуск, очистка очереди документов, приостановка печати документа; для маршрутизатора — конфигурирование и т. д.

**Система контроля доступа** решает, какие операции разрешены для данного субъекта по отношению к данному объекту. Для автоматизированного контроля доступа необходимо, чтобы для каждой пары субъект-объект были однозначно определены *правила доступа*, на основании которых система могла бы принимать решение о разрешении или запрете выполнения каждой из предусмотренных для данного объекта операций.

Важнейшими элементами управляемого доступа являются процедуры идентификации, аутентификации и авторизации.

> **Идентификация** — это присвоение объектам и субъектам ИС уникальных имен — **идентификаторов**.

Только при наличии уникальных идентификаторов система получает возможность распознавать и оперировать субъектами и объектами. Одни идентификаторы автоматически генерируются операционной системой (ОС) и приложениями (идентификаторы процессов, идентификаторы логических сетевых соединений), другие назначаются администратором компьютерной сети (идентификаторы пользователей, адреса компьютеров, доменные имена сетевых сервисов), третьи порождаются обычными сетевыми пользователями, обладающими таким правом (выбор собственного имени, назначение имен файлам).

**Идентификация пользователей** представляет собой процедуру, выполняемую при логическом входе в систему, когда пользователь в ответ на выведенное на экране приглашение

печатает свой идентификатор (имя), а система, сверяясь со своими данными, определяет, входит ли данное имя в число имен зарегистрированных (легальных) пользователей.

Пользователь может быть представлен в системе в виде нескольких субъектов и, соответственно, иметь несколько пользовательских идентификаторов. Например, один идентификатор он может применять во время сетевой регистрации, а другой — для работы с корпоративной базой данных.

**Аутентификация**[1] — это процедура доказательства субъектом/объектом того, что он есть то, за что (кого) себя выдает.

Аутентификация, или, другими словами, процедура установления подлинности, может применяться как к пользователям, так и другим объектам и субъектам, в частности, к данным, программам, приложениям, устройствам, документам (рис. 26.2).

**Рис. 26.2.** «В Интернете никто не узнает, что ты собака, если успешно пройдешь аутентификацию» (рисунок Питера Штайнера)

**Аутентификация данных** означает доказательство их подлинности, то есть того, что они поступили в неизмененном виде и именно от того человека, который объявил об этом.

В процедуре аутентификации участвуют две стороны:

- *аутентифицируемый* доказывает свою аутентичность, предъявляя некоторое доказательство — *аутентификатор*;
- *аутентифицирующий* проверяет эти доказательства и принимает решение.

Аутентификация бывает односторонней и двусторонней (взаимной).

Так, мы имеем дело с *односторонней аутентификацией*, в частности, при выполнении логического входа в защищенную систему. После того как пользователь сообщает системе свой идентификатор, он должен пройти процедуру аутентификации, то есть доказать, что именно ему принадлежит введенный им идентификатор (имя пользователя). Аутентифи-

[1] Термин «аутентификация» (authentication) происходит от латинского слова authenticus, которое означает «подлинный», «достоверный», «соответствующий самому себе».

кация предотвращает доступ к сети нежелательных лиц и разрешает вход для легальных пользователей.

В качестве аутентификатора аутентифицируемый может продемонстрировать знание некоего общего для обеих сторон секрета, например слова (пароля), или обладание неким уникальным предметом (физического ключа) либо предъявить собственные биохарактеристики (допустим, отпечатки пальцев).

В некоторых случаях односторонней аутентификации оказывается недостаточно, и тогда используют *двустороннюю аутентификацию*. Например, пользователь, обращающийся с запросом к корпоративному веб-серверу, должен доказать ему свою легальность, но он также должен убедиться сам, что ведет диалог действительно с веб-сервером своего предприятия. Другими словами, сервер и клиент должны пройти процедуру взаимной аутентификации. Здесь мы имеем дело с двусторонней *аутентификацией на уровне приложений*. При установлении сеанса связи между двумя устройствами также часто предусматриваются процедуры взаимной *аутентификации устройств* на более низком, канальном, уровне.

**Авторизация**[1] — это процедура контроля доступа субъектов (пользователей, вычислительных процессов, устройств) к объектам (например, файлам, приложениям, сервисам, устройствам) и предоставления каждому из них именно тех прав, которые для них определены правилами доступа.

В отличие от аутентификации, которая позволяет распознать легальных и нелегальных пользователей, авторизация касается только *легальных* пользователей, успешно прошедших процедуру аутентификации.

Доступ к объектам, полученный в обход разрешений системы контроля доступа, называется *несанкционированным* или *неавторизованным*.

# Модели информационной безопасности

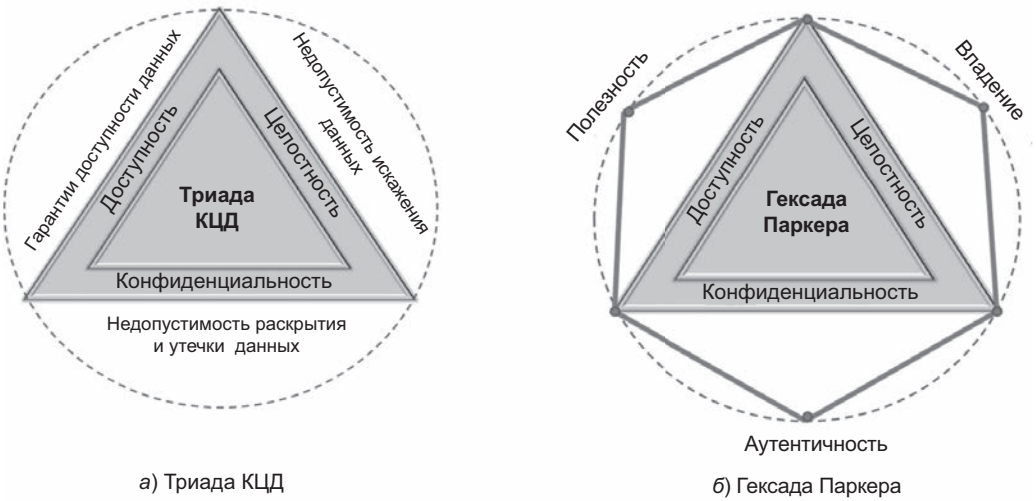
Понятие информационной безопасности может быть пояснено с помощью так называемых **моделей безопасности**, суть которых в следующем: множество всех видов нарушений безопасности делится на несколько базовых групп таким образом, чтобы любое возможное нарушение обязательно можно было отнести по крайней мере к одной из этих групп. Затем система объявляется безопасной, если она способна противостоять каждой из этих групп нарушений.

## Триада «конфиденциальность, доступность, целостность»

Одной из первых и наиболее популярных по сей день моделей безопасности является модель, предложенная Зальцером (Saltzer) и Шредером (Schroeder). Авторы постулировали, что все возможные нарушения информационной безопасности всегда могут быть отнесены

[1] Термин «авторизация» (authorization) происходит от латинского слова auctoritas, показывающего уровень престижа человека в Древнем Риме и соответствующие этому уровню привилегии.

по меньшей мере к одной из трех групп: нарушения конфиденциальности, нарушения целостности или нарушения доступности (рис. 26.3, *а*).




*а*) Триада КЦД

*б*) Гексада Паркера

**Рис. 26.3.** Модели безопасности

Соответственно, ИС находится в *состоянии безопасности*, если она защищена от нарушений конфиденциальности, целостности и доступности, где:

- **конфиденциальность** (confidentiality) — состояние ИС, при котором информационные ресурсы доступны только тем пользователям, которым этот доступ разрешен;
- **целостность** (integrity) — состояние системы, при котором информация, хранящаяся и обрабатываемая этой ИС, а также процедуры обработки информации не могут быть изменены, удалены или дополнены неавторизованным образом;
- **доступность** (availability) — состояние системы, при котором услуги, оказываемые системой, могут гарантированно и с приемлемой задержкой быть предоставлены пользователям, имеющим на это право.

Для ссылки на триаду иногда используют аббревиатуру КЦД (*к*онфиденциальность, *ц*елостность, *д*оступность), в англоязычной форме — CIA.

Требования к безопасности могут меняться в зависимости от назначения ИС, характера используемых данных и типа возможных угроз. Трудно представить систему, для которой нарушения целостности и доступности не представляли бы опасности. Вместе с тем обеспечение конфиденциальности не всегда является обязательным.

Например, если вы публикуете информацию в Интернете на веб-сервере и вашей целью является сделать ее доступной для самого широкого круга людей, то конфиденциальность не требуется. Однако требования целостности и доступности остаются актуальными. Действительно, если вы не предпримете специальных мер по обеспечению целостности системы, то злоумышленник может изменить данные на вашем сервере и нанести этим ущерб вашему предприятию. Преступник может, например, внести изменения в помещенный на веб-сервере прайс-лист, что негативно отразится на конкурентоспособности вашего пред-

приятия, или испортить коды свободно распространяемого вашей фирмой программного продукта, что, безусловно, скажется на ее деловой репутации. Если бы модифицированные данные были к тому же секретными, то в таком случае имело бы место не только нарушение их целостности, но и их конфиденциальности.

Не менее важным в данном примере является и обеспечение доступности данных. Затратив немалые средства на создание и поддержание сервера в Интернете, предприятие вправе рассчитывать на отдачу: увеличение числа клиентов, количества продаж и т. д. Однако существует вероятность того, что злоумышленник предпримет атаку, в результате которой помещенные на сервер данные станут недоступными для тех, кому они предназначались. Примером таких злонамеренных действий может служить «бомбардировка» сервера пакетами, каждый из которых в соответствии с логикой работы соответствующего протокола вызывает тайм-аут сервера, что, в конечном счете, делает его недоступным для всех остальных запросов.

Некоторые виды нарушений безопасности могут быть приведены к модели КДЦ только путем расширенного толкования основополагающих понятий конфиденциальности, доступности и целостности. В частности, свойство конфиденциальности по отношению, например, к устройству печати можно интерпретировать так, что доступ к устройству имеют те и только те пользователи, которым этот доступ административно разрешен, причем они могут выполнять только те операции с устройством, которые для них определены. Свойство доступности устройства означает его готовность к работе всякий раз, когда в этом возникает необходимость. В свою очередь, свойство целостности может быть интерпретировано как свойство неизменности параметров данного устройства.

## Гексада Паркера

За почти полвека, прошедших с момента публикации статьи Зальцера и Шредера, информационные системы и среда, в которой они функционируют, претерпели революционные изменения, поэтому неудивительно, что появились новые типы нарушений, которые намного труднее (если вообще возможно) трактовать в терминах КДЦ. Рассмотрим, например, ситуацию, когда легальный клиент банка посылает по электронной почте запрос на снятие со счета крупной суммы, а затем заявляет, что этот запрос, который хотя и был послан от его имени, он не отправлял. Является ли это нарушением безопасности? Да. Были ли при этом нарушены конфиденциальность, доступность или целостность? Нет. Следовательно, список свойств безопасной системы следует расширить, добавив к КЦД еще одно свойство — неотказуемость. **Неотказуемость** (non-repudiation) — это такое состояние системы, при котором обеспечивается невозможность отрицания пользователем, выполнившим какие-либо действия, факта их выполнения, в частности, отрицания отправителем информации факта ее отправления и/или отрицания получателем информации факта ее получения.

Одной из наиболее популярных альтернатив триаде КЦД является так называемая **гексада Паркера** (Parkerian Hexad), в которой определено шесть базовых видов нарушений, в число которых, помимо нарушений конфиденциальности, доступности и целостности, входят еще три вида нарушений: аутентичности, владения и полезности (рис. 26.3, *б*).

**Аутентичность** (authenticity) — это состояние системы, при котором пользователь не может выдать себя за другого, а документ всегда имеет достоверную информацию о его

источнике (авторе). Из этого определения видно, что аутентичность является аналогом *неотказуемости*.

**Владение** (possession) — это состояние системы, при котором физический контроль над устройством или другой средой хранения информации предоставляется только тем, кто имеет на это право.

**Полезность** (utility) — это такое состояние ИС, при котором обеспечивается удобство практического использования как собственно информации, так и связанных с ее обработкой и поддержкой процедур. В безопасной системе меры, предпринимаемые для защиты системы, не должны неприемлемо усложнять работу сотрудников, иначе последние будут воспринимать меры защиты как помеху и пытаться при всякой возможности их обойти.

Российский государственный стандарт[1] дает определение информационной безопасности на основе гексады Паркера:

> **Информационная безопасность** — все аспекты, связанные с определением, достижением и поддержанием конфиденциальности, целостности, доступности, неотказуемости, подотчетности, аутентичности и достоверности информации или средств ее обработки.

# Уязвимость, угроза, атака

**Уязвимость** (vulnerability) — это слабое звено ИС, которое, став известным злоумышленнику, может позволить ему нарушить ее безопасность. Уязвимостями являются, например, ошибка в программе, примитивный пароль, неправильное назначение прав доступа к файлу с важными данными и множество других дефектов в разработке, эксплуатации или настройке системы.

Уязвимости системы могут быть скрытыми, то есть еще не обнаруженными, известными, но только теоретически, или же общеизвестными и активно используемыми злоумышленниками. Для общеизвестных уязвимостей в программных продуктах производители регулярно выпускают исправления, называемые **патчами** (patch — «заплатка»). Так, компания Microsoft даже назначила специальный день — каждый второй вторник каждого месяца, когда она объявляет о новых исправлениях в семействе ОС Windows. Многие из этих исправлений направлены на устранение уязвимостей. Однако к этой рутинной процедуре — регулярному внесению исправлений — не все и не всегда относятся с должным вниманием, из-за чего общеизвестные, но неисправленные ошибки в программном обеспечении являются одним из самых распространенных типов уязвимостей.

Другим типом уязвимостей, которым часто пользуются злоумышленники, являются ошибки в конфигурировании программных и аппаратных средств. Например, имена «администратор» и «гость», установленные по умолчанию во многих ОС, могут облегчить злоумышленникам доступ к системе, поэтому уже при начальном конфигурировании ОС они должны быть заменены другими, менее очевидными именами. С этой же целью администратор должен настроить подсистему интерактивного входа на то, чтобы она не

---

[1] ГОСТ 13335-1:2006 «Информационная технология. Методы и средства обеспечения безопасности. Часть 1. Концепция и модели менеджмента безопасности информационных и телекоммуникационных технологий».

показывала последнего имени пользователя, систему аудита (то есть настроить ее таким образом, чтобы подсистема фиксировала все успешные и неуспешные попытки входа пользователей), а также выполнить другие, столь же простые, но необходимые настройки.

Поиск уязвимостей — важная часть задачи обеспечения безопасности. Эта работа включает в себя регулярное тестирование системы. В любой момент времени для любой системы можно указать множество различных видов уязвимостей. Например, для ОС и приложений новые уязвимости появляются чуть ли не каждый день. Выявлять их вручную — задача очень трудоемкая, поэтому для автоматизации поиска уязвимостей используют различные программные инструменты — **средства сканирования уязвимостей**, такие, например, как McAfee, Nessus и др. Сканирование заключается в последовательном (адрес за адресом узла, или номер за номером порта, или идентификатор за идентификатором сетевого соединения) направлении запросов целевой системе. Затем на основании полученных ответов генерируется «информационный отпечаток», и, наконец, сравнением «отпечатка» с записями в базе данных выполняется идентификация уязвимости.

Другими базовыми понятиями информационной безопасности являются угроза и атака.

**Угроза** (threat) — набор обстоятельств и действий, которые *потенциально* могут привести к нарушению безопасности системы (то есть к нарушению ее конфиденциальности, целостности и доступности, если пользоваться моделью КЦД).

**Атака** (attack) — это реализованная угроза.

Мы в основном ограничимся рассмотрением только *технических* угроз, то есть угроз, исходящих из искусственно созданного человеком мира техники и технологий (в частности, из Интернета), не принимая во внимание угрозы, возникающие от природных катаклизмов, военных действий, террористических атак или экономических потрясений.

Атака может произойти только тогда, когда одновременно существуют уязвимость и направленная на использование этой уязвимости угроза (рис. 26.4).

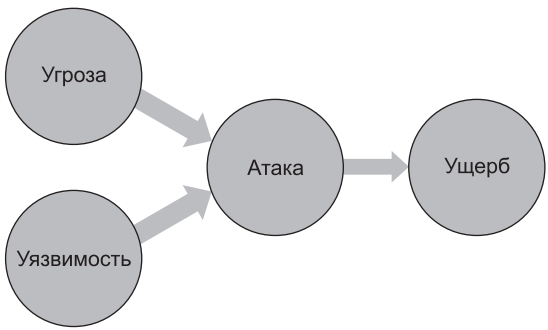



**Рис. 26.4.** Логическая связь между понятиями «уязвимость», «угроза», «атака», «ущерб»

То есть вполне возможна ситуация, когда система имеет некую уязвимость, но эта уязвимость еще не стала известной злоумышленникам. В подобном случае соответствующая угроза отсутствует, а значит, и атака не может быть проведена. Аналогично существование общеизвестной угрозы не влечет никакой опасности для системы, в которой нет соот-

ветствующей уязвимости. Например, появление информации о некоторой ошибке в коде ОС Windows может породить угрозу, но атака не осуществится, если эта уязвимость будет быстро устранена.

Таким образом, любая угроза направлена на поиск и/или использование уязвимостей системы. В некоторых случаях злоумышленник работает «на ощупь», пытаясь обнаружить тот или иной дефект системы. Система реагирует на такого рода угрозы выдачей сообщений о мелких, но странных неполадках, а также флуктуациями в статистических характеристиках работы системы, на основании которых администратор сети или специалист по безопасности может заподозрить подготовку атаки.

Другие угрозы выражаются в четкой последовательности действий и имеют формализованное воплощение в виде **эксплойта** (exploit) — программы или просто последовательности командных строк, некоторой порции данных и/или пошаговом описании действий, которые, будучи выполненными, позволяют злоумышленнику воспользоваться некоторой конкретной уязвимостью ИС в своих интересах. Особая опасность эксплойта состоит в том, что, имея его в своем распоряжении, даже малоподготовленный хакер способен провести успешную атаку. Для этого ему достаточно зайти на один из многочисленных сайтов, снабжающих всех желающих своей «продукцией». Более того, в придачу к инструкциям и программам в Интернете можно найти даже предложения о сдаче в аренду целых **ботсетей**, готовых к реализации мощных кибер-атак. В то же время наличие у эксплойтов фиксированных признаков, таких, например, как специфические кодовые последовательности, облегчают распознавание и отражение соответствующих атак.

Угрозы могут исходить как от легальных пользователей сети, так и от внешних злоумышленников. Примерно 2/3 от общего числа всех наиболее серьезных инцидентов, связанных с безопасностью, составляют нарушения или ошибки легальных пользователей сетей: сотрудников и клиентов предприятий, студентов, имеющих доступ к сети учебного заведения, и др.

Угрозы со стороны легальных пользователей могут быть как умышленными, так и неумышленными. К *умышленным* угрозам относятся, например, доступ и похищение конфиденциальных данных, мониторинг системы с целью получения информации об ее устройстве, посещение запрещенных веб-сайтов, вынос за пределы предприятия съемных носителей и т. п.

Безопасность может быть нарушена и в результате *непреднамеренных* нарушений пользователей и обслуживающего персонала — ошибок, приводящих к повреждению сетевых устройств, данных, программного обеспечения, ОС и приложений, беспечность в сохранении секретности паролей и др. Известно, что правильное конфигурирование устройств является одним из мощных средств обеспечения безопасности. Но, будучи выполненной с ошибками, эта операция способна обернуться своей противоположностью — угрозой. Как выяснилось, некоторые «атаки» на ИС были на самом деле не атаками, а ошибками администраторов сетей при выполнении конфигурирования элементов системы. Например, широко известен случай неверного конфигурирования протокола маршрутизации BGP в сети клиента провайдера AS7007, который привел к отказам работы большой части Интернета в 1997 году[1].

Угрозы внешних злоумышленников **(хакеров)** по определению являются умышленными и обычно квалифицируются как преступления. Отметим, что среди внешних нарушителей

---

[1] http://www.merit.edu/mail.archives/nanog/1997-04/msg00444.html.

безопасности встречаются люди, занимающиеся этой деятельностью как профессионально, так и из хулиганских побуждений.

# Ущерб и риск. Управление рисками

Известно, что абсолютная безопасность ИС не может быть обеспечена никакими средствами: всегда есть вероятность появления ошибок и проведения новых атак со стороны злоумышленников. Поэтому целью обеспечения информационной безопасности является не исключение, а *минимизация* возможного негативного влияния, которое могут оказать на ИС существующие угрозы. Из этого также следует, что надо каким-то образом ранжировать угрозы, чтобы решить, какими угрозами можно пренебречь, а на какие обратить основное внимание. Естественной мерой опасности атак и угроз является возможный ущерб, связанный с каждым из этих нарушений.

**Ущерб** (loss, impact) — это негативное влияние на систему, оказываемое проведенной атакой.

Подчеркнем, что в качестве ущерба рассматриваются не только и не столько потери, связанные с восстановлением работы ИС, в частности серверов, файловой системы или системы аутентификации, — главное внимание должно быть уделено потерям, которые в результате этих нарушений понесло предприятие, строящее свой бизнес на базе этой ИС.

Важнейшей задачей обеспечения информационной безопасности является управление рисками. Здесь **риск** определяется как оценка ущерба от атаки с учетом вероятностной природы атаки. Другими словами, риск характеризуется парой:

{Ущерб от атаки, Вероятность атаки}.

Суть **управления рисками** — это системный анализ угроз, прогнозирование и оценка их последствий для предприятия, ранжирование угроз по степени их вероятного осуществления и опасности последствий и, наконец, выбор на приоритетной основе контрмер, направленных на смягчение или исключение возможного негативного воздействия этих нарушений на деятельность предприятия.

Управление рисками включает три укрупненных этапа (рис. 26.5):

1. Анализ уязвимостей.
2. Оценка рисков.
3. Управление рисками, или риск-менеджмент (принятие конкретных мер).

*Анализ уязвимостей* — объективное обследование реально существующих компьютерной сети, административных процедур и персонала. Угрозы определяются по отношению к *активам* предприятия, то есть ресурсам предприятия, представляющим для него ценность и являющимся объектом защиты (оборудование, недвижимость, транспортные средства, вычислительные устройства, ПО, документация и др.). Перечень угроз формулируется предположительно, то есть с использованием вероятностных категорий.

*Оценка рисков* — ранжирование возможных атак по степени опасности. Для этого вычисляются соответствующие риски — вероятностные оценки ущерба, который может быть нанесен предприятию каждой из атак в течение некоторого периода времени. Риск атаки тем выше, чем больше ущерб от нее и чем выше ее вероятность.


**Рис. 26.5.** Управление рисками

*Риск-менеджмент* — по каждому риску предпринимаются меры из следующего списка:

- *Принятие риска*. Этот вариант касается неизбежных атак, наносящих приемлемый ущерб.
- *Устранение риска*. Данный вариант имеет место, когда существующий риск можно свести на нет устранением либо уязвимости (например, сделать код коммерческого программного продукта открытым), либо угрозы (допустим, установить антивирусную систему).
- *Снижение риска*. Если риск невозможно ни принять, ни устранить, то предпринимаются действия по его снижению. Например, всегда существует некоторая вероятность проникновения злоумышленников в систему путем подбора паролей. В таком случае риск несанкционированного доступа можно снизить, установив более строгие требования к длине и сменяемости паролей.
- *Перенаправление риска*. Если риск невозможно ни принять, ни устранить, ни даже существенно снизить, то риск может быть перенаправлен страховой компании.

# Типы и примеры атак

## Пассивные и активные атаки

Атаки разделяют на активные и пассивные.

**Активные атаки** включают явные воздействия на систему, изменяющие ее состояние. Это могут быть зловредный программный код-вирус, внедренный в исполняемую системой программу, искажения данных на страницах взломанного веб-сайта, блокировка сетевого сервиса путем «бомбардировки» его ложными запросами или внедренное в коммуника-

ционный протокол ложное сообщение. Главной отличительной чертой активных атак является то, что после своего завершения они, как правило, оставляют следы.

Многие активные кибератаки относят к типу *взламывания* (breaking-in) по аналогии с бытовыми ограблениями со взломом, когда хозяин заходит в свой дом и сразу обнаруживает поврежденные замки, опустошенные ящики и разбросанные на полу вещи. В компьютерной системе после активного проникновения злоумышленника тоже остаются следы «взлома» — например, изменяется содержимое памяти, поступают странные диагностические сообщения, приложения начинают выполняться неправильно, замедленно или вообще зависают, в характеристиках сетевого трафика и в других статистических данных о работе системы появляются необъяснимые всплески активности.

**Пассивные атаки** не нарушают нормальной работы ИС. Они связаны со сбором сведений об ИС, например, прослушиванием внутрисетевого трафика или перехватом сообщений, передаваемых по линиям связи. Во многих случаях пассивные атаки не оставляют следов, поэтому их очень сложно выявить (часто они так и проходят незамеченными). Если использовать военную терминологию, то это разведка (но не боем).

Противопоставление активной и пассивной форм атаки является некоторой идеализацией. На практике мы редко имеем дело с активной или пассивной атакой «в чистом виде». Чаще всего атака включает подготовительный этап сбора информации об атакуемой системе, а затем на основе собранных данных осуществляется активное вмешательство в ее работу. К полезной для хакера информации относятся типы ОС и приложений, IP-адреса, номера портов, имена и пароли пользователей. Часть информации такого рода может быть получена при анализе открытой информации или простом общением с персоналом (это называют **социальным инжинирингом**), а часть — с помощью тех или иных программ. В последнем случае мы сталкиваемся с другой последовательностью этапов: сначала выполняется активная фаза внедрения на атакуемый компьютер подслушивающей программы, затем период пассивного сбора информации (например, паролей пользователей), а после этого — снова активная фаза проникновения в компьютер.

## Отказ в обслуживании

К числу активных атак относятся две весьма распространенные атаки: отказ в обслуживании и распределенная атака отказа в обслуживании.

Смысл атаки **отказа в обслуживании** (Denial of Service, DoS) прямо следует из ее названия. Система, предназначенная для выполнения запросов легальных пользователей, вдруг перестает это делать или делает с большими задержками, что эквивалентно отказу. Очевидный пример такой системы — веб-сайт. Наверняка 17 млн британских болельщиков Энди Марри «обрушили» бы сайт BBC, если бы трансляция финального теннисного матча Уимблдона в 2013 году шла только в Интернете (к счастью, параллельно шла телевизионная передача). Такие всплески запросов являются экстраординарными, и правильно спроектированные серверы справляются с нагрузкой, на которую они рассчитаны. Однако отказ в обслуживании может наступить в результате не только резкой флюктуации интенсивности запросов, но и злонамеренных действий, когда перегрузка создается искусственно — допустим, когда на атакуемый компьютер посылается интенсивный поток запросов, сгенерированных средствами атакующего компьютера. Этот поток «затопляет» атакуемый компьютер, вызывая его перегрузку, и в конечном счете делает его недоступным. Блоки-

ровка происходит в результате исчерпания ресурсов либо процессора, либо операционной системы, либо канала связи (полосы пропускания).

Злоумышленник может многократно усилить эффект от проведения атаки отказа в обслуживании путем кражи чужой вычислительной мощности. Для этого он получает контроль над атакуемым компьютером, загружает в него вредительское программное обеспечение и активирует его. Таким образом, злоумышленник незаметно от владельца «ответвляет» часть вычислительной мощности, заставляя компьютер работать на себя. При этом владельцу компьютера не наносится никакого другого вреда, кроме снижения производительности его компьютера. Для проведения мощной атаки злоумышленник захватывает контроль над некоторым множеством компьютеров (рис. 26.6), организует их согласованную работу и направляет суммарный, многократно усилившийся поток запросов с множества компьютеров-«зомби» на компьютер-жертву. Говорят, что в таких случаях имеет место **распределенная атака отказа в обслуживании** (Distributed Denial of Service, DDoS), или *DDoS*-атака.

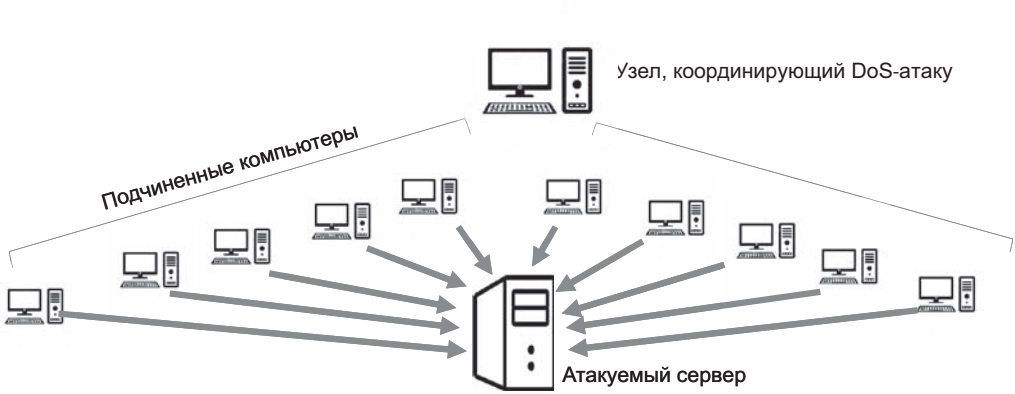



**Рис. 26.6.** Схема DDoS-атаки

При проведении атак злоумышленнику важно не только добиться своей цели, заключающейся в причинении ущерба атакуемому объекту, но и уничтожить все следы своего участия в этом. Одним из основных приемов, используемых злоумышленниками для «заметания следов», является *подмена содержимого пакетов*, или **спуфинг** (*spoofing*). В частности, для сокрытия места нахождения источника вредительских пакетов (например, при атаке отказа в обслуживании) злоумышленник изменяет значение поля адреса отправителя в заголовках пакетов. Поскольку адрес отправителя генерируется автоматически системным программным обеспечением, злоумышленник вносит изменения в соответствующие программные модули так, чтобы они давали ему возможность отправлять со своего компьютера пакеты с любыми IP-адресами. Еще труднее определить адрес источника распределенной атаки, так как непосредственными исполнителями выступают «зомбированные» компьютеры и именно их адреса содержатся в поле адреса отправителя пакетов, «бомбардирующих» компьютер-жертву. И хотя ничего не подозревающие владельцы компьютеров-исполнителей становятся участниками распределенной атаки помимо своей воли, большая часть ответственности ложится и на них. Ведь именно их недоработки в деле обеспечения безопасности собственных систем сделали возможной эту атаку.

## Внедрение вредоносных программ

Многочисленная группа активных атак связана с внедрением в компьютеры **вредоносных программ** (malware — сокращение от malicious software). К этому типу программ относятся троянские и шпионские программы, руткиты, черви, вирусы, спам, логические бомбы и др. (рис. 26.7).

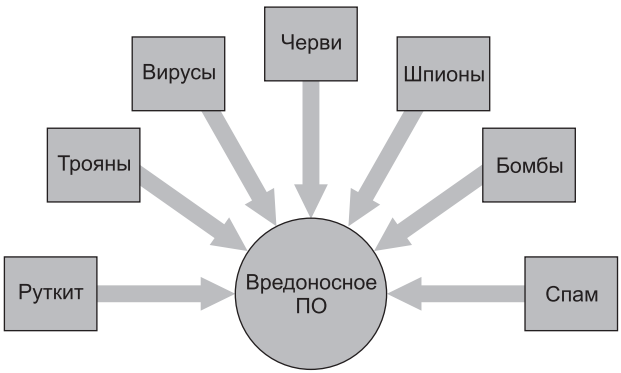



**Рис. 26.7.** Вредоносные программы

Эти программы могут проникать на атакуемые компьютеры разными путями. Самый простой из них — «самодоставка», когда пользователь загружает файлы из непроверенных источников (съемных носителей или веб-сайтов) либо беспечно открывает подозрительный файл, пришедший к нему как приложение по электронной почте. Существуют и более сложные представители вредоносных программ, обладающие собственными механизмами «размножения», копии таких программ распространяются по компьютерам сети без участия пользователей.

Одним из примеров вредоносных программ являются **шпионские программы** (spyware), которые тайно (как правило, удаленно) устанавливаются злоумышленниками на компьютеры ничего не подозревающих пользователей, чтобы отслеживать и фиксировать все их действия. В число таких действий может входить введение имени и пароля во время логического входа в систему, посещение тех или иных веб-сайтов, обмен информацией с внешними и внутренними пользователями сети и пр. Собранная информация пересылается злоумышленнику, который применяет ее в преступных целях.

Заметим, что в качестве шпионских программ могут использоваться не только созданные специально для этих целей вредоносные программы, но и программы легального назначения. Так, опасным средством шпионажа могут стать легальные системы мониторинга сети, такие, например, как популярные сетевые мониторы Wireshark или Microsoft Network Monitor. Исходное назначение этих программ состоит в том, чтобы дать администратору сети возможность следить за сетевым трафиком, в частности, захватывать пакеты, используя механизм фильтрации, просматривать их содержимое, собирать статистику по загрузке устройств. В руках злоумышленника такая программа превращается в мощный инструмент взлома сети, который позволяет перехватывать пакеты с паролями и другой секретной информацией. Потери, вызванные вредоносными программами, могут заключаться не только в уничтожении, искажении или похищении информации, приведении в нерабочее

состояние программного обеспечения, а значит, и компьютера в целом, но и в значительных затратах времени и сил администраторов на обнаружение и распознавание атак, фильтрацию внешних сообщений, тестирование и перезагрузку систем.

## Кража личности, фишинг

По мере развития услуг, оказываемых через Интернет, все более популярными становятся аферы, когда один человек выдает себя за другого. Действительно, ведь в этом случае не требуется личное присутствие в офисе, и индивидуум доказывает свою идентичность, передавая обслуживающему центру свои персональные данные по телефону или используя интерактивную систему веб-сайта. Злоумышленник решает выдать себя за другого, чтобы, например, взять кредит на чужое имя, получить доступ к чужому счету, рассчитаться за покупку чужой карточкой, получить именное приглашение на закрытое мероприятие. Такую *пассивную* атаку, заключающуюся в сборе данных о другом человеке, называют **кражей личности** (identity theft).

**Фишинг** (phishing — искаженное fishing) используется мошенниками для «выуживания» персональных данных. К примеру, вы отвечаете на телефонный звонок, а человек, представившийся сотрудником банка или государственной налоговой службы, работником ЖКХ или представителем провайдера мобильной связи, начинает выспрашивать у вас персональные данные. Угроза может прийти и по электронной почте. Будущая жертва получает сообщение, в котором, к примеру, говорится о якобы произведенной ею покупке, как правило, достаточно дорогой. Далее говорится, что при снятии средств за эту покупку у банковской системы возникли некие проблемы. Для разрешения ситуации клиенту предлагается срочно пройти по ссылке на сайт банка. Жертва, взволнованная тем, что никакой такой покупки она не совершала, торопится прояснить ситуацию, щелкает на предложенной ссылке и видит на экране знакомый логотип своего банка и интерактивную форму, запрашивающую персональные данные клиента, ИНН, номер счета, девичью фамилию матери и другие данные, которые нужны злоумышленнику.

Чем больше людей узнает о приемах выуживания информации, тем более изощренные методы обмана применяют преступники. Они создают поддельные сайты, выглядящие как настоящие, и используют доменные имена, очень похожие на настоящие. К примеру, когда вам предлагают посетить сайт международной платежной системы PayPal, а в адресной строке браузера появляется адрес www.p*e*ypal.com, вы можете и не заметить подмены. Когда же наученные горьким опытом пользователи Интернета стали более внимательными, мошенники научились, используя несовершенства браузеров, помещать в поле адресной строки браузера имя настоящего сайта, в нашем случае — paypal.com. В такой ситуации даже самый внимательный пользователь может потерять бдительность и перейти на подставной сайт. И хотя эта уязвимость браузера была вскоре устранена, расслабляться нельзя — преступники продолжают совершенствовать приемы фишинга.

Следующим изобретением стали всплывающие окна. Предположим, клиент получает доступ к сайту своего банка (действительному, не поддельному) в результате прохождения стандартной процедуры идентификации и аутентификации. Он просматривает страницы сайта, причем у него нет никаких сомнений в том, что это реальный сайт. В какой-то момент на экране появляется всплывающее окно, которое стилистически выглядит как неотъемлемая часть сайта. В этом окне размещена интерактивная форма, запрашивающая персональные данные. Клиент чувствует себя в полной безопасности и вводит все запра-

шиваемые данные. Однако в действительности «настоящий» сайт банка является только фоном, на котором располагаются окна-ловушки злоумышленника.

# Иерархия средств защиты

Обычно первое, что ассоциируется с информационной безопасностью, — это антивирусные программы, файерволы, системы шифрования, аутентификации, аудита и другие технические средства защиты. Бесспорно, роль этих средств в обеспечении безопасности велика, однако не меньшее, а иногда и большее влияние на безопасность системы оказывают средства, построенные на качественно иной основе.

Видеокамера и надежный замок в офисе, продуманная процедура приема сотрудников на работу, закон, угрожающий хакеру уголовным преследованием, стандарт, помогающий провести анализ возможного ущерба из-за действия нарушителя, — все эти мало схожие между собой средства одинаково важны для обеспечения безопасности.

Успех в области информационной безопасности может принести только *системный подход*, при котором средства защиты разных типов применяются совместно и под централизованным управлением.

Общепризнанным является представление множества разных средств защиты в виде четырех иерархически организованных уровней: законодательного, административного, процедурного и технического уровней. Средства каждого из них могут быть использованы на разных этапах жизненного цикла системы обеспечения информационной безопасности.

**Средства безопасности законодательного уровня.** К этому уровню средств безопасности относятся правовое регулирование, стандартизация, лицензирование и морально-этические нормы, принятые в обществе. Законодательство может прямо влиять на концепцию построения защиты. Например, выход Федерального закона «О персональных данных», регламентирующего меры по обеспечению безопасности персональных данных при их обработке, потребовал от многих предприятий пересмотра и внесения принципиальных изменений в процедуры и инфраструктуру обработки информации. Важным направлением законодательства в области безопасности является и стандартизация. Стандартные процедуры оценки систем дают возможность их сопоставления и сравнения, на основании результатов которых может выполняться **сертификация систем** на соответствие определенным требованиям. К числу самых известных сертификационных стандартов относят **Оранжевую книгу**[1]. Этот самый заслуженный и популярный стандарт оценивает степень защищенности ОС.

**Средства безопасности административного уровня** базируются на *политике безопасности*, которая определяет стратегические направления информационной защиты предприятия — очерчивает круг критически важных информационных ресурсов предприятия, защита которых представляет наивысший приоритет, предлагает возможные меры устранения или уменьшения связанных с этими ресурсами рисков. На основе найденной стратегии разрабатывается программа обеспечения безопасности ИС, планируется совокупный бюджет, необходимый для выполнения программы, назначаются руководители и очерчивается зона их ответственности.

---

[1] Формальное название этого стандарта: «Министерство обороны США, Критерии оценки доверенных компьютерных систем» (Department of Defence, Trusted Computer System Evaluation Criteria).

**Средства безопасности процедурного уровня** решают задачи, поставленные вышележащим административным уровнем, с использованием технических средств, предоставляемых нижележащим техническим уровнем. В качестве основного средства процедурного уровня выступает человек, выполняющий взаимосвязанную последовательность действий, направленную на решение той или иной задачи обеспечения безопасности. Любой аспект информационной безопасности предполагает использование средств процедурного уровня. Даже простое поддержание нормального режима работы ИС осуществляется за счет выполнения множества повседневных процедур: резервного копирования, управления программным обеспечением, профилактических работ и т. п. К средствам процедурного уровня относится также управление персоналом, пропускной режим на территорию предприятия, охрана границ территории и др.

**Средства безопасности технического уровня** можно разделить на программные, аппаратные и программно-аппаратные.

Программные средства включают защитные инструменты операционных систем (подсистемы аутентификации и авторизации пользователей, средства управления доступом, аудит и др.) и прикладные программы, предназначенные для решения задач безопасности (системы обнаружения и предотвращения вторжений, антивирусные средства, прокси-серверы).

Примером аппаратных средств, специализирующихся на информационной защите, являются источники бесперебойного питания, генераторы напряжения, средства контроля доступа в помещения и пр.

К аппаратно-программным средствам относятся, например, некоторые анализаторы сетевого трафика и межсетевые экраны. И хотя данный уровень средств называется техническим, к нему также относят математические методы (методы криптографии), алгоритмы (эвристический алгоритм расчета времени оборота в протоколе TCP), абстрактные модели (модели контроля доступа) и т. п. Именно техническому уровню средств безопасности уделяется основное внимание в этой книге.

# Принципы защиты информационной системы

Далее рассмотрены принципы построения системы обеспечения информационной безопасности, многие из которых имеют универсальный характер и применимы для защиты систем самой разной природы.

## Подход сверху вниз

Подход «сверху вниз», где понятие «верх» означает руководство предприятия, а «низ» — уровень рядовых сотрудников, соответствует универсальному принципу движения от общего к частному. При таком подходе все принципиальные решения принимаются топменеджментом, затем руководители промежуточных уровней преобразуют их в более развернутые планы и частные решения, которые, наконец, доводятся в виде инструкций и в разной степени формализованных процедур до уровня исполнителей. Именно руководители предприятия определяют стратегически важные объекты защиты, оценивают риски,

которые может понести предприятие в результате разрушения тех или иных информационных активов, намечают стратегию защиты информационных ресурсов.

Противоположный подход — «от частного к общему», или «снизу вверх», успешно используемый в некоторых сферах деятельности (например, в научных исследованиях), совершенно неприменим для проектирования сложных технических систем, к которым относится система обеспечения безопасности. Решения, принимаемые на уровне специалистов отдельных подразделений, могут оказаться несогласованными и не способствовать достижению глобальной цели. Например, системный администратор на свой страх и риск, приложив большие усилия и потратив значительные средства, обеспечил надежную защиту базы данных, а в ней, как впоследствии оказалось, хранится легко восстанавливаемая информация, которая не представляет для бизнеса особой ценности.

## Защита как процесс

*Защита должна представлять собой непрерывный, циклический, проактивный процесс.*

Задача информационной защиты не может быть решена раз и навсегда — напротив, работа по защите ИС должна идти непрерывно на протяжении всего существования защищаемой системы. Все системы безопасности уникальны, поскольку отражают специфику конкретных предприятий, для защиты которых они предназначены. Но какие бы различные цели ни преследовались при их создании, какие бы различные технологии ни использовались, какие бы индивидуальные решения ни принимались, общий ход проектирования для всех правильно построенных систем защиты должен иметь циклический характер. Цикл должен включать анализ угроз и уязвимостей защищаемой системы, оценку рисков, разработку политики безопасности всех уровней и реализацию принятых решений, направленных на снижение рисков.

Процесс обеспечения безопасности по возможности должен иметь *проактивный* (упреждающий), а не реактивный характер. При реактивном подходе защита заключается в принятии мер уже после того, когда нарушение безопасности произошло. Очевидно, что для успешности отражения атаки в первую очередь важны правильно выбранные действия и скорость их выполнения, а как раз этого трудно ожидать в ситуации кризиса. Поэтому более предпочтительным является проактивный подход, когда для защиты от вероятных угроз в спокойной обстановке проводится основательная подготовка оборонительных мер: устанавливаются необходимые технические средства, продумываются действия персонала, составляются и документируются инструкции — то есть делается все, что только может быть сделано заранее.

## Эшелонированная защита

*Эффективная защита обеспечивается путем многократного резервирования средств безопасности.*

Надежность решения любой задачи повышается, если использовать резервирование. Задача обеспечения безопасности не является здесь исключением. Так, например, для обес-

печения физической сохранности важного документа могут применяться самые разные средства защиты: дверные замки, датчики разбития окон, противопожарные сигнальные устройства, тревожная кнопка, сейф и масса других полезных приспособлений.

Информационная система существует в окружении гораздо более изощренных и многообразных угроз, здесь тем более невозможно найти панацею — одно-единственное средство, которое могло бы со стопроцентной надежностью противостоять всем видам атак. Поэтому на пути к защищаемому информационному ресурсу, как правило, устанавливают несколько барьеров. Вместе с тем возникает резонный вопрос: если ни одно из средств обеспечения безопасности не является абсолютно надежным и в принципе может быть преодолено злоумышленником, то в чем смысл нескольких защитных рубежей? Ответ: многократное резервирование в системах защиты служит не столько для того, чтобы какое-то из защитных средств продублировало отказавшее, а главным образом для того, чтобы заставить преступника *потратить как можно больше времени* на преодоление череды защитных барьеров. Замедление атаки повышает шанс ее обнаружения и принятия адекватных мер.

Рассмотрим, например, как реализуется принцип эшелонированной защиты в случае, когда необходимо обеспечить безопасность данных, хранящихся на одном из хостов внутренней локальной сети предприятия. На рис. 26.8 концентрическими окружностями представлены рубежи обороны, каждый из которых добавляет к уже накопленному защитному потенциалу собственные средства защиты (некоторые виды этих средств обеспечения безопасности рассмотрены в последующих главах).


**Рис. 26.8.** Рубежи обороны ИТ-системы

Самый внешний слой (организационно-административный) решает задачу безопасности данных, затрудняя злоумышленникам физический доступ к данным. С этой целью разрабатываются и применяются административные и организационные меры безопасности: проверка персонала при приеме на работу, взаимный контроль персонала, ограничение использования переносных портативных носителей и др.

Следующий слой также направлен на защиту от физического проникновения, но другими средствами — средствами физической защиты: ограждения, освещение, видеокамеры, контроль входа в здание, двери с кодовыми замками и т. п.

Далее вступают в действие технические средства безопасности, которые для сети с типовой структурой включают следующие рубежи защиты:

- внешняя сеть — для защиты от проникновения применяются средства регистрации входа, аудит, защитные свойства VPN;
- периметр внутренней сети — защита усиливается за счет файервола и прокси-серверов;
- внутренняя сеть — добавляются системы обнаружения и предотвращения вторжений сетевого уровня;
- хост — дополнительно проводятся процедуры аутентификации и авторизации, работают программный файервол, антивирус, системы обнаружения и предотвращения вторжений уровня хоста;
- данные — механизм контроля доступа и шифрование.

## Сбалансированная защита

*Степень защищенности системы измеряется защищенностью ее самого слабого звена.*

Этот принцип можно сформулировать и несколько по-другому: при построении системы безопасности необходимо обеспечить баланс стойкости всех ее компонентов. Например, если в сети все сообщения шифруются, но ключи легкодоступны, то эффект от шифрования окажется нулевым.

Из данного принципа можно сделать и еще одно заключение: если у злоумышленника существует несколько путей нанести урон системе и один из этих путей имеет слабую защиту, то нет смысла добиваться высокого качества защиты других путей. То есть если внешний трафик сети, подключенной к Интернету, проходит через мощный сетевой экран, но пользователи имеют возможность связываться с узлами Интернета по коммутируемым линиям через локально установленные модемы, то деньги (как правило, немалые), потраченные на сетевой экран, можно считать выброшенными на ветер. В таких случаях оказывается полезным еще один принцип — *принцип единого контрольно-пропускного пункта*, который заключается в том, что весь входящий во внутреннюю сеть и выходящий во внешнюю сеть трафик проходит через единственный узел сети, например межсетевой экран.

Необходимость баланса стойкости разных компонентов системы безопасности особенно ярко иллюстрируется провалами силовых ведомств, когда мощные организации, располагающие гигантскими ресурсами защиты, допускают существование явных прорех в своих системах безопасности. Так, известен случай, когда дисковый накопитель с классифицированными данными вооруженных сил США был случайно обнаружен продаю-

щимся на базаре в Ираке. Причина — использование ненадежных процедур утилизации данных и аппаратуры, хотя для остальных стадий существования данных — хранения и передачи — использовались мощные алгоритмы шифрования. Еще два примера связаны с масштабными утечками сверхсекретных данных — их главными действующими лицами были Мэннинг (2010 г.) и Сноуден (2013 г.). В обоих случаях очевидным слабым звеном стало управление персоналом: несмотря на то, что незадолго до инцидентов в поведении потенциальных нарушителей их коллегами отмечались «странности», а также то, что в карьере каждого из них произошли события, которые обычно квалифицируются как провоцирующие факторы, в отношении них не было предпринято никаких расследований. Кроме того, в обоих случаях не сработал и контроль использования портативных запоминающих устройств.

## Компромиссы системы безопасности

Система обеспечения безопасности создается в результате компромисса между *качеством защиты*, с одной стороны, и *затратами* на разработку этой системы — с другой. Под качеством здесь понимается комплекс характеристик: функциональное разнообразие, надежность защиты, удобство работы сотрудников, поддерживающих систему безопасности, сотрудников других подразделений предприятия.

Пусть, например, на предприятии внедряется система защиты. Ее назначение — сократить прогнозируемый совокупный ущерб, который мог бы быть нанесен предприятию, если бы система защиты отсутствовала. При внедрении системы защиты возможный ущерб от атак снизился, однако в позиции «убытки» у предприятия добавились затраты на внедрение системы безопасности. Кроме того, к убыткам предприятия должны быть отнесены те потери, которые предприятие понесло из-за снижения производительности в результате внедрения системы безопасности. Подобное снижение может быть вызвано как дополнительными затратами вычислительных ресурсов, так и необходимостью выполнения сотрудниками предприятия дополнительных процедур, связанных с безопасностью.

> Решение о внедрении системы безопасности можно считать экономически обоснованным только в том случае, если потери от риска в совокупности с затратами на систему безопасности и потерями из-за снижения производительности окажутся меньше исходного значения ущерба от риска.

Очевидно, что создание абсолютно непроницаемой защиты невозможно, так как у атакующих всегда остается теоретическая возможность взломать любую защиту — обычно это только вопрос времени и средств, которыми располагают злоумышленники. А это значит, что перед защищающейся стороной рано или поздно встанет дилемма: продолжать ли вкладывать деньги или остановиться на текущем уровне безопасности. Для ответа на этот вопрос разработчикам системы безопасности предлагается встать на место злоумышленника и попытаться оценить, какой уровень защиты злоумышленник мог бы посчитать неприемлемым для себя. Так, например, вряд ли имеет смысл браться за добычу конфиденциальных данных, если эта работа настолько длительная, что к тому времени, когда секретная информация попадет в руки, она уже устареет и не будет представлять никакой ценности. Аналогично, никто (из экономически мотивируемых преступников) не будет заниматься взломом системы, если выгоды от обладания защищаемым ресурсом меньше, чем средства, потраченные на проведение атаки. Исходя из этих соображений,

можно сформулировать следующие утверждения, каждое из которых представляет собой вариацию *принципа разумной достаточности*:

- Затраты на обеспечение безопасности информации должны быть, по крайней мере, не больше, чем величина потенциального ущерба от ее утраты.
- Стойкость системы безопасности считается достаточной, если время преодоления защиты превосходит время старения информации.
- Стойкость системы безопасности считается достаточной, если стоимость ее преодоления злоумышленниками превосходит стоимость полученной ими выгоды.

Таким образом, проектирование системы безопасности требует нахождения множества компромиссов между возможными затратами и возможными рисками. Так, в некоторых случаях можно отказаться от дорогостоящего файервола в пользу стандартных средств фильтрации обычного маршрутизатора, в других же приходится идти на беспрецедентные затраты.

# Шифрование — базовая технология безопасности

## Основные понятия и определения

Шифрование является краеугольным камнем всех служб информационной безопасности, будь то система аутентификации или авторизации, защищенный канал или средства безопасного хранения данных. Прежде чем перейти к конкретным методам и алгоритмам шифрования, давайте определим некоторые базовые понятия криптографии.

**Шифрование** — обратимое преобразование информации в целях обеспечения конфиденциальности данных. **Дешифрование** — процедура, которая, будучи примененной к зашифрованному тексту[1], снова приводит его в исходное состояние.

Пара процедур — шифрование и дешифрование — называется **криптосистемой**. Обычно криптосистема предусматривает наличие специального элемента — **секретного ключа**, в качестве которого может выступать некоторый предмет, например книга, число или рисунок. Простейший метод шифрования — замена букв в шифруемом тексте в соответствии с тем или иным правилом. Например, каждой букве алфавита может ставиться в соответствие другая буква этого алфавита, сдвинутая на некоторое число позиций влево или вправо. В качестве секретного ключа здесь выступает число, определяющее сдвиг.

Криптосистема считается *раскрытой*, если найдена процедура, позволяющая подобрать ключ за реальное время. Методы раскрытия криптосистемы, процедуры выявления уязвимости криптографических алгоритмов, выяснение секретного ключа называют **криптоанализом** или взломом шифра. Попытку раскрытия конкретного шифра с применением методов криптоанализа называют **криптографической атакой**.

Например, классическим методом криптоанализа, применяемым для раскрытия шифров, основанных на перестановке или замене букв, является частотный анализ. Для текстов, на-

---

[1] Информацию, над которой выполняются функции шифрования и дешифрования, мы будем условно называть «текстом», учитывая, что это может быть также числовой массив или графические данные.

писанных на определенном языке, относящихся к определенной сфере знаний, существуют устойчивые статистические данные о частоте, с которой встречается в тексте та или иная буква или последовательность букв, включая некоторые слова. Обладая такими данными и проведя статистический анализ зашифрованного текста, можно выполнить обратную замену символов.

Сложность алгоритма раскрытия является одной из важных характеристик криптосистемы и называется **криптостойкостью**. В криптографии принято **правило Керкгоффса**, заключающееся в том, что *стойкость шифра должна определяться только секретностью ключа*. Так, все стандартные алгоритмы шифрования (например, AES, DES, PGP) широко известны[1], их детальное описание содержится в легкодоступных документах, но от этого их эффективность не снижается. Система остается защищенной, даже если злоумышленнику известно все об алгоритме шифрования, но он не знает секретный ключ.

Существующие криптосистемы можно разделить на два класса — **симметричные** и **асимметричные**. В симметричных схемах шифрования (классическая криптография) секретный ключ шифрования совпадает с секретным ключом дешифрования. В асимметричных схемах шифрования (криптография с открытым ключом) ключ шифрования не совпадает с ключом дешифрования.

## Симметричное шифрование

На рис. 26.9 приведена модель симметричной криптосистемы. В данной модели три участника: два абонента, желающих обмениваться шифрованными сообщениями, и злоумышленник, который хочет перехватить и каким-либо образом расшифровать передаваемые сообщения.

**ПРИМЕЧАНИЕ**

При объяснении алгоритмов шифрования здесь и далее мы будем называть участников обмена Алисой и Бобом, а злоумышленника, старающегося перехватить их сообщения, — Евой. Эти имена традиционно используются в криптографии.

В распоряжении Алисы и Боба имеется незащищенный канал передачи сообщений, который в принципе может прослушиваться злоумышленником. Поэтому они договариваются использовать шифрование и для этого им нужен секретный ключ, известный только им двоим. Этот ключ им был передан (или один из них послал его другому) заранее по другому каналу — надежному. Боб и Алиса, получив ключ, находятся в абсолютно равном (симметричном) положении, каждый из них может как посылать шифрованные сообщения, так и получать и расшифровывать их. Для определенности на рисунке показана схема передачи сообщений со стороны Боба.

Боб зашифровывает свое сообщение — открытый текст $X$ — функцией шифрования $F$ с секретным ключом $k$ и передает в открытый канал результат — шифрованный текст $Y$. Алиса получает $Y$ и передает его на вход функции дешифрования $F'$, которая выполняет в обратном порядке все действия, выполненные ранее функцией $F$. Это может быть сделано

[1] Вместе с тем существует немало фирменных алгоритмов, описание которых не публикуется для того, чтобы усилить защиту.

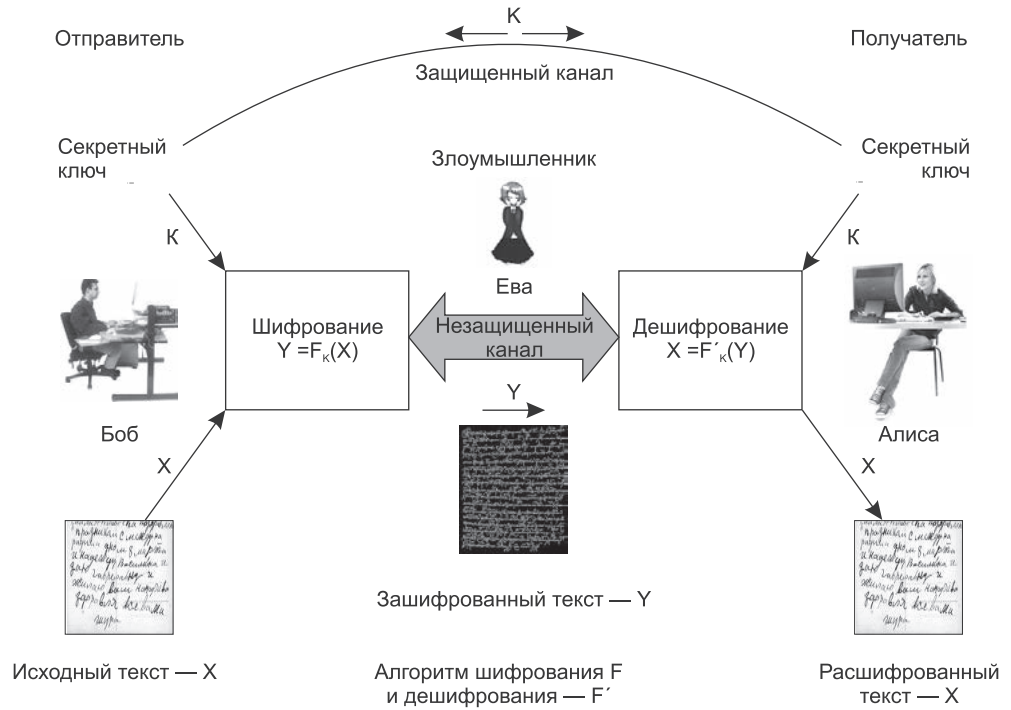



**Рис. 26.9.** Модель симметричного шифрования

только в том случае, если на вход функции *F* будет подано то же самое значение параметра — значение ключа *k*. Алиса имеет секретный ключ и поэтому получает расшифрованное значение. При необходимости передавать шифрованные сообщения Бобу Алиса должна действовать аналогичным образом.

До недавнего времени наиболее популярным стандартным симметричным алгоритмом шифрования данных был **DES** (Data Encryption Standard). Для шифрования используется циклическая последовательность операций над битами шифруемого текста — перестановки, подстановки, логические операции двоичной арифметики. Эти операции применяются блочно, размер блока равен 64 битам. Некоторые операции над блоками шифруемых данных связаны со значениями секретного ключа, также имеющего длину 64 бита. Промежуточный результат шифрования складывается по модулю 2 (операция XOR) с преобразованной двоичной последовательностью ключа. Полный шифрованный текст получается слиянием результатов шифрования для всех блоков исходного текста.

Процедура дешифрования выполняется в обратном порядке. Поскольку собственно алгоритм DES не является секретом и широкодоступен, в том числе доступны все таблицы, описывающие перестановки, то стойкость алгоритма (степень сложности дешифрования) определяется только сложностью подбора ключа, которая прямо зависит от его длины.

Криптостойкость всех симметричных алгоритмов зависит от качества ключа, что предъявляет повышенные требования к службе генерации ключей, а также к надежности канала обмена секретными ключами между участниками секретных переговоров.

Чтобы повысить криптостойкость алгоритма DES, был разработан его усиленный вариант, называемый **тройным алгоритмом DES**, который включает троекратное шифрование с использованием двух разных ключей. При этом можно считать, что длина ключа увеличивается с 56 до 112 бит, а значит, криптостойкость алгоритма существенно повышается. Но за это приходится платить производительностью — тройной алгоритм DES требует в три раза больше времени на реализацию, чем «обычный».

В 2001 году был стандартизован симметричный алгоритм шифрования **AES** (Advanced Encryption Standard). AES обеспечивает лучшую защиту, так как использует 128-битные ключи (а также может работать со 192- и 256-битными ключами) и имеет более высокую скорость работы, кодируя за один цикл 128-битный блок, в отличие от 64-битного блока DES. В настоящее время, помимо AES, распространенным симметричным алгоритмом шифрования является алгоритм Blowfish. Утвержденные в качестве государственного стандарта РФ шифры «Магма» и «Кузнечик» также являются симметричными блочными методами шифрования с ключом 256 бит, при этом в первом из них используется блок размером 64 бита, а во втором — 128 бит.

**(S)** *Симметричный блочный алгоритм шифрования, использующий преобразования Фейстеля.*

## Проблема распределения ключей

Симметричный подход к шифрованию изначально несет в себе очевидную проблему, называемую проблемой **распределения ключей** (key distribution), которая состоит в следующем. Отправитель и получатель хотят обмениваться секретными сообщениями, но в их распоряжении имеется незащищенный открытый канал. Поэтому они вынуждены использовать шифрование, но чтобы послать зашифрованное сообщение, нужно предварительно обменяться секретной информацией о значении ключа. Однако секретный ключ нельзя передать по открытому каналу. Если его зашифровать другим ключом, то опять возникает проблема доставки второго ключа. Получается замкнутый круг.

Единственным по-настоящему надежным решением этой проблемы является передача ключа при личной встрече абонентов. Однако при активном обмене требуется часто менять ключи, чтобы не дать возможности криптоаналитику собрать большое количество шифрованного материала — известно, что чем больше зашифрованных сообщений окажется в руках криптоаналитика, тем легче ему раскрыть криптосистему. Кроме того, если злоумышленник перехватывает и сохраняет сообщения, зашифрованные одним и тем же ключом, то при раскрытии данного ключа они *все* окажутся скомпрометированными. Следовательно, необходимы частые личные встречи абонентов для обмена ключами, что, во-первых, не всегда возможно, а во-вторых, вообще делает бессмысленным обмен данными по каналу связи — действительно, зачем шифровать данные, если их можно лично передать при встрече.

Менее надежным способом распределения ключей является использование курьеров или других вариантов защищенной доставки ключей, но это решение тоже имеет очевидные изъяны. Существуют и другие приемы, не решающие, но смягчающие проблему распределения ключей. Например, у абонента может быть несколько секретных ключей, имеющих разное назначение. Один ключ выдается ему на долгий срок. Этот ключ применяется только для шифрования (дешифрования) других ключей — кратковременных, каждый из

которых действителен только на время одного сеанса связи. И хотя в этом случае все равно остается проблема доставки долговременного ключа, уже нет необходимости его частой смены, так как этот ключ используется относительно редко и шифрует небольшие порции данных — сеансовые ключи.

Несмотря на различные усовершенствования процедуры распределения ключей, они не могут полностью устранить коренной изъян симметричных методов — *необходимость доставки секретного ключа по незащищенному каналу*.

Если проблема с ключами возникает в системе с двумя абонентами, то она многократно усугубляется в системе с большим числом абонентов. Пусть, например, $n$ абонентов желают обмениваться секретными данными по принципу «каждый с каждым» — значит, в этом случае потребуется $n(n-1)/2$ ключей и все они должны быть сгенерированы и распределены надежным образом. То есть *количество требуемых ключей пропорционально квадрату количества абонентов*, что при большом числе абонентов делает задачу чрезвычайно сложной. Но именно такая ситуация наблюдается во всех современных сетях связи — телефонных, радио и компьютерных. Все это сделало проблему распределения ключей чрезвычайно актуальной.

## Метод Диффи—Хеллмана передачи секретного ключа по незащищенному каналу

В середине 70-х годов американские ученые Мартин Хеллман и Уилтфилд Диффи нашли способ, с помощью которого абоненты могли безопасно обмениваться секретными ключами без передачи их по каналу связи. Особенность этого открытия состоит в том, что оно противоречит всем интуитивным представлениям человека, делая возможным то, что кажется «очевидно» невозможным.

Метод Диффи—Хеллмана основан на использовании свойств односторонних функций.

**Односторонняя функция** (one-way function) — это функция $y = F(x)$, которая легко вычисляется для любого входного значения $x$, но обратная задача — определение $x$ по заданному значению функции $y$ — решается очень трудно. Примером односторонней функции может служить простейшая функция двух аргументов $F(p,q) = pq$, представляющая собой произведение двух простых чисел $p$ и $q$, она вычисляется сравнительно просто, даже если числа $p$ и $q$ очень большие. Но чрезвычайно сложно решить обратную задачу (называемую факторизацией) — по произведению подобрать исходные два простых числа. Другой пример — функция $Y(x) = D^x \bmod P$, которая при некоторых ограничениях на параметры $D$ и $P$ является односторонней, то есть зная $Y$, а также параметры $D$ и $P$, нельзя без экстраординарных вычислительных усилий найти аргумент $x$.

Итак, пусть Алиса и Боб решили обмениваться шифрованными сообщениями, но в их распоряжении имеется только незащищенный открытый канал связи, при этом никаких возможностей встретиться или передать секретный ключ через кого-нибудь другого у них нет. В соответствии с алгоритмом Диффи—Хеллмана для успешного решения задачи Алиса и Боб должны выполнить следующие действия. Прежде всего они *открыто* договариваются о том, что будут использовать одностороннюю функцию $Y = D^x \bmod P$. Затем они договариваются о значениях параметров $D$ и $P$. Пусть, например, они договорились, что $D = 7$ и $P = 13$, то есть функция имеет вид $Y = 7^x \bmod 13$. Еще раз подчеркнем, что в соответствии с алгоритмом Диффи—Хеллмана вся эта информация не является

секретной, и даже если переговоры будут подслушаны Евой, это не даст ей возможности прочитать сообщения Алисы и Боба. Дальнейшие действия участников обмена описываются в табл. 26.1.

**Таблица 26.1.** Действия Алисы и Боба в соответствии с алгоритмом Диффи—Хеллмана

| | **Действия Алисы** | | **Действия Боба** | |
|---|---|---|---|---|
| 1 | Алиса секретным образом выбирает произвольное число $A$ (закрытый ключ Алисы) | Пусть, например, $A = 2$ | Боб также секретно выбирает произвольное число $B$ (закрытый ключ Боба) | Пусть, например, $B = 4$ |
| 2 | Алиса вычисляет значение $a$ односторонней функции $Y$, используя в качестве аргумента свое секретное число $A$: то есть $a = D^A \bmod P$ (открытый ключ Алисы) | $a = 7^2 \bmod 13 = 10$ | Боб также вычисляет значение $b$ односторонней функции $Y$, используя в качестве аргумента свое секретное число $B$: $b = D^B \bmod P$ (открытый ключ Боба) | $b = 7^4 \bmod 13 = 2401 \bmod 13 = 9$ |
| 3 | Алиса посылает Бобу свой открытый ключ $a$ | | Боб посылает Алисе свой открытый ключ $b$ | |
| 4 | Алиса, получив от Боба число $b$, вычисляет по формуле $K = b^A \bmod P$ (разделяемый секретный ключ) | $K = 9^2 \bmod 13 = 81 \bmod 13 = 3$ | Боб, получив от Алисы число $a$, вычисляет по формуле $K = a^B \bmod P$ (разделяемый секретный ключ) | $K = 10^4 \bmod 13 = 10000 \bmod 13 = 3$ |
| 5 | По правилам модульной арифметики $b^A \bmod P = (D^B \bmod P)A \bmod P = D^{BA} \bmod P$ | $K = 3$ | По правилам модульной арифметики $a^B \bmod P = (D^A \bmod P)\, B \bmod P = D^{AB} \bmod P$ | $K = 3$ |

В результате описанной процедуры на шаге 4 Алиса и Боб получили одно и то же число 3! Математические преобразования показывают, что вычисления Алисы и Боба всегда будут давать одинаковые результаты. Полученные в результате числа они могут использовать в качестве известного только им ключа для различных симметричных методов шифрования.

Посмотрим, может ли Ева подобрать разделяемый секретный ключ Алисы и Боба. Пусть на шаге 3, когда Алиса и Боб посылали друг другу свои открытые ключи $a$ (10) и $b$ (9), Ева смогла перехватить эти числа (ведь канал является открытым) и теперь пытается вычислить разделяемый секретный ключ. Зная число $a$, которое Алиса послала Бобу, Ева хочет повторить действия Боба и вычислить разделяемый секретный ключ по формуле $10^B \bmod 13$. Для этого ей требуется закрытый ключ Боба $B$, который он, однако, хранит секретно от всех. Зато Ева знает, что Боб использовал свой закрытый ключ $B$, когда вычислял значение своего открытого ключа — $b$. То есть задача будет решена, если Ева сможет подобрать такое значение $B$, чтобы значение $7^B \bmod 13$ равнялось 9. Но именно это практически неразрешимо, поскольку функция $7^B \bmod 13$ является односторонней. Таким образом, Алиса и Боб действительно получили секретный ключ.

Для того чтобы усложнить решение обратной задачи, то есть для восстановления закрытого ключа Алисы или Боба по открытому, на параметры алгоритма накладываются некоторые ограничения, в том числе следующие:

- все параметры $D$, $P$, $A$, $B$ должны быть целыми положительными числами;
- $A$ и $B$ должны быть большими числами порядка $10^{100}$;

- $P$ должно быть большим простым числом порядка $10^{300}$, причем желательно, чтобы $(P - 1)/2$ также было простым числом;
- число $D$ не обязательно должно быть большим, обычно оно выбирается меньше десяти, $D < P$.

Хотя алгоритм Диффи—Хеллмана стал прорывом в области криптографии, в его исходном состоянии он представлял скорее теоретическую, нежели практическую ценность. Устранив препятствие в виде необходимости надежного закрытого канала для передачи ключа, этот метод не снял проблемы квадратичной зависимости числа ключей от числа абонентов. Решение пришло очень скоро — уже через год после появления алгоритма Диффи—Хеллмана была теоретически доказана возможность принципиально нового подхода к шифрованию — асимметричного шифрования, при использовании которого (помимо прочих преимуществ) кардинально упрощается задача распределения ключей.

## Концепция асимметричного шифрования

До сравнительно недавнего времени понятие «симметричное шифрование» не существовало просто потому, что все методы, которые использовались человечеством на протяжении нескольких тысяч лет, по современной классификации могли быть отнесены к классу симметричных, а других просто не было. Более того, все эти тысячи лет существовала твердая убежденность, что в принципе никогда не может быть иных схем, кроме симметричной, когда отправитель шифрует сообщение с помощью секретного ключа, а получатель с помощью этого же ключа сообщение расшифровывает!

Революция свершилась в конце 60-х — середине 70-х, когда с разницей в несколько лет две группы ученых, одна из которых — уже знакомые нам Диффи и Хеллман, а другая — сотрудники секретной правительственной лаборатории Великобритании[1] Эллис, Кокс и Уильямсон, независимо друг от друга изобрели принципиально новый подход к шифрованию, открывающий глобальные перспективы в области современных коммуникаций. Предельно упрощая, этот подход можно описать фразой: «отправитель шифрует сообщение с помощью одного ключа, а получатель расшифровывает его с помощью другого ключа». Как видим, здесь на двух сторонах обменного канала используются разные ключи, то есть присутствует асимметрия — соответственно все методы, основанные на таком подходе, стали называть «асимметричными».

Конечно, удивительно, что за несколько тысяч лет не было ни одной известной науке попытки изобретения асимметричного метода шифрования, и вдруг, практически одновременно, две независимые группы ученых совершают это открытие! Возможно, причина кроется в том, что к концу 60-х годов совпало два обстоятельства: во-первых, возникла острая потребность в новом типе шифрования, а во-вторых, появились технические возможности реализации этой идеи.

Потребность была продиктована зрелостью таких видов массовых коммуникаций, как телефон, радио, компьютерные сети, для которых, во-первых, особенно важна секретность ввиду слабой защищенности публичных средств связи, а во-вторых, неприемлемы ограничения традиционных методов шифрования, выражающихся в необходимости обмена секретным

[1] Известно, что исторически первыми были британские криптографы, которые открыли асимметричное шифрование на 6 лет раньше, чем Диффи и Хеллман, однако до 1997 года они не могли обнародовать свои результаты, так как их работа имела гриф секретности.

ключом для каждой пары абонентов. К концу 60-х годов стали отчетливо вырисовываться перспективы использования Интернета как мировой сети связи, и одновременно с этим стало приходить осознание того, что глобальная публичная сеть может выполнить свою миссию только в том случае, если миллионам ее пользователей будет предоставлена возможность защищенного обмена сообщениями.

Эти темы особенно волновали военных разных стран, которых очень привлекала возможность распределенного управления вооруженными силами, но пугала невозможность гарантировать секретность передаваемых директив. И если в недалеком прошлом проблема распределения секретных ключей хотя и существовала, но была преодолимой, то в новых условиях она стала принципиальным препятствием.

К этому времени созрели технические возможности реализации вычислительно емких алгоритмов шифрования, к которым могут быть отнесены асимметричные алгоритмы. Массовое распространение получили компьютеры, обладающие такой вычислительной мощностью, которой до сих пор могли похвастаться только уникальные модели суперкомпьютеров. Это сделало шифрование обыденной операцией, которая может быть выполнена на обычном персональном компьютере.

Вот на таком историческом фоне и была предложена концепция асимметричной криптосистемы, называемой также **шифрованием с открытым ключом**.

На рис. 26.10 представлена модель асимметричной криптосистемы. Так же, как и в модели симметричного шифрования (см. рис. 26.9), здесь показаны три участника: отправитель (Боб), получатель (Алиса) и злоумышленник (Ева). В отличие от симметричной схемы шифрования, в которой наличие разделяемого секретного ключа автоматически означает

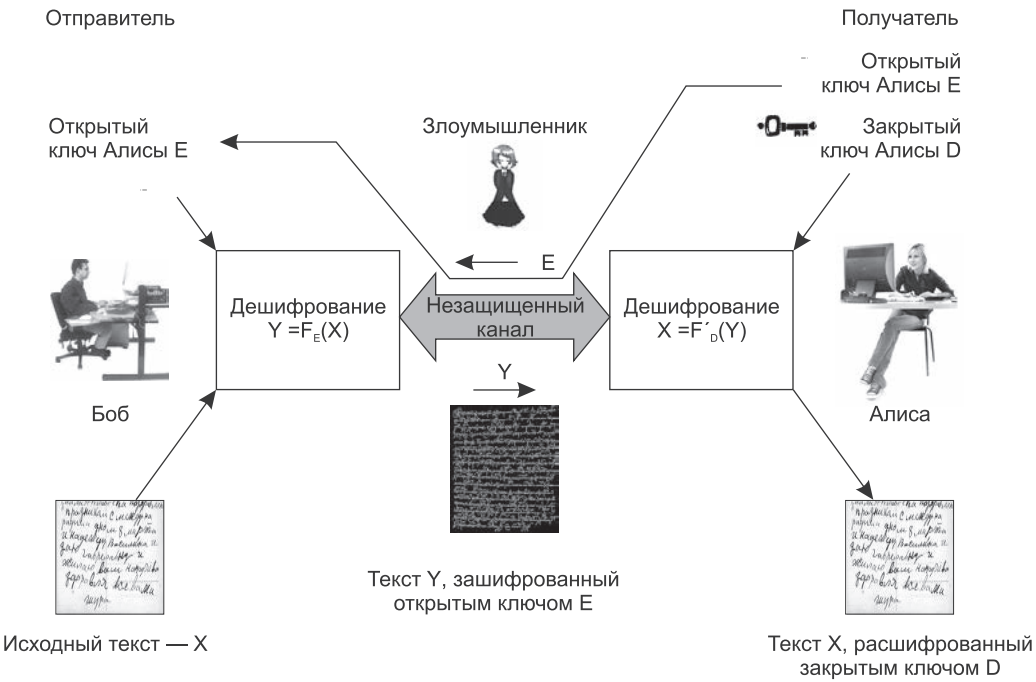



**Рис. 26.10.** Схема асимметричного шифрования

возможность двустороннего защищенного обмена, здесь существует отдельная процедура для передачи зашифрованных сообщений в каждую из сторон. На рисунке показан вариант, когда зашифрованные сообщения могут быть посланы только Бобом в сторону Алисы, но не наоборот.

1. Итак, Алиса пожелала, чтобы Боб посылал ей зашифрованные сообщения. Для этого она сгенерировала пару ключей: **открытый ключ** (public key) *E* и **закрытый ключ** (private key) *D*. Для шифрования текста служит открытый ключ, но расшифровать этот текст можно только с помощью закрытого ключа. Алиса не хочет, чтобы кто-либо читал ее почту, поэтому она сохраняет закрытый ключ *D* (часто называемый также личным ключом) в секрете. Открытый же ключ *E* Алиса свободно передает всем, от кого хочет получать зашифрованные сообщения. Открытый ключ не представляет никакого секрета, Алиса может поместить его на своей странице в социальной сети или обнародовать в рекламе на телевидении. Все, кто хотят посылать Алисе зашифрованные сообщения, используют один и тот же ключ *E*, но при этом никто из них не может прочитать сообщения друг друга.
2. Алиса передает Бобу свой открытый ключ *E* по незащищенному каналу в незашифрованном виде.
3. Боб шифрует свое сообщение *X* открытым ключом Алисы *E* и посылает зашифрованный текст $Y = F_E(X)$ по открытому каналу. Никто не может прочитать это сообщение. Даже сам Боб, если бы ему вдруг захотелось перечитать, что он там написал, не смог бы этого сделать, потому что для этого нужен закрытый ключ Алисы, которого у него нет.
4. Алиса получает шифрованное сообщение $Y = F_E(X)$ и расшифровывает его своим закрытым ключом *D*: $X = F'_D(Y)$.

Для того чтобы в сети все *n* абонентов имели возможность не только принимать зашифрованные сообщения, но и сами посылать таковые, каждый абонент должен обладать собственной парой ключей *E* и *D*. Всего в сети будет $2n$ ключей: *n* открытых ключей для шифрования и *n* секретных ключей для дешифрования. Таким образом решается проблема масштабируемости: *квадратичная зависимость количества ключей от числа абонентов в симметричных алгоритмах заменяется линейной зависимостью в асимметричных алгоритмах*. Решается и проблема доставки ключа; поскольку теперь он не является секретом, его можно без опаски передавать по открытому каналу. А злоумышленнику нет смысла стремиться завладеть открытым ключом, поскольку это не дает возможности расшифровать текст или вычислить закрытый ключ.

## Алгоритм асимметричного шифрования RSA

Открыватели асимметричного подхода к шифрованию показали *концептуальную возможность* существования функций, позволяющих построить криптографическую систему, в которой текст шифруется одним ключом, а расшифровывается другим. Они также обрисовали те перспективы, которые открывает этот подход в деле решения проблемы распределения ключей. Ими были сформулированы два принципиальных требования, которым должны удовлетворять функции асимметричной криптосистемы:

- зашифрованное сообщение должно быть результатом вычислений односторонней функции, чтобы никто не мог выполнить обратные преобразования и получить исходный текст;

- эта односторонняя функция должна быть сконструирована таким образом, чтобы у нее был некоторый секретный элемент, зная который получатель шифровки мог бы легко выполнить обратное преобразование.

Функции, которые удовлетворяют данным требованиям, назвали **односторонними функциями с потайным входом** (trapdoor function). Некоторое время ученым не удавалось найти функций, удовлетворяющих этим критериям, поэтому идея асимметричного шифрования не находила практического применения. Наконец, в 1978 году трое американских ученых, Ривест, Шамир и Адлеман, предложили долгожданный **алгоритм асимметричного шифрования RSA**, названный так по первым буквам их фамилий — Rivest, Shamir, Adleman. В табл. 26.2 описываются основные шаги алгоритма RSA.

**Таблица 26.2.** Последовательность действий участников обмена данными в соответствии с алгоритмом RSA

| **Действия Алисы и Боба** | **Числовой пример** |
|---|---|
| Алиса произвольно выбирает два случайных простых числа $P$ и $Q$. Они должны быть очень большими — от этого зависит стойкость алгоритма шифрования | В примере для простоты расчетов берутся очень маленькие числа. Пусть $P = 7$ и $Q = 13$ |
| Алиса вычисляет два произведения:<br>$N = PQ$<br>$M = (P - 1)(Q - 1)$ | $N = 91$<br>$M = 6 \times 12 = 72$ |
| Алиса выбирает случайное целое число $E$, меньшее $M$ и не имеющее с ним общих сомножителей | $E = 5$ |
| Пара $(E, N)$ — это открытый ключ Алисы, который она передает всем, от кого хочет получать шифрованные сообщения. Алиса посылает Бобу и всем остальным, с кем она желает вести защищенную переписку, свой открытый ключ $(E, N)$ | (5, 91) |
| Алиса находит $D$ такое, что $DE = 1 \bmod M$.<br>Пара $(D, N)$ — это закрытый ключ Алисы, который она не показывает никому. С этого момента она готова получать зашифрованные сообщения от Боба | $D \times 5 = 1 \bmod 72$<br>$D = 29$ (это число легко находится подбором, если учитывать признаки делимости на 5) |
| Боб получил открытый ключ Алисы и так же, как все остальные, имеющие доступ к этому ключу, может посылать Алисе зашифрованные сообщения. Он представляет свое сообщение в любом цифровом формате и разбивает его на блоки $X$ таким образом, чтобы $0 < X < N$ | Пусть секретный текст, посылаемый Бобом, состоит из одного символа $R$, который в коде ASCII имеет значение 1010010, или 82 в десятичном коде |
| Боб шифрует сообщение $X$ открытым ключом $(E, N)$: $C = X^E \bmod N$ и посылает Алисе зашифрованное сообщение $C$ | $C = 82^5 \bmod 91 = =\{82^3 \bmod 91 \times 82^2 \bmod 91\} \bmod 91 = 10$<br>Вычисление модуля от степени числа упрощается при использовании следующего правила: $(Y^{a+b+c}) \bmod P = (Y^a \bmod P \times Y^b \bmod P \times Y^c \bmod P) \bmod P$ |
| Алиса получает сообщение $C$ и расшифровывает его своим закрытым ключом $(D, N)$: $X = C^D \bmod N$ | $X = 10^{29} \bmod 91 = \{10^1 \bmod 91 \times 10^4 \bmod 91 \times 10^6 \bmod 91 \ldots\} \bmod 91$ (внутри фигурной скобки четыре раза повторяется последний сомножитель — $10^6 \bmod 91$) $10 \bmod 91 = 10$; $10^4 \bmod 91 = 81$; $10^6 \bmod 91 = 1$<br>$X = \{10 \times 81 \times 1\} \bmod 91 = 82$ |
| Результат расшифровки $X = 82$ совпадает с исходным секретным сообщением | |

Еве для того, чтобы прочитать перехваченное сообщение $C$, требуется закрытый ключ Алисы ($D$, $N$). Но в ее распоряжении имеется только открытый ключ ($E$, $N$). Теоретически, зная открытый ключ, можно вычислить значение закрытого ключа. Однако необходимым промежуточным действием в этом преобразовании является нахождение простых чисел $P$ и $Q$, для чего нужно разложить на простые множители очень большое число $N$, а это является чрезвычайно трудоемкой процедурой. Таким образом, здесь мы имеем дело с односторонней функцией $N = P \times Q$. Но для Алисы это же действие — разложение большого числа на два простых множителя — не представляет никакого труда, потому что она знает, как сконструировано это число $N$, она сама его вычислила, произвольно выбрав два сомножителя. Другими словами, Алисе известен «потайной вход» этой односторонней функции. Именно с огромной вычислительной сложностью разложения большого числа $N$ на простые множители $P$ и $Q$ связана высокая криптостойкость алгоритма RSA.

Хотя информация об открытом ключе не является секретной, ее нужно защищать от *подлогов*, чтобы злоумышленник под именем легального пользователя не навязал свой открытый ключ, после чего с помощью своего закрытого ключа он мог бы расшифровывать все сообщения, посылаемые легальному пользователю, и отправлять свои сообщения от его имени. Решение проблемы дает технология **цифровых сертификатов**[1] — электронных документов, которые связывают конкретных пользователей с конкретными открытыми ключами.

## Хеш-функции. Односторонние функции шифрования. Проверка целостности

В области информационной безопасности особое место занимает специальный класс односторонних функций, называемых хеш-функциями.

**Хеш-функцией** (hash function) называют одностороннюю функцию, которая, будучи примененной к некоторым данным, дает в результате значение, состоящее из фиксированного сравнительно небольшого и не зависящего от длины исходных данных числа байтов. Результат работы хеш-функции называют **хеш-кодом** или **дайджестом**. Рассмотрим, например, функцию взятия модуля $Y(x) = x \bmod n$, где операция $x \bmod n$ (то есть $x$ по модулю $n$) дает в результате остаток от деления $x$ на $n$. Эта функция, во-первых, является односторонней, так как, зная остаток от деления $x$ на $n$, невозможно однозначно определить значение аргумента $x$, во-вторых, она относится к классу хеш-функций, поскольку ее результат не зависит от аргумента $x$ и всегда находится в диапазоне от 0 до $(n - 1)$.

Хеш-функции называют также **односторонними функциям шифрования** (**ОФШ**), где в качестве шифрованного представления исходных данных выступает дайджест. При этом знание дайджеста *не позволяет* и даже *не предполагает* восстановления исходных данных. Односторонние функции шифрования используют в разных целях, в том числе для обеспечения целостности и аутентичности информации. Пусть, например, требуется обеспечить целостность сообщения, передаваемого по сети. Отправитель и получатель договорились, что они будут использовать одностороннюю функцию $H$ с секретным числом — ключом $K$ — в качестве параметра. Прежде чем отправить сообщение $X$, отправитель вычисляет для него дайджест $M = H(X, K)$ и отправляет его вместе с сообщением $X$ адресату

---

[1] См. раздел «Аутентификация на основе цифровых сертификатов» главы 27.

(рис. 26.11). Адресат, получив данные *X* и *M*, применяет ту же самую ОФШ к переданному в открытом виде исходному сообщению *X*, используя известный ему секретный ключ *K*: *M'* =*H*(*X*, *K*). Если значения дайджестов вычисленного локально *M'* и полученного по сети *M* совпадают, то содержимое сообщения не было изменено во время передачи.

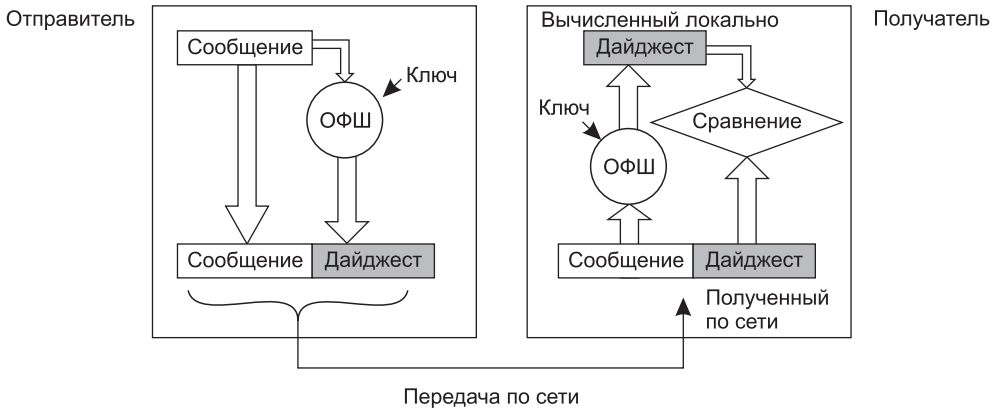



**Рис. 26.11.** Использование параметрической односторонней функций шифрования для контроля целостности

Хеш-функции широко используются в сетевых протоколах, алгоритмах электронно-цифровой подписи, механизмах аутентификации на основе паролей. Наиболее популярной в системах безопасности в настоящее время является серия хеш-функций MD2, MD4, MD5. Все они генерируют дайджесты фиксированной длины в 16 байт. Адаптированным вариантом MD4 является американский стандарт SHA, длина дайджеста в котором составляет 20 байт. Компания IBM поддерживает односторонние функции MDC2 и MDC4, основанные на алгоритме шифрования DES.

# ГЛАВА 27 Технологии аутентификации, авторизации и управления доступом

## Технологии аутентификации

Как отмечено, аутентификация применительно к вычислительной системе — это доказательство подлинности различных элементов данной системы при их взаимодействии. Пользователь при входе в систему должен предъявить системе доказательства, что он именно тот пользователь, идентификатор которого он вводит. Таким доказательством может служить пароль. Документ, полученный пользователем по электронной почте, должен сопровождаться дополнительной информацией, убеждающей пользователя, что документ не был изменен при передаче и что автором этого документа является именно тот человек, от имени которого это письмо было послано. Здесь доказательством может служить электронная подпись. Устройства, взаимодействующие по сети, должны доказать друг другу, что ни одно из них не подменено злоумышленником с целью ответвления или прослушивания трафика. Для этого в протоколе взаимодействия устройств должна быть предусмотрена процедура взаимной аутентификации. Взаимная аутентификация требуется и для организации безопасного сеанса пользователя и серверного приложения. Аутентификация может проводиться не только по отношению к отдельному пользователю, но и к группе пользователей. Методы аутентификации различаются в зависимости от того, что служит аутентификатором, а также от того, каким образом организован обмен аутентификационными данными между аутентифицируемым и аутентифицирующим элементами системы.

## Факторы аутентификации человека

Абсолютно надежная аутентификация человека представляет собой теоретически неразрешимую задачу. Нет такого аутентификатора, который со стопроцентной надежностью доказывал бы аутентичность человека. Пароль можно перехватить, электронный ключ — украсть, отпечаток пальца — подделать, радужную оболочку глаза — подменить качественным изображением. Более того, не существует научного доказательства невозможности совпадения у разных людей отпечатков пальцев или радужных оболочек глаза. Даже совпадение результатов анализа ДНК при современном уровне развития техники не может служить абсолютным доказательством аутентичности человека.

Однако на практике при аутентификации пользователей в вычислительных системах ограничиваются некоторым не стопроцентным, хотя и достаточно высоким уровнем достовер-

ности доказательства аутентичности человека. Аутентификаторы, которые используются при этом, разделяют на три класса:

- «что-то, что знаю» — к этому типу относятся многоразовые и одноразовые пароли, правила преобразования информации;
- «что-то, что имею» — различные миниатюрные устройства, называемые аппаратными аутентификаторами/ключами;
- «что-то, чем являюсь» — различные биометрические показатели аутентифицируемого.

Класс аутентификаторов называют *фактором*. Если в процедуре аутентификации предусматривается предъявление аутентифицируемым нескольких аутентификаторов, относящихся к разным классам, то такую аутентификацию называют многофакторной. Наибольшее распространение в настоящее время получила **двухфакторная аутентификация**, при которой пользователь предъявляет многоразовый пароль («что-то, что знаю») и аппаратный ключ («что-то, что имею»). Следует заметить, что в некоторых случаях термин «многофакторная аутентификация» служит для обозначения процедур **многоступенчатой аутентификации**, построенных на использовании нескольких аутентификаторов, относящихся к одному и тому же классу. Примером такой процедуры является аутентификация владельца банковского счета при его звонке в банк: сначала его просят назвать несколько букв из его пароля, а затем задают несколько вопросов с заранее согласованными и зафиксированными в базе данных аутентифицирующей организации ответами, например, о его памятном географическом пункте, о марке первого автомобиля и т. п.

# Аутентификация на основе паролей

**Пароль** — это используемая при аутентификации сохраняемая в секрете последовательность символов, либо выбранная пользователем, либо сгенерированная программным или аппаратным средством, либо назначенная администратором.

Пароли бывают одноразовыми и многоразовыми. **Многоразовые пароли**, как это следует из их названия, могут использоваться для доказательства аутентичности многократно. В процедурах аутентификации, основанных на **одноразовых паролях**, аутентифицируемый должен каждый раз предъявлять новое значение пароля. Обычно для генерации одноразовых паролей применяются специальные программы или аппаратные устройства (см. далее).

## Недостатки многоразовых паролей

Механизмы аутентификации на основе многоразовых паролей, обладая простотой и логической ясностью, традиционно являются самым популярным средством аутентификации. Однако им свойственны и недостатки. Это, во-первых, возможность раскрытия и разгадывания паролей, во-вторых, возможность «подслушивания» пароля при его передаче по сети путем анализа сетевого трафика. В-третьих, обладатели паролей могут стать жертвами социального инжиниринга. Так, например, беглый экс-сотрудник Агентства национальной безопасности США Эдвард Сноуден, работая системным администратором разведывательной базы США на Гавайях, использовал логины и пароли более 20 своих сослуживцев, чтобы получить доступ к секретным файлам. Он получал эти данные, объясняя, что они необходимы ему для работы.

Для снижения уровня угрозы раскрытия паролей администраторы сети, как правило, применяют встроенные программные средства, служащие для формирования *политики назначения и использования паролей*: задание максимального и минимального сроков действия пароля, хранение списка уже использованных паролей, управление поведением системы после нескольких неудачных попыток логического входа и т. п.

Многие пользователи пренебрегают угрозами, которые несут в себе легко угадываемые пароли. Так, червь Mumu, поразивший компьютерные сети в 2003 году, искал свои жертвы, подбирая пароли из очень короткого списка: password, passwd, admin, pass, 123, 1234, 12345, 123456 и пустая строка. Такая на удивление примитивная стратегия дала прекрасные (с точки зрения атакующей стороны) результаты — множество компьютеров было взломано.

В списке наиболее популярных паролей, применяемых пользователями Интернета при доступе к веб-серверам, опубликованном в августе 2013 года компанией Google, места в первой десятке занимают имена и даты рождения членов семьи и близких друзей, названия мест рождения, даты свадьбы, клички домашних животных, что-либо связанное с любимой футбольной командой и слово «password». Как видно из приведенного списка, для заинтересованного человека не составит большого труда подобрать эти пароли.

Но даже при выборе менее предсказуемого пароля вы все же рискуете, что он будет разгадан простым перебором всех возможных символов — такой метод часто называют **брутфорс-атакой**[1]. В табл. 27.1 приведены данные, характеризующие стойкость паролей, состоящих из 6 и 8 знаков, сформированных из разных наборов символов. Время определялось для специальной программы подбора паролей, выполняемой на компьютере со средними характеристиками[2]. Обратите внимание, насколько возрастает время подбора пароля при увеличении его длины всего лишь на два знака. Так, при использовании только букв латинского алфавита (строчных и прописных) время подбора пароля из 8 знаков в 3000 раз больше, чем из 6 знаков!

**Таблица 27.1.** Сравнение стойкости паролей

| Множество символов | Количество комбинаций | | Время подбора пароля | |
|---|---|---|---|---|
| | 6 знаков | 8 знаков | 6 знаков | 8 знаков |
| Цифры от 1 до 9 | 1 миллион комбинаций | 100 миллионов комбинаций | Практически мгновенно | 10 секунд |
| 26 только прописных или только строчных букв латинского алфавита | 309 миллионов комбинаций | 200 миллиардов комбинаций | 30 секунд | Менее 6 часов (в 720 раз дольше, чем для 6 знаков) |
| Смесь 52 прописных и строчных букв латинского алфавита | 19 миллиардов комбинаций | 53 триллиона комбинаций | Полчаса | Два месяца (почти в 3000 раз дольше, чем для 6 знаков) |
| Прописные и строчные буквы, цифры и все символы (точка, двоеточие и т. п.) | 782 миллиарда комбинаций | 7,2 квадриллиона комбинаций | 22 часа | 57 лет (примерно в 22 700 раз дольше, чем для 6 знаков) |

[1] Brute-force (англ.) — решать что-либо «в лоб», методом грубой силы.

[2] Данные взяты из статьи http://www.lockdown.co.uk/?pg=combi.

Серьезной проблемой использования многоразовых паролей является их **ручная синхронизация**. В обычной жизни нам требуется не один, а несколько паролей: для входа в сеть предприятия, на котором мы работаем, для доступа к «личному кабинету» провайдера мобильной связи, для доступа к банковскому счету и к другим самым разным интернет-сайтам. Часто во всех этих случаях применяется *один и тот же пароль* (возможно, с небольшими вариациями), потому что у нас нет времени придумывать и, главное, запоминать новый пароль для доступа к новому ресурсу. Выполнив регистрацию на сайте, не заслуживающем доверия, вы сообщаете его владельцам свой пароль, который теперь может быть использован для доступа к другим вашим данным, возможно, имеющим для вас критическое значение.

Слабостью паролей является и процедура реакции аутентифицирующего компьютера на неправильно введенный пароль. На первый взгляд, естественным приемом, направленным на противодействие подбору паролей, кажется *блокирование учетной записи*, с которой было проведено некоторое количество (обычно не более трех) неудачных попыток входа. Однако такой подход дает злоумышленнику прекрасную возможность быстро заблокировать работу предприятия. Действительно, идентификаторы пользователей являются менее защищенной информацией, чем пароли, к тому же они часто легко угадываемы (ADMIN, Guest, IVANOV и т. п.) и их легче подсмотреть, так как они выводятся на экран. Поэтому злоумышленник может легко подобрать имена, выполнить по три неудачные попытки аутентификации для каждой учетной записи, вызвать их блокировку и привести таким образом систему в недоступное состояние. Снятие блокировок с учетных записей может стать серьезной проблемой, если таких записей очень много.

Наряду с паролями существует и другой вариант использования аутентификаторов из класса «что-то, что знаю». Администратор сообщает пользователю заранее безопасным образом некоторое правило, например *правило преобразования* последовательности чисел в другие символы. Во время процедуры аутентификации система выводит на экран случайную последовательность чисел. Пользователь в соответствии с известным только ему и системе правилом преобразует их в другую последовательность символов, которую вводит в качестве пароля. Поскольку система также «знает» правило преобразования, она может проверить правильность введенного пароля. То есть здесь в качестве разделяемого секрета выступает правило преобразования.

## Строгая аутентификация в компьютерной сети на основе многоразовых паролей

Как правило, аутентификация пользователей в компьютерных сетях строится на основе централизованной схемы. На одном из серверов сети поддерживается база данных, в которой хранятся *учетные данные* обо всех пользователях сети. Учетные данные содержат наряду с другой информацией идентификаторы и пароли пользователей. Когда пользователь осуществляет логический вход в сеть, он набирает на клавиатуре компьютера свои идентификатор и пароль, которые передаются на сервер. По идентификатору пользователя в централизованной базе данных, хранящейся на сервере, находится соответствующая запись, из нее извлекается пароль и сравнивается с тем, который ввел пользователь. Если они совпадают, то аутентификация считается успешной, пользователь получает легальный статус и те права, которые определены для него системой авторизации.

Однако такая упрощенная схема имеет большой изъян. А именно при передаче пароля с клиентского компьютера на сервер, выполняющий процедуру аутентификации, этот

пароль может быть перехвачен злоумышленником. Поэтому в разных системах аутентификации применяются разные приемы, чтобы избежать передачи пароля по сети в незащищенном виде.

Аутентификация, в процессе которой используются методы шифрования, а аутентификатор не передается по сети, называется **строгой аутентификацией**.

Рассмотрим пример строгой аутентификации пользователей, реализуемой средствами ОС[1]. Пусть аутентификация пользователей сети выполняется на основе их паролей, хранящихся в зашифрованном виде в централизованной базе SAM (Security Accounts Manager). Пароли зашифровываются с помощью односторонней функции шифрования при занесении их в базу данных во время процедуры создания учетной записи для нового пользователя (рис. 27.1). Введем обозначение для этой односторонней функции — ОФШ1. Таким образом, пароль $P$ хранится в базе данных SAM в виде дайджеста $d(P)$, при этом знание дайджеста не позволяет восстановить исходный текст.

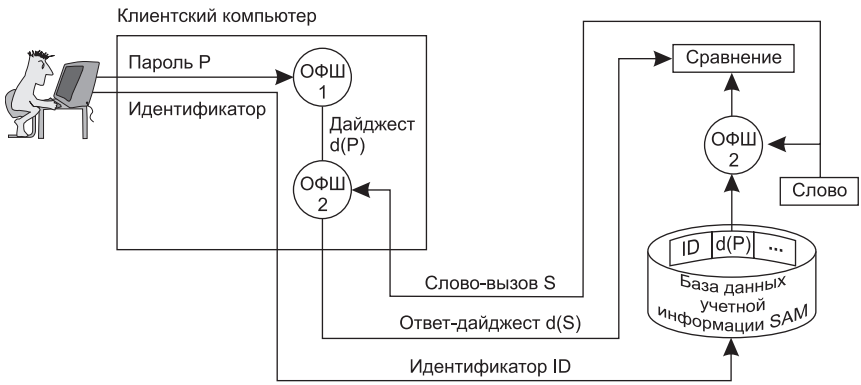



**Рис. 27.1.** Схема сетевой аутентификации на основе многоразового пароля

При логическом входе пользователь локально вводит в свой компьютер имя-идентификатор (ID) и пароль $P$. Клиентская часть подсистемы аутентификации, получив эти данные, передает запрос по сети на сервер, хранящий базу SAM. В этом запросе в открытом виде содержится идентификатор пользователя, но пароль в сеть ни в каком виде *не передается*.

К паролю на клиентской станции применяется та же односторонняя функция ОФШ1, которая была использована при записи пароля в базу данных SAM, то есть динамически вычисляется дайджест пароля $d(P)$.

В ответ на поступивший запрос серверная часть службы аутентификации генерирует случайное число $S$ случайной длины, называемое **словом-вызовом** (challenge). Это слово передается по сети с сервера на клиентскую станцию пользователя. К слову-вызову на клиентской стороне применяется односторонняя функция шифрования ОФШ2. В отличие от функции ОФШ1, функция ОФШ2 является параметрической и получает в качестве параметра дайджест пароля $d(P)$. Полученный в результате ответ $d(S)$ передается по сети на сервер базы SAM.

---

[1] Аутентификация по данной схеме, наряду с другими методами, выполняется в ОС семейства Windows.

Параллельно этому на сервере слово-вызов *S* аналогично шифруется с помощью той же односторонней функции ОФШ2 и дайджеста пароля пользователя *d*(*P*), извлеченного из базы SAM, а затем сравнивается с ответом, переданным клиентской станцией. При совпадении результатов считается, что аутентификация прошла успешно. Таким образом, аутентификация проходит без передачи пароля по каналам связи.

Заметим, что при каждом запросе на аутентификацию генерируется новое слово-вызов, так что перехват ответа *d*(*S*) клиентского компьютера не может быть использован в ходе другой процедуры аутентификации.

## Строгая аутентификация в протоколе CHAP

Другим примером строгой аутентификации может служить *аутентификация по квитированию вызова* (Challenge Handshake Authentication Protocol, CHAP), применяемая в протоколе PPP. Протокол PPP предусматривает два режима аутентификации:

- аутентификация по протоколу PAP, когда пароль передается по линии связи в открытом виде;
- аутентификация по протоколу CHAP, при которой пароль по линии связи не передается и, следовательно, обеспечивается более высокий уровень безопасности.

Рассмотрим применение протокола CHAP при аутентификации удаленных пользователей, подключенных к Интернету по коммутируемому каналу. Здесь аутентифицирующей стороной является сервер провайдера, а аутентифицируемой — клиентский компьютер (рис. 27.2). При заключении договора клиент получает от провайдера пароль (пусть, например, это будет слово parol). Этот пароль хранится в базе данных провайдера в виде дайджеста $Z = d(\text{parol})$, полученного путем применения к паролю односторонней хеш-функции MD5.

В протоколе CHAP предусмотрено четыре типа сообщений: *Success* (успех), *Challenge* (вызов), *Response* (ответ), *Failure* (ошибка).

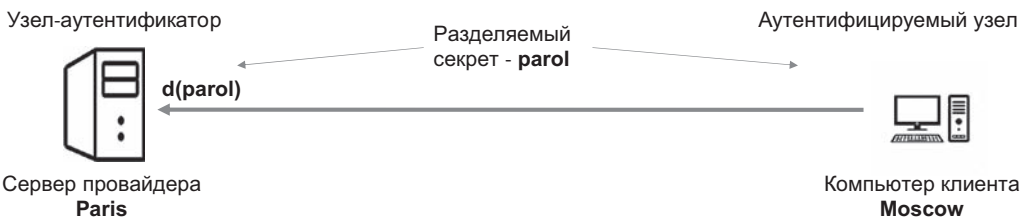



**Рис. 27.2.** Аутентификация по протоколу CHAP

Аутентификация выполняется в следующей последовательности:

1. Пользователь-клиент активизирует некоторую программу удаленного доступа к серверу провайдера, вводя назначенные ему имя и пароль. Имя (на рисунке это «Moscow») передается по сети провайдеру в составе запроса на соединение, но пароль не передается в сеть ни в каком виде.
2. Сервер провайдера, получив запрос от клиента, генерирует псевдослучайное слово-вызов (пусть это будет слово «goodmorning») и передает его клиенту вместе со значением, идентифицирующим сообщение в рамках данного сеанса (ID), и собственным именем

(здесь — «Paris»). Это сообщение типа *Challenge*: (ID, goodmorning), Paris. Для защиты от перехвата ответа аутентификатор должен использовать разные значения слова-вызова при каждой процедуре аутентификации.

3. Программа клиента, получив этот пакет, извлекает из него слово-вызов, добавляет к нему идентификатор и вычисленный локально дайджест $Z = d(\text{parol})$, а затем вычисляет с помощью все той же функции MD5 дайджест $Y = d\{(\text{ID, goodmorning, } d(\text{parol})\}$ от всех этих трех значений. Результат клиент посылает серверу провайдера в пакете *Response*.
4. Сервер провайдера сравнивает полученный по сети дайджест $Y$ с тем значением, которое он получил, локально применив ту же хеш-функцию к набору аналогичных компонентов, хранящихся в его памяти.
5. Если результаты совпадают, то аутентификация считается успешной и аутентификатор посылает партнеру пакет *Success*.

Способ аутентификации, при котором многоразовые пароли пользователей хранятся в базе данных сервера в виде дайджестов, кажется вполне безопасным. Ведь если злоумышленник и сможет получить к ним доступ, то он даже теоретически не сможет восстановить исходное значение паролей по дайджесту. Однако создатель первого червя Роберт Моррис решил эту проблему. Он разработал довольно простую программу, которая генерировала возможные варианты паролей как используя слова из словаря, так и последовательным перебором символов. Для каждого сгенерированного слова вычислялся дайджест, который затем сравнивался с дайджестами из файла паролей. Удивительно, но такая стратегия оказалась весьма эффективной — хакеру удалось завладеть несколькими паролями.

## Аутентификация на основе аппаратных аутентификаторов

Алгоритмы аутентификации, основанные на многоразовых паролях, не очень надежны. Пароли можно подсмотреть, разгадать или просто украсть. Более надежными оказываются схемы на основе программных или аппаратных **генераторов одноразовых паролей** (рис. 27.3).

Независимо от того, какую реализацию системы аутентификации на основе одноразовых паролей выбирает пользователь, он, как и в системах аутентификации с применением многоразовых паролей, сообщает системе свой идентификатор, однако вместо того чтобы вводить каждый раз один и тот же пароль, он указывает последовательность цифр, сообщаемую ему аппаратным или программным ключом. Через определенный небольшой период времени ключ генерирует другую последовательность — новый пароль. Сервер аутентификации проверяет введенную последовательность и разрешает пользователю осуществить логический вход. Сервер аутентификации может представлять собой отдельное устройство, выделенный компьютер или программу, выполняемую на обычном сервере.

Следует иметь в виду, что, как правило, системы аутентификации на основе одноразовых паролей рассчитаны на проверку *удаленных*, а не локальных пользователей.

Рассмотрим схему использования аппаратного генератора одноразовых паролей, в основе которой лежит **синхронизация по времени**. Этот популярный алгоритм аутентификации

был разработан компанией Security Dynamics. Идея метода состоит в том, что аппаратный ключ и аутентифицирующий сервер по одному и тому же алгоритму вычисляют некоторое значение — одноразовый пароль. Алгоритм имеет два параметра:

- *разделяемый секретный ключ*, представляющий собой 64-разрядное число, уникально назначаемое каждому пользователю и хранящееся как в аппаратном ключе, так и в базе данных сервера аутентификации;
- *значение текущего времени*.


*а* *б*

**Рис. 27.3.** Аппаратные ключи, генерирующие одноразовые пароли: *а* — ключ клиентов банка Barklays для доступа к своим счетам; *б* — аппаратный ключ компании SecurID

Если вычисленные значения совпадают, то аутентификация считается успешной.

Итак, пусть удаленный пользователь пытается совершить логический вход в систему с персонального компьютера (рис. 27.4). Аутентифицирующая программа предлагает ему ввести его личный персональный номер (PIN), состоящий из четырех десятичных цифр (на рисунке — 2360), а также одноразовый пароль — шесть цифр случайного числа, отображаемого в тот момент на дисплее аппаратного ключа (на рисунке — 112511). На основе PIN-кода сервер извлекает из базы данных информацию о пользователе, а именно — его секретный ключ. Затем сервер выполняет вычисления по тому же алгоритму, которой заложен в аппаратном ключе, используя в качестве параметров секретный ключ и значение текущего времени, проверяя, совпадает ли сгенерированное число с числом, введенным пользователем. Если они совпадают, то пользователю разрешается логический вход.

Потенциальной проблемой этой схемы является *временная синхронизация* сервера и аппаратного ключа. Вопрос согласования часовых поясов решается просто, но гораздо сложнее обстоит дело с постепенным рассогласованием внутренних часов сервера и аппаратного ключа, тем более что потенциально аппаратный ключ может работать несколько лет. Компания Security Dynamics решает эту проблему двумя способами. Во-первых, при производстве аппаратного ключа измеряется отклонение частоты его таймера от номинала. Далее эта величина учитывается в виде параметра алгоритма сервера. Во-вторых, сервер отслеживает коды, генерируемые конкретным аппаратным ключом, и если таймер данного ключа постоянно спешит или отстает, то сервер динамически подстраивается под него.

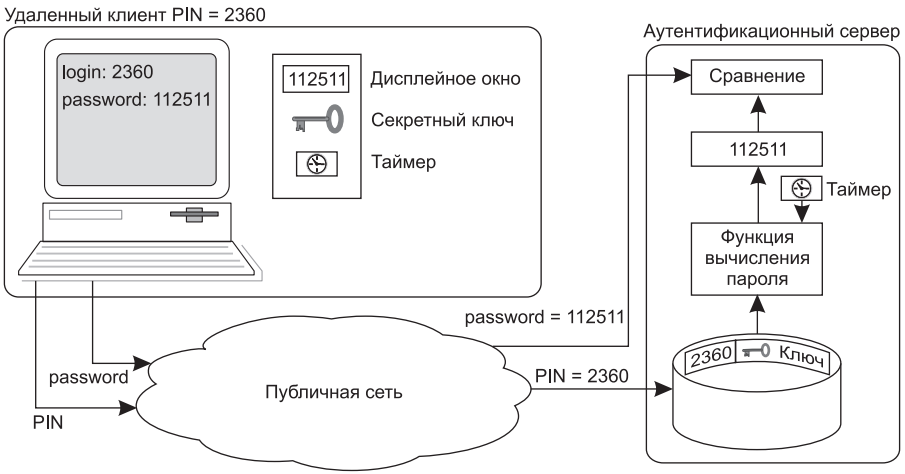



**Рис. 27.4.** Аутентификация на основе временной синхронизации

Существует, однако, еще одна проблема, связанная со схемой временной синхронизации. Одноразовый пароль, генерируемый аппаратным ключом, действителен в течение некоторого интервала времени (от нескольких десятков секунд до нескольких десятков минут), то есть в течение этого времени одноразовый пароль, в сущности, является многоразовым. Поэтому теоретически возможно, что очень проворный хакер сможет перехватить PIN-код и одноразовый пароль с тем, чтобы также получить доступ в сеть в течение этого интервала.

Схема временной синхронизации *не требует наличия компьютера* на стороне аутентифицируемого — для этих целей можно ограничиться простым терминалом или факсом. Пользователи могут даже вводить свой пароль с телефонной клавиатуры, когда звонят в сеть для получения голосовой почты.

## Аутентификация по схеме «запрос-ответ»

Другая схема применения аппаратных ключей, называемая часто **запрос-ответ**, основана на идее, очень сходной с идеей строгой аутентификации, рассмотренной в предыдущем разделе. В том и другом случаях применяется слово-вызов. Когда пользователь пытается осуществить логический вход, аутентификационный сервер передает ему запрос в виде некоторого случайного числа (см. слово-вызов на рис. 27.5). Аппаратный ключ пользователя зашифровывает это случайное число (например, по алгоритму DES) и секретный ключ пользователя. Секретный ключ пользователя хранится в базе данных сервера и в памяти аппаратного ключа. В зашифрованном виде слово-вызов возвращается на сервер. Сервер, в свою очередь, также зашифровывает сгенерированное им самим случайное число с помощью того же алгоритма шифрования и того же секретного ключа пользователя, а затем сравнивает результат с числом, полученным от аппаратного ключа. Как и в методе временной синхронизации, в случае совпадения этих двух чисел пользователю разрешается вход в сеть.

Механизм со словом-вызовом имеет свои ограничения — он обычно требует наличия компьютера на каждом конце соединения, так как аппаратный ключ должен иметь возможность как получать, так и отправлять информацию. Схема «запрос-ответ» уступает схеме временной синхронизации по простоте использования. Для логического входа

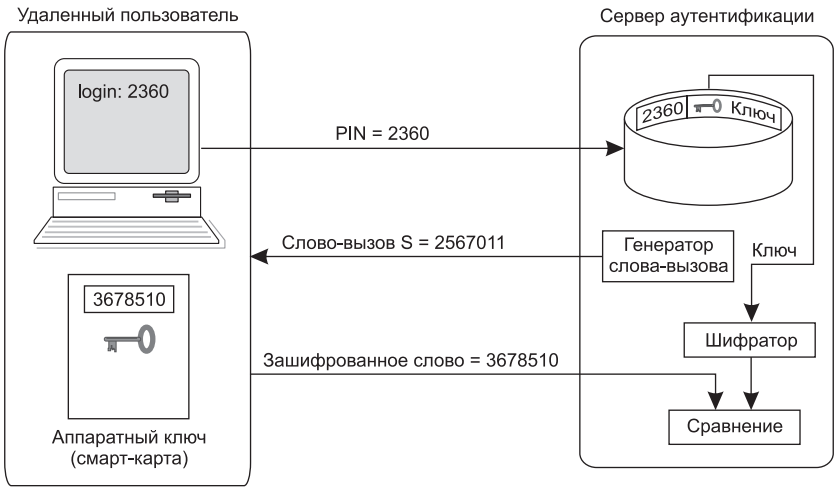



**Рис. 27.5.** Аутентификация по схеме «запрос-ответ»

с помощью схемы временной синхронизации пользователю достаточно набрать 10 цифр. Схемы «запрос-ответ» могут потребовать от пользователя выполнения большего числа ручных действий. В некоторых схемах «запрос-ответ» пользователь должен сам ввести секретный ключ, а затем набрать на клавиатуре компьютера полученное с помощью аппаратного ключа зашифрованное слово-вызов.

## Аутентификация информации. Электронная подпись

Аутентификация данных включает:

- подтверждение *целостности* хранящихся и переданных по сети данных и программ, то есть установление факта того, что они не подвергались модификации;
- доказательство *авторства* сообщения (документа, программы), в том числе и для недопущения отказа от авторства;
- доказательство *легальности* приобретения программного обеспечения.

Все эти задачи в той или иной мере могут быть решены посредством электронной подписи.

Согласно терминологии, утвержденной Международной организацией по стандартизации (ISO), под термином «**электронная (цифровая) подпись**» понимаются методы, позволяющие устанавливать подлинность автора сообщения (документа) при возникновении спора относительно авторства. Основная область применения цифровой подписи — финансовые документы, сопровождающие электронные сделки, документы, фиксирующие международные договоренности, и т. п. Подчеркнем, что электронная подпись не ставит задачи обеспечения конфиденциальности сообщений.

Хотя для получения подписи могут использоваться симметричные алгоритмы, более распространенными являются алгоритмы на основе открытого и закрытого ключей. На рис. 27.6 показана схема формирования цифровой подписи по алгоритму RSA. Каждый

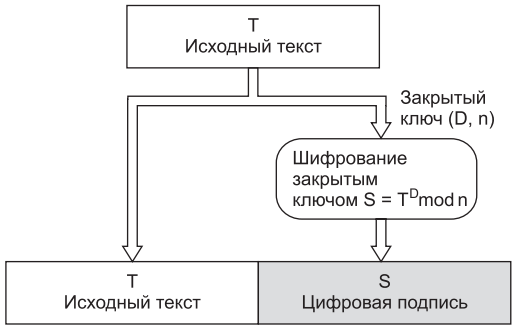



**Рис. 27.6.** Схема формирования цифровой подписи по алгоритму RSA

пользователь сети имеет свой закрытый ключ ($D$, $n$), необходимый для формирования подписи, а соответствующий этому секретному ключу открытый ключ ($E$, $n$), предназначенный для проверки подписи, известен всем другим пользователям сети. Подписанное сообщение состоит из двух частей: незашифрованной части, в которой содержится исходный текст $T$, и зашифрованной части, представляющей собой цифровую подпись. Цифровая подпись $S$ вычисляется с помощью закрытого ключа ($D$, $n$) по формуле

$$S = T^D \bmod n.$$

Сообщение посылается в виде пары ($T$, $S$). Каждый пользователь, имеющий соответствующий открытый ключ ($E$, $n$), получив сообщение, отделяет открытую часть $T$, расшифровывает цифровую подпись $S$ и проверяет равенство

$$T = S^E \bmod n.$$

Если результат расшифровки цифровой подписи совпадает с открытой частью сообщения, то считается, что документ подлинный, не претерпел никаких изменений в процессе передачи, а автором его является именно тот человек, который передал свой открытый ключ получателю.

К недостаткам данного алгоритма можно отнести то, что длина подписи в этом случае равна длине сообщения, что не всегда удобно. Для уменьшения «длины» электронной подписи вместо $S = T^D \bmod n$ используются формула

$$S = (H(T))^D \bmod n.$$

Здесь $H(T)$ — хеш-функция, преобразующая исходное сообщение в короткий дайджест. В этом случае получатель сообщения ($T$, $S$) должен сначала применить к открытому тексту $T$ хеш-функцию $H$ и получить дайджест $H(T)$, а затем приступить к расшифровке подписи $S$ открытым ключом. Если расшифрованная подпись совпадает с дайджестом, то авторство сообщения доказано. Использование хеш-функций дает выигрыш не только в объеме сообщения, но и во времени получения электронной подписи.

Если помимо проверки аутентичности документа, обеспечиваемой цифровой подписью, надо обеспечить его конфиденциальность, то после применения к тексту цифровой подписи перед передачей его по каналу связи выполняют совместное шифрование исходного текста и цифровой подписи любым способом шифрования, согласованным отправителем и получателем.

## Аутентификация на основе цифровых сертификатов

Аутентификация с применением цифровых сертификатов является альтернативой применению паролей и представляется естественным решением в условиях, когда число пользователей сети (пусть и потенциальных) измеряется миллионами. В таких обстоятельствах процедура предварительной регистрации пользователей, связанная с назначением и хранением их паролей, становится крайне обременительной, опасной, а иногда и просто нереализуемой. При наличии сертификатов сеть, которая дает пользователю доступ к своим ресурсам, не хранит никакой информации о своих пользователях — они ее предоставляют сами в своих запросах в виде сертификатов, удостоверяющих личность пользователей. Сертификаты выдаются специальными уполномоченными сертифицирующими организациями (СО) — **центрами сертификации** (Certificate Authority, CA), или **удостоверяющими центрами**. Поэтому задача хранения секретной информации (закрытых ключей) возлагается на самих пользователей, что делает это решение гораздо более масштабируемым, чем вариант с централизованной базой паролей.

**Сертификат** представляет собой электронную форму, в которой содержится следующая информация:

- открытый ключ владельца данного сертификата;
- сведения о владельце сертификата, такие, например, как имя, адрес электронной почты, наименование организации, в которой он работает, и т. п.;
- наименование сертифицирующей организации, выдавшей данный сертификат;
- электронная подпись сертифицирующей организации, то есть зашифрованные закрытым ключом этой организации данные, содержащиеся в сертификате.

Использование сертификатов основано на предположении, что сертифицирующих организаций немного и их открытые ключи широкодоступны, например, из публикаций в журналах.

Сертификаты могут быть представлены в трех формах (рис. 27.7):

- *в открытой форме* сертификат содержит всю информацию в незашифрованном виде;
- *в форме из двух частей* — открытой, содержащей всю информацию в незашифрованном виде, и закрытой, представляющей собой ту же информацию, но зашифрованную закрытым ключом сертифицирующей организации;
- *в форме из трех частей* — во-первых, открытой, во-вторых, зашифрованной закрытым ключом сертифицирующей организации, в-третьих, части, представляющей собой первые две части, зашифрованные закрытым ключом владельца.

Когда пользователь хочет подтвердить свою личность, он предъявляет свой сертификат в двух формах: открытой (то есть такой, в которой он получил его в сертифицирующей организации) и зашифрованной (с применением своего закрытого ключа). Сторона, проводящая аутентификацию, берет из незашифрованного сертификата открытый ключ пользователя и расшифровывает с его помощью зашифрованного сертификата. Совпадение результата с открытым сертификатом подтверждает, что предъявитель действительно является владельцем закрытого ключа, соответствующего указанному открытому.

Затем с помощью известного открытого ключа указанной в сертификате организации проводится расшифровка подписи этой организации в сертификате. Если в результате полу-

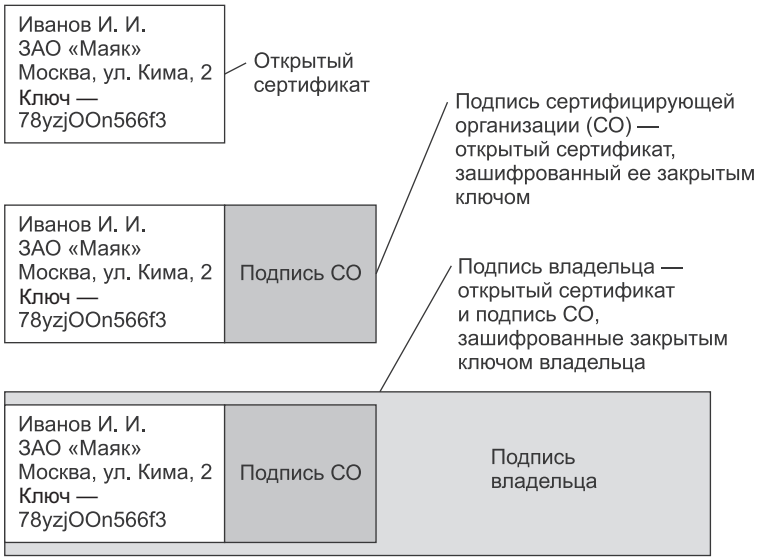



**Рис. 27.7.** Формы представления цифрового сертификата

чается тот же сертификат с тем же именем пользователя и его открытым ключом, значит, он действительно прошел регистрацию в сертификационном центре, является тем, за кого себя выдает, и указанный в сертификате открытый ключ действительно принадлежит ему.

Сертификаты можно применять не только для аутентификации, но и *для предоставления прав доступа к ресурсам*. Для этого в сертификат могут вводиться дополнительные поля, в которых указывается принадлежность его владельцев к той или иной категории пользователей. Эта категория назначается сертифицирующей организацией в зависимости от условий, на которых выдается сертификат. Например, организация, поставляющая через Интернет на коммерческой основе информацию, может выдавать сертификаты определенной категории пользователям, оплатившим годовую подписку на некоторый бюллетень. В этом случае веб-сервер будет предоставлять доступ к страницам бюллетеня только пользователям, предъявившим сертификат данной категории.

Подчеркнем тесную связь открытых ключей с сертификатами. Сертификат является удостоверением не только личности, но и принадлежности открытого ключа.

*Цифровой сертификат устанавливает и гарантирует соответствие между открытым ключом и его владельцем.*

Это предотвращает угрозу подмены открытого ключа. Если некоторый абонент *А* получает по сети сертификат от абонента *Б*, то он может быть уверен, что открытый ключ, содержащийся в сертификате, гарантированно принадлежит абоненту *Б*, адрес и другие сведения о котором содержатся в этом сертификате. Это значит, что абонент *А* может без опасений использовать открытый ключ абонента *Б* для секретных посланий в адрес последнего.

При наличии сертификатов отпадает необходимость хранить на серверах корпораций списки пользователей с их паролями, вместо этого достаточно иметь на сервере список

имен и открытых ключей сертифицирующих организаций. Может также понадобиться некоторый механизм для установления соответствия категорий владельцев сертификатов традиционным группам пользователей, чтобы можно было в неизменном виде задействовать механизмы управления избирательным доступом большинства операционных систем или приложений.

Сертификат является средством аутентификации пользователя при его обращении к сетевым ресурсам, роль аутентифицирующей стороны играют при этом информационные серверы корпоративной сети или Интернета. В то же время и сама процедура получения сертификата также включает этап аутентификации, когда аутентификатором выступает сертифицирующая организация. Для получения сертификата клиент должен сообщить сертифицирующей организации свой открытый ключ и те или иные сведения, удостоверяющие его личность. Все эти данные клиент может отправить по электронной почте или принести на съемном носителе лично. Перечень необходимых данных зависит от типа получаемого сертификата. Сертифицирующая организация проверяет доказательства подлинности, помещает свою цифровую подпись в файл, содержащий открытый ключ, и посылает сертификат обратно, подтверждая факт принадлежности данного конкретного ключа конкретному лицу. После этого сертификат может быть встроен в любой запрос на использование информационных ресурсов сети (рис. 27.8).

Практически важным является вопрос о том, кто имеет право выполнять функции сертифицирующей организации. Во-первых, задачу обеспечения своих сотрудников сертификатами может взять на себя само предприятие. В этом случае упрощается процедура первичной аутентификации при выдаче сертификата. Предприятия достаточно осведомлены о своих сотрудниках, чтобы брать на себя задачу подтверждения их личности. Для автоматизации процесса генерации, выдачи и обслуживания сертификатов предприятия могут использовать готовые программные продукты. Например, компания Netscape Communications выпустила сервер сертификатов, который организации могут у себя устанавливать для выпуска своих сертификатов. Во-вторых, эти функции могут выполнять независимые центры по выдаче сертификатов, работающие на коммерческой основе, например сертифицирующий центр компании Verisign. Сертификаты компании Verisign выполнены в соответствии с международным стандартом X.509 и используются во многих продуктах, ориентированных на защиту данных, в том числе в популярном протоколе защищенного канала SSL. Любой желающий может обратиться с запросом на получение сертификата на веб-сервер этой компании.

Механизм получения пользователем сертификата хорошо автоматизируется в сети в модели «клиент-сервер», когда браузер исполняет роль клиента, а в сертифицирующей организации установлен специальный сервер выдачи сертификатов. Браузер генерирует для пользователя пару ключей, оставляет закрытый ключ у себя и передает частично заполненную форму сертификата серверу. Чтобы неподписанный еще сертификат нельзя было подменить при передаче по сети, браузер ставит свою электронную подпись, зашифровывая сертификат выработанным закрытым ключом. Сервер сертификатов подписывает полученный сертификат, фиксирует его в своей базе данных и возвращает его каким-либо способом владельцу. Очевидно, что при этом может выполняться еще и неформальная процедура подтверждения пользователем своей личности и права на получение сертификата, требующая участия оператора сервера сертификатов. Это могут быть доказательства оплаты услуги, доказательства принадлежности к той или иной организации — все случаи жизни предусмотреть и автоматизировать нельзя. После получения сертификата браузер

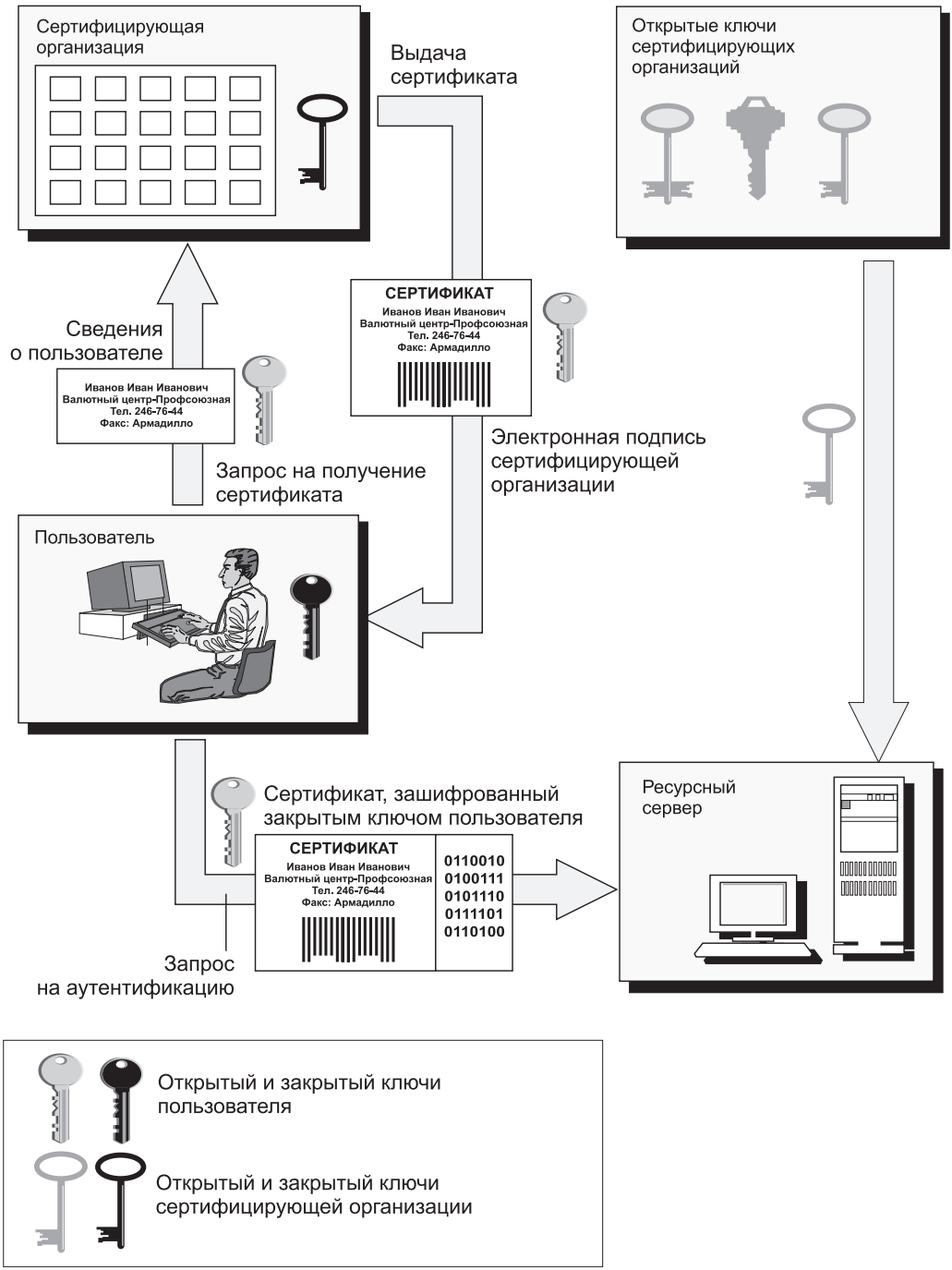



**Рис. 27.8.** Схема аутентификации пользователей на основе сертификатов

сохраняет его вместе с закрытым ключом и использует при аутентификации на тех серверах, которые поддерживают такой процесс. В настоящее время существует большое количество протоколов и продуктов, применяющих сертификаты. В частности, практически все браузеры и операционные системы реализуют поддержку сертификатов.

Несмотря на активное использование технологии цифровых сертификатов во многих системах безопасности, эта технология еще не решила целый ряд серьезных проблем. Это прежде всего поддержание базы данных о выпущенных сертификатах. Сертификат выдается не навсегда, а на некоторый вполне определенный срок. По истечении срока годности сертификат должен либо обновляться, либо аннулироваться. Кроме того, необходимо предусмотреть возможность досрочного прекращения полномочий сертификата. Все заинтересованные участники информационного процесса должны быть вовремя оповещены о том, что некоторый сертификат уже недействителен. Для этого сертифицирующая организация должна оперативно поддерживать список отозванных сертификатов.

Имеется также ряд проблем, связанных с тем, что сертифицирующие организации существуют не в единственном числе. Все они выпускают сертификаты, но даже если эти сертификаты соответствуют единому стандарту (сейчас это, как правило, стандарт X.509), то все равно остаются нерешенными многие вопросы. Все ли сертифицирующие центры заслуживают доверия? Каким образом можно проверить полномочия того или иного сертифицирующего центра? Можно ли создать иерархию сертифицирующих центров, когда сертифицирующий центр, стоящий выше, мог бы сертифицировать центры, расположенные в иерархии ниже? Как организовать совместное использование сертификатов, выпущенных разными сертифицирующими организациями?

Для решения этих и многих других проблем, возникающих в системах, использующих технологии шифрования с открытыми ключами, оказывается необходимым комплекс программных средств и методик, называемый *инфраструктурой с открытыми ключами* (Public Key Infrastructure, PKI).

Рассмотренная схема аутентификации включает три основных элемента — это пользователи, цифровые сертификаты и центры сертификации. Чтобы данная схема работала надежно и эффективно, в нее должны быть включены дополнительные элементы, которые в совокупности с основными и образуют PKI.

В число дополнительных элементов может входить, например, регистрационный центр (Registration Authority), который служит посредником между пользователем, запросившим сертификат, и центром сертификации. Пользователь обычно обращается к регистрационному центру с помощью веб-интерфейса и сообщает данные о себе. Регистрационный центр проверяет эту информации и в случае ее подлинности передает данные о пользователе, подписанные собственным закрытым ключом, центру сертификации. Регистрационный центр может обслуживать несколько центров сертификации. При отсутствии регистрационного центра его функции выполняет центр сертификации.

Другим типом дополнительных элементов PKI являются разнообразные хранилища сертификатов, содержащие информацию о действующих, отозванных и истекших сертификатах.

## Аутентификация программных кодов

Электронная подпись и сертификаты могут применяться для доказательства аутентичности (подлинности) программ. Пользователю важно быть уверенным, что программа,

которую он загрузил с какого-либо сервера Интернета, действительно содержит коды, разработанные определенной компанией. Компания Microsoft предложила для этих целей технологию **аутентикода** (authenticode).

Организация, желающая подтвердить свое авторство на программу, должна встроить в распространяемый код так называемый **подписывающий блок** — аутентикод (рис. 27.9). Этот блок состоит из двух частей. Первая часть — это сертификат организации-разработчика данной программы, полученный обычным образом от какого-либо сертифицирующего центра. Вторую часть образует зашифрованный дайджест, полученный в результате применения хеш-функции к распространяемому коду. Шифрование дайджеста выполняется с помощью закрытого ключа организации.

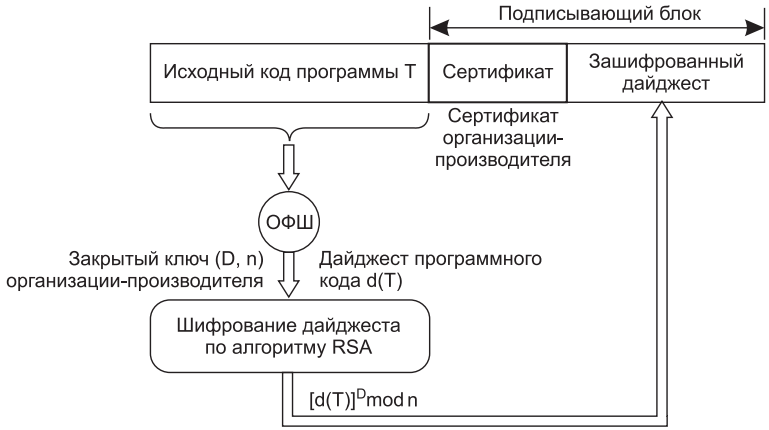



**Рис. 27.9.** Схема получения аутентикода

Компания-разработчик может потребовать от пользователя программы доказательство легальности ее приобретения — допустим, запросить регистрационный номер программы (Product ID или Serial Number), называемый также *лицензионным ключом активации*. Обычно этот номер пишется на отдельном бланке, прилагаемом к поставляемой программе, наносится на упаковку или высылается по электронной почте при покупке программы через Интернет.

Другим способом доказательства легальности приобретения и законности использования программных продуктов являются миниатюрные электронные устройства — *электронные замки*, подобные уже рассмотренным нами аппаратным аутентификаторам. Эти устройства поставляются вместе с защищаемыми от нелегального использования программами. Перед запуском программы электронный замок должен быть подключен к компьютеру, например, через USB-порт. Инициирующий блок программы обращается к данному устройству с запросом и, получив «правильный» ответ, начинает работать. Если же ожидаемый ответ не поступает, то выполнение программы блокируется. Таким образом, электронный замок действует как специфический аутентификатор пользователя, доказывающий то, что он является законным владельцем программы.

### (S) *Биометрическая аутентификация*

# Аутентификация пользователей ОС

Существует две принципиально отличные схемы аутентификации, реализуемые операционными системами и специальными сетевыми службами. В одной из них, которую мы будем называть **локальной системой аутентификации**, ОС работает в пределах одного компьютера: она задействует базу аутентификационных данных пользователей, причем результаты аутентификации могут применяться только для доступа к ресурсам этого компьютера.

По другой схеме работает так называемая **система аутентификации домена**: она базируется на центральной базе аутентификационных данных пользователей группы компьютеров (*домена аутентификации*), хранящейся на одном из серверов сети, и результаты аутентификации служат для доступа к ресурсам данного домена.

Локальная система аутентификации ОС работает при *логическом входе пользователя* как с терминала *компьютера*, так и *через* сеть. Первый вариант называют **интерактивным** логическим входом, второй — **удаленным** (сетевым, или неинтерактивным). Понятно, что при удаленном логическом входе риски безопасности выше, так как аутентификационные данные передаются через сеть — корпоративную или Интернет — и их легче перехватить. Перехват данных аутентификации представляет собой угрозу даже в случае строгой аутентификации, когда пароль не передается в открытом виде по сети или же не передается вовсе — при наличии большого массива аутентификационных данных, то есть данных перехватов большого количества процедур входа одного и того же пользователя, пароль может быть вычислен по имеющимся результатам его ввода.

Хотя локальные системы аутентификации ОС поддерживают все распространенные методы аутентификации — на основе многоразовых и одноразовых паролей (аппаратных и программных), биометрических данных и цифровых сертификатов, — основным методом аутентификации пользователей является метод на базе *многоразового пароля*. Практически все универсальные ОС (MS Windows, Unix/Linux и MAC OS X) предлагают этот метод по умолчанию.

*Одноразовые пароли*, обеспечивающие более надежную аутентификацию, чем многоразовые, чаще применяются при удаленном логическом входе через соединения VPN с шифрованием информации, где передача аутентификационной информации идет через Интернет и, следовательно, риск ее перехвата и взлома особенно велик. Одноразовые пароли могут сочетаться с многоразовыми при двухфакторной аутентификации.

Аутентификация на основе *сертификатов* применяется чаще всего для удаленно работающих пользователей, которые предъявляют сертификаты, выданные сервером сертификации, организации, к которой принадлежит пользователь.

Аутентификация на основе *биометрических данных* штатными средствами универсальных ОС обычно не поддерживается, так как их повышенная надежность нужна только в особо защищенных системах. Кроме того, для поддержки биометрической аутентификации требуется приобрести и установить соответствующее специальное программное обеспечение и специальные устройства.

Необходимо отличать процедуру аутентификации *пользователя ОС* от процедуры аутентификации *пользователя серверной части* некоторого приложения. Многие серверные приложения имеют собственную систему аутентификации пользователей, никак не связанную с системой аутентификации ОС, под управлением которой они работают. Неза-

висимость системы аутентификации сервера приложений имеет как положительные, так и отрицательные стороны. Преимуществом здесь является разграничение по умолчанию пользователей ОС, которым потенциально может понадобиться доступ к любому ресурсу компьютера, и пользователей некоторого сервиса, которым нужен доступ только к ресурсам, относящиеся к данному сервису, например, только к файлам, хранящимся в корневом каталоге FTP-сервера. К недостаткам же можно отнести низкую защищенность протокола аутентификации некоторых приложений, а также необходимость запоминания двух различных имен и паролей для одного и того же пользователя, ошибки администраторов ОС и сервисов из-за дублирования учетных записей и т. п.

# Технологии управления доступом и авторизации

После того как пользователь, пройдя аутентификацию, доказал свою легальность, ему предоставляется некоторый набор прав по отношению к защищаемым системой ресурсам.

> Наделение легальных пользователей правами доступа к ресурсам называется **авторизацией**. Процедура приведения авторизации в действие называется **управлением доступом** (access control).

Если, например, субъект пытается использовать ресурс с запрещенным для него типом доступа, то механизм управления доступом должен отклонить эту попытку и, возможно, уведомить систему об этом инциденте с целью генерации сигнала тревоги.

## Формы представления ограничений доступа

При решении задачи управления доступом необходимо руководствоваться *принципом минимальных привилегий*. В соответствии с ним каждому субъекту в системе должен быть назначен минимально возможный набор прав, достаточный для решения именно тех задач, на которые он уполномочен. Применение этого принципа ограничивает те возможные потери, которые могут быть нанесены в результате неумышленных ошибок или неавторизованных действий.

Ограничение доступа может задаваться в форме **правил**. На основании правила система управления доступом в любой момент времени *динамически* решает вопрос о предоставлении или непредоставлении доступа. Правило может строиться с учетом различных факторов, в том числе длительности сеанса связи (ограничение доступа по времени использования ресурса), возраста человека (ограничение для детей на доступ к некоторым сайтам), времени суток (разрешение на использование ресурсов и сервисов Интернета только в рабочие часы). Популярной мерой ограничения доступа в Интернет является *капча* (capcher) — субъекту, обратившемуся с запросом к ресурсу, предлагается ввести символы, выведенные на экран в таком искаженном виде, в котором их сможет распознать только человек, — таким образом исключается доступ к ресурсам искусственных субъектов (программных систем).

Для ограничения доступа используются также **контентно**- и **контекстно-зависимые правила**. Например, в компании может быть принято правило, в соответствии с которым некоторым категориям пользователей запрещается доступ к документам, содержащим те или иные ключевые слова или фразы: «для ограниченного использования», «секретно», слова, обозначающие кодовое название проекта, и др. Ограничения могут быть наложены и на доступ к ресурсам, содержащим текст на иностранном языке. Это примеры контентно-зависимых правил. В контекстно-зависимых правилах принимаются во внимание некоторые факторы, характеризующие текущее состояние среды и/или предысторию (контекст) запроса. Простейшим правилом такого рода является отказ в доступе пользователю, который сделал подряд три безуспешные попытки аутентификации.

Эффективным средством ограничения доступа является **конфигурирование пользовательского интерфейса**. Таким путем пользователь может быть лишен возможности не только обращаться к тем или иным каталогам и файлам, но и возможности видеть на своем экране часть структуры файловой системы, доступ к которой ему запрещен. Администратор может настроить систему меню пользовательского интерфейса так, что некоторые пункты этих меню не будут выводиться на экран, что исключит принципиальную возможность запуска пользователем части функций.

**Матрица прав доступа** является универсальной и наиболее гранулированной (то есть тонко дифференцированной) формой представления политики контроля доступа, она директивно, «в лоб» описывает для каждого пользователя набор конкретных операций, которые ему разрешается выполнять по отношению к каждому объекту (рис. 27.10).

| Субъекты / Объекты | User 1 | User 2 | User 3 |
|---|---|---|---|
| File 1 | Читать и записывать | Читать и записывать | Читать |
| File 2 | Читать и записывать | Нет доступа | Читать |
| File 3 | Записывать | Нет доступа | Читать |

**Рис. 27.10.** Матрица прав доступа

Матричный способ описания прав доступа теоретически дает возможность отразить все многообразие отношений субъектов и объектов системы для всех возможных сочетаний {субъект, объект, назначенные права}. Однако этот универсальный способ представления, как правило, очень сложно реализовать на практике из-за громоздкости матрицы, учитывая огромное число элементов — как субъектов, так и объектов — в вычислительной системе.

Особенностью матрицы прав доступа является не только ее большая размерность, но и наличие большого числа *нулевых* элементов. Такой вид матриц в математике называют разряженными. Нулевое значение здесь говорит о том, что для данного сочетания {субъект, объект} права доступа не определены, а именно такие сочетания составляют большинство в реальных системах. Свойство разреженности матрицы может быть использовано для более компактного представления правил доступа.

С каждым объектом можно связать **список управления доступом** (Access Control List, ACL), в котором указаны только те субъекты (пользователи), которые имеют разрешения на доступ к данному объекту (файлу). Ясно, что количество субъектов в данном списке будет значительно меньше общего числа субъектов системы. Такие списки должны быть созданы для всех ресурсов. Способ описания прав доступа набором списков столь же универсальный и гибкий, как матрица, но вместе с тем имеет и более компактный вид, поскольку не включает пустые элементы матрицы (рис. 27.11, *а*).

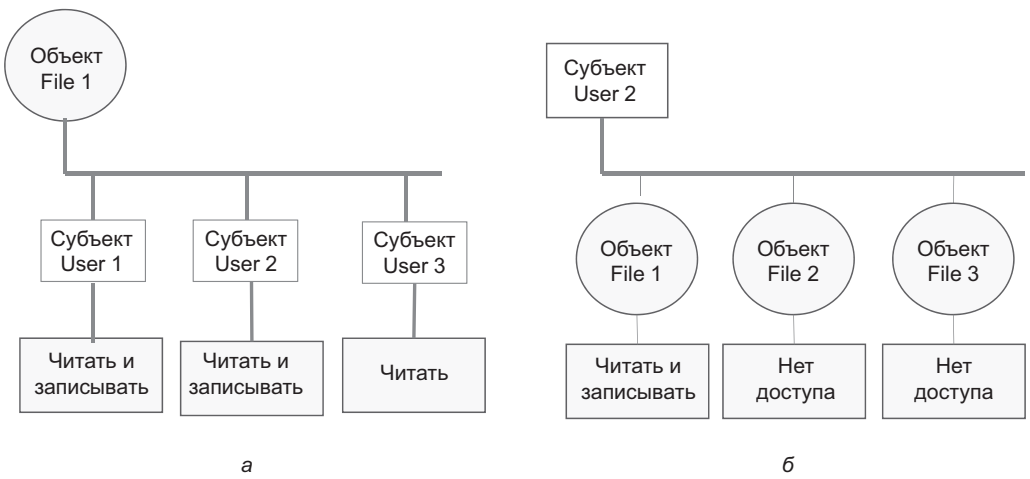



**Рис. 27.11.** *а* — список управления доступом к объекту; *б* — список разрешений пользователя User 2

Права доступа могут быть определены как по отношению к ресурсам, так и по отношению к пользователям. В последнем случае его называют **списком разрешений** (capability). На рис. 27.11, *б* показан список разрешений, которые имеет пользователь User 2 по отношению к ресурсам File 1, File 2 и File 3.

Очевидно, что совокупность списков управления доступом ко всем ресурсам системы несет ту же самую информацию, что и совокупность списков разрешений для всех пользователей, так как и те и другие являются разными проекциями одной и той же матрицы. В одних реализациях систем управления доступом (например, в большинстве операционных систем) применяются ограничения, заданные для объекта (ACL), а в других (например, в некоторых расширениях системы Kerberos, включающих авторизацию) — ограничения для субъекта (списки разрешений).

Другим способом «сжатия» матрицы является определение прав доступа *для групп субъектов* по отношению к *группам объектов*. Такое представление возможно, когда многие элементы матрицы имеют одинаковое значение, что соответствует ситуации в реальной системе, когда некоторая группа пользователей имеет одинаковые права. Это дает возможность компактно описать права доступа с помощью матрицы меньшей размерности.

В некоторых случаях, если существует простое *правило* определения прав доступа, хранение матрицы вообще не требуется, поскольку значения ее элементов могут вычисляться системой управления доступом *динамически*. Например, пусть все объекты и субъекты

системы изначально снабжены метками из одного и того же множества. Кроме того, предположим для простоты изложения, что для всех объектов определен только один вид операции доступа. Допустим также существование следующего правила: доступ к объекту разрешен, если метки субъекта и объекта совпадают, и не разрешен, если не совпадают. Имея такое правило, нет смысла заранее создавать и хранить матрицу — проще вычислять соответствующий элемент при каждой попытке доступа.

Ранее мы рассматривали различные подходы к хранению и представлению информации о правах доступа, не придавая значения тому, каким именно образом они были назначены. Однако способ назначения прав — *авторизация* — существенно влияет на способ управления доступом.

Существует два основных подхода к авторизации:

- для авторизации выделяется особый *полномочный орган* (authority), который принимает все решения о наделении пользователей правами относительно всех объектов;
- функции принятия решений по авторизации *делегируются* некоторым субъектам.

Каждый из этих двух подходов управления доступом может быть реализован множеством различных способов, отражающих разные методы задания и приведения в исполнение ограничений, однако большинство реализуемых на практике способов может быть отнесено к одной из следующих категорий:

- **дискреционный метод доступа** (Discretionary Access Control[1], **DAC**), называемый также избирательным или произвольным;
- **мандатный метод доступа** (Mandatory Access Control[2], **MAC**), называемый также принудительным;
- **ролевой доступ** (Role-based Access Control, **RBAC**), называемый также недискреционным методом доступа (nondiscretionary access control).

Помимо этих методов «в чистом виде», система управления доступом может базироваться и на их комбинации.

## Дискреционный метод управления доступом

Одно из первых систематических изложений принципов DAC было предпринято в 1987 году в документе NCSC-TG-003-87, «Руководство по дискретному управлению доступом». В то время модель DAC была самой распространенной схемой управления доступом и, кстати, таковой она остается и по сегодняшний день — большинство универсальных ОС реализуют дискреционную модель. В документе NCSC дается следующее определение метода DAC:

> **Дискреционный метод** представляет собой средство ограничения доступа к объектам, базирующееся на уникальных идентификаторах субъекта и/или групп, к которым этот субъект относится. Управление доступом в методе DAC является дискреционным, или произвольным, — в том смысле, что субъект, обладающий некоторыми разрешениями на доступ к объектам, может по своему усмотрению передать часть своих полномочий (иногда прямо, а иногда — опосредованно) другим субъектам.

---

[1] Discretionary — действующий по своему усмотрению.

[2] Mandatory — обязательный, принудительный.

Отсюда следуют две главные особенности дискреционного метода:

- Права доступа в методе DAC описываются в виде *списков ACL*, которые дают возможность гибкого и гранулированного определения набора разрешенных операций для каждого отдельного пользователя по отношению к каждому отдельному ресурсу, причем и пользователи, и ресурсы задаются уникальными идентификаторами.
- В методе DAC право назначать права на доступ к объектам *делегируются* отдельным пользователям — владельцам объектов. То есть им разрешается действовать по своему усмотрению и назначать другим пользователям права на доступ к тем объектам, владельцами которых они являются.

Таким образом, процедура авторизации является распределенной между множеством пользователей-владельцев. Владельцами считаются пользователи, создавшие объект, или пользователи, которые были назначены владельцами другими уполномоченными на то пользователями или системными процессами. Владелец имеет полный контроль над созданным им объектом и несет всю полноту ответственности за управление доступом к нему. Вместе с тем он может назначать права доступа к своим объектам, руководствуясь некоторым правилом, принятым на предприятии.

Основным достоинством метода DAC является его гибкость, обусловленная свободой пользователей наделять правами или аннулировать права других пользователей на доступ к своим ресурсам, а также возможностями тонкой настройки набора разрешенных операций. Однако это достоинство имеет свою оборотную сторону.

Как и всякая распределенная система, система управления доступом по методу DAC страдает от *невозможности гарантированно проводить общую политику*, осуществлять надежный контроль действий пользователей. Любая политика безопасности, принятая на предприятии, может быть нарушена в результате ошибочных или вредительских действий пользователей.

Другой недостаток дискреционного метода связан с тем, что здесь *права на доступ определяются по отношению к объекту*, а не его содержимому. Это означает, что любой пользователь (точнее, его процесс), имеющий доступ к файлу согласно некоторому списку ACL1, может скопировать его содержимое в другой файл, характеризуемый другим списком ACL2. Это показывает, что системы с контролем доступа по методу DAC не могут применяться там, где требуется очень высокий уровень защиты информации.

**(S)** *Недостатки метода DAC*

## Мандатный метод управления доступом

**Мандатный доступ** позволяет реализовать системы, отвечающие самым строгим требованиям безопасности, — как правило, они используются в правительственных и военных учреждениях или в других организациях, для которых чрезвычайно важен высокий уровень защиты данных.

К основным чертам мандатного метода управления доступом можно отнести следующие:

- авторизацию и управление доступом осуществляет *центральный полномочный орган*, отвечающий за безопасность (обычно в роли такого органа выступает ОС);

- решение о предоставлении права доступа принимается ОС динамически на основе простого *правила*, которое разрабатывается уполномоченными на то лицами на основе политики безопасности.

Простота правил достигается тем, что и субъекты, и объекты разбиваются на небольшое число групп. Каждой группе объектов присваивается **уровень (гриф) секретности**, а группам субъектов — **уровни допуска** к объектам того или иного уровня секретности. В разных системах могут быть приняты разные правила, но все они базируются на сравнении уровня секретности объекта и уровня допуска субъекта. Например, правило может быть следующим: субъекту разрешается доступ к объекту, если уровень его допуска равен или выше уровня секретности объекта. На рис. 27.12 это правило представлено в виде матрицы.

| Уровень допуска субъектов / Уровень секретности объектов | Уровень от «совершенно секретно» и ниже | Уровень от «секретно» и ниже | Уровень данных для служебного пользования |
|---|---|---|---|
| Совершенно секретно | Доступ разрешен | 0 | 0 |
| Секретно | Доступ разрешен | Доступ разрешен | 0 |
| Данные для служебного пользования | Доступ разрешен | Доступ разрешен | Доступ разрешен |

**Рис. 27.12.** Правило мандатного доступа, представленное в виде матрицы

Пользователи должны принимать решение системы как данность, они лишены возможности управлять доступом к своим ресурсам или передавать свои права другим пользователям. В отличие от систем DAC, мандатный доступ *имеет централизованный характер и позволяет жестко проводить принятую политику безопасности*.

Элементы, описывающие уровни секретности объектов или уровни допуска субъектов, называют **метками безопасности** (security labels). Мандатный метод управления доступом предусматривает назначение меток безопасности всем без исключения субъектам и объектам системы с тем, чтобы в дальнейшем они использовались системой для принятия решений о допуске.

В большинстве случаев для адекватного отражения политики безопасности невозможно сформулировать правило, основанное на учете только уровней секретности и допусков. К одному и тому же уровню секретности могут быть отнесены самые разные материалы, а в соответствии с принципом минимальных привилегий пользователь должен получать доступ только к той информации, которую ему необходимо знать. Чтобы сделать возможным более специфическое задание прав доступа, в метки безопасности объекта и субъекта добавляется информация о конкретном виде данных, к которому относится данный объект или к которому разрешен доступ данному субъекту соответственно.

Таким образом, каждая метка безопасности состоит из двух частей (рис. 27.13):

- часть, отражающая уровень секретности/допуска, называется *классификацией*;
- часть, характеризующая специфику информации, называется *категорией*.


**Рис. 27.13.** Структура метки безопасности объекта/субъекта

Категория относит данные к определенному виду информации. Например, разные категории могут быть присвоены материалам, относящимся к разным проектам, разным административным подразделениям, разным профессиональным группам. Одному и тому же объекту/субъекту может быть присвоено несколько категорий. Так, отчет о завершении этапа некоторой антитеррористической операции может быть отнесен не только к категории материалов, касающихся данной операции, но и дополнительно к категории материалов подразделения, занимающегося этой работой. Объекты одной категории могут быть классифицированы по-разному, например, одна часть отнесена к более высокому уровню секретности, а другая часть — к более низкому.

Правило, определяющее право доступа, строится на анализе обеих частей меток безопасности объекта и субъекта. Доступ разрешается, если выполняются следующие два условия:

- классификация субъекта равна или выше классификации объекта;
- по меньшей мере, одна из категорий объекта, к которому пытается получить доступ субъект, совпадает хотя бы с одной из категорий данного субъекта.

Рисунок 27.14 иллюстрирует соотношение между классификацией и категорией. Здесь цвет кружков служит для обозначения разных категорий объектов. На рисунке показано три уровня классификации: «совершенно секретно», «секретно» и «для служебного пользования». Объекты одной категории могут принадлежать разным уровням классификации. В метке безопасности субъекта указана классификация «секретно» и перечислены две категории, к которым ему разрешен доступ. Стрелками показаны три попытки доступа. Попытка обращения к уровню «совершенно секретно» была заблокирована системой из-за недостаточно высокого уровня допуска субъекта. Обращение к объекту уровня «секретно» была разрешена, так как классификация субъекта равна классификации объекта, а категория объекта совпала с одной из категорий, указанных в метке безопасности субъекта. Попытка доступа к объекту уровня «для служебного пользования» была пресечена, хотя субъект и имеет более высокий уровень допуска («секретно»). В данном случае ограничением служит категория объекта, которая не совпадает ни с одной из категорий субъекта.

Мандатный доступ, как уже отмечалось, является более безопасным, чем дискреционный, но для его эффективной реализации требуется большой объем подготовительной работы, а после запуска системы необходимо поддерживать в актуальном состоянии метки безопасности существующих объектов, а также назначать метки новым ресурсам и пользователям.

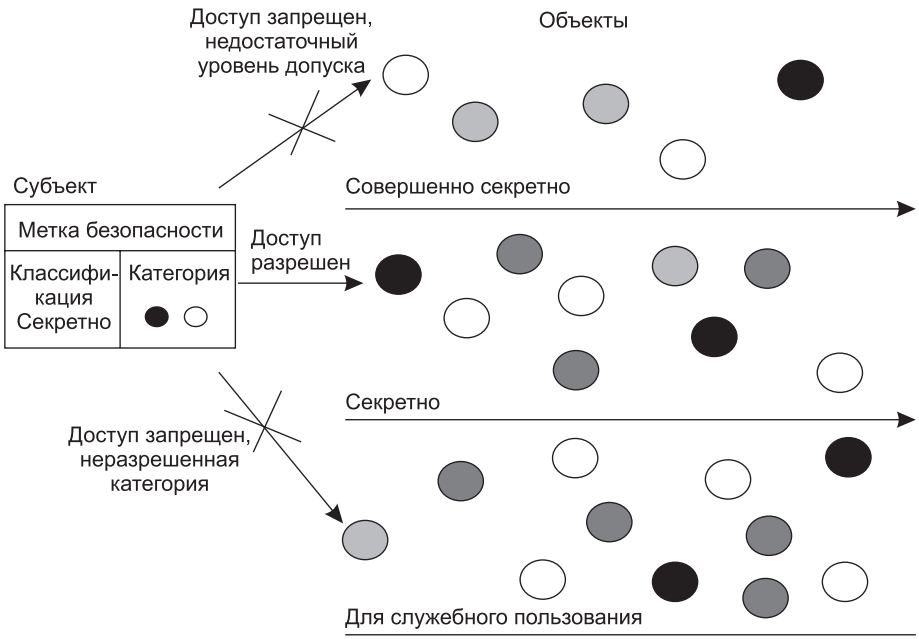



**Рис. 27.14.** Правило мандатного доступа

## Ролевое управление доступом

**Ролевой метод управления доступом** более приближен к реальной жизни, чем дискреционный и мандатный методы. Как видно из названия, основным его свойством является использование «ролей».

Понятие «роль» в данном контексте ближе всего к понятию «должность» или «круг должностных обязанностей». Поскольку одну и ту же должность может занимать несколько людей, то и одна и та же роль может быть приписана разным пользователям. Роли устанавливаются для целей авторизации. Набор ролей должен некоторым образом (не однозначно) соответствовать перечню различных должностей, существующих на предприятии, к которому эта система относится. Ролевая система доступа лучше всего работает в организациях, в которых существует четкое распределение должностных обязанностей.

*Разрешения приписываются ролям*, а не отдельным пользователям или группам пользователей (рис. 27.15). Затем те или иные роли приписываются пользователю. Например, в системе управления доступом, развернутой в банке, всем юристам приписана роль «юрист», трейдерам — роль «трейдер», менеджерам — роль «менеджер» и т. д. Процесс определения ролей должен включать тщательный анализ того, как функционирует организация, какой набор функций должен выполнять работник, имеющий ту или иную должность. Каждой из ролей назначаются права доступа, необходимые и достаточные пользователям для выполнения служебных обязанностей, обусловленных приписыванием к данной роли.

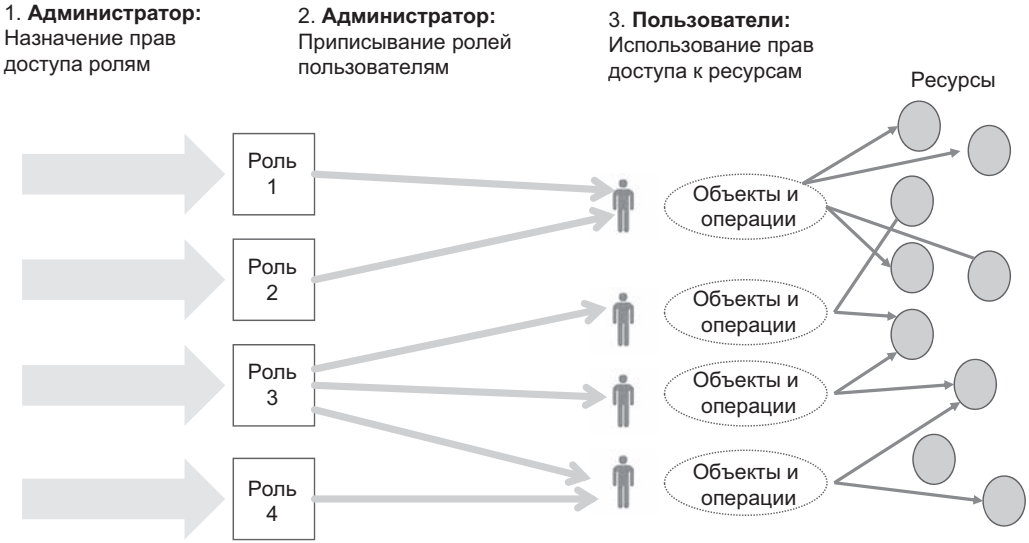



**Рис. 27.15.** Схема авторизации в системах управления доступом на основе ролей

Каждому пользователю может быть приписано *несколько* ролей (с некоторыми ограничениями, о которых рассказано далее). Во время сеанса работы пользователя все роли, которые ему назначены, становятся активными и пользователь получает права доступа, являющиеся результатом объединения прав доступа всех этих групп.

Все пользователи, играющие одну и ту же роль, имеют идентичные права. Изменение производственной ситуации: расширение бизнеса, внедрение новых технологий, продвижение сотрудника по служебной лестнице, перевод сотрудника в другое подразделение и др. — все это может вызвать аннулирование одной роли пользователя и приписывание ему другой роли. Такой подход упрощает администрирование прав доступа: вместо необходимого в дискреционном и мандатном методах отслеживания и обновления прав каждого отдельного пользователя, в ролевом методе достаточно изменить роль или заменить одну роль другой. Таким образом, к особенностям ролевого управления доступом можно отнести следующее:

- Ролевой метод управления доступом *сочетает в себе черты мандатного и дискреционного методов*.
- Ролевую систему управления доступом *легче администрировать* и контролировать, чем дискреционную. В дискреционной системе права назначаются пользователю «мелкими порциями»: запись в определенный файл, чтение другого файла, запуск некоторой программы и т. п. Такой способ позволяет с ювелирной точностью создавать и индивидуально настраивать комплекс прав доступа пользователя, однако он является очень трудоемким, вследствие чего возрастает возможность ошибок. В ролевой системе права доступа выдаются в виде «глыбы» — интегрированного набора разрешений, рассчитанных на возможность выполнения некоторых относительно сложных операций: заполнение кредитного документа, генерация отчетов и др.
- Ролевой доступ является *централизованным* методом — так же, как и в мандатном методе, пользователь лишен возможности управлять назначением прав. Назначение

пользователю роли можно считать некоторым аналогом приписывания уровня допуска пользователю мандатной системы. Однако ролевой доступ является более гибким, чем мандатный, а по возможностям настройки прав доступа ролевой доступ ближе к дискреционному.

- Ролевой метод доступа нельзя отнести к хорошо *масштабируемым*. Он эффективно работает в пределах единой системы или приложения, таких, например, как FreeBSD, СУБД Oracle, MS Active Directory, но на больших предприятиях, численность персонала которых составляет тысячи сотрудников, поддержание множества ролей становится сложной и запутанной задачей. Занимая промежуточное положение между мандатным и дискреционным методами, ролевое управление доступом уступает им обоим в масштабируемости. В мандатном методе централизованный характер принятия решений (который не способствует масштабируемости) компенсируется простотой выполняемого алгоритма назначения прав. В дискреционном же методе, напротив, сложность механизма наделения правами компенсируется распределенным характером процедуры принятия решений.

## Управление доступом в операционных системах

Что касается систем управления доступом в *универсальных ОС*, то там доминирует *дискреционная модель* управления доступом; согласно этой модели владелец ресурса (пользователь, который его создал или которому передано владение) самостоятельно определяет, кто имеет доступ к этому ресурсу и какие операции с ним он может выполнять. Практически все популярные сегодня семейства универсальных ОС — Unix/Linux/CentOS/Ubuntu, Mac OS X, MS Windows — опираются на дискреционную модель доступа как основную.

*Мандатная модель* — это особенность *специализированных ОС*, рассчитанных на применение в среде с повышенными требованиями к безопасности. Тем не менее существует набор модулей ядра Linux под названием SELinux (Security Enhanced Linux), который реализует многие свойства мандатной модели в среде Linux.

*Ролевая модель* применяется в универсальных ОС частично в виде механизма встроенных групп с предопределенными правами, сосуществуя с моделью дискреционного доступа для индивидуальных пользователей и групп.

**(S)** *Особенности аутентификации в ОС семейства Unix и управление доступом в ОС семейства Windows*

# Централизованные системы аутентификации и авторизации

Традиционный способ аутентификации с помощью многоразовых паролей отлично подходит для случая, когда пользователь все время работает с единственным компьютером, обращаясь только к его ресурсам и ресурсам Интернета, не требующим аутентификации. Такому пользователю нужно запоминать и периодически менять только один пароль. Однако более типичным является случай, когда пользователю приходится работать на разных географически рассредоточенных компьютерах — рабочем стационарном ком-

пьютере, домашнем стационарном компьютере, личном планшете, гостевом компьютере предприятия-партнера — и при этом получать доступ к различным серверам: серверам своего предприятия, серверам предприятия-партнера, к защищенным веб-сайтам Интернета и др.

В том случае, когда каждый компьютер и каждый сервер требует отдельной аутентификации с помощью многоразового пароля, пользователю приходится помнить и обновлять довольно много паролей, и с этой задачей многие пользователи справляются не очень успешно. Согласно исследованию, проведенному Network Applications Consortium, около 70 % звонков пользователей в службу ИТ-поддержки связано с просьбой восстановления забытого пароля, а в среднем пользователь крупной корпоративной сети тратит на процедуры логического входа 44 часа в год. Неудивительно, что большие усилия затрачиваются на разработку процедур так называемого единого логического входа.

Целью **единого логического входа** (Single Sign On, SSO) является создание такого порядка аутентификации, при котором пользователь выполняет вход в сеть только один раз, доказывая свою аутентичность с помощью любого способа аутентификации, а затем результат этой аутентификации прозрачным для пользователя способом применяется каждый раз, когда ему нужно доказывать свою аутентичность какому-либо другому серверу или приложению.

В настоящее время не существует системы аутентификации, реализующей концепцию единого логического входа, которая бы работала со всеми типами операционных систем, приложений и при этом учитывала бы разнообразные отношения между организациями, к которым принадлежат пользователи и информационные ресурсы. Однако имеются системы, позволяющие организовать единый логический вход для однородной в каком-то отношении информационной системы, например для сети, использующей только одну определенную ОС или один определенный протокол аутентификации, либо для группы организаций, доверяющих друг другу при аутентификации своих пользователей. Так, например, свойство однородности операционных систем является условием применимости системы единого входа на основе справочной службы Microsoft Active Directory.

Схема, иллюстрирующая идею систем единого логического входа, представлена на рис. 27.16.

В этой схеме имеются три элемента:

- *Пользователь*, который располагает некоторой информацией, достаточной для его аутентификации. Это может быть информация любого типа из упомянутых ранее — многоразовый пароль, одноразовый пароль, цифровой сертификат, биометрические данные и т. п. На рисунке в качестве примера показан вариант аутентификации на основе многоразового пароля, здесь ID — идентификатор, PW — *многоразовый пароль*.
- *Провайдер идентичности* (Identity Provider) — это система, которая может аутентифицировать пользователя на основе базы данных учетных записей пользователей. Этот элемент может иметь и другие названия, например *сервер аутентификации*.
- *Провайдер сервисов* (Service Provider), называемый также *ресурсным сервером*, — это система, предоставляющая сервисы пользователям. Такими сервисами могут быть файловый сервис, почтовый сервис, веб-сервис, сервис баз данных и т. п. Предполагается, что сервис предоставляется только аутентифицированным пользователям.

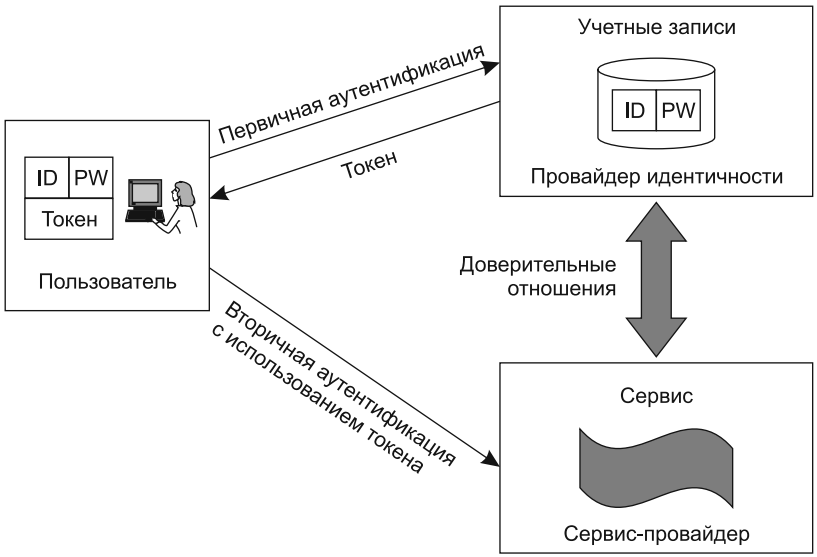



**Рис. 27.16.** Схема единого логического входа

Особенность схемы — в том, что база учетных данных имеется только у провайдера идентичности, а провайдер сервисов доверяет результатам аутентификации пользователей, выполненной провайдером идентичности. Говорят, что в таком случае существуют *доверительные отношения* (trust relationships) между провайдером идентичности и провайдером сервисов.

Пользователь выполняет логический вход в сеть, обращаясь к провайдеру идентичности. Если пользователь смог подтвердить свою аутентичность, то провайдер идентичности предоставляет пользователю некоторую информационную структуру — *токен доступа*, который пользователь хранит в своей базе данных. При необходимости получения доступа к некоторому сервису пользователь предъявляет токен доступа ресурсному серверу. Токен доступа защищен криптографически таким образом, что ресурсный сервер имеет возможность убедиться, что токен был выдан пользователю сервером аутентификации, которому ресурсный сервер доверяет аутентифицировать пользователей. Говорят, что в этом случае происходит *вторичная аутентификация* пользователя, но для самого пользователя она прозрачна, так как предъявлением токена доступа занимается программное обеспечение его компьютера.

Токен доступа обычно имеет ограниченное время действия, например сутки, поэтому пользователь должен его возобновлять, повторяя процедуру с сервером аутентификации.

Примером системы аутентификации, реализующей концепцию единого логического входа, является протокол и основанная на нем сетевая служба Kerberos.

**(S)** *Классификация систем единого логического входа*

**(S)** *Система Kerberos*

# ГЛАВА 28 Технологии безопасности на основе анализа трафика

## Фильтрация

Под **фильтрацией трафика** понимается обработка IP-пакетов маршрутизаторами и файерволами, приводящая к отбрасыванию некоторых пакетов или изменению их маршрута. Фильтрация трафика позволяет либо предотвратить атаку на сеть, заранее блокируя доступ к ней для некоторых внешних сетей и хостов, либо, если источник атаки не был предварительно заблокирован, остановить ее.

Условия фильтрации бывают самыми разными, поэтому не всегда удается найти простой признак, по которому одни пакеты нужно пропускать, а другие — отбрасывать. К тому же такое условие почти всегда является компромиссом между предотвращением атаки и поддержанием должной функциональности защищаемого узла — чем больше потенциальных атак мы предотвращаем, тем больше урезаем функции узла TCP/IP, связанные с его обычной работой. Поэтому фильтрацию можно рассматривать как инструмент, который может быть как полезным, так и опасным, поэтому им надо уметь пользоваться. Еще одним аргументом в пользу тщательного рассмотрения условий фильтрации перед ее применением является потенциальное замедление продвижения трафика при его фильтрации, так как проверка условий фильтрации — это дополнительное действие, и маршрутизатор или файервол может не справиться с такой дополнительной работой вовремя.

## Виды фильтрации

Выборочная передача кадров/пакетов маршрутизатором осуществляется на основе стандартных и дополнительных правил, называемых также **фильтрами**.

**Стандартные правила фильтрации** присущи не только маршрутизаторам, но и другим коммуникационным устройствам — концентраторам и коммутаторам. Эти правила определяют способ функционирования устройств. Так, стандартное (функциональное) правило фильтрации для *концентратора* заключается в том, что кадр, поступивший на любой его интерфейс, независимо от адреса назначения кадра *повторяется* на всех остальных его интерфейсах. *Коммутатор* же функционирует в соответствии с правилом, когда кадр, имеющий некоторый адрес назначения, повторяется только на том интерфейсе, к которому подключена подсеть, имеющая в своем составе узел с данным адресом. Что касается стандартных правил фильтрации для *маршрутизатора*, то они состоят в том, что пакет, поступивший на входной интерфейс, перемещается на тот или иной интерфейс (или отбрасывается) *на основе адресной таблицы* маршрутизатора, в которой учитываются параметры маршрутов и пакетов.

**Дополнительные правила фильтрации**, или **пользовательские фильтры**, задаются сетевыми администраторами, исходя из политики безопасности либо с целью изменения стандартных маршрутов. Дополнительные правила фильтрации маршрутизаторов[1] могут учитывать:

- IP-адреса источника и приемника;
- MAC-адреса источника и приемника;
- идентификаторы интерфейсов, с которых поступают пакеты;
- типы протоколов, сообщения которых несут IP-пакеты (TCP, UDP, ICMP, OSPF);
- номера портов TCP/UDP (то есть типы протоколов прикладного уровня).

При наличии пользовательского фильтра маршрутизатор сначала сравнивает описываемые этим фильтром условия с признаками пакета и при положительной проверке выполняет над пакетом ряд нестандартных действий. Например, пакет может быть отброшен; направлен следующему маршрутизатору, отличающемуся от того, который указан в таблице маршрутизации; помечен как вероятный кандидат на отбрасывание при возникновении перегрузки. Одним из таких действий может быть и обычная передача пакета в соответствии с записями таблицы маршрутизации.

Фильтрация может преследовать *разные цели*, в том числе логическую структуризацию сети, нестандартную маршрутизацию, защиту сети от вредительского трафика.

**Логическая структуризация**, то есть разделение компьютерной сети на подсети и сегменты, самым непосредственным образом влияет на ее эффективность. Инструментом логической структуризации является фильтрация *пользовательского трафика*, проходящего через интерфейсы сетевых устройств на основе стандартных правил.

**Нестандартная маршрутизация** реализуется за счет фильтрации *маршрутных объявлений*. Протоколы маршрутизации, обмениваясь маршрутными объявлениями, создают таблицы маршрутизации, на основе которых любой узел составной сети может связываться с любым другим узлом. Благодаря этому принципу дейтаграммных сетей каждый пользователь Интернета может получить доступ к любому публичному сайту (напомним, в сетях, основанных на технике виртуальных каналов, действует другое правило: взаимодействие произвольных узлов невозможно без предварительной процедуры установления между ними виртуального канала). Однако такая всеобщая достижимость узлов в IP-сетях не всегда отражает потребности их владельцев, поэтому многие маршрутизаторы поддерживают фильтрацию объявлений протоколов маршрутизации, что позволяет дифференцированно управлять достижимостью узлов. Подчеркнем, фильтрация трафика в целях безопасности является важным средством **защиты от атак**. Функцию фильтрации поддерживают файерволы разного типа, в том числе файерволы на базе маршрутизаторов.

## Правила фильтрации маршрутизаторов Cisco

Рассмотрим примеры пользовательских фильтров, написанных на командном языке маршрутизаторов Cisco. Эти фильтры, называемые **списками доступа** (access list)[2], явля-

---

[1] О фильтрации трафика коммутаторами локальных сетей см. главу 11.

[2] Напомним, что термин «список доступа» употребляется и при описании механизмов контроля доступа в ОС, но в ином смысле.

ются очень распространенным средством ограничения пользовательского трафика в IP-маршрутизаторах. Существует два типа списков доступа Cisco:

- **стандартный список доступа** (Standard), позволяющий задавать условия фильтрации, учитывающие только IP-адрес источника;
- **расширенный список доступа** (Extended), позволяющий использовать в условиях фильтрации IP-адреса источника и приемника, порты TCP и UDP источника и приемника, а также типы сообщений некоторых других протоколов, например ICMP.

Как стандартный, так и расширенный список доступа может состоять из нескольких условий, каждое из которых записывается в виде отдельной строки. Условия применяются к пакету в том порядке, в котором они перечисляются в списке доступа до первого совпадения (оставшиеся условия не проверяются).

Стандартный список доступа имеет следующий формат:

```
access-list номер_списка_доступа { deny | permit } {адрес_источника [ метасимволы_
источника ] |any }
```

**Служебные слова** стандартного списка доступа:

- `access-list` — служебное слово, с которого начинается каждая запись;
- `deny` — запрет прохождения пакета, если условие выполняется;
- `permit` — разрешение прохождения пакета, если условие выполняется;
- `any` — служебное слово, которое говорит о том, что условие должно быть применено к любому значению адреса источника.

**Числовые параметры** стандартного списка доступа:

- *номер_списка_доступа* — всем условиям одного и того же списка доступа присваивается один и тот же номер из диапазона 1–99;
- *адрес_источника* — IP-адрес источника;
- *метасимволы_источника* используются аналогично маске, которая накладывается на поля IP-адреса источника поступившего пакета и сравнивается с параметром *адрес_источника*.

Пример стандартного списка доступа:

```
access-list 1 deny 192.78.46.0 0.0.0.255
```

Здесь:

- 1 — номер списка доступа;
- `deny` — пакет, который удовлетворяет условию данного списка доступа, должен быть отброшен;
- 192.78.46.0 — адрес источника;
- 0.0.0.255 — метасимволы источника.

Этот фильтр запрещает передачу пакетов, у которых в старших трех байтах адреса источника имеется значение 192.78.46.0.

Список доступа может включать более одного условия. В этом случае он состоит из нескольких строк с ключевым словом access-list с одним и тем же номером. Так, если мы хотим разрешить прохождение через маршрутизатор пакетов хоста 192.78.46.12, запрещая

передачу пакетов, отправляемых любым другим хостом подсети 192.78.46.0/24, то список доступа будет выглядеть следующим образом:

```
access-list 1 permit 192.78.46.12 0.0.0.0
access-list 1 deny 192.78.46.0 0.0.0.255
```

Расширенный список доступа имеет следующий формат:

```
access-list номер_списка_доступа { deny | permit } ключевое_слово_протокола
{адрес_источника метасимволы_источника [ операция порт_источника] | any }
{ адрес_приемника метасимволы_приемника [ операция порт_приемника ] | any }
```

Параметры:

- *номер_списка_доступа* — номер списка доступа из диапазона 100–199;
- *ключевое_слово_протокола* — `ip`, `tcp`, `udp` или `icmp`;
- *операция*: `eq`, `lt`, `gt` (позволяет задать порт, диапазон портов UDP/TCP или тип пакета ICMP).

Расширенный список дает возможность фильтровать пакеты определенных приложений на основе известных портов TCP/UDP их серверной части. Рассмотрим несколько примеров. Пример 1:

```
access-list 105 permit tcp any host 210.135.17.101 eq 21
```

Эта запись разрешает прием запросов от любого хоста, направленных FTP-серверу (TCP-порт 21) с адресом 210.135.17.101 (используется дополнительное служебное слово `host` вместо маски 0.0.0.0). Пример 2:

```
access-list 101 deny ICMP any 192.78.46.0 0.0.0.255 eq 8
```

Эта запись запрещает передачу эхо-запросов (ping-запросов) от любого хоста к хостам подсети 192.78.46.0/24. Пример 3:

```
access-list 105 permit tcp any eq 80 any gt 1023 established
```

Эта запись разрешает клиентам веб-службы (они всегда имеют порт TCP > 1023) получать ответы от любых веб-серверов (порт 80), с которыми у них уже установлено TCP-соединение (служебное слово `established` оговаривает это, маршрутизатор проверяет данный факт по наличию признака ACK в пакете).

Список доступа можно применять к любому интерфейсу маршрутизатора и в любом направлении: если список применяется с ключевым словом `in`, то он действует на входящие в интерфейс пакеты. В этом случае говорят, что выполняется **входная фильтрация** (ingress filtering). Например, написанный нами список доступа 1 можно применить к некоторому интерфейсу для обработки входящего трафика, используя следующую команду:

```
access-list 1 in
```

Если же применить список доступа с ключевым словом `out`, то он будет воздействовать на пакеты, исходящие из интерфейса, и в этом случае будет выполняться **выходная фильтрация** (egress filtering).

Для обеспечения подотчетности необходимо *протоколирование событий*, связанных с фильтрацией пакетов. Маршрутизаторы Cisco могут помещать сообщения об обработке пакетов, удовлетворяющих условию некоторой записи списка доступа, в системный

журнал маршрутизатора syslog. По умолчанию такая опция для каждой записи списка доступа неактивна — это сделано для уменьшения нагрузки на маршрутизатор. Для активизации протоколирования необходимо добавить к записи ключевое слово `log`, например:

```
access-list 102 permit TCP any 21 any log
```

В заключение отметим, что приведенный здесь пример языка для списков доступа маршрутизаторов Cisco является хотя и фирменной, но достаточно типичной реализацией, хорошо иллюстрирующей возможности применения маршрутизаторов как файерволов. Отсутствие фильтрации с запоминанием состояния связано со стремлением не создавать слишком большую нагрузку на маршрутизатор и «не отвлекать» его от основных обязанностей. Это ограничение является главным отличием маршрутизаторов от программных и программно-аппаратных файерволов.

**(S)** *Фильтрация маршрутных объявлений*

# Файерволы

## Функциональное назначение файервола

**Файервол** (**межсетевой экран**, или **брандмауэр**) — это комплекс программно-аппаратных средств, осуществляющий информационную защиту одной части компьютерной сети от другой путем анализа и фильтрации проходящего между ними трафика.

Файерволы осуществляют экранирование защищаемого объекта и формируют его внешнее представление. В сетевой среде файерволы часто являются первым и самым мощным рубежом обороны.

**ПРИМЕЧАНИЕ**

Исходным значением термина «файервол» (от англ. firewall) является элемент конструкции дома, а именно — стена, сделанная из огнеупорного материала и препятствующая распространению огня между частями дома (обычно принадлежащими разным собственникам). Термин «брандмауэр» (от нем. brandmauer) много лет назад пришел в русский язык из немецкого. Изначально он обозначал перегородку в поезде, отделяющую область топки паровоза от пассажирского отделения. Интересно, что немецкие специалисты в области безопасности для обозначения межсетевого экрана используют англоязычный термин firewall. В русском языке для термина «файервол» используются и другие транслитерации: файрволл, файрвол, фаервол.

Файервол (рис. 28.1) защищает внутреннюю сеть (например, локальную сеть предприятия или, как вырожденный случай, отдельный компьютер пользователя) от угроз, исходящих из внешней сети (в общем случае — из Интернета). Файервол может также защищать одну внутреннюю сеть предприятия от другой, если в соответствии с принципом минимума полномочий пользователям этих сетей не требуется полный взаимный доступ к ресурсам друг друга.

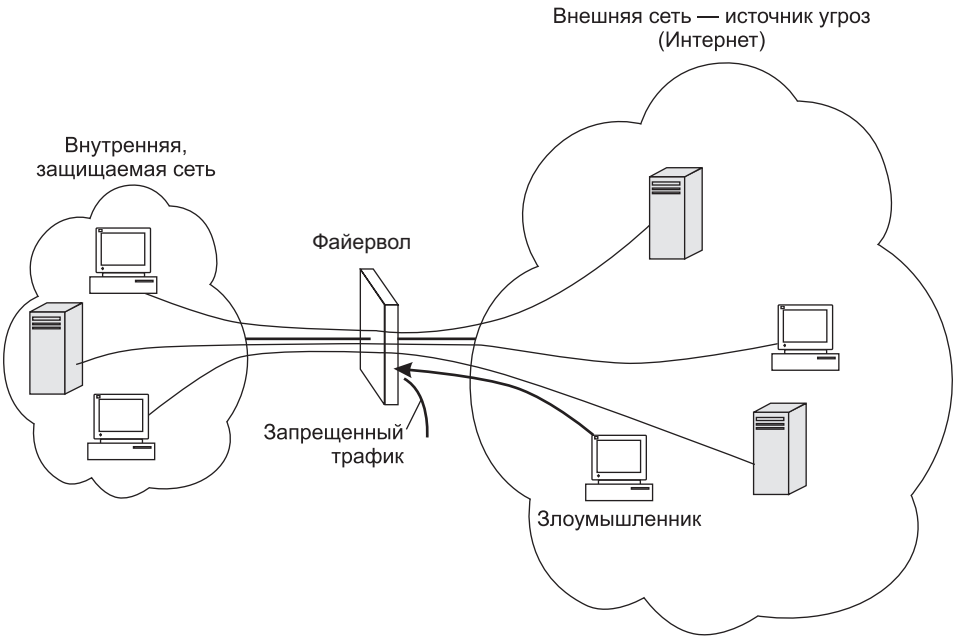



**Рис. 28.1.** Файервол защищает внутреннюю сеть от угроз, исходящих из внешней сети

Для эффективного выполнения файерволом его главной функции — анализа и фильтрации трафика — необходимо, чтобы через него проходил *весь* трафик, которым обмениваются узлы защищаемой части сети с узлами Интернета. Если сеть связана с внешними сетями несколькими линиями связи, то каждая такая линия должна быть защищена файерволом.

*Основными функциями* файервола являются:

- фильтрация трафика в целях защиты внутренних ресурсов сети;
- аудит — файервол должен фиксировать все события, связанные с обнаружением и блокировкой подозрительных пакетов.

Наряду с этими двумя базовыми функциями на файервол могут быть возложены и другие *вспомогательные функции* защиты, в частности:

- антивирусная защита;
- шифрование трафика;
- логическое посредничество между внутренними клиентами и внешними серверами (функция прокси-сервера);
- трансляция сетевых адресов (NAT);
- фильтрация сообщений по содержимому, включая типы передаваемых файлов, имена DNS и ключевые слова;
- предупреждение и обнаружение вторжений и сетевых атак;
- функции VPN.

Как можно заметить, большинство из перечисленных функций часто реализуются в виде отдельных продуктов или в составе систем защиты других типов. Так, функции пакетной фильтрации встроены практически во все маршрутизаторы; задача обнаружения вирусов решается множеством разнообразных программ; шифрование трафика — неотъемлемый элемент технологий защищенных каналов и т. д. и т. п. Прокси-серверы часто поставляются в виде приложений, более того, они сами иногда интегрируют в себе функции, свойственные межсетевым экранам, такие, например, как фильтрация по содержимому (контенту) или трансляция сетевых адресов.

Отсюда возникают сложности при определении понятия «файервол». Например, довольно распространено мнение, что файервол — это пограничное устройство, выполняющее фильтрацию пакетов (то есть маршрутизатор), а прокси-сервер — это совершенно отличный от файервола инструмент защиты. Другие настаивают, что прокси-сервер является непременным и неотъемлемым атрибутом любого файервола, третьи — что файерволом может быть названо только такое программное (аппаратное) устройство, которое способно отслеживать состояние потока пакетов в рамках соединения. Мы же в этой книге будем придерживаться следующей точки зрения:

**Файервол** — это программно-аппаратный комплекс, выполняющий разнообразные функции по защите внутренней сети, набор которых может меняться в зависимости от типа, модели и конкретной конфигурации файервола, при этом минимальный набор функций должен включать фильтрацию трафика для предотвращения сетевых атак и аудит событий, связанных с фильтрацией.

### Пример-аналогия

Функционально сетевой экран можно сравнить с системой безопасности современного аэропорта. Аналогии здесь достаточно очевидные (рис. 28.2) — самолет соответствует защищаемой внутренней сети, а внешняя сеть, из которой приходит потенциально опасный трафик, — внешнему миру, откуда прибывают будущие пассажиры самолета, готовящегося к полету, при этом не все они приезжают с чистыми и ясными намерениями.

В потоке пассажиров, постоянно входящих в здание аэропорта, могут встречаться различные злоумышленники. Наиболее зловещие — террористы — пытаются пронести на борт взрывчатку (в сетевом мире — пакеты, несущие во внутреннюю сеть вирусы, способные «взорвать» серверы и компьютеры пользователей) или оружие для захвата самолета в воздухе (атака по захвату управления удаленным компьютером). Контрабандисты несут с собой незадекларированные ценности (запрещенный контент), а некоторые личности пытаются попасть в самолет по поддельным документам (несанкционированный доступ к внутренним ресурсам сети).

Чтобы отфильтровать трафик пассажиров, система безопасности аэропорта пропускает всех пассажиров и их багаж через единственно возможный путь — зону контроля. Так же поступают при защите сети, направляя весь входящий трафик через файервол. В зоне контроля аэропорта применяются разнообразные средства проверки пассажиров и их багажа. Аутентификация происходит путем сличения паспортов с компьютерной базой данных, а лиц пассажиров — с фотографиями в паспортах. Сюда же можно отнести просвечивание сумок и чемоданов, проход пассажиров через металлодетекторы. Между злоумышленниками и службой безопасности постоянно происходит состязание в коварстве, с одной стороны, и находчивости — с другой. Например, использование террористами флаконов для маскировки жидких компонентов бомбы лишило пассажиров возможности брать с собой в салон шампуни и другие жидкости в больших объемах.

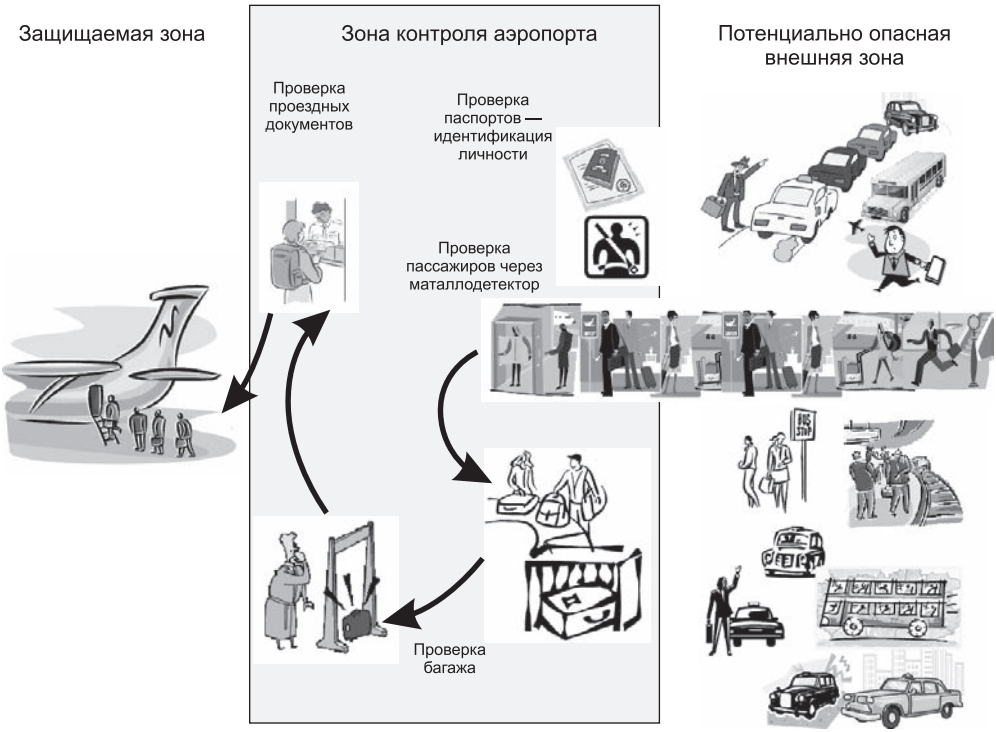



**Рис. 28.2.** Зона контроля аэропорта как аналогия файервола

Файерволы тоже пытаются использовать все возможные средства и методы для противостояния разнообразным угрозам. С помощью паролей и цифровых сертификатов они проверяют аутентичность внешних узлов, пытающихся установить соединения с внутренними; отслеживают логику обмена пакетами для того, чтобы отразить атаки, основанные на искажении этой логики; «просвечивают» содержимое электронных писем и загружаемых документов, пытаясь блокировать запрещенный контент; сканируют загружаемые программы, проверяя их на наличие известных вирусов. Так же, как и в зоне контроля аэропорта, здесь постоянно идет соревнование между хакерами, все время изобретающими новые методы атак, и разработчиками файерволов, старающихся эти атаки обнаружить и пресечь.

## Типы файерволов

Файерволы можно классифицировать по самым разным критериям. Остановимся в этой связи на способе реализации, способе фильтрации и уровне модели OSI.

*Способ реализации файервола* так же многовариантен, как и его функциональность. В качестве аппаратной составляющей сетевого экрана может выступать маршрутизатор или комбинация маршрутизаторов, компьютер или комбинация компьютеров, комбинация маршрутизаторов и компьютеров, наконец, это может быть и какое-то специализированное устройство. Таким же разнообразием отличается и программная составляющая сетевого экрана, имеющая гибкую структуру и включающая в себя различные модули, функции

которых могут широко варьироваться. В самом общем виде по способу реализации различают программный, аппаратный и программно-аппаратный файерволы:

- **программный файервол** реализован как программная система, работающая под управлением универсальной ОС, такой как Microsoft Windows, Linux или Mac OS (возможно, имеющая версии для нескольких универсальных ОС);
- **аппаратный файервол** реализован как набор дополнительных функций маршрутизатора, относящихся к фильтрации (реже — Ethernet-коммутатора);
- **программно-аппаратный файервол** представляет собой программную систему, работающую на базе специализированной платформы. Обычно в качестве такой специализированной платформы выступает аппаратный сервер, сертифицированный для работы с программным обеспечением файервола, плюс установленная на нем универсальная ОС с набором специфических настроек, обеспечивающих максимальный уровень безопасности.

В зависимости от *способа фильтрации* различаются файерволы без запоминания состояния и файерволы с запоминанием состояния:

- **файерволы без запоминания состояния** (stateless) выполняют фильтрацию на основе статических правил, при этом не отслеживаются состояния соединений (сеансов);
- **файерволы с запоминанием состояния** (stateful) принимают решения динамически, с учетом текущего состояния сеанса и его предыстории.

Файерволы с запоминанием состояния для каждого сеанса, который удовлетворяет некоторым условиям, создают динамическую структуру данных в специальной таблице состояний файервола. После прихода очередного пакета контролируемого сеанса состояние сеанса корректируется и принимается решение о выполнении заданного действия с пакетом — пропускать или отбрасывать. Отслеживание для протоколов состояний сеансов требует больших объемов ресурсов, именно поэтому файерволы с запоминанием состояния создаются на программно-аппаратных платформах, имеющих большую оперативную память для хранения таблицы состояний сеансов и быстродействующие процессоры для обработки в реальном времени поступающих пакетов. Если же ресурсов такого файервола оказывается недостаточно, то вместо пользы он может принести вред — например, в ситуации, когда внутренние серверы оказываются недоступными не из-за атак на них, а из-за заторов трафика на интерфейсах файервола.

Теперь, когда мы обсудили такую особенность файерволов, как способность работать в режимах с запоминанием и без запоминания состояния, можно сказать, что именно отсутствие у маршрутизаторов режима фильтрации с запоминанием состояния является *наиболее существенным их отличием* от программных и программно-аппаратных файерволов. Это ограничение связано со стремлением не создавать слишком большую нагрузку на маршрутизатор, чтобы «не отвлекать» его от выполнения основных обязанностей. В то же время существуют и исключения из этого правила, например, модели Juniper SRX являются, с одной стороны, файерволами, поддерживающими фильтрацию с запоминанием состояния и работающими на всех уровнях, включая прикладной, а с другой — они поддерживают все функции «нормального» маршрутизатора, а не только их усеченный набор, как это часто происходит с программно-аппаратными файерволами.

Одной из наиболее важных характеристик файервола является *уровень протокола модели OSI*, на котором он работает. По этому признаку различают файерволы сетевого, сеансового

и прикладного уровней. Если же файервол анализирует и фильтрует трафик на нескольких уровнях, то его относят к самому высокому из всех этих уровней.

Уровень протокола, на котором работает файервол, часто используют в качестве интегральной характеристики, поскольку с ней коррелируют другие признаки файервола. Например, файерволы, работающие на сеансовом и прикладном уровнях, чаще относятся к разряду файерволов с запоминанием состояния, а более простые файерволы сетевого уровня — без запоминания. Далее мы приводим типичные сочетания характеристик файерволов, упорядоченные по уровням.

Управляемые коммутаторы, обладающие расширенным набором функций, в том числе с возможностью фильтрации кадров канального уровня на основе задаваемых администратором списков доступа, могут условно быть отнесены к **файерволам канального уровня**.

**Файерволы сетевого уровня**, называемые также **файерволами с фильтрацией пакетов** (packet filtering firewall), в полном соответствии со своим названием решают задачу фильтрации пакетов по IP-адресам (как источника, так и приемника), а также по значению поля протокола верхнего уровня — в пакет сетевого уровня могут быть вложены сообщения протоколов TCP, UDP, ICMP и др. Более того, несмотря на свое название, такие файерволы работают и на более высоком, транспортном уровне, то есть на уровне портов TCP и UDP, но только на основе статических правил, при которых не отслеживаются состояния соединений, то есть в режиме без запоминания состояния. Поэтому с помощью файервола сетевого уровня можно заблокировать доступ к определенному приложению, запретив прохождение пакетов с определенными номерами портов TCP или UDP, но нельзя защитить сеть от искаженного сеанса TCP или HTTP, потому что это требует отслеживания последовательности шагов в сеансе и, следовательно, запоминания состояния сеанса, а этого файерволы сетевого уровня делать не умеют.

Этому типу файерволов соответствуют маршрутизаторы, поддерживающие пользовательские фильтры, а также программные персональные файерволы операционных систем. Опытный администратор может задать достаточно изощренные правила фильтрации, учитывающие многие требования, касающиеся защиты ресурсов внутренней сети. Тем не менее этот тип сетевых экранов уступает по степени защиты другим типам. *Преимуществами* файерволов сетевого уровня являются простота, невысокая стоимость и минимальное влияние на производительность сети (то есть их дополнительная работа по фильтрации трафика не замедляет маршрутизацию пакетов между двумя сетями).

**Файерволы сеансового уровня** являются файерволами с *запоминанием состояний* соединений на уровнях ниже прикладного. Файерволы сеансового уровня эффективно противодействуют тем видам атак на протокол TCP, в которых нарушается *трехшаговая процедура установления соединения*. Мы подробно обсудим этот случай в следующей главе.

Файерволы сеансового уровня используются для защиты и от других типов атак, в том числе таких, для распознавания которых требуется анализ не отдельных пакетов, а их последовательности. Например, атаку Ping flood можно распознать по слишком маленькому интервалу между эхо-запросами от одного и того же источника, для чего устанавливается предельно допустимый минимальный интервал между эхо-запросами, а затем фиксируется время прихода очередного запроса. Если оно оказывается меньше предельного, то пакет отбрасывается. Таким образом, запоминание состояния сеансов может обобщаться и на протоколы, работающие *без установления соединения* (ICMP, UDP, DNS), а это означает,

что, в отличие сетевых файерволов, файерволы сеансового типа способны защитить сеть от некоторых видов DoS-атак, даже если в этих атаках не используется протокол TCP.

**Файерволы прикладного уровня** способны интерпретировать, анализировать и контролировать *содержимое сообщений*, которыми обмениваются приложения. Они также работают на основе фильтрации с запоминанием состояния, но анализируют состояния не только протоколов нижних уровней (вплоть до транспортного), но и прикладного уровня — таких, как протоколы SSH, HTTP, FTP, SQL, SMTP, POP3, IMAP, FTP, SSH, SQL и др.

**ПРИМЕЧАНИЕ**

Особым типом файерволов этого уровня является прокси-сервер, который перехватывает запросы клиентов к внешним серверам с тем, чтобы потом отправить их от своего имени. Этот тип сетевых экранов обеспечивает самый высокий уровень защиты, хотя и имеет свои недостатки, например, требует больших вычислительных затрат. Кроме того, прокси-сервер может скрывать адрес «доверившегося» ему клиента, что снижает эффективность других средств защиты.

Следует отличать функции по блокировке приложений, реализуемые файерволами сетевого уровня, от защиты приложений внутренней сети файерволами прикладного уровня. Файервол сетевого уровня понимает структуру заголовков пакетов TCP и UDP, за счет чего может запретить или разрешить прохождение пакетов с определенным номером программного порта TCP или UDP, а так как этот номер присвоен серверной части некоторого приложения, то блокируется *весь трафик извне* к этому приложению.

Файервол прикладного уровня действует более гибко. Он контролирует сеанс некоторого приложения, разрешая или запрещая *определенные виды взаимодействия* между внутренней и внешней частями этого приложения в соответствии с заданными правилами. Так, при контроле веб-службы файервол может разрешить использование только определенных команд протокола HTTP, запретив остальные. В список запрещенных команд могут попасть опасные для веб-сервера команды PUT и DELETE. Аналогично при контроле почтовой службы файервол прикладного уровня может не пропускать вовне письма, не подписанные цифровой подписью отправителя, если это предусмотрено политикой безопасности предприятия.

Файервол прикладного уровня, используемый в качестве корпоративного межсетевого экрана, чаще всего является интегрированным продуктом с модульной структурой, которая позволяет менять набор поддерживаемых функций фильтрации в зависимости от потребностей конкретной сети. За счет дополнительных модулей файерволы прикладного уровня могут поддерживать самые разные функции защиты программного обеспечения, например:

- *трансляция внутренних IP-адресов* пользователей на основе стандарта NAT;
- *антивирусный контроль* загружаемых пользователем файлов и получаемых писем «на лету»;
- *контроль контента*, заключающийся, например, в ограничении доступа пользователей к внешним веб-сайтам, страницы которых содержат заданные ключевые слова; такой же контроль может применяться к электронным письмам, отправляемым вовне;
- *транзитная аутентификация пользователей*, обращающихся к некоторому приложению на внутреннем сервере, — эта функция полезна для тех приложений, которые либо не выполняют аутентификацию пользователей совсем, либо делают это незащищенным способом, как, например, FTP-сервер, который принимает пароли пользователей в от-

крытом виде: файервол перехватывает обращение пользователя к FTP-серверу (команду USER) и организует сеанс логического входа пользователя, например, с сервером аутентификации Kerberos, если именно такой способ аутентификации применяется в корпоративной сети;

- *централизованное шифрование* электронных писем пользователей, что избавляет пользователей от необходимости конфигурировать такую функцию на своих клиентских компьютерах (для этого файервол должен хранить цифровые сертификаты пользователей);
- *функции шлюза VPN* с удаленными подразделениями предприятия и удаленными пользователями.

## Программные файерволы хоста

**Программные файерволы хоста** являются частью его программного обеспечения, реализуя наряду с файерволом сети двухступенчатый контроль трафика. Программный файервол работает в режиме ядра ОС, контролируя сетевые интерфейсы хоста и перехватывая пакеты до передачи их протоколам стека TCP/IP.

Программные файерволы хоста являются, как правило, *файерволами сетевого уровня без запоминания состояния сеанса*. Отслеживание состояния сеанса требует значительных вычислительных ресурсов компьютера, и поддержка файерволом хоста такой функции могла бы привести к существенному замедлению выполнения основных его функций. Как и файерволы сетевого уровня на основе маршрутизаторов, программные файерволы хоста позволяют применять правила, учитывающие номера портов TCP/UDP. Это означает, что пользователь хоста может разрешать или запрещать доступ по сети к определенным приложениям хоста, пользующимися закрепленными за ними портами. Посмотрим, как можно блокировать доступ по сети к приложениям с помощью программного файервола iptables, имеющегося практически во всех версиях Unix/Linux. Этот файервол запускается как Unix-демон и работает на основе правил, записанных в текстовом виде в файле /etc/sysconfig/iptables. Правила состоят из трех секций:

- `INPUT` — правила фильтрации входящего трафика;
- `OUTPUT` — правила фильтрации исходящего трафика;
- `FORWARD` — правила фильтрации транзитного трафика в том случае, когда хост работает как IP-маршрутизатор, имея два сетевых интерфейса.

На рис. 28.3 показан пример правил секции INPUT. Как можно заметить, их синтаксис похож на синтаксис правил маршрутизаторов Cisco (см. начало главы).

Рассмотрим, например, такое правило:

```
ACCEPT tcp - - anywhere anywhere state NEW tcp dpts:vnc-server:5903
```

Оно говорит о том, что пакеты, удовлетворяющие условию правила, должны быть приняты (`ACCEPT`). Этому правилу удовлетворяют пакеты протокола TCP (`tcp`) с любым адресом источника (`anywhere`) и любым адресом приемника (`anywhere`), относящиеся к новому сеансу TCP (`state NEW`, то есть к пакетам с признаком SYN) и имеющие в поле порта назначения

значения в диапазоне от 5900 (это стандартный порт сервиса vnc-server, поэтому порт задан с помощью своего имени) до 5903.

```
[root@ganymede sysconfig]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere            udp dpt:domain
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere            udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:bootps
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpts:vnc-server:5903
ACCEPT     udp  --  anywhere             anywhere            state NEW udp dpts:vnc-server:5903
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:oa-system
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:webcache
ACCEPT     udp  --  anywhere             anywhere            state NEW udp dpt:webcache
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:pcsync-https
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:msgsrvr
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ddi-tcp-1
REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited
```

**Рис. 28.3.** Правила секции INPUT файервола iptables

Список включает еще несколько аналогичных правил, а завершает его правило `REJECT all anywhere anywhere`, запрещающее все, что не разрешено явно.

## Влияние DHCP на работу файервола

Назначение IP-адресов может происходить вручную в результате выполнения процедуры конфигурирования интерфейса либо автоматически с помощью протокола динамического конфигурирования хостов DHCP. Этот протокол работает в локальных сетях, так как основан на широковещательных запросах клиентов к DHCP-серверам.

В сети, где адреса назначаются динамически, нельзя быть уверенными в адресе, который в данный момент имеет тот или иной узел. Такое непостоянство IP-адресов влечет за собой некоторые проблемы, прежде всего в работе DNS-серверов и файерволов. Как отмечено, фильтрация трафика во многих случаях происходит на основе IP-адреса защищаемого хоста, а при использовании DHCP этот адрес выдается хосту временно, например на сутки, после чего процедура назначения адреса повторяется, причем нет никакой гарантии, что адрес останется тем же. Поэтому в локальной сети, защищенной файерволом и использующей протокол DHCP, возможны два подхода:

- Применение DHCP только для конфигурирования *клиентских* компьютеров, для которых обычно не требуется отображение доменного имени в IP-адрес, и конфигурирование серверов с постоянными, статическими IP-адресами.
- Применение DHCP и для клиентов, и для серверов, но настройка сервера DHCP производится таким образом, чтобы он для серверов создавал *статические записи* DHCP, в которых MAC-адрес запроса связывается с IP-адресом ответа сервера. При поступлении запроса с MAC-адресом, имеющимся в какой-либо статической записи сервера DHCP, последний помещает в ответ IP-адрес из этой статической записи, а не из динамического пула IP-адресов.

# Прокси-серверы

## Функции прокси-сервера

**Прокси-сервер** (proxy server) — особый тип приложения, выполняющего функции посредника между клиентскими и серверными частями распределенных сетевых приложений, причем предполагается, что клиенты принадлежат внутренней (защищаемой) сети, а серверы — внешней (потенциально опасной) сети. Роль транзитного узла позволяет прокси-серверу логически разорвать прямое соединение между клиентом и сервером с целью контроля процесса обмена сообщениями между ними.

Подобно файерволу, прокси-сервер может эффективно выполнять свои функции только при условии, что контролируемый им трафик не пойдет обходным путем.

Прокси-сервер может быть установлен не только на платформе, где работают все остальные модули файервола (рис. 28.4, *а*), но и на любом другом узле внутренней сети или сети демилитаризованной зоны (рис. 28.4, *б*). В последнем случае программное обеспечение клиента должно быть сконфигурировано таким образом, чтобы у него не было возможности установить прямое соединение с ресурсным сервером, минуя прокси-сервер.

Когда клиенту необходимо получить ресурс от какого-либо сервера (файл, веб-страницу, почтовое сообщение), он посылает свой запрос соответствующему прокси-серверу. Последний анализирует этот запрос и на основании заданных ему администратором правил решает, каким образом он должен быть обработан: отброшен, передан без изменения ресурсному серверу, модифицирован тем или иным способом перед передачей, немедленно обработан силами самого прокси-сервера и др.

В качестве правил, которыми руководствуется прокси-сервер, могут выступать условия пакетной фильтрации. Правила могут быть достаточно сложными — например, в рабочие часы блокируется доступ к тем или иным узлам и/или приложениям, а доступ к другим узлам разрешается только определенным пользователям, причем для FTP-серверов пользователям разрешается делать лишь загрузку, тогда как выгрузка запрещается. Прокси-серверы могут также фильтровать почтовые сообщения по типу пересылаемого файла (например, запретить получение приложений формата MP3) и по их контенту. К разным пользователям могут применяться разные правила фильтрации, поэтому часто на прокси-серверы возлагается и задача аутентификации пользователей. Если после всесторонней оценки запроса от приложения прокси-сервер констатирует, что запрос удовлетворяет условиям прохождения дальше во внешнюю сеть, то он по поручению приложения-клиента, но от своего имени выполняет процедуру соединения с сервером, затребованным данным приложением.

Помимо основных функций, многие прокси-серверы могут выполнять другие полезные операции, например, *обнаруживать вирусы* еще до того, как они попали во внутреннюю сеть, или *собирать статистические данные* о работе пользователей сети в Интернете. В некоторых случаях прокси-сервер может изменять запрос клиента. Например, если в него встроена функция трансляции сетевых адресов (см. далее раздел «Файерволы с функцией NAT»), то он может в пакете запроса *подменять IP-адреса* и/или номера портов TCP и UDP отправителя. Поэтому многие атаки, построенные на знании злоумышленником адресов узлов внутренней сети, становятся нереализуемыми.

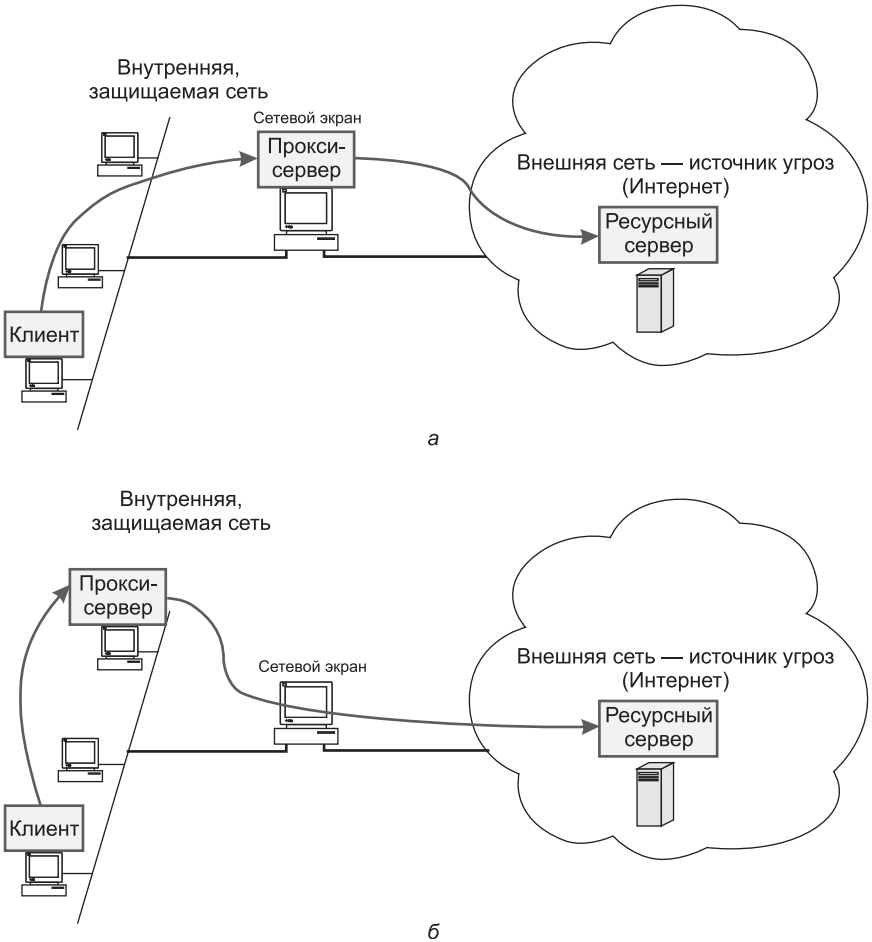



**Рис. 28.4.** Варианты размещения прокси-сервера

Прокси-сервер, выступая посредником между клиентом и сервером, взаимодействующими по определенному протоколу, не может не учитывать специфику этого протокола. Так, для каждого из протоколов HTTP, HTTPS, SMTP/POP, FTP, telnet существует особый прокси-сервер, ориентированный на использование соответствующими приложениями: веб-браузером, программой электронной почты, FTP-клиентом, клиентом telnet. Каждый из этих посредников принимает и обрабатывает пакеты только того типа приложений, для обслуживания которого он был создан. Обычно несколько разных прокси-серверов объединяют в один программный продукт.

Посмотрим, как учитывает специфику протокола *прокси-сервер, ориентированный на веб-службу*. Этот тип прокси-сервера может, например, выполнить собственными силами запрос веб-клиента, не отсылая его к соответствующему веб-серверу. Работая транзитным узлом при передаче сообщений между браузерами и веб-серверами Интернета, прокси-сервер не только передает клиентам запрашиваемые веб-страницы, но и сохраняет их

в своей *кэш-памяти* на диске. В соответствии с алгоритмом кэширования на диске прокси-сервера оседают наиболее часто используемые веб-страницы. При получении запросов к веб-серверам прокси-сервер прежде всего проверяет, есть ли запрошенная страница в его кеше. Если есть, то она немедленно передается клиенту, а если нет, то прокси-сервер обычным образом делает запрос от имени своего доверителя. Прокси-сервер веб-службы может осуществлять *административный контроль* проходящего через него контента, в частности, ограничивать доступ клиента к сайтам, имеющим IP-адреса или DNS-имена из «черных списков». Более того, он может *фильтровать сообщения* на основе ключевых слов. Различают прокси-серверы прикладного и сеансового уровней:

- **Прокси-сервер прикладного уровня**, как это следует из его названия, умеет «вклиниваться» в процедуру взаимодействия клиента и сервера по одному из прикладных протоколов, например HTTP, HTTPS, SMTP/POP, FTP или telnet. Чтобы выступать в роли посредника на прикладном уровне, прокси-сервер должен «понимать» смысл команд, «знать» форматы и последовательность сообщений, которыми обмениваются клиент и сервер соответствующей службы. Это позволяет прокси-серверу проводить анализ содержимого сообщений, делать заключения о подозрительном характере того или иного сеанса.
- **Прокси-сервер сеансового уровня** выполняет свою посредническую миссию на транспортном уровне, контролируя TCP-соединение. Очевидно, что, работая на более низком уровне, прокси-сервер обладает гораздо меньшим «интеллектом» и имеет меньше возможностей для выявления и предупреждения атак. Но он обладает и очень важным преимуществом перед прокси-сервером прикладного уровня — *универсальностью*, то есть он может быть использован любыми приложениями, работающими по протоколу TCP (а в некоторых случаях и UDP).

## «Проксификация» приложений

Список приложений (точнее, их клиентских частей), которые должны передавать свои запросы во внешнюю сеть исключительно через прокси-сервер, определяется администратором. Чтобы эти приложения поддерживали такой режим выполнения, их программы должны быть соответствующим образом написаны.

> Приложения должны быть оснащены средствами, которые распознавали бы запросы к внешним серверам и перед отправкой преобразовывали эти запросы так, чтобы все они попадали на соответствующий прокси-сервер, а не передавались в соответствии со стандартным протоколом прямо на сервер-адресат.

Эти средства должны также поддерживать *протокол обмена сообщениями приложения-клиента с прокси-сервером*. В последние годы в большинстве приложений, ориентированных на работу через Интернет, предусмотрена встроенная поддержка прокси-сервера. Такой поддержкой, например, оснащены все веб-браузеры и все клиенты электронной почты, которыми мы сейчас пользуемся.

«Проксификация» приложения, изначально не рассчитанного на работу через прокси-сервер, требует изменения исходного кода с последующей перекомпиляцией — очевидно, что такая работа не представляет сложностей для разработчиков данного приложения,

но администратор сети не всегда может ее выполнить, например, из-за отсутствия исходного кода или же необходимой квалификации программиста. Задача администратора заключается в приобретении готовых приложений, совместимых с используемым в сети прокси-сервером. Однако даже приобретение готового «проксифицированного» клиента не делает его готовым к работе — необходимо еще конфигурирование, в частности, нужно сообщить клиенту адрес узла сети, на котором установлен соответствующий прокси-сервер.

Как можно было ожидать, процедура «проксификации» значительно упрощается для прокси-сервера сеансового уровня. Для «проксификации» приложения в этом случае достаточно внести простейшие исправления в исходный текст, которые сводятся к замене всех *стандартных вызовов сетевых функций* версиями этих функций из библиотеки процедур соответствующего прокси-сервера, а затем выполнить перекомпиляцию его программы.

Еще один подход к «проксификации» — встраивание поддержки прокси-сервера в операционную систему. В этом случае приложения могут оставаться в полном «неведении» о существовании в сети прокси-сервера — все необходимые действия за них выполнит ОС.

# Трансляция сетевых адресов

Маршрутизация осуществляется на основе адресов назначения, которые помещены в заголовки пакетов и, как правило, остаются неизменными с момента их формирования отправителем до момента поступления на узел получателя. Но из этого правила есть и исключения. Например, в широко применяемой сегодня технологии **трансляции сетевых адресов** (Network Address Translation, **NAT**) предполагается продвижение пакета во внешней сети (в Интернете) на основании адресов, отличающихся от тех, которые используются для маршрутизации пакета во внутренней (корпоративной) сети.

Одной из причин использования технологии NAT является *дефицит IPv4-адресов*. Если по каким-либо причинам предприятию, у которого имеется потребность подключиться к Интернету, не удается получить у поставщика услуг необходимое количество глобальных IPv4-адресов, то оно может прибегнуть к технологии NAT. В этом случае для адресации внутренних узлов служат специально зарезервированные для этих целей **частные адреса**. Мы уже обсуждали их в главе 13. Чтобы узлы с частными адресами могли связываться через Интернет между собой или с узлами, имеющими глобальные адреса, необходимо использовать технологию NAT.

Технология NAT также оказывается полезной, когда предприятие из соображений *безопасности* желает скрыть адреса узлов своей сети, чтобы не дать возможности злоумышленникам составить представление о структуре и масштабах корпоративной сети, а также о структуре и интенсивности исходящего и входящего трафиков.

Именно технология NAT стояла у истоков зарождения файерволов как отдельного класса продуктов. В начале 90-х годов, когда дефицит адресов IPv4 еще мало ощущался, несколько специалистов основали компанию Network Translation и разработали программный продукт PIX, который позволял транслировать сетевые адреса. Позднее эту компанию приобрела компания Cisco, а программный продукт стал знаменитым фаерйволом Cisco PIX Firewall, являющимся одним из флагманов средств защиты этого класса.

## Традиционная технология NAT

Технология трансляции сетевых адресов имеет несколько разновидностей, наиболее популярная из которых — **традиционная технология трансляции сетевых адресов** — позволяет узлам из частной сети прозрачным для пользователей образом получать доступ к узлам внешних сетей. Подчеркнем, что в данном варианте NAT решается проблема организации только тех сеансов связи, которые *исходят* из частной сети. Направление сеанса в данном случае определяется положением инициатора: если обмен данными инициируется приложением, работающем на узле внутренней сети, то сеанс называется исходящим (несмотря на то что в его рамках в сеть могут поступать данные извне)[1].

Идея технологии NAT — в следующем. Пусть сеть предприятия образует тупиковый домен, узлам которого присвоены частные адреса (рис. 28.5). На маршрутизаторе, связывающем сеть предприятия с внешней сетью, установлено программное обеспечение NAT. NAT-устройство динамически отображает набор частных адресов {IP*} на набор глобальных адресов {IP}, полученных предприятием от поставщика услуг и присвоенных внешнему интерфейсу маршрутизатора предприятия.

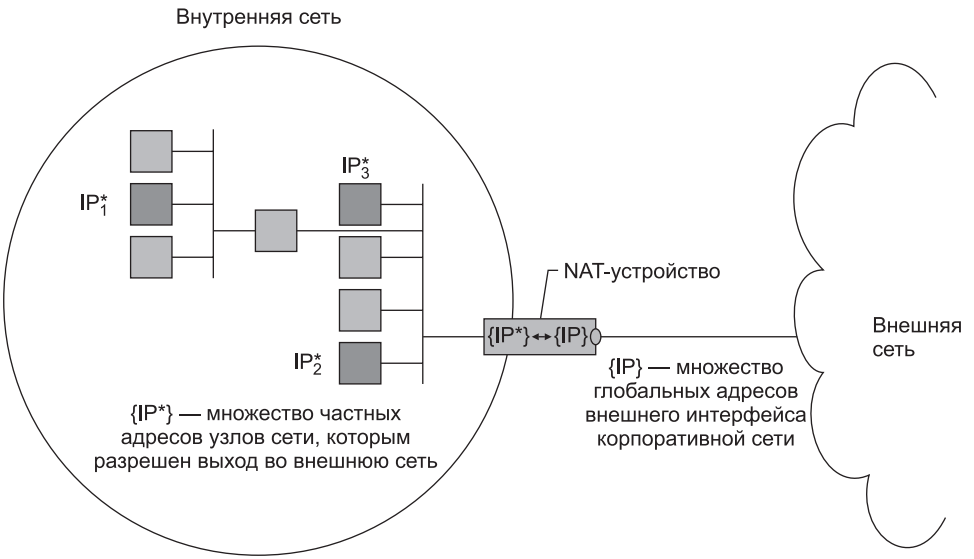



**Рис. 28.5.** Схема действия традиционной технологии NAT

Важным для работы NAT-устройства является правило распространения маршрутных объявлений через границы частных сетей. Объявления протоколов маршрутизации о внешних сетях «пропускаются» пограничными маршрутизаторами во внутренние сети и обрабатываются внутренними маршрутизаторами. Обратное утверждение неверно — маршрутизаторы внешних сетей не получают объявлений о внутренних сетях, объявления

[1] Традиционная технология NAT в виде исключения допускает сеансы обратного направления, заранее выполняя статическое взаимно однозначное отображение внутренних и внешних адресов для некоторого ограниченного набора узлов.

о них отфильтровываются при передаче на внешние интерфейсы. Поэтому внутренние маршрутизаторы «знают» маршруты ко всем внешним сетям, а внешним маршрутизаторам ничего не известно о существовании частных сетей. Традиционная технология NAT подразделяется на технологии **базовой трансляции сетевых адресов** (Basic Network Address Translation, Basic NAT) и **трансляции сетевых адресов и портов** (Network Address Port Translation, NAPT). В технологии Basic NAT для отображения используются только IP-адреса, а в технологии NAPT — еще и так называемые транспортные идентификаторы, в качестве которых чаще всего выступают порты TCP и UDP.

## Базовая трансляция сетевых адресов

Если количество локальных узлов, которым необходимо обеспечить выход во внешнюю сеть, меньше или равно имеющемуся количеству глобальных адресов, то для каждого частного адреса гарантировано однозначное отображение на глобальный адрес. В каждый момент времени количество внутренних узлов, которые получают возможность взаимодействовать с внешней сетью, ограничивается количеством адресов в глобальном наборе. В этой ситуации целью трансляции является не столько решение проблемы дефицита адресов, сколько обеспечение безопасности.

Частные адреса некоторых узлов могут отображаться на глобальные адреса *статически*. К таким узлам можно обращаться извне, используя закрепленные за ними глобальные адреса. Соответствие внутренних адресов внешним задается таблицей, поддерживаемой маршрутизатором или другим устройством (например, файерволом), на котором установлено программное обеспечение NAT.

В нескольких тупиковых доменах могут быть совпадающие частные адреса. Например, в сетях *A* и *B* на рис. 28.6 для внутренней адресации применяется один и тот же блок адресов 10.0.1.0/24. В то же время адреса внешних интерфейсов обеих сетей (181.230.25.1/24, 181.230.25.2/24 и 181.230.25.3/24 в сети *A* и 185.127.125.2/24 185.127.125.3/24 185.127.125.4/24 в сети *B*) уникальны глобально, то есть никакие другие узлы в составной сети их не используют. В данном примере в каждой из сетей только три узла имеют возможность «выхода» за пределы сети своего предприятия. Статическое соответствие частных адресов этих узлов глобальным адресам задано в таблицах пограничных устройств обеих сетей.

Когда узел 10.0.1.4 сети *A* посылает пакет хосту 10.0.1.2 сети *B*, он помещает в заголовок пакета в качестве адреса назначения глобальный адрес 185.127.125.3/24. Узел-источник по умолчанию направляет пакет своему маршрутизатору R1, которому известен маршрут к сети 185.127.125.0/24. Маршрутизатор передает пакет на пограничный маршрутизатор R2, которому также известен маршрут к сети 185.127.125.0/24. Перед отправкой пакета модуль NAT, работающий на данном пограничном маршрутизаторе, используя таблицу отображения, заменяет в поле адреса источника частный адрес 10.0.1.4 соответствующим ему глобальным адресом 181.230.25.1/24. Когда пакет после путешествия по внешней сети поступает на внешний интерфейс NAT-устройства сети *B*, глобальный адрес назначения 185.127.125.3/24 преобразуется в частный адрес 10.0.1.2. Пакеты, передаваемые в обратном направлении, проходят аналогичную процедуру трансляции адресов.

Заметим, что в описанной операции не требуется участия узлов отправителя и получателя, то есть она прозрачна для пользователей.

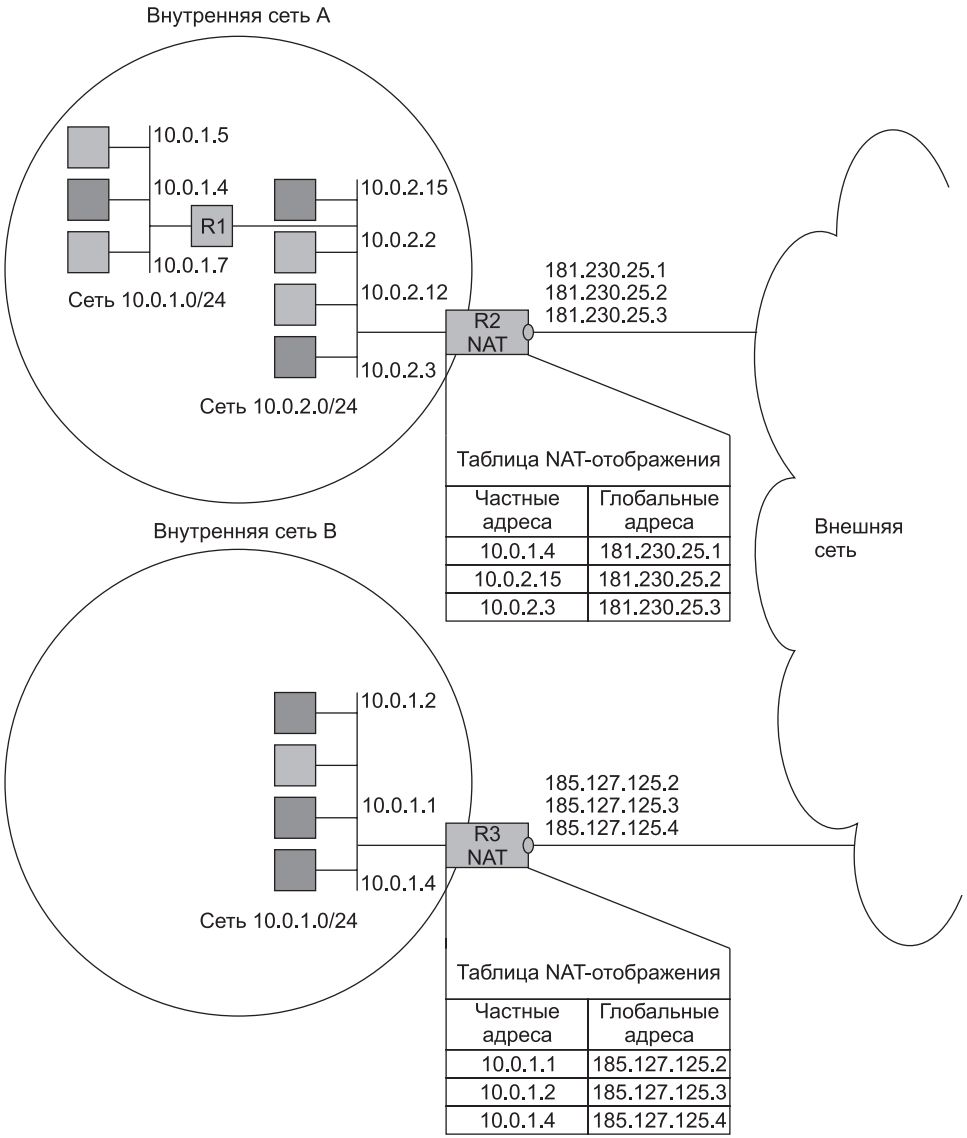



**Рис. 28.6.** Базовая трансляция сетевых адресов для исходящих сеансов

## Трансляция сетевых адресов и портов

Пусть некоторая организация имеет частную IP-сеть и глобальную связь с поставщиком услуг Интернета. Внешнему интерфейсу пограничного маршрутизатора R2 назначен глобальный адрес, остальным узлам сети организации — частные адреса. NAPT позволяет *всем* узлам внутренней сети одновременно взаимодействовать с внешними сетями, используя единственный зарегистрированный IP-адрес. Возникает законный

вопрос: как внешние пакеты, поступающие *в ответ* на запросы из частной сети, находят узел-отправитель, если в поле адреса источника всех пакетов, отправляемых во внешнюю сеть, помещается один и тот же адрес — адрес внешнего интерфейса пограничного маршрутизатора?

Для однозначной идентификации узла-отправителя привлекается дополнительная информация. Если в IP-пакете находятся данные протокола UDP или TCP, то в качестве такой информации выступает номер порта UDP или TCP соответственно. Но и это не вносит полной ясности, поскольку из внутренней сети может исходить несколько запросов с совпадающими номерами портов отправителя, а значит, опять возникает вопрос об однозначности отображения единственного глобального адреса на набор внутренних адресов. Решение состоит в том, что при прохождении пакета из внутренней во внешнюю сеть каждой паре {внутренний частный адрес; номер порта TCP или UDP отправителя} ставится в соответствие пара {глобальный IP-адрес внешнего интерфейса; назначенный номер порта TCP или UDP}. Назначенный номер порта выбирается произвольно, однако должно быть выполнено условие его уникальности в пределах всех узлов, получающих выход во внешнюю сеть. Соответствие фиксируется в таблице. Эта модель при наличии единственного зарегистрированного IP-адреса, полученного от поставщика услуг, удовлетворяет требованиям по доступу к внешним сетям для большинства сетей средних размеров. На рис. 28.7 приведен пример, когда в тупиковой сети *A* используются внутренние адреса из блока 10.0.0.0. Внешнему интерфейсу маршрутизатора этой сети поставщиком услуг назначен адрес 181.230.25.1.

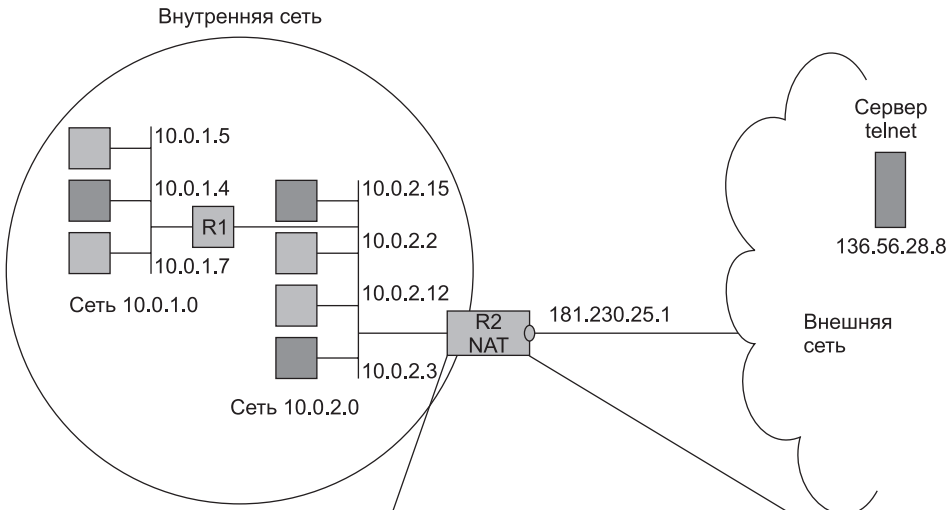



Таблица NAT-отображения

| Частный адрес | Порт | Глобальный адрес | Назначенный порт |
|---|---|---|---|
| 10.0.1.4 | 1245 | 181.230.25.1 | 3451 |
| 10.0.2.15 | 1245 | 181.230.25.1 | 3452 |
| 10.0.2.3 | 1045 | 181.230.25.1 | 3455 |

**Рис. 28.7.** Трансляция сетевых адресов и портов для исходящих сеансов TCP и UDP

Когда хост 10.0.1.4 внутренней сети посылает во внешнюю сеть пакет серверу telnet, то он в качестве адреса назначения использует его глобальный адрес 136.56.28.8. Пакет поступает маршрутизатору R1, который знает, что путь к сети 136.56.0.0/16 идет через пограничный маршрутизатор R2. Модуль NAPT маршрутизатора R2 транслирует адрес 10.0.1.4 и порт TCP 1245 источника в глобально-уникальный адрес 181.230.25.1 и уникально назначенный TCP-порт, в приведенном примере — 3451. В таком виде пакет отправляется во внешнюю сеть и достигает сервера telnet. Когда получатель генерирует ответное сообщение, то он в качестве адреса назначения указывает единственный зарегистрированный глобальный адрес внутренней сети, являющийся адресом внешнего интерфейса NAPT-устройства. В поле номера порта получателя сервер помещает назначенный номер TCP-порта, взятый из поля порта отправителя пришедшего пакета. При поступлении ответного пакета на NAPT-устройство внутренней сети именно по номеру порта в таблице трансляции выбирается нужная строка. По ней определяется внутренний IP-адрес соответствующего узла и действительный номер порта. Эта процедура трансляции полностью прозрачна для конечных узлов.

**ПРИМЕЧАНИЕ**

Заметьте, что в таблице имеется еще одна запись с номером порта 1245, причем такая ситуация вполне возможна: операционные системы на разных компьютерах независимо присваивают номера портов клиентским программам. Именно для разрешения такой неоднозначности и привлекаются уникально назначенные номера портов.

В технологии NAPT разрешаются только исходящие из частной сети сеансы TCP и UDP. Но бывают ситуации, когда нужно обеспечить доступ к некоторому узлу внутренней сети извне. В простейшем случае, когда служба зарегистрирована, то есть ей присвоен хорошо известный номер порта (например, WWW или DNS), и, кроме того, эта служба представлена во внутренней сети в единственном экземпляре, задача решается просто — служба и узел, на котором она работает, однозначно определяются хорошо известным зарегистрированным номером порта службы.

Завершая рассмотрение технологии NAT, заметим, что помимо традиционной технологии NAT существуют и другие ее варианты, например технология двойной трансляции сетевых адресов, когда модифицируются оба адреса — и источника, и приемника (в отличие от традиционной технологии NAT, когда модифицируется только один адрес). Двойная трансляция сетевых адресов необходима, когда частные и внешние адресные пространства имеют коллизии. Наиболее часто это происходит, когда внутренний домен имеет некорректно назначенные публичные адреса, которые принадлежат другой организации. Подобная ситуация может возникнуть, если сеть организации была изначально изолированной и адреса назначались произвольно, причем из глобального пространства. Или же такая коллизия может быть следствием смены поставщика услуг, причем организация хотела бы сохранить старые адреса для узлов внутренней сети.

# Системы мониторинга трафика

Файервол может успешно защитить внутреннюю сеть от разнообразных атак при условии, что его фильтры правильно сконфигурированы. Однако даже правильные фильтры кон-

фигурируются статически, так что для подлинно эффективной защиты требуется *заранее предвидеть* все возможные атаки, что в принципе невозможно. Любой новый тип атаки имеет все шансы «просочиться» через файервол и достичь внутренних серверов защищаемой сети. Обнаружить следы атак, которые смогли преодолеть барьер файервола, можно путем мониторинга сетевого трафика.

**Мониторинг сетевого трафика** — непрерывный процесс инструментального автоматизированного наблюдения за отдельными параметрами трафика с целью проверки соблюдения SLA, планирования сети, а также предотвращения негативных событий — таких, как технические аварии, угрозы и атаки злоумышленников.

Здесь мы рассмотрим следующие средства мониторинга сетевого трафика:

- *анализаторы протоколов*, или *сетевые сниферы*, позволяют захватывать трафик локальных сетей, представлять его в удобном для анализа виде, но собственно анализ данных оставляют администратору;
- *маршрутизаторы, поддерживающие протокол NetFlow*, собирают обобщенные данные о трафике глобальных сетей, передавая его для анализа программным системам NetFlow, которые автоматизируют поиск атак и угроз;
- *системы обнаружения вторжений* (Intrusion Detection Systems, IDS) специализируются на автоматическом распознавании вторжений и угроз в прослушиваемом трафике локальных сетей.

# Анализаторы протоколов

**Анализаторы протоколов** способны на основе некоторых заданных оператором логических условий захватывать отдельные пакеты и декодировать их, то есть показывать в удобной для специалиста форме вложенность пакетов протоколов разных уровней друг в друга с расшифровкой содержания полей каждого пакета. Дружественный интерфейс, обычно присущий этому классу устройств, позволяет пользователю выводить результаты анализа интенсивности трафика; получать мгновенную и усредненную статистическую оценку производительности сети; задавать определенные события и критические ситуации для отслеживания их возникновения.

## Зеркализация портов и неразборчивый режим

Анализатор протоколов представляет собой либо самостоятельное специализированное устройство, либо персональный компьютер, обычно переносной класса Notebook, оснащенный *специальной сетевой картой* и соответствующим программным обеспечением. Программное обеспечение анализатора протоколов состоит из ядра, поддерживающего работу сетевого адаптера и декодирующего получаемые данные, и дополнительного программного кода, зависящего от типа исследуемой сети. В состав некоторых анализаторов может входить также *экспертная система*, способная выдавать пользователю рекомендации о том, какие эксперименты следует проводить в данной ситуации, что могут означать те или иные результаты измерений, как устранить некоторые виды неисправностей в сети и пр.

Анализатор протоколов подключается к сети точно так же, как обычный узел. В примере на рис. 28.8 анализатор подключен к порту P4 коммутатора. Предположим, предстоит анализировать трафик клиентов сети, проходящий через порт коммутатора P2. Так как сеть сегментирована, трафик порта P2 не появляется на порту P4 и анализатор остается без работы, поскольку нужного трафика на его порту просто нет.

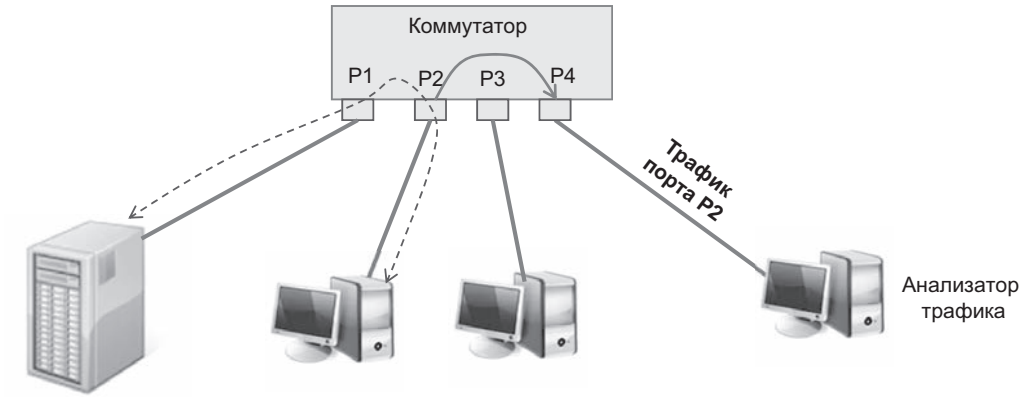



**Рис. 28.8.** Зеркализация трафика

Для того чтобы анализатор мог работать по назначению, нужно каким-то образом обеспечить прохождение трафика порта P2 через порт P4, например, посредством специальной дополнительной функции, которую поддерживают современные коммутаторы и маршрутизаторы, — функции **зеркализации портов**. Она состоит в том, что трафик какого-либо порта копируется в буфер другого порта. В нашем примере как входной, так и выходной трафик порта P2 копируется в порт P4 — в таком случае говорят, что порт P2 зеркализирован в порт P4.

Однако мы все еще не достигли поставленной цели. Несмотря на наличие необходимого трафика на порту P4, сетевой адаптер анализатора не принимает появляющиеся на нем кадры, так как они адресованы не ему (у кадров другие MAC-адреса назначения). Для преодоления этого препятствия сетевой адаптер анализатора протоколов должен быть переведен в **режим неразборчивого захвата пакетов** (promiscuous mode). В неразборчивом режиме адаптер захватывает все пакеты, биты которых появляются на его входе.

## Анализатор протоколов Wireshark

В качестве примера рассмотрим популярный, свободно распространяемый программный анализатор протоколов **Wireshark**, позволяющий анализировать захваченный трафик, используя иерархическое представление полей пакетов (рис. 28.9).

Верхняя панель окна результатов Wireshark показывает основные параметры каждого захваченного пакета: порядковый номер, время, адреса источника и отправителя и др. Расположенная ниже панель позволяет рассмотреть один из пакетов (в данном случае с номером 581) более детально, при этом поля, состоящие из нескольких подполей, можно раскрывать рекурсивно, добираясь до самого дна иерархии признаков пакета, например до признаков заголовка IP или TCP. Wireshark поддерживает весьма длинный список про-

токолов от канального до прикладного уровня — на практике это означает возможность раскрытия заголовков протоколов с пояснениями назначения каждого поля.

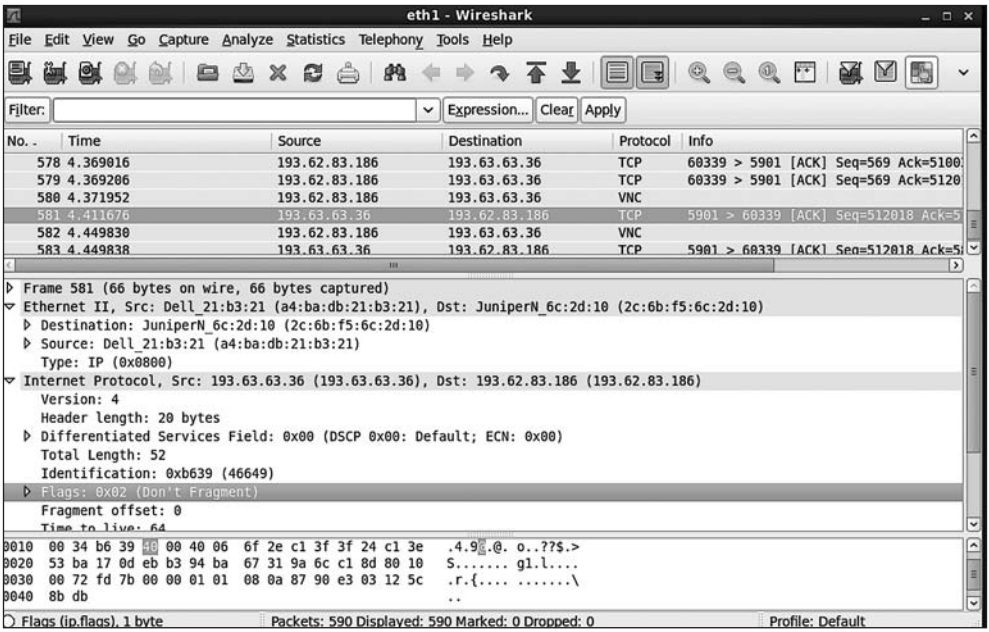



**Рис. 28.9.** Анализ трафика с помощью программы Wireshark

Wireshark дает возможность гибко задавать фильтры двух типов: *фильтр захвата пакетов* и *фильтр отображения пакетов*; практически любое поле любого протокола может быть использовано в условиях этих фильтров. Захваченные кадры помещаются в файл с расширением .pcap (*p*acket *cap*ture), стандартизованный формат которого сегодня поддерживают практически все программные и программно-аппаратные средства мониторинга трафика.

Применение анализаторов протоколов для обнаружения атак требует значительного опыта, так как за десятками сеансов различных протоколов, часто несущих избыточную информацию, не так-то просто увидеть подозрительную активность. Поэтому часто анализ данных, собранных в файле формата PCAP, автоматизируют с помощью какой-нибудь из доступных *программ анализа трафика*[1]. В то же время предоставляемая анализаторами протоколов принципиальная возможность получить полную картину проходящего через сеть трафика дает шанс специалисту по безопасности разобраться с ситуацией «вручную» и обнаружить атаку даже в том случае, когда атака является совершенно новой, не известной автоматическим средствам анализа трафика.

### (S) *Анализатор протоколов Tcpdump*

[1] Не путать с анализатором протоколов.

## Система мониторинга NetFlow

**Система мониторинга NetFlow** является сегодня основным средством учета и анализа трафика, проходящего через маршрутизаторы и коммутаторы сети. Поддерживающие протокол NetFlow сетевые узлы не только выполняют свою основную работу (передачу пакетов в соответствии с адресом назначения), но и собирают статистику о проходящих через них потоках данных, периодически отправляя собранную информацию в *коллекторы* для хранения и обработки. Практически все ведущие производители сетевого оборудования поддерживают протокол NetFlow, так что для превращения вашего маршрутизатора в источник информации о проходящем трафике достаточно активизировать на нем систему NetFlow. Собранную статистику можно использовать в том числе и для *распознавания сетевых атак*. NetFlow собирает статистику не об отдельных пакетах, а о *потоках* пакетов, определяя поток как последовательность пакетов, объединенных набором общих признаков, в число которых чаще всего входят:[1]

- IP-адрес источника;
- IP-адрес назначения;
- порт TCP/UDP источника;
- порт TCP/UDP приемника;
- тип протокола, переносимого IP-пакетом (полезно в тех случаях, когда это не TCP или UDP; например, это может быть ICMP или OSPF);
- индекс интерфейса, на который получен пакет;
- качество обслуживания — значения байта ToS/DiffServ.

NetFlow собирает разнообразную статистику о потоке: время начала и окончания потока, объем данных, переданных с момента начала потока, средняя скорость передачи данных, ну и, естественно, все параметры, определяющие поток, то есть адреса, порты и т. д. Собранная статистика передается в коллекторы (на один или несколько серверов) по окончании потока или же по истечении определенного периода времени.

Маршрутизатор может собирать данные NetFlow в двух режимах: непрерывном, когда обрабатывается каждый пакет, поступающий в маршрутизатор, и выборочный, когда обрабатывается только каждый *n*-й пакет. Выборочный режим менее надежен для распознавания атак, зато он создает гораздо меньше дополнительной нагрузки на маршрутизатор, а для магистрального маршрутизатора, через который проходят десятки, а иногда и сотни тысяч потоков, это существенно.

Типичная система мониторинга на базе NetFlow включает следующие функциональные компоненты (рис. 28.10):

- **Экспортер потока**, называемый также **сенсором**, агрегирует пакеты в потоки и передает статистические данные об этих потоках в один или несколько коллекторов. Экспортером чаще всего является маршрутизатор или коммутатор, хотя могут быть использованы и отдельно стоящие устройства, получающие данные путем зеркалирования порта коммутатора.

---

[1] Такой набор параметров потока определен в NetFlow версии 5, являющейся на момент написания книги наиболее распространенной. В последующих версиях, например в версии 9, определено более 50 параметров потока, имеется возможность собирать статистику о протоколах MPLS, BGP, IPv6.

- **Коллектор** отвечает за прием, хранение и предварительную обработку данных о потоках, полученных от экспортера потока. Реализуется одним или несколькими серверами.
- **Программа-анализатор** анализирует полученные данные о потоках с целью распознавания возможных атак или возникновения перегрузок сети, установления состава и тенденций изменения трафика в сети.

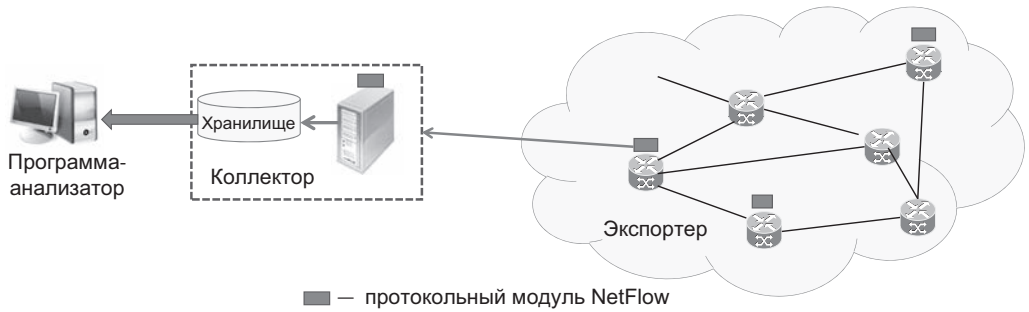



**Рис. 28.10.** Типичная система мониторинга на базе NetFlow

Важно подчеркнуть, что, в отличие от анализаторов трафика и систем обнаружения атак, NetFlow собирает так называемые **метаданные** о трафике, не заглядывая в поля данных пакетов. Часто статистику NetFlow сравнивают с телефонным счетом, показывающим, с кем и сколько разговаривал данный абонент, но не раскрывающим, о чем он говорил. Однако знания метаданных часто бывает достаточно для того, чтобы распознать атаку. Для этого применяется общий принцип мониторинга сети — сравнение ее текущего поведения с «нормальным», то есть таким, которое устойчиво повторялось в прошлом, и при этом мы знаем, что атак в сети не наблюдалось.

> Устойчивые значения статистических характеристик «нормального» поведения сети и ее узлов, которые получены на основании мониторинга сети на довольно значительном периоде времени (недели, месяцы), называются **базовым уровнем** (baseline) характеристик сети.

Другими словами, данные NetFlow служат для поиска аномалий в характере метаданных. Этот же прием используют службы безопасности банков — если, допустим, вы обычно снимаете деньги в банкоматах Москвы, то снятие денег в Рейкьявике является для вас аномалией и ее нужно проверить: может быть, вы просто полетели посмотреть на гейзеры, а может быть, у вас украли данные вашей карточки.

Атака обычно генерирует не совсем обычный трафик, поэтому существуют рекомендации для распознавания таких аномалий. Перечислим основные из них:

- *Выявление узлов с необычно большим числом запросов на установление соединений* (Top N Sessions). Если какой-либо узел вдруг вошел в число $N$ узлов, наиболее активных в отношении установления сеансов, то это должно вызывать подозрения (значение $N$ обычно выбирается не очень большим, к примеру 10). Такая активность характерна для DoS/DDoS-атак, узлов, зараженных червями, атак сканирования портов и некоторых других видов злоумышленной деятельности. Так, спам-хост будет пытаться отослать

как можно больше писем и поэтому устанавливать большое количество соединений в единицу времени с портом 25 (SMTP-порт, на который отправляется почта).

- *Выявление узлов с необычно интенсивным трафиком* (Top N Data). В этом случае хост, который обычно не входил в число *N* самых активных, начинает посылать или получать необычно большое количество данных в единицу времени. Это может быть DoS-атака или же активность червя, пытающегося заразить другие хосты.
- *Анализ SYN и других флагов заголовка TCP*. Наличие необычно большого числа пакетов с установленным флагом SYN или другими флагами заголовка TCP может свидетельствовать о DoS-атаке.
- *Анализ ICMP-сообщений*. Большое количество ICMP-сообщений «Порт/хост/сеть недоступен» может свидетельствовать о сканировании злоумышленником или вирусом хостов и портов.

Другим эффективным методом анализа трафика является проверка значений некоторых полей пакетов на предмет совпадения со значениями, используемыми в известных типах атак (*сравнение с образцами*). Чаще всего образцами атаки являются значения *портов TCP/UDP* и *IP-адресов*. Например, червь SQL Slammer чаще всего использует TCP-порт 1434, а червь W32/Netsky.c всегда использует DNS-сервер с адресом из списка конкретных IP-адресов. Подчеркнем, подход к анализу данных NetFlow должен быть адаптивным, основанным на постоянном обновлении и пополнении базы признаков атак, то есть аналитик должен стараться «идти в ногу» с разработчиками вирусов, ботов и другого вредоносного программного обеспечения.

## Системы обнаружения вторжений

**Система обнаружения вторжений** (Intrusion Detection System, **IDS**) — это программное или аппаратное средство, которое выполняет непрерывное наблюдение за сетевым трафиком и деятельностью субъектов системы с целью предупреждения, выявления и протоколирования атак. В отличие от файерволов, которые строят защиту сети исключительно на основе анализа сетевого трафика, системы обнаружения вторжений учитывают в своей работе различные подозрительные *события*, происходящие в системе.

Существуют ситуации, когда сетевой экран оказывается проницаемым для злоумышленника, например, когда атака идет через туннель VPN из взломанной сети, когда инициатором атаки является пользователь внутренней сети и т. п. И дело здесь не в плохой конфигурации межсетевого экрана, а в самом принципе его работы. Несмотря на то что файервол обладает памятью и анализирует последовательность событий, он конфигурируется на блокирование трафика с *заранее предсказуемыми признаками*, например, по IP-адресам или протоколам. Так что факт взлома внешней сети, с которой у него был установлен защищенный канал и которая прежде вела себя вполне корректно, в правилах экрана отразить нельзя. Точно так же, как и неожиданную попытку легального внутреннего пользователя скопировать файл с паролями или повысить уровень своих привилегий. Подобные подозрительные действия может обнаружить только система, оснащенная агентами, встроенными во многие точки сети, причем она должна следить не только за трафиком, но и за всеми обращениями к критически важным ресурсам отдельных компьютеров, а также обладать информацией о перечне подозрительных действий (сигнатур атак) пользователей. Тако-

вой и является система обнаружения вторжений. Она *не дублирует* действия файервола, а *дополняет* их, производя, кроме того, автоматический анализ всех журналов событий, имеющихся у сетевых устройств и средств защиты, чтобы попытаться найти следы атаки, если ее не удалось зафиксировать в реальном времени.

Другим важным отличием IDS от файерволов является то, что в обязанности IDS *не входит блокировка* подозрительного трафика. IDS только пытается выявить подозрительную активность и поднять тревогу — обычно путем предупреждения администратора сети электронным сообщением. Кроме поднятия тревоги IDS протоколирует подозрительные пакеты, помещая их в журнал.

Типовая система IDS включает следующие функциональные элементы (рис. 28.11):

- источники данных;
- датчики;
- анализатор;
- администратор;
- оператор;
- менеджер.

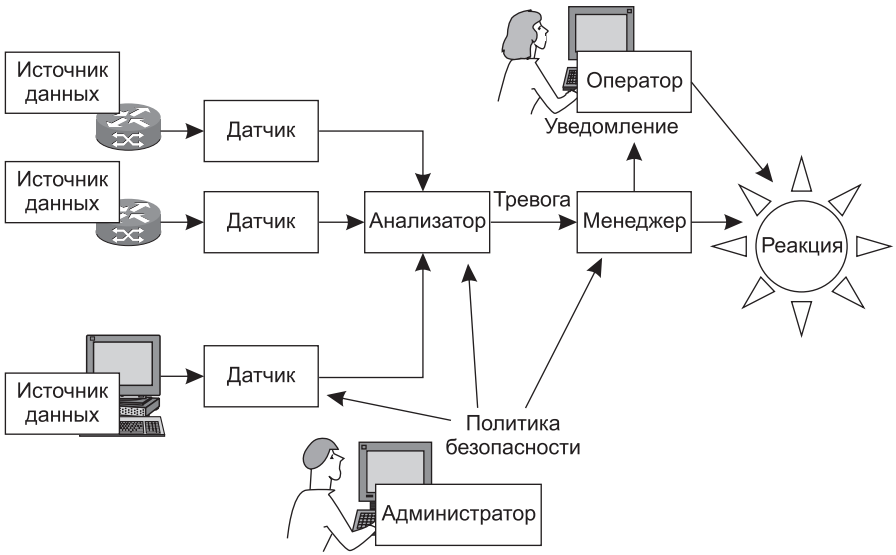



**Рис. 28.11.** Элементы функциональной архитектуры IDS

*Источниками данных* для системы обнаружения вторжений являются маршрутизаторы, коммутаторы и хосты локальной сети, словом, все элементы сети, которые передают, генерируют и принимают трафик.

*Датчик* копирует пакеты, циркулирующие в сети, и передает их анализатору для выявления подозрительной активности. Датчик может представлять собой отдельный компьютер, подключенный к *зеркализованному* порту коммутатора, или же это может быть программный компонент маршрутизатора, который имеет доступ к пакетам, буферизуемым на его

интерфейсах. Датчик может осуществлять первичную фильтрацию пакетов, отбирая только те пакеты, которые удовлетворяют некоторым очевидным критериям, например пакеты, направленные к атакуемым наиболее часто публичным веб-серверам.

*Анализатор* является «мозгом» IDS — он получает данные от датчиков и проверяет их на наличие угроз и подозрительной активности в сети. Анализатор работает на основе правил, составленных *администратором* системы безопасности предприятия в соответствии с политикой безопасности. При выполнении условия одного из правил анализатор вырабатывает сообщение тревоги и передает его *менеджеру* системы IDS — программному компоненту, который хранит конфигурацию IDS и поддерживает удобный интерфейс с оператором IDS. Менеджер IDS оповещает оператора IDS о тревоге в виде некоторого уведомления, привлекающего внимание, например в виде текстовой строки на экране с мерцающим символом, в виде звукового сигнала, продублированного электронным письмом, и т. п.

*Оператор* системы IDS на основе данных уведомления принимает решение о реакции сети на подозрительную активность — это может быть отключение сетевого интерфейса, через который поступает подозрительный трафик, изменение правил файервола для блокировки определенных пакетов или же игнорирование уведомления, если оператор считает, что вероятность вторжения очень мала. В любом случае все данные о потенциальном вторжении протоколируются в журнале менеджера и могут быть использованы впоследствии для повторного анализа ситуации. Если же IDS выполняет также функции *системы предупреждения вторжений*, то менеджер может автоматически передать команды на маршрутизатор или файервол для блокировки подозрительного трафика.

Описанная схема является функциональной, в реальной системе IDS эти функции не обязательно реализуются в отдельных блоках или модулях системы. В минимальном варианте все функции IDS могут быть сосредоточены в программном обеспечении единственного компьютера, сетевой адаптер которого исполняет роль датчика за счет того, что присоединен к зеркализованному порту коммутатора или маршрутизатора. Правда, в этом случае контролироваться будет только один сегмент корпоративной сети (или несколько сегментов, если на порт коммутатора, к которому подключен компьютер с IDS, зеркализуется несколько рабочих портов коммутатора).

Более масштабируемой является реализация IDS с несколькими датчиками, подключенными к различным сегментам сети и посылающими захваченный трафик центральному анализатору. В качестве таких датчиков может выступать дополнительное программное обеспечение маршрутизатора или коммутатора или же отдельные аппаратные устройства.

Наряду с системами обнаружения вторжений существуют **системы предупреждения вторжений** (Intrusion Prevention Systems, **IPS**), которые выполняют автоматические действия по прекращению атаки в случае ее обнаружения. Часто такие системы перепоручают эту работу файерволу, передавая ему новое правило для блокировки подозрительного трафика.

В IDS для обнаружения вторжений применяются нескольких типов правил:

- *Правила, основанные на сигнатуре* (*подписи*) *атаки* (signature rules), используют характерную для той или иной атаки последовательность символов в данных пакета. Например, правило может диктовать поиск строки «user root» в полях FTP-пакета — как известно, этот протокол передает пароли пользователей в открытом виде и применение его суперпользователем root считается грубым нарушением политики безопасности предприятия, так что система IDS должна отслеживать такие случаи. Для эффективной работы система IDS должна иметь обширную постоянно пополняемую базу сигнатур атак.

- *Правила, основанные на анализе протоколов* (protocol rules), связаны с проверкой логики работы протокола и фиксацией отклонений от него. Так как каждый протокол обладает специфической логикой, IDS обычно имеет библиотеку программных модулей, каждый из которых может анализировать поведение определенного протокола. Правила анализа протоколов гораздо сложнее, чем анализа сигнатуры, они требуют высокого быстродействия IDS, чтобы она могла работать в реальном времени.
- *Правила, основанные на статистических аномалиях трафика*, аналогичны тем, которые были рассмотрены в описании технологии анализа данных NetFlow.

# Аудит событий безопасности

**Аудит** (auditing) — это набор процедур учета и анализа всех событий, представляющих потенциальную угрозу для безопасности системы. Аудит является одной из задач, решаемых в целях обеспечения важнейшего требования безопасности — **подотчетности**.

В государственном стандарте подотчетность определяется как свойство безопасной системы, обеспечивающее «*однозначное прослеживание действий любого логического объекта*». Возможность фиксировать деятельность объектов системы, а затем ассоциировать их с индивидуальными идентификаторами пользователей позволяет выявлять нарушения безопасности и определять ответственных за эти нарушения.

Для обеспечения свойства подотчетности в компьютерных сетях используются различные программно-аппаратные средства, в том числе файерволы, системы обнаружения и предотвращения вторжений, анализаторы протоколов, антивирусные системы, а также некоторые подсистемы ОС.

Аудит позволяет «шпионить» за выбранными объектами и выдавать сообщения тревоги, когда, например, какой-либо рядовой пользователь пытается прочитать или модифицировать системный файл. Если кто-то пытается выполнить действия, отслеживаемые системой безопасности, то система аудита пишет сообщение в журнал регистрации, идентифицируя пользователя.

**Журнал регистрации** — это совокупность хронологически упорядоченных записей о специально отобранных событиях. Данные журнала регистрации должны быть надежно защищены от модификации и разрушения неавторизованными субъектами. Таким образом, аудит по своей природе является *реактивным* (а не проактивным) действием, то есть записи становятся достоянием специалиста по безопасности уже по прошествии некоторого времени после того, как события произошли.

*Поскольку никакая система безопасности не гарантирует стопроцентную защиту, именно система аудита оказывается последним рубежом в борьбе с нарушениями.*

Эту мысль лаконично отражает популярное изречение «Prevention is ideal, but detection is a must!», которое говорит о том, что, бесспорно, предупреждение нарушений — это желательная, но, увы, часто недостижимая цель, в то время как обнаружение и фиксация нарушений — это то, что должно быть сделано обязательно. Действительно, после того как

злоумышленнику удалось провести успешную атаку, пострадавшей стороне не остается ничего другого, как обратиться к службе аудита. Если при настройке службы аудита были правильно заданы события, которые требуется отслеживать, то подробный анализ записей в журнале может дать много полезной информации, которая, возможно, позволит найти злоумышленника или, по крайней мере, предотвратить повторение подобных атак устранением уязвимых мест в системе защиты. Аудит действует и как *угроза* потенциальным нарушителям, предупреждая их о том, что в случае несанкционированных действий они легко могут быть выведены на чистую воду.

Функциональный компонент аудита, обеспечивающий *непрерывное* наблюдение за параметрами системы, называется подсистемой **мониторинга**. Мы уже обсуждали мониторинг сетевого трафика. Кроме него объектами мониторинга могут выступать загрузка процессора, статус сетевого интерфейса (активный или пассивный), включенное или выключенное устройство печати.

Ведение журнала событий, особенно в режиме мониторинга, может потребовать слишком много ресурсов вычислительной системы (дискового пространства, вычислительной мощности процессора), а также определенных затрат рабочего времени персонала. Поэтому важно соблюдать баланс между количеством различных видов событий, подлежащих регистрации, с одной стороны, и затрачиваемыми на их протоколирование ресурсами и возможностью их анализа — с другой.

Задачу формирования правил *отбора событий*, подлежащих регистрации, выполняет администратор сети. В ОС, например, к числу таких событий относят попытки успешного и неуспешного логического входа в систему, запуска программ, доступа к защищаемым ресурсам, изменения атрибутов объектов и полномочий пользователей и др. Практически всегда фиксируются события, важные для ОС в целом: рестарт системы, очистка журнала регистрации событий, изменение системного времени и др.

Даже самые простые журналы событий содержат такое огромное количество сведений, что их практически невозможно анализировать «вручную», без специальных средств. Для автоматизации обработки и анализа данных журнала регистрации могут быть использованы различные программные средства, в том числе *средства предварительной обработки данных аудита*, предназначенные для сжатия информации журнала регистрации за счет удаления из него малоинформативных записей, которые только создают ненужный «шум». При реализации аудита в больших сложных системах, состоящих из множества подсистем с собственными средствами протоколирования событий, иногда необходимо приложить специальные усилия по «синхронизации» журналов регистрации. В частности, поскольку в разных частях большой системы одни и те же объекты могут иметь разные имена и, напротив, разные объекты — одинаковые имена, администраторам, возможно, придется принять специальное соглашение об однозначном именовании объектов.

# Типовые архитектуры сетей, защищаемых файерволами

## Логическая сегментация защищаемой сети

Мы рассмотрели функциональные возможности файерволов по защите одной сети от возможных атак, исходящих от другой сети. В простейшем случае первая сеть — это един-

ственная внутренняя сеть предприятия, а внешняя представлена всеми сетями Интернета, соединенными с внутренней сетью через единственную линию связи предприятия с провайдером Интернета. В реальности ситуация оказывается сложнее — сеть предприятия может состоять из нескольких сетей, при этом серверы и хосты этих сетей нуждаются в защите различного типа. Например, если в одной сети находится почтовый сервер и веб-сервер предприятия, а в другой — сервер базы данных клиентов предприятия, то доступ к ним должен регулироваться в соответствии с разными правилами. Если добавить к этому, что многие предприятия соединяют свои сети с Интернетом несколькими линиями связи и, возможно, через нескольких провайдеров, то защита сети предприятия приобретает еще одно измерение — защиту всего периметра сети с помощью нескольких файерволов, при этом их правила защиты должны быть согласованными. Под **сетью периметра** понимается совокупность всех связей корпоративной сети с внешними сетями — сетями провайдеров или корпоративными сетями других предприятий.

Для надежной и эффективной защиты корпоративной сети она должна быть *логически сегментирована* таким образом, чтобы ресурсы каждой подсети в отношении мер защиты были подобными. Ресурсы корпоративной сети, к которым обращаются внешние пользователи, безусловно, составляют в отношении мер безопасности отдельную группу — в нее входят почтовый сервер, веб-сервер, DNS-сервер. Повсеместной практикой является выделение таких ресурсов в отдельную группу и размещение их в подсети, которая получила название **демилитаризованной зоны**[1] (demilitarized zone, **DMZ**). В каком-то смысле зона DMZ подобна транзитной зоне аэропорта, потому что пассажирам разрешается использовать только ресурсы этой зоны, а доступ к внутренним ресурсам авиапредприятия им закрыт.

Рассмотрим особенности организации защиты DMZ на примере сети, показанной на рис. 28.12. В этой сети на рубеже защиты установлено два маршрутизатора, между которыми располагается демилитаризованная зона. Маршрутизаторы играют роль файерволов сетевого уровня. В данном случае сеть DMZ является и сетью периметра, так как только она соединяет внутреннюю сеть предприятия с внешними сетями.

В сети DMZ расположены два общедоступных сервера — внешний DNS-сервер и внешний веб-сервер предприятия. В этой зоне могут быть размещены также прокси-серверы. Учитывая, что само назначение этих компьютеров предполагает практически никак не ограничиваемый доступ к ним внешних пользователей (а значит, и злоумышленников), их необходимо защищать особенно тщательно. Главными задачами при защите этих компьютеров (называемых иногда **компьютерами-бастионами**) является обеспечение целостности и доступности размещенных на них данных для пользователей внешней сети. Эту задачу решают «индивидуальные» средства защиты, устанавливаемые на компьютерах-бастионах: антивирусные программы, фильтры спама и т. п. Кроме того, каждый сервер, к которому разрешено обращение внешних пользователей, должен быть сконфигурирован на поддержку минимально необходимой функциональности. Например, публичный DNS-сервер предприятия не должен быть открытым для любых запросов, так как он может стать инструментом DDoS-атаки.

Чтобы пояснить, каким образом сеть периметра усиливает защиту внутренней сети, посмотрим, что произойдет, если какой-либо злоумышленник сможет «взломать» первый рубеж защиты — внешний маршрутизатор — и начнет прослушивать трафик подключенной

---

[1] Иногда термины «сеть периметра» и «демилитаризованная зона» используются как синонимы.

к нему сети периметра. Очевидно, что он получит доступ только к трафику общедоступных серверов, который не является секретным.

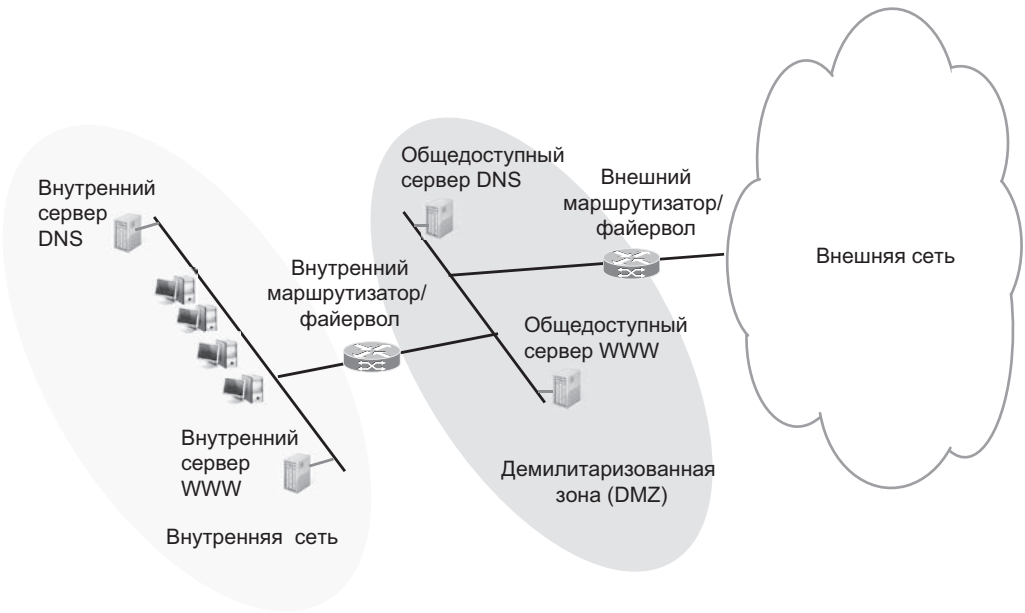



**Рис. 28.12.** Файервол на базе двух маршрутизаторов

Внешний маршрутизатор призван фильтровать трафик с целью защиты сети периметра и внутренней сети. Однако строгая фильтрация в этом случае оказывается невостребованной. Общедоступные серверы по своей сути предназначены для свободного доступа. Что касается защиты внутренней сети, правила фильтрации для доступа к ее узлам и сервисам являются одними и теми же для обоих маршрутизаторов, поэтому внешний маршрутизатор может просто положиться в этом деле на внутренний маршрутизатор. Обычно внешний маршрутизатор находится в зоне ведения провайдера, и администраторы корпоративной сети ограничены в возможностях его оперативного реконфигурирования. Это является еще одной причиной, по которой функциональная нагрузка на внешний маршрутизатор обычно невелика.

Основная работа по обеспечению безопасности локальной сети возлагается на внутренний маршрутизатор, который защищает ее как от внешней сети, так и от сети периметра. Правила, определенные для узлов сети периметра по доступу к ресурсам внутренней сети, часто бывают более строгими, чем правила, регламентирующие доступ к этим ресурсам внешних пользователей. Это делается для того, чтобы в случае взлома какого-либо компьютера-бастиона уменьшить число узлов и сервисов, которые впоследствии могут быть атакованы с этого компьютера. Именно поэтому внутренний маршрутизатор должен отбрасывать все пакеты, следующие во внутреннюю сеть из сети DMZ, исключая пакеты нескольких протоколов (например, HTTP, SMTP, DNS), абсолютно необходимых пользователям внутренней сети для обращения к внешним серверам, установленным в сети DMZ или же за пределами корпоративной сети. Для исключения возможности обращения внешних пользователей к серверам внутренней сети можно разрешить пропуск к ней только тех

TCP-пакетов, которые относятся к TCP-сеансам, установленным по инициативе внутренних пользователей. Например, для маршрутизаторов Cisco это можно сделать с помощью такой строки списка доступа:

```
access-list 200 permit tcp any 201.15.0.0 0.0.255.255 established
```

Здесь 201.15.0.0/16 — диапазон адресов внутренней сети, а список доступа применяется во входном направлении к интерфейсу внутреннего маршрутизатора, к которому подключена сеть DMZ.

Защиту внутренней сети можно усилить, если в ней имеются аналоги внешних серверов, то есть в нашем примере это веб-сервер и DNS-сервер. В подобной конфигурации только этим серверам в случае необходимости разрешается взаимодействовать с серверами зоны DMZ, внутренние же пользователи работают напрямую лишь с внутренними серверами. Например, DNS-сервером по умолчанию для пользователей сети должен быть назначен внутренний DNS-сервер, и только ему позволено изнутри обращаться к внешнему DNS-серверу в том случае, когда он не может разрешить запрос самостоятельно.

Защиту внутренних серверов можно усилить за счет использования частных IP-адресов во внутренней сети. В этом случае внутренний маршрутизатор при трансляции частных адресов должен поддерживать режим NAPT на своем интерфейсе, связывающем его с сетью DMZ.

## Архитектура сети с защитой периметра и разделением внутренних зон

Демилитаризованная зона является практически обязательным элементом защищенной архитектуры любой корпоративной сети, однако в общем случае такая сеть должна быть разбита на большее число сегментов со сходными требованиями к защите на основе фильтрации и анализа трафика. Этот подход иллюстрируется архитектурой сети, показанной на рис. 28.13, — достаточно крупная корпоративная сеть разделена на 6 сегментов, каждый из которых представляет отдельную IP-сеть.

Каждый из сегментов сети представляет собой отдельную зону безопасности. Сети зон соединены друг с другом через *корпоративный файервол*, непосредственной связи между этими сетями нет — такая архитектура позволяет надежно реализовывать правила политики безопасности для каждой зоны. Корпоративный файервол выполнен в виде двух устройств, работающих в режиме горячего резервирования, когда каждое из устройств реализует одни и те же правила фильтрации трафика, и в случае отказа одного из устройств работоспособное устройство без разрыва имеющихся соединений может продолжить обслуживать трафик, проходивший ранее через отказавшее устройство.

Корпоративный файервол также контролирует интернет-трафик предприятия, который проходит через две линии связи с различными интернет-провайдерами — такое соединение достаточно типично для крупных корпоративных сетей, так как оно обеспечивает высокую надежность интернет-связи.

Посмотрим на состав и требования к защите каждой из зон.

*Зона 1* представляет собой демилитаризованную зону предприятия с открытыми для публичного доступа серверами. Ее особенности мы уже рассмотрели. Иногда эту зону разделяют на две зоны, выделяя веб-серверы в отдельную зону, так как защита веб-сервисов

на прикладном уровне существенно отличается от защиты почтового сервера или DNS-сервера.

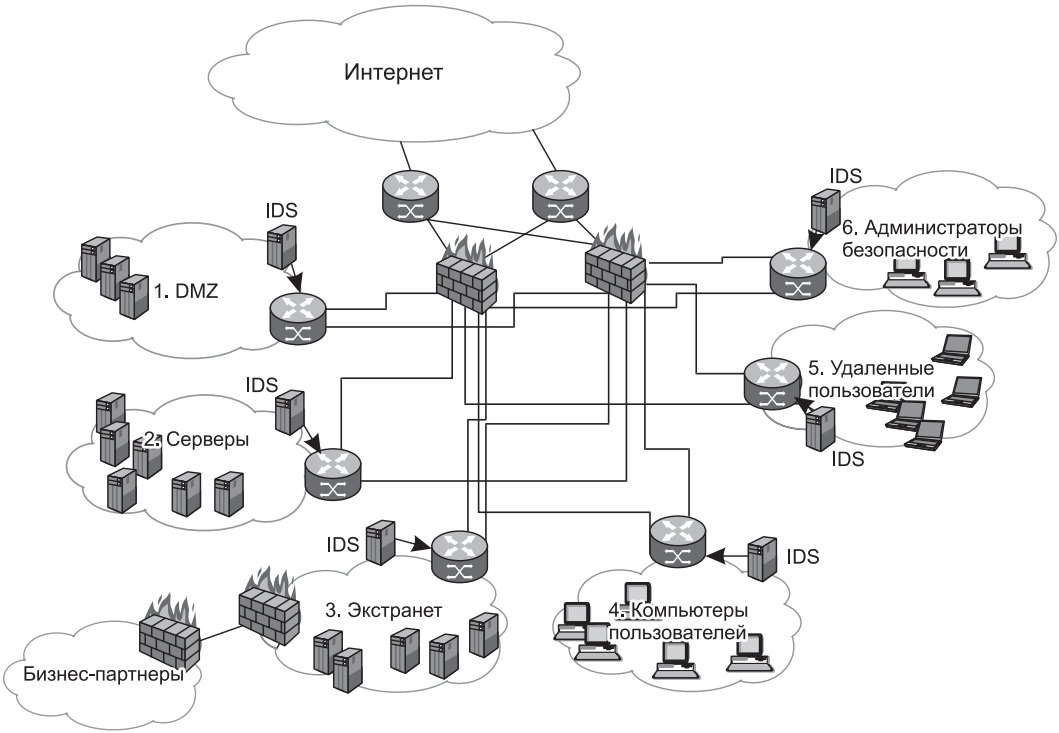



**Рис. 28.13.** Архитектура сети с несколькими зонами защиты

*Зона 2* — это зона внутрикорпоративных серверов, иногда называемая корпоративным порталом. Здесь сосредоточены все информационные ресурсы предприятия, к которым обращаются сотрудники в свое работе: внутренний веб-сервер, серверы баз данных, внутренний почтовый сервер, серверы приложений управления предприятием и т. п. К этой зоне должны иметь доступ только пользователи предприятия, доступ к ней внешних пользователей должен быть заблокирован. Но и для внутренних пользователей доступ к ресурсам этой зоны должен быть ограничен файерволом в соответствии с принципом минимальных привилегий. На уровне файервола он означает, что пользователям должен быть разрешен доступ только к портам тех приложений, которые им нужны в работе, а все остальные порты должны быть файерволом заблокированы.

*Зона 3* — это зона *экстранет* (extranet), где сосредоточены ресурсы, содержащие конфиденциальные данные, к которым разрешен доступ только сотрудникам предприятий-партнеров, а доступ из публичного домена Интернета — запрещен. Зона экстранет в нашем примере соединена с предприятиями-партнерами отдельной линией связи, возможно, через отдельного провайдера, и контролируется отдельным файерволом. В других случаях доступ к экстранет может проходить через общие для всех зон линии связи с Интернетом и контролироваться тем же файерволом. Для обеспечения конфиденциальности данных

может быть применена технология *виртуальных частных сетей* — в этом случае файервол должен выполнять также функции VPN-шлюза.

*Зона 4* — это внутренняя зона предприятия, в ней находятся клиентские компьютеры сотрудников предприятия. Для этой зоны разрешается установление соединений с корпоративным порталом (зона 2) и внешними серверами Интернета. Установление соединений с компьютерами этой зоны извне, то есть из Интернета и из любой другой зоны предприятия, запрещается.

*Зона 5* объединяет *мобильных сотрудников* предприятия, то есть сотрудников, пользующихся удаленным доступом из дома или из сетей других предприятий или публичных провайдеров Интернета (например, из зон Wi-Fi на вокзалах, аэропортах, кафе и т. п.). Обычно для хостов этой зоны устанавливаются те же правила доступа, что и для пользователей зоны 4, то есть они имеют доступ к ресурсам зоны 2 и могут устанавливать соединения с ресурсами Интернета. Для обеспечения конфиденциальности, как и в случае с экстранет, доступ мобильных пользователей осуществляется через защищенные каналы VPN.

*Зона 6* объединяет серверы и клиентские компьютеры, используемые для администрирования средств безопасности предприятия. Здесь сосредоточены серверы политики файерволов, антивирусной защиты, приложений обеспечения безопасности, таких как анализаторы трафика.

К сети каждой зоны подключен сервер IDS, который выполняет анализ трафика этой сети (или, по крайней мере, ее наиболее критичных сегментов) и предупреждает оператора систем безопасности о подозрительной активности.

Файерволы корпоративной сети должны быть сконфигурированы так, чтобы их правила отражали политику безопасности предприятия. Собственно, эта политика и должна определять структуризацию ресурсов сети на зоны, приведенный пример — только один из вариантов этой политики, хотя и достаточно типичный. Возможно и другое разбиение ресурсов на зоны — как более детальное, так и более укрупненное. Например, зона 5, объединяющая в нашем примере всех пользователей предприятия, работающих в его локальной сети (то есть не удаленно), может быть разбита на несколько зон с учетом организационной структуры предприятия — так, в отдельную зону безопасности может быть выделен финансовый отдел.

# ГЛАВА 29 Атаки на транспортную инфраструктуру сети

## Атаки на транспортные протоколы

### TCP-атаки

Протокол TCP используется злоумышленниками и как инструмент для организации атак (обычно — атак отказа в обслуживании), и как цель нападения — нарушение TCP-сеанса атакуемого приложения, например, путем подделки сегмента.

#### Затопление SYN-пакетами

Этот тип DoS-атаки активно применяется злоумышленниками на протяжении многих лет; впервые он был подробно описан (с приведением кода атаки) в 1996 году, и в том же году началось его практическое «применение», продолжающееся по сей день. Атакуемым является конечный узел — как правило, сервер, работающий с клиентами по протоколу TCP.

**Атака затоплением SYN-пакетами** (SYN Flood) использует уязвимость процедуры установления логического соединения протокола TCP. Как отмечено в главе 15, эта процедура основана на использовании флагов *SYN* и *ACK*, переносимых в заголовке каждого TCP-сегмента (рис. 29.1, *а*). Для реализации атаки злоумышленник организует передачу на сервер массированного потока пакетов с флагом *SYN*, каждый из которых инициирует создание нового TCP-соединения (рис. 29.1, *б*). Получив пакет с флагом *SYN*, сервер выделяет для нового соединения необходимые ресурсы и в полном соответствии с протоколом отвечает клиенту пакетом с флагами *ACK* и *SYN*. После этого, установив тайм-аут, он ожидает от клиента завершающий пакет с флагом *ACK*, который, увы, так и не приходит.

Аналогичным образом создается множество других «недоустановленных» соединений. Обычно ОС сервера имеет лимит на количество одновременно поддерживаемых «недоустановленных» TCP-соединений (глобально или для каждого программного порта отдельно), так как каждое открытое соединение требует выделения памяти ядра ОС для нового **блока TCB** (Transmit Control Block). Этот блок содержит данные о состоянии соединения: сокет клиента, номер ожидаемого сегмента, указатель на положение сегмента в буфере и др. Блок TCB имеет размер от 280 до 1300 байт в зависимости от типа ОС. При достижении лимита ОС начинает отвергать все последующие запросы на установление TCP-соединений и, следовательно, отказывает в обслуживании всем, в том числе легальным клиентам сервера. По истечении тайм-аута ОС удаляет из памяти блоки TCB «недоустановленных» соединений и начинает устанавливать новые соединения повторно.

Для осуществления атаки затоплением SYN-пакетами атакующий должен заблокировать нормальную реакцию своего компьютера на получение от атакуемого сервера сегмента с флагами *SYN/ACK*. Нормальная реакция состоит в том, что в соответствии с протоколом

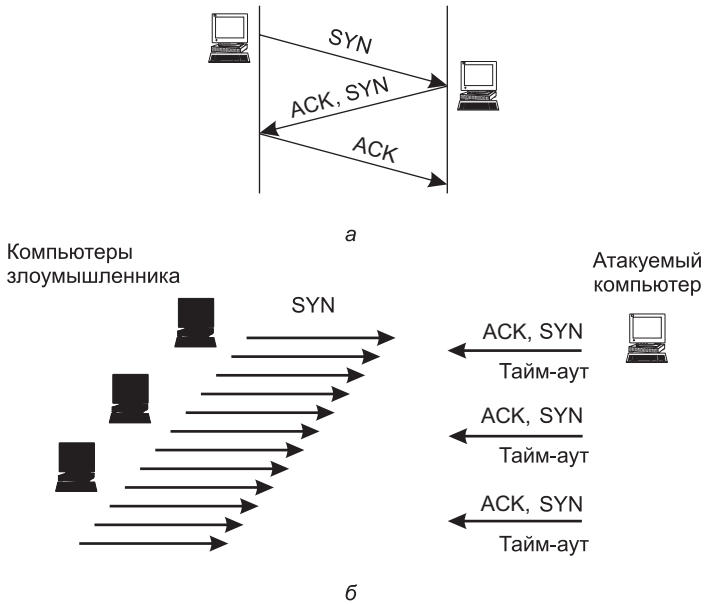



**Рис. 29.1.** Проведение DoS-атаки, в которой используются особенности протокола TCP: *а* — нормальный порядок установления TCP-соединения; *б* — DoS-атака путем создания множества незакрытых TCP-соединений

TCP атакующий должен отправить в ответ сегмент с флагом *ACK*. Но если это произойдет, то атакуемый сервер посчитает процедуру установления TCP-соединения завершенной, удалит соответствующий блок TCB из списка «недоустановленных» соединений и начнет принимать новые соединения. Таким образом, атака не удастся. Поэтому атакующий фильтрует входящий трафик, отсеивая ответы *SYN/ACK* от атакуемого сервера. Обычно атака затоплением SYN-пакетами обнаруживается посредством выявления в трафике большого количества SYN-сегментов без соответствующего количества ACK-сегментов, идущих от того же источника. При этом заметного всплеска сетевого трафика может и не быть, так как лимит «недоустановленных» соединений сам по себе не столь велик. Главным средством борьбы с атакой затоплением SYN-пакетами является *фильтрация* трафика, поступающего от источника атаки. Для этого нужно определить адрес атакующего узла, что в ряде случаев сделать непросто — атакующий может помещать SYN-сегменты в IP-пакеты с «поддельным» адресом отправителя, применяя *спуфинг*. Спуфинг помогает атакующему преодолеть защитный фильтр и избавиться от вредящих ему ответных SYN/ACK-сегментов атакуемого сервера. Для этого ему достаточно выбрать в качестве «поддельных» адреса, которые не будут реагировать на SYN/ACK-сегменты, — например, адреса несуществующих узлов.

### ПРИМЕЧАНИЕ

Спуфинг IP-адресов источника используется во многих типах атак, поэтому борьба с ним — естественный элемент обеспечения сетевой безопасности. Основным средством борьбы является применение на маршрутизаторах техники проверки обратного пути (Reverse Path Check, RPC). Идея этой проверки достаточно проста — пакет должен передаваться маршрутизатором в соответствии с его

адресом назначения только в том случае, если его адрес источника имеется в таблице маршрутизации для интерфейса, с которого этот пакет получен. Действительно, если компьютер злоумышленника подключен к сети 212.100.100.0/24, но генерирует пакеты с адресом источника 25.0.30.18, то маршрутизатор провайдера, к которому подключена сеть 212.100.100.0/24, легко может проверить, что через интерфейс, на который был получен пакет с подделанным адресом, достичь сети 25.0.30.18 нельзя, а значит, пакет нужно отбросить. Однако техника RPC работает не всегда. В тех случаях, когда сеть злоумышленника имеет несколько подключений к сетям разных провайдеров, может произойти отбрасывание легитимных пакетов.

Преодоление атаки путем фильтрации также осложняется, когда поток SYN-сегментов поступает на атакуемый сервер сразу от сотен зараженных компьютеров какой-нибудь сети ботов, то есть когда имеет место **распределенная атака затоплением SYN-пакетами** (DDoS SYN Flood).

Другим способом борьбы с атакой затоплением SYN-пакетами является *изменение параметров протокола TCP* — увеличение предельного числа «недоустановленных» соединений, уменьшение тайм-аута вытеснения старых «недоустановленных» соединений, усложнение логики самой процедуры установления соединения, например, введения специальных *cookie-блоков SYN*. В этом методе при приеме запроса SYN-сервер не запоминает блок TCB в своей оперативной памяти, а посылает его (в сжатом виде) клиенту вместе с SYN/ACK-ответом. При нормальном ходе установления соединения клиент отвечает ACK-сегментом, в котором повторяет сжатый блок TCB. Сервер, получив этот ACK-сегмент, а с ним и все параметры устанавливаемого соединения, создает соответствующий блок TCB в памяти своего ядра. Поскольку в этой модифицированной процедуре на начальном этапе установления соединения ресурсы на сервере не выделяются, то и атака затоплением SYN-пакетами не удается. Разновидностью TCP-атаки затоплением SYN-пакетами является **TCP-атака затоплением ACK-пакетами**, выполняемая путем *отражения*. Злоумышленник посылает SYN-пакеты, в поле адреса источника которых помещен адрес жертвы, на большое количество серверов. Последние отвечают на SYN-пакеты пакетами с установленным битом ACK, «бомбардирующими» атакуемый компьютер и исчерпывающими пропускную способность его входного интерфейса. Этот прием превращает DoS-атаку в DDoS-атаку без использования сети ботов, так как все компьютеры, отвечающие на SYN-запросы, не заражаются предварительно каким-либо вирусом, а работают в полном соответствии со стандартной версией протокола TCP.

## Подделка TCP-сегмента

Протокол TCP предназначен для надежной транспортировки сообщений. Для этого каждый сегмент данных сопровождается порядковым номером первого байта сегмента, причем начальные значения этих номеров для каждой из двух сторон, обменивающихся данными, выбирается случайным образом. При приеме очередного сегмента протокол TCP проверяет, находится ли его порядковый номер в пределах окна приема, и только в случае положительного результата такой проверки добавляет принятые данные к байтам, принятым ранее в ходе данного TCP-сеанса. Описанная проверка предназначена для защиты сегментов некоторого TCP-сеанса от смешивания с сегментами других сеансов, но этот механизм защиты не так уж надежен, чем и пользуются злоумышленники.

**Атака подделкой TCP-сегмента** состоит в генерации TCP-сегментов, все атрибуты которых имеют значения, легитимные для некоторого существующего TCP-сеанса атакуемого

компьютера, то есть IP-адреса, номера TCP-портов источника и приемника, а также порядковые номера из текущего диапазона окна приема. Принимающая сторона не может отличить такие поддельные сегменты от настоящих и помещает информацию злоумышленника в поток пользовательских данных, а значит, злоумышленник может добиться желаемого эффекта, например, поместить ложную информацию в базу данных, заразить атакуемый компьютер вирусом и т. п.

Чтобы «поддельный» сегмент выглядел как настоящий, атакующий может либо прослушивать трафик, либо просто перебирать все возможные значения адресов, портов и порядковых номеров сегментов. Прослушивание трафика представляет собой отдельную нетривиальную задачу, включающую перенаправление трафика (об атаках такого типа см. далее). В то же время перебор параметров TCP-сеанса требует большой вычислительной мощности компьютера атакующего. В обоих случаях атаковать проще длительные TCP-сеансы, например сеансы загрузки больших видеофайлов (короткие сеансы веб-серфинга намного менее уязвимы).

Разновидностью подделки TCP-сегментов является их *повторное использование*. Если злоумышленник смог каким-то образом перехватить трафик между двумя участниками TCP-сеанса, то впоследствии он может просто посылать участникам сеанса дубликаты перехваченных сегментов. Этот прием может применяться злоумышленником для разных целей — например, для нарушения работы некоторого приложения за счет представления устаревшей (перехваченной) информации в качестве новой.

### Сброс TCP-соединения

**Атака сбросом TCP-соединения** используется для разрыва TCP-соединений легальных пользователей. Для проведения атаки злоумышленник должен подделать заголовок TCP-сегмента. При поступлении TCP-сегмента с установленным флагом *RST* узел должен немедленно завершить сеанс, к которому относится этот сегмент, и удалить все данные, полученные в ходе сеанса. Разработчики протокола TCP ввели этот флаг для обработки аварийных ситуаций. Например, если в одном из узлов во время TCP-сеанса происходит сбой, то после восстановления системы он может послать сегмент с этим признаком, чтобы уведомить узел-собеседник о невозможности продолжения сеанса. Сброс соединения используется также некоторыми файерволами для прекращения атаки.

Борьба с атаками подделкой TCP-сегмента и сбросом TCP-соединения может вестись по двум направлениям. Первое направление связано с предотвращением прослушивания трафика. Второе направление основано на изменении поведения самого протокола TCP — например, путем включения дополнительной процедуры аутентификации каждого TCP-сегмента с использованием цифровой подписи. Как известно, цифровая подпись не обеспечивает конфиденциальности (содержимое защищаемых полей не шифруется), но она гарантирует, что TCP-сегмент не был изменен третьей стороной.

## ICMP-атаки

**Атака перенаправлением трафика** может быть осуществлена в самых разных целях (одна из них раскрыта в процессе рассмотрения атаки подделкой TCP-сегментов). При этом существует несколько способов перенаправления трафика. Так, в пределах локальной сети эту задачу можно решить *с помощью протокола ICMP*. В соответствии с данным про-

токолом маршрутизатор посылает хосту непосредственно присоединенной локальной сети ICMP-сообщение о перенаправлении маршрута при отказе этого маршрута или в тех случаях, когда обнаруживает, что для некоторого адреса назначения хост использует нерациональный маршрут. На рис. 29.2 применяемый по умолчанию маршрутизатор R1, получив от хоста H1 пакет, адресованный хосту H2, определяет, что наилучший маршрут к хосту H2 пролегает через маршрутизатор R2. Маршрутизатор R1 отбрасывает полученный пакет и помещает его заголовок в ICMP-сообщение о перенаправлении маршрута, которое посылает хосту H1. В сообщении содержится IP-адрес альтернативного маршрутизатора R2, который теперь должен использовать хост, посылая данные хосту H2. Хост H1 вносит изменения в свою таблицу маршрутизации и с этого момента отправляет пакеты хосту H2 по новому скорректированному маршруту.

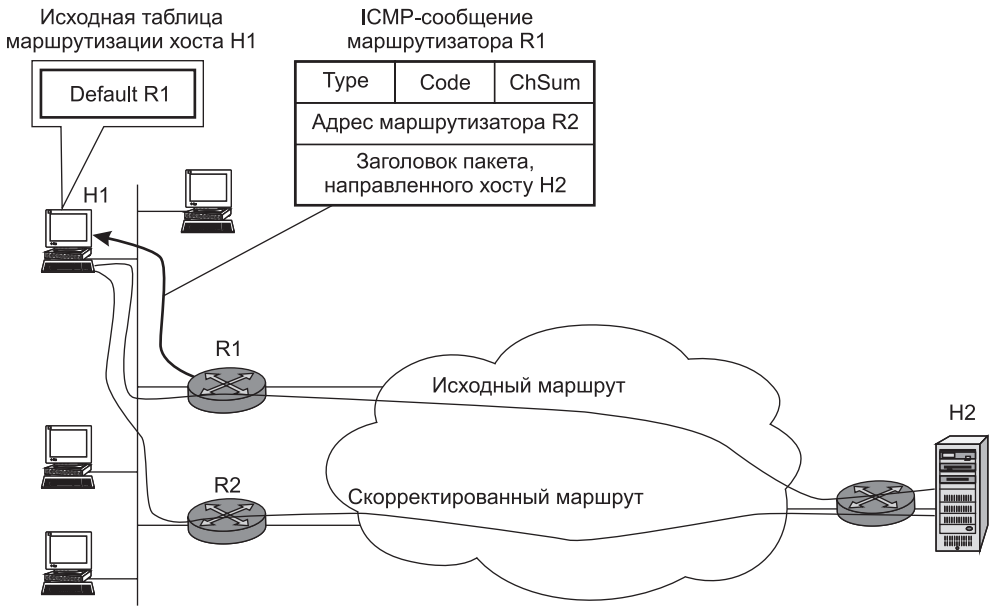



**Рис. 29.2.** Перенаправление маршрута предлагаемым по умолчанию маршрутизатором

Для перехвата трафика, направляемого хостом H1 хосту H2, злоумышленник должен сформировать и послать хосту H1 пакет, маскирующийся под ICMP-сообщение о перенаправлении маршрута (рис. 29.3). В этом сообщении содержится запрос о корректировке таблицы маршрутизации хоста H1, предусматривающей установку во всех пакетах с адресом $IP_{H2}$ в качестве адреса следующего маршрутизатора адреса $IP_{HA}$, являющегося де-факто адресом хоста-злоумышленника HA.

Чтобы хост «поверил» этому сообщению, в поле IP-адреса отправителя должен быть помещен адрес предлагаемого по умолчанию маршрутизатора R1. Когда пакеты, передаваемые введенным в заблуждение хостом, начнут поступать на узел злоумышленника, он может либо захватывать и не передавать эти пакеты дальше, имитируя для поддержания диалога приложение, которому эти пакеты предназначались, либо организовать транзитную передачу данных по указанному адресу назначения $IP_{H2}$. Читая трафик между узлами H1 и H2,

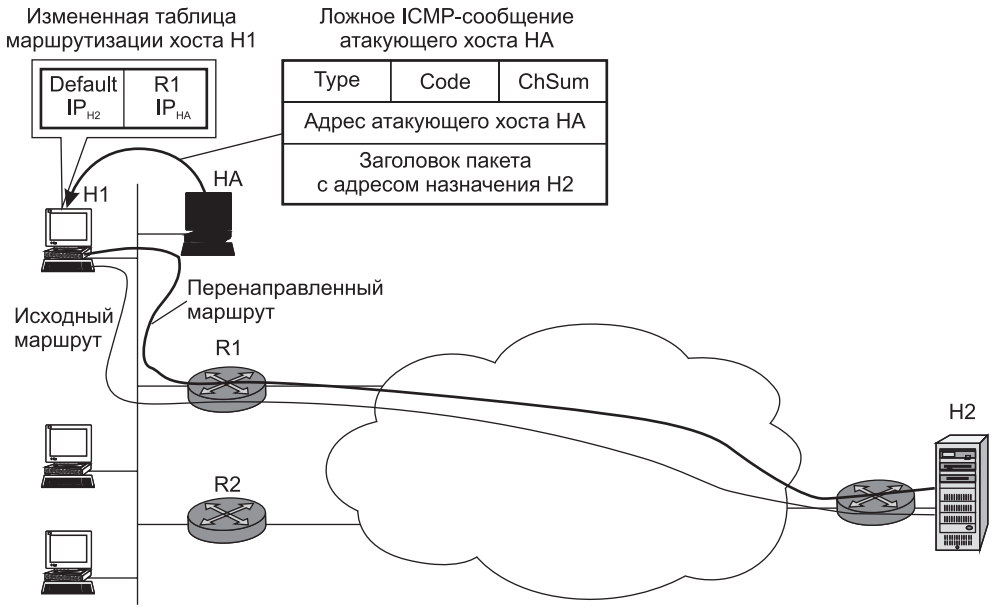



**Рис. 29.3.** Перенаправление маршрута злоумышленником

злоумышленник получает всю необходимую информацию для несанкционированного доступа к серверу H2. Сами маршрутизаторы также могут реагировать на ICMP-сообщения о перенаправлении маршрута, но обычно провайдеры отключают эту опцию для предотвращения атак данного типа.

Заметим, что простейший вариант перенаправления трафика в локальной сети может быть осуществлен путем отправки в сеть *ложного ARP-ответа*. В данном случае схема очевидна: получив широковещательный ARP-запрос относительно некоторого IP-адреса, злоумышленник посылает ложный ARP-ответ, в котором сообщается, что данному IP-адресу соответствует его собственный MAC-адрес.

**ICMP-атака Smurf** — это DDoS-атака, использующая функцию *эхо-запроса* протокола ICMP. Название атаки произошло от имени файла smurf.c, содержащего код атаки и получившего распространение в 1998 году.

Эхо-запросы и эхо-ответы протокола ICMP больше известны по утилите *ping*[1], с помощью которой можно проверить достижимость узла Интернета. Для проверки достижимости утилита ping посылает тестируемому узлу ICMP-пакет, в котором в качестве типа сообщения указан код 8 (эхо-запрос). Получив его, тестируемый узел отправляет в обратном направлении ICMP-пакет с кодом 0 (эхо-ответ). Атака Smurf тоже строится на возможности отправки эхо-запроса не только по индивидуальному, но и по *широковещательному* (broadcast) адресу некоторой сети. Например, если у сети адрес 200.200.100.0/24, то ее широковещательный адрес — 200.200.100.255, и эхо-запрос должен быть доставлен всем узлам этой сети (рис. 29.4).

---

[1] См. раздел «Утилита ping» в главе 14.

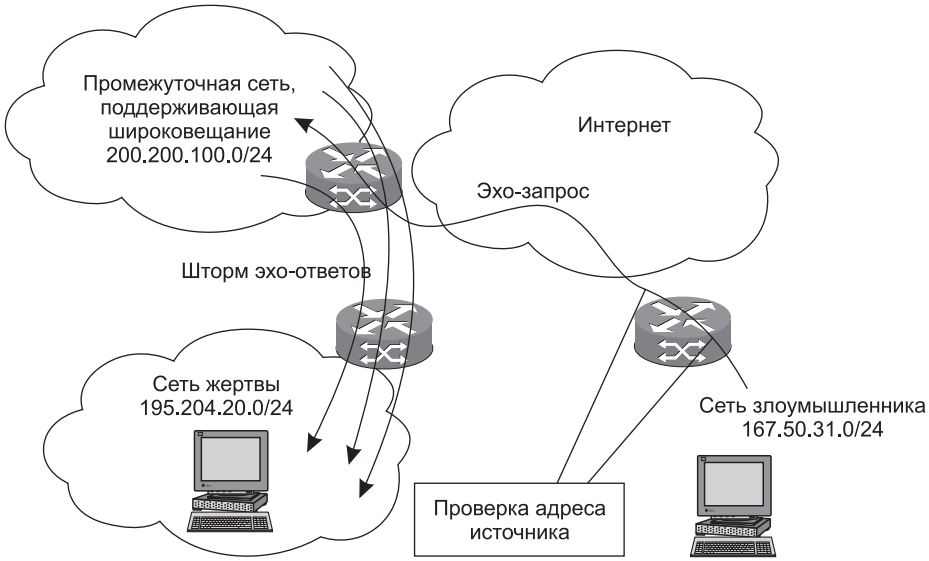



**Рис. 29.4.** Компоненты ICMP-атаки Smurf

Компьютер злоумышленника с адресом 167.50.31.17 находится в сети 167.50.31.0/24, а атакуемый компьютер имеет адрес 195.204.20.145 и подключен к сети 195.204.20.0/24. Компьютер злоумышленника генерирует эхо-запросы с адресом приемника 200.200.100.255 и адресом источника 195.204.20.145. Эхо-запросы передаются через Интернет в сеть 200.200.100.0.24 и принимаются всеми узлами этой сети, которые отвечают на ICMP-запросы эхо-ответами. В том случае, когда в сети 200.200.100.255 имеется достаточно большое количество активных узлов (понятно, что их не может быть более 254), на атакуемый узел 195.204.20.145 приходит интенсивный поток эхо-ответов, так как именно его адрес указан в эхо-запросах как адрес источника. В результате сетевой интерфейс атакуемого компьютера оказывается затопленным эхо-ответами и при превышении интенсивности этого потока некоторой величины его пропускная способность оказывается исчерпанной.

В ICMP-атаке Smurf используется характерный прием — *усиление атаки за счет отражения* посланного пакета большим количеством компьютеров. Необходимо, впрочем, отметить, что сегодня ICMP-атака Smurf представляет скорее исторический интерес. До 1999 года передача через Интернет IP-пакета с широковещательным адресом была обязательной для маршрутизаторов Интернета, но из-за атак, подобных Smurf, в стандарты было внесено изменение, и сегодня предлагаемым по умолчанию режимом является *фильтрация пакетов с широковещательными адресами*. Кроме того, промежуточная сеть, узлы которой используются для отражения эхо-запроса, может быть экранирована с помощью файервола от эхо-запросов, приходящих из внешних сетей.

Атака с драматическим названием «**Пинг смерти**» (Ping of Death) состоит в отправке на атакуемый компьютер эхо-запроса в IP-пакете, длина которого превышает его допустимый размер, составляющий, согласно стандарту, 65 535 байт. Поскольку соответствующий буфер ядра ОС не рассчитан на такую длину пакета, ОС терпит крах, отсюда и название атаки, основанной на превышении размера буфера при сборке фрагментированного IP-пакета

и, таким образом, являющейся частным случаем атак, использующих IP-фрагментацию (см. далее). К слову, база для атаки «Пинг смерти» давно ликвидирована — разработчики ОС еще в середине 90-х годов ввели в стек TCP/IP проверку длины собираемого фрагментированного IP-пакета.

Другая атака, называемая **Ping-затоплением**, также является достаточно простой — злоумышленник использует утилиту ping своей ОС для отправки эхо-запросов на атакуемый компьютер с максимально возможной частотой. Если быстродействие сетевого интерфейса его компьютера выше, чем у атакуемого компьютера, то атака удается, так как вся входная пропускная способность интерфейса атакуемого компьютера оказывается исчерпанной. К тому же атакуемый компьютер будет успевать отвечать на часть эхо-запросов эхо-ответами, что приведет к частичному исчерпанию пропускной способности в выходном направлении, а также к замедлению работы программ из-за отвлечения центрального процессора на обработку эхо-запросов.

## UDP-атаки

**Атака UDP-затоплением** относится к DoS-атакам и имеет целью исчерпание пропускной способности интерфейса атакуемого компьютера. Она подобна только что рассмотренной атаке Ping-затопления, когда злоумышленник просто направляет интенсивный поток UDP-дейтаграмм на атакуемый компьютер. Поскольку протокол UDP работает без установления соединения, то атакуемый компьютер обязан принимать все направляемые ему UDP-дейтаграммы, так как не может, как это делается при обмене данными по протоколу TCP, заставить передающий компьютер ограничить интенсивность потока направляемых ему пакетов, уменьшив размер окна приема. Злоумышленник может использовать аппаратный генератор трафика для того, чтобы генерировать UDP-трафик с максимально возможной скоростью выходного интерфейса, игнорируя ответные ICMP-сообщения в тех случаях, когда у атакуемого компьютера программный порт, указанный в UDP-пакетах, не открыт.

Слабым местом такого вида атак является то, что их интенсивность принципиально ограничена производительностью интерфейса атакующего компьютера. Имея стандартный для пользовательского компьютера интерфейс 1 Гбит/с, невозможно затопить UDP-пакетами сервер с интерфейсом 10 Гбит/с. Злоумышленник может преодолеть это ограничение, если в его распоряжении имеется сеть ботов. Именно такой подход был использован в 2007 году, когда была осуществлена массированная DDoS-атака UDP-затоплением на корневые DNS-серверы, при этом трафик создавался примерно пятью тысячами ботов (подробнее см. раздел «Атаки на DNS»).

**Атака ICMP/UDP-затоплением** имеет двойное имя, так как в ней используется два протокола. Злоумышленник направляет интенсивный поток UDP-пакетов, в которых в качестве адреса источника указан адрес компьютера-жертвы, на программные порты компьютеров, находящиеся в пассивном состоянии (то есть в данный момент с этими портами не связаны приложения, слушающие сеть). При получении UDP-пакета с номером пассивного порта компьютер в соответствии с логикой работы стека TCP/IP отвечает ICMP-сообщением о недостижимости порта назначения, которое направляется атакуемому компьютеру. Как видно из описания, в атаке имеет место отражение от компьютеров промежуточной сети; в случае использования широковещательного адреса она становится DDoS-атакой. Для предотвращения этой атаки применяют те же меры, что и для предотвращения ICMP-атаки Smurf — дополнительно реализуется пропуск файерволом только тех UDP-пакетов, порты

которых соответствуют активным приложениям компьютеров сети. Кроме того, можно ограничить интенсивность сообщений о недостижимости порта назначения компьютеров сети.

**Атака UDP/echo/chargen-затоплением** похожа на описанную выше — в ней также имеет место отражение UDP-пакетов, но при этом пакеты отправляются с номером порта 7 или 19. Эти порты обычно активны, они поддерживают сервисы echo (порт 7) и chargen (порт 19), использующие протокол UDP. Сервис chargen в ответ на запрос генерирует строку случайных символов случайной длины от 0 до 512 и посылает ее обратившемуся хосту. Этот сервис был встроен в ОС Unix для отладки ее сетевых функций. Аналогичное назначение имеет сервис echo (не путать с эхо-запросами и эхо-ответами протокола ICMP), возвращающий строку любого запроса по адресу обратившегося хоста. В простейшем случае атакующий посылает UDP-пакеты на порт 7 и/или 19 некоторого промежуточного хоста и указывает обратный адрес атакуемого хоста. Промежуточный хост начинает «бомбардировать» атакуемый хост ответами сервисов chargen и/или echo. При этом усиления атаки не происходит, так как объем ответных сообщений невелик; для усиления может быть использован широковещательный адрес промежуточной сети. Более интересной выглядит атака, когда атакующий посылает пакет с портом 19 и указывает в нем исходный порт 7. В этом случае единственный пакет атакующего вызывает бесконечный обмен пакетами между сервисом chargen промежуточного хоста и сервисом echo атакуемого хоста.

## IP-атаки

Протокол IP сам по себе не предоставляет злоумышленникам особых шансов для атак, так как работает без установления соединения и достаточно прост в реализации. Тем не менее некоторые возможности для IP-атак существуют. Рассмотрим их.

**Атака на IP-опции** представляет собой DoS-атаку на маршрутизаторы, в которой используется поле дополнительных опций протокола IP.

В *IPv4* заголовок IP-пакета может включать поле опций, задающих некоторую *нестандартную обработку* пакета маршрутизатором. Например, существует опция строгой маршрутизации от источника, позволяющая отправителю IP-пакета задать точный список адресов промежуточных маршрутизаторов, через которые должен проходить маршрут доставки пакета, в то время как опция свободной маршрутизации от источника задает лишь некоторые из промежуточных маршрутизаторов маршрута. Опция фиксации маршрута требует от маршрутизаторов фиксации в пакете адресов промежуточных маршрутизаторов, передающих пакет. Отметим, у производителей маршрутизаторов имеется возможность определять свои типы опций.

Поскольку у большинства IP-пакетов поле опций отсутствует, для продвижения пакетов маршрутизатор задействует специализированные *процессоры портов*, выполняющих эту операцию очень быстро и экономно. Но если встречается пакет с полем опций, то процессор порта его обработать не может и передает пакет *центральному процессору* маршрутизатора. В результате обработка трафика замедляется. Поэтому если на маршрутизатор поступает интенсивный поток пакетов, у которых присутствует одна или несколько опций, то его работа может существенно замедлиться, вплоть до отказа в обслуживании нормальных пакетов. Ситуация усугубляется, когда в пакете указаны две взаимоисключающие опции, например, строгой маршрутизации от источника и свободной маршрутизации от источника с разными промежуточными адресами.

Спецификация *IPv6* допускает наличие нескольких заголовков в пакете — основного и нескольких дополнительных. Вместо полей опций в пакете IPv6 могут присутствовать дополнительные заголовки, одним из которых является заголовок пошаговых опций (Hop-by-hop Options). Как и в случае опций IPv4, опции дополнительного заголовка пошаговых опций IPv6 обрабатываются центральным процессором маршрутизатора. Помещение в такой заголовок большого числа опций неопределенного типа будет замедлять работу маршрутизатора IPv6. Обычная практика борьбы с подобной атакой — фильтрация (отбрасывание) всех пакетов, в заголовке которых имеются опции. Возможно также игнорирование всех или некоторых опций.

**Атака на фрагментацию** направлена на конечные узлы IP-сетей, в обязанность которых входит сборка фрагментированного IP-пакета. Для подобных атак используются некоторые уязвимости, присущие операциям сборки, например:

- *Превышение максимальной длины пакета* (переполнение буфера сборки). Этот способ атаки уже был упомянут при описании атаки «Пинг смерти». Максимальное значение смещения фрагмента равно $(2^{13} - 1) \times 8 = 8191 \times 8 = 65\,528$. Так как максимальная длина IP-пакета равна 65 535 байт, очевидно, что последний фрагмент не должен иметь длину более 7 байт. Задавая фрагмент с максимальным смещением и размером в 8 и более байт, злоумышленник переполняет буфер ядра ОС, что может привести к падению ОС.
- *Перекрытие сегментов за счет специального подбора смещений и длин фрагментов*. Некоторые ОС не справляются со сборкой таких пакетов и падают. Например, эта уязвимость используется в атаке Teardrop.
- *Замещение фрагментов*. Эта DoS-атака используется для обмана таких защитных средств, как файерволы и системы обнаружения вторжений. Пакеты атаки фрагментируются и посылаются вместе с фрагментами-дубликатами, в которых содержится безобидная информация. Первым посылается безобидный фрагмент, а следом — фрагмент, содержащий код атаки, но с такими же смещением и длиной. В результате фрагмент атаки замещает безобидный фрагмент. Не все файерволы и системы обнаружения вторжений распознают фрагментированную таким образом атаку.

## Сетевая разведка

Как можно увидеть из описания атак на транспортные протоколы, многие из них требуют предварительных знаний об атакуемой сети и ее хостах. Например, для проведения ICMP-атаки Smurf нужно найти промежуточную сеть с большим количеством хостов, отвечающих на эхо-запросы, при этом такая сеть должна быть достижима для пакетов с широковещательным адресом этой сети, посланных из сети злоумышленника; безусловно, должен быть известен и *IP-адрес* атакуемого компьютера. Если злоумышленник хочет задействовать сеть ботов, зараженных вирусом определенного типа, то ему понадобится просканировать большое количество компьютеров на отклик по определенному *порту*, который используется этим вирусом для получения команд от контроллера атаки. Дело в том, что вирусы стараются распространиться на возможно большее число компьютеров, но нельзя сказать заранее, будет ли успешным такое внедрение для какого-то определенного хоста — это зависит от конфигурации средств защиты и других параметров ОС хоста. Поэтому злоумышленник заранее не знает, какие хосты он может использовать в качестве членов сети ботов, даже если именно он инициировал распространение этого вируса. А возможно, он

просто решит воспользоваться известным вирусом, распространенным другими лицами, и поэтому ему нужно собрать сведения о зараженных компьютерах.

Вот почему почти любую атаку предваряет **сетевая разведка**, при которой злоумышленник пытается собрать необходимые для атаки сведения. Конкретный набор сведений зависит от типа атаки, но чаще всего сетевая разведка включает сбор следующих данных: IP-адреса активных (то есть включенных, отвечающих на сетевой трафик) хостов; номера активных TCP-портов; номера активных и пассивных UDP-портов хостов; тип и версии ОС и приложений.

Обнаружение IP-адресов активных хостов сети называют **сканированием сети** (network scanning), а активных и пассивных портов — **сканированием портов** (port scanning). Сам термин «сканирование» говорит о том, что злоумышленник тестирует один за другим все возможные значения IP-адресов некоторой подсети (например, для подсети с маской /24 — 254 значения) или номера портов (65 535 номеров и для TCP, и для UDP). Вот наиболее распространенные приемы сканирования сети:

- **Пинг**[1] **TCP SYN** к одному из публично доступных портов, чаще всего к порту 80 (порт веб-сервера), который с большой степенью вероятности (но, конечно, не обязательно) открыт для внешнего доступа. Если хост отвечает пакетом SYN/ACK, то сканер считает, что *хост активен*, и завершает TCP-соединение пакетом с признаком RST.
- **Пинг TCP ACK** позволяет во многих случаях обойти файервол, если тот блокирует выбранный порт. Обычной практикой конфигурирования файервола является разрешение трафика *уже установленных* TCP-соединений, а признаком принадлежности пакета к такому соединению является наличие установленного флага ACK. В том случае, когда пинг TCP SYN к некоторому порту не проходит, а TCP ACK проходит, результатом сканирования становится заключение: *хост активен, но защищен файерволом* — такая информация может быть ценной для злоумышленника.
- **Пинг UDP**. На тестируемый хост направляется UDP-пакет с номером порта, который, как рассчитывает злоумышленник, с большой степенью вероятности является *пассивным*. В том случае, когда компьютер включен и этот порт пассивен, сканер получает в ответ ICMP-сообщение о недоступности порта; если же компьютер отключен, то злоумышленник получает сообщение от маршрутизатора о недоступности хоста.
- **Пинг ICMP**. Администраторы сетей чаще всего блокируют эхо-запросы протокола ICMP, однако для проверки активности хоста злоумышленник может использовать *другие типы ICMP-запросов*, например запрос о длине маски IP-адреса (код 17) или запрос синхронизации времени протокола ICMP (код 13).
- **Пинг IP**. На исследуемый компьютер направляется IP-пакет с кодом протокола, отличным от кодов протоколов TCP, UDP и ICMP. Скорее всего, такой тип протокола не поддерживается стеком TCP/IP данного компьютера, и в том случае, если хост активен, в ответ будет послано ICMP-сообщение о недостижимости протокола.

Подобные же методы применяются и для *сканирования портов*. Здесь предпочтение отдается SYN-сканированию протокола TCP, так как это самый быстрый способ, что в данном случае имеет значение — в отличие от сканирования хостов здесь проверяются десятки тысяч портов (по 65 535 портов для TCP и UDP). Сканирование портов часто осуществляется

---

[1] В этом перечне процедур термин «пинг» использован в широком смысле — он не указывает конкретно на утилиту ping, работающую по протоколу ICMP, а говорит о том, что данные процедуры, подобно утилите ping, тестируют активность хоста с определенным IP-адресом.

с помощью тех же специализированных программных средств, что и для инвентаризации сети и аудита ее защищенности.

Сканирование сети и портов обычно не проходит незамеченным — очень вероятно, что средства протоколирования событий ОС и файерволов зафиксируют этот процесс, и администратор сканируемой сети начнет расследовать инцидент. И первый вопрос, возникающий при этом: с какого адреса выполнялось сканирование? Чтобы избежать раскрытия, злоумышленники часто используют *спуфинг IP-адреса*. На первый взгляд кажется, что в сетевой разведке этот прием не может сработать, так как злоумышленнику нужно получать ответы на свой компьютер, иначе он не может получать информацию. Тем не менее спуфинг IP-адреса возможен и при сканировании. Одним из приемов является маскировка его среди множества других адресов: тестовые сканирующие пакеты отправляются с действительного IP-адреса наряду с множеством таких же пакетов, но с поддельными адресами. Расчет здесь делается на то, что при расследовании факта сканирования трудно будет установить, кто являлся истинным организатором сканирования, а кого просто использовали как прикрытие. Еще более изощренным является так называемое «пустое» сканирование, когда истинный адрес никогда не указывается, а результаты сканирования злоумышленники пытаются понять по реакции третьего компьютера, адрес которого подделывается.

# Атаки на DNS

Центральная роль службы DNS делает ее, с одной стороны, желанной *целью* атакующего, поскольку нарушение работы DNS наносит огромный ущерб работе сети, а с другой — мощным *средством* для проведения атак на другие сетевые механизмы, потому что многие из них оказываются безоружными перед этой глобальной службой.

## DNS-спуфинг

В этой атаке DNS является не целью, а средством (рис. 29.5). Рассмотрим пример, в котором злоумышленник использует **DNS-спуфинг** для получения доступа к корпоративному серверу www.example.com. Для этого ему нужны аутентификационные данные какого-нибудь его клиента.

Он решает перенаправить поток данных, которые легальный корпоративный клиент посылает корпоративному серверу, на свой компьютер. Для этого нужно опередить ответ DNS-сервера резольверу клиента и навязать ему свой вариант ответа, в котором вместо IP-адреса корпоративного сервера (в примере — 193.25.34.125) злоумышленник указывает IP-адрес атакующего хоста (203.13.1.123). На пути реализации этого плана имеется несколько серьезных препятствий.

Прежде всего необходимо задержать ответ DNS-сервера, например, подвергнув его DoS-атаке. Другая проблема связана с определением номера порта DNS-клиента, который необходимо указать в заголовке пакета, чтобы данные дошли до приложения, так как если серверная часть DNS имеет постоянно закрепленный за ней так называемый хорошо известный номер порта 53, то клиентская часть протокола DNS получает номер порта динамически при запуске, причем ОС выбирает его из достаточно широкого диапазона. Эту задачу злоумышленник решает путем прямого перебора всех возможных номеров.

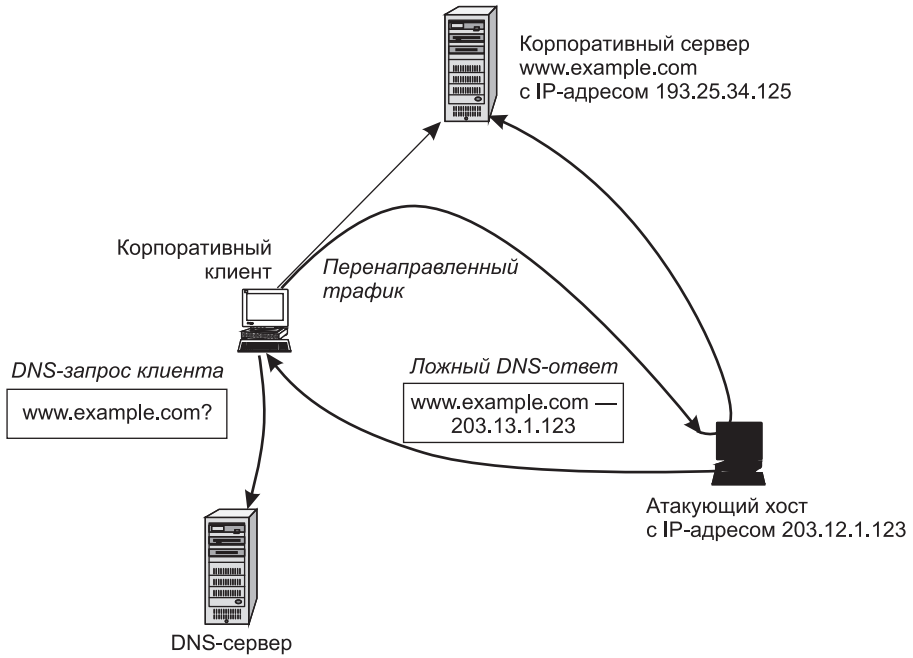



**Рис. 29.5.** Схема перенаправления трафика путем использования ложных DNS-ответов

Также путем перебора возможных значений преодолевается и проблема определения идентификаторов DNS-сообщений. Эти идентификаторы передаются в DNS-сообщениях и служат для того, чтобы DNS-клиент мог установить соответствие поступающих ответов посланным запросам. Итак, злоумышленник «бомбардирует» клиентскую машину ложными DNS-ответами, перебирая все возможные значения идентифицирующих полей так, чтобы клиент, в конце концов, принял один из них за истинный DNS-ответ. Как только это происходит, цель злоумышленника можно считать достигнутой: пакеты от клиента направляются на адрес атакующего хоста, злоумышленник получает в свое распоряжение имя и пароль легального пользователя, а с ними — и доступ к корпоративному серверу.

## Атаки на корневые DNS-серверы

Наиболее мощными и ощутимыми по своим последствиям были DDoS-атаки на корневые серверы, случившиеся 21 октября 2002 года и 6–7 февраля 2007-го.

Подробности **атаки 21 октября 2002 года** приводятся в отчете специалистов, администрирующих корневые серверы[1]:

- Атака длилась чуть больше часа и была направлена на все 13 корневых серверов.
- Атака была комбинированной: использовались методы ICMP-атаки ping-затоплением, TCP-атаки затоплением SYN-пакетами, атаки фрагментированными IP-пакетами и атаки UDP-затоплением.

[1] http://d.root-servers.org/october21.txt.

- Интенсивность атаки на сервер — 50–100 Мбит/с; суммарная интенсивность — 900 Мбит/с.
- В атаке использовался спуфинг IP-адресов, отследить реальные источники атаки не удалось.

Служба DNS показала хорошую устойчивость к атаке — пользователи замечали только небольшое увеличение времени ожидания при открытии сайта в браузере; все корневые серверы продолжали работать, и на все принятые ими запросы были даны ответы, но из-за перегрузки входных интерфейсов некоторых серверов не все запросы были приняты. После этой атаки были проведены дополнительные работы по повышению устойчивости службы DNS, которые включали повышение скорости интерфейсов и линий связи, соединяющих корневые серверы с Интернетом, и увеличение числа корневых серверов. Кроме того, корневые серверы были более равномерно распределены по автономным системам и географическим регионам.

**Атака 6–7 февраля 2007 года** длилась 24 часа и была намного мощнее, чем атака 21 октября 2002 года, — интенсивность трафика достигала 1 Гбит/с на один пул корневых серверов, но атаковано было только четыре из них. В атаке было использовано 4500–5000 компьютеров под управлением Microsoft Windows, причем члены этой сети ботов были распределены по сетям нескольких стран. При атаке имело место затопление корневых серверов UDP-пакетами, направленными на порт 53 (порт DNS), то есть атака относилась к типу UDP-затопления, а использование порта 53 помогало пакетам добраться до серверов, так как у файерволов, защищающих DNS-серверы, этот порт всегда открыт, иначе сервер не смог бы выполнять свою работу. Атака привела к почти полному исчерпанию пропускной способности двух из четырех атакованных пулов серверов, но два других пострадали не так существенно и смогли отвечать на большую часть запросов.

Атака практически немедленно была обнаружена центрами, ответственными за администрирование атакованных пулов корневых серверов (по предупреждающим сообщениям самих серверов и данным хостов, выполняющих постоянный мониторинг корневых серверов путем отправки на них контрольных запросов). Для снижения эффекта атаки был предпринят ряд мер, первой из которых явилась блокировка любых DNS-запросов, длина которых превышала 300 байт (обычно длина DNS-запроса — не больше 100 байт), а в атакующих сообщениях для усиления эффекта затопления размеры полей данных UDP-потока доходили до 1023 байт. Однако такая блокировка помогла только частично — последующий анализ показал, что размер поля данных трафика атаки менялся случайным образом от 0 до 1023 байт, при этом атакующие компьютеры не использовали спуфинг IP-адресов, что дало возможность отследить размещение ботов: Южная Корея — 65 %, США — 19 %, Канада — 3,5 %, Китай — 2,5 %, остальные страны — 10 %. Хост, координирующий атаку, находился в США, хосты сети ботов обращались к нему по протоколу HTTP.

Причины атаки остались неясными; в отчете ICANN предполагается, что атака — проявление тщеславия хакеров: ведь попытка остановить весь Интернет является вызовом для любого хакера.

Мы остановились на этих двух примерах, так как они дают хорошее представление о масштабах современных DDoS-атак и о том, что защититься от них очень сложно даже таким опытным специалистам, которые обслуживают корневые DNS-серверы. В то же время такое архитектурное решение, как виртуализация серверов, когда логический сервер представлен большим пулом физических серверов, рассредоточенных по разным сетям и автономным системам, способно гасить эффект даже очень интенсивной DDoS-атаки.

В эффективности такого подхода мы еще раз убедимся в главе 30 при рассмотрении безопасности облачных вычислений.

## DDoS-атаки отражением от DNS-серверов

Основная идея **атаки отражением от DNS-серверов** базируется на следующем. В Интернете работают миллионы DNS-серверов, отвечающих за отправку ответов на запросы клиентов. При этом ответ может по объему намного превосходить запрос — например, если запрос относится к передаче файла зоны (запрос типа AXFR) и зона включает большое количество записей. В марте 2013 года атаке такого рода[1] подвергся веб-сервер компании Spamhouse — некоммерческой организации, борющейся со спамом (рис. 29.6).

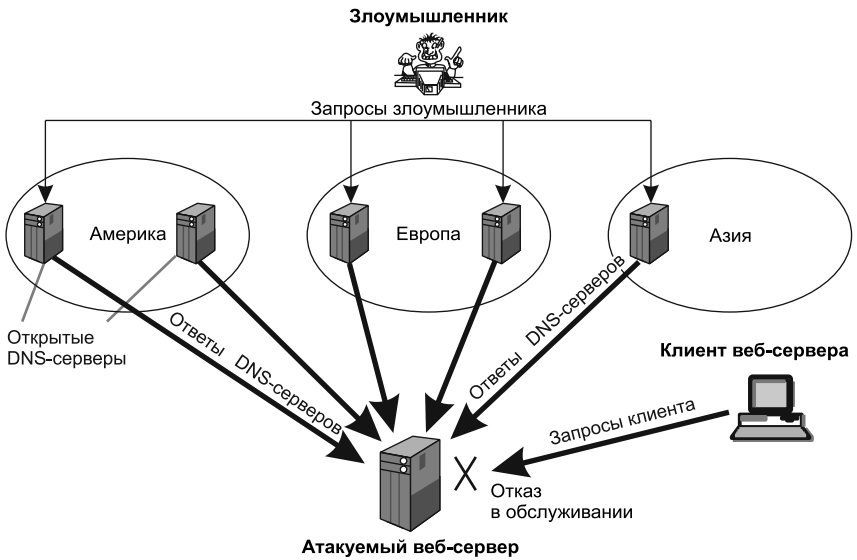



**Рис. 29.6.** Атака отражением от DNS- серверов

Для организации этой атаки было использовано около 30 000 DNS-серверов, работающих в открытом рекурсивном режиме, то есть отвечающих на запросы любых пользователей и при этом дающих полный (рекурсивный) ответ. Рекурсивный режим здесь является важным элементом атаки, так как нерекурсивные DNS-серверы только перенаправляют запрашивающего на другой DNS-сервер — их ответ является коротким и не может усилить атаку. Общей практикой является поддержание рекурсивного режима ответов только для «своих» клиентов — сотрудников предприятия для корпоративного DNS-сервера или же подписчиков сервиса для интернет-провайдера. Поскольку в Интернете имеется около 28 миллионов открытых рекурсивных DNS-серверов, найти объекты для атаки оказалось не так уж трудно.

Злоумышленником был послан поток запросов на 30 000 открытых DNS-серверов, в качестве адреса отправителя которых указывался адрес атакуемого веб-сервера. Ответы от 30 000 DNS-серверов обрушились на веб-сервер компании Spamhouse.

---

[1] http://blog.cloudflare.com/the-ddos-that-knocked-spamhaus-offline-and-ho.

Для усиления атаки использовались не обычные запросы на разрешение имени, а запросы на передачу объемного файла со всеми записями зоны ripe.net (RIPE NCC — это региональный информационный интернет-центр по Европе). Файл зоны ripe.net имеет размер около 3000 байт, так что при размере запроса в 28 байт коэффициент усиления составил около 100. Такое мощное усиление позволило, используя поток запросов к одному DNS-серверу интенсивностью всего в 2,5 Мбит/с, создать атаку с общей интенсивностью в 75 Гбит/с. Для отдельного DNS-сервера такой поток запросов не является чем-то необычным, так что владельцы этих серверов, скорее всего, эту атаку не заметили, а вот результирующий поток атаки в 75 Гбит/с вывел веб-сервер компании Spamhouse из строя.

Точнее, веб-сервер Spamhouse был выведен из строя только до определенного момента, пока его владельцы не перевели его «под крыло» CloudFlare — провайдера облачных сервисов, к тому же специализирующегося на защите от DDoS-атак. Перевод помог, так как распределенная виртуальная структура CloudFlare, использующая технику anycast и файерволы, смогла абсорбировать большую часть трафика атаки, поле чего веб-сервер Spamhouse вновь стал доступен пользователям Интернета.

## Методы защиты службы DNS

Существует ряд мер предосторожности, которые повышают защищенность DNS-серверов от атак или использования их в качестве инструмента атаки:

- *Защита ОС хоста.* Так как DNS — это приложение ОС, то сама ОС должна быть надежно защищена всеми возможными способами.
- *Разделение пользователей на внутренних и внешних.* Рекурсивные неполномочные ответы должны предоставляться только внутренним пользователям как вызывающим большее доверие.
- *Передача файла зоны из первичного сервера только вторичным серверам этой зоны* с использованием для передачи защищенных протоколов, например SFTP или SCP.
- *Использование DNSSEC.* **DNSSEC** представляет собой набор стандартов, обеспечивающих аутентификацию ответов DNS-серверов с помощью цифровой подписи и системы публичных ключей. DNSSEC-клиент может проверить, что полученный ответ действительно пришел от полномочного сервера зоны, а не от сервера, который просто утверждает, что он полномочен, а на самом деле может таковым и не являться. DNSSEC затрудняет злоумышленникам спуфинг-атаки, так как для этого требуется подделывать цифровую подпись сервера. С 2010 года все корневые серверы, а также многие серверы верхнего уровня и крупных провайдеров поддерживают DNSSEC.

# Безопасность маршрутизации на основе BGP

## Уязвимости протокола BGP

Маршрутизация между автономными системами на основе протокола BGP является (наряду со службой DNS) одним из наиболее уязвимых элементов Интернета. Это объясняется, во-первых, *тяжелыми последствиями*, к которым приводит ошибочная работа

BGP-маршрутизаторов сети провайдеров: маршруты ко многим частям Интернета вдруг исчезают или оказываются ложными для значительной части пользователей. Во-вторых, протокол BGP принципиально *менее защищен*, чем внутренние протоколы маршрутизации OSPF и IS-IS, так как «собеседники» BGP-маршрутизатора находятся за пределами административной ответственности его организации, и поэтому возможностей для проверки достоверности маршрутных объявлений протокола BGP намного меньше, чем в случае внутренних протоколов.

Мы сознательно не использовали в заголовке этого раздела слово «атаки». Информация о случаях неправильной работы протокола BGP часто скрывается провайдерами Интернета, избегающими раскрытия деталей работы своей сети по разным причинам, в том числе и по причине безопасности. Поэтому большая часть крупных инцидентов, произошедших в Интернете по «вине» протокола BGP, остается инцидентами, а не атаками, так как трудно сказать, произошел тот или иной инцидент из-за ошибки конфигурирования маршрутизатора персоналом провайдера или же это была спланированная и осуществленная атака. Во многих статьях и документах, описывающих уязвимости этого протокола маршрутизации, появилось новое действующее лицо — *провайдер-злоумышленник* (malicious ISP), который вольно или невольно создает проблемы для остальных провайдеров.

Инциденты с маршрутизацией в Интернете являются следствием того, что маршрутное объявление протокола BGP формируется шаг за шагом многими провайдерами, при этом достоверность информации каждого шага проверить невозможно, так что у провайдера имеется полная свобода действий при обработке маршрутного объявления и передаче его соседним провайдерам. Вместо корректного объявления о префиксе своей сети с указанием номера своей автономной системы как исходной или добавления номера своей AS к уже имеющейся последовательности AS-номеров при трансляции объявления о чужой сети, он может выполнить ряд некорректных манипуляций с маршрутным объявлением, например:

- поместить адрес чужой сети с номером своей автономной системы в качестве исходной, чтобы ложно направить трафик к этой сети в свою автономную систему. При этом адрес чужой сети в некорректном объявлении BGP должен быть *более специфическим*, чем уже существующие корректные записи об этой сети в маршрутизаторах других провайдеров, тогда новая ложная запись станет более приоритетной и будет использоваться вместо корректных. Такой тип инцидента называется *захватом префикса*;
- выбросить из последовательности какую-то определенную автономную систему, чтобы обойти политику некоторой третьей автономной системы, которая по финансовым или иным соображениям блокирует все маршруты, проходящие через эту автономную систему;
- добавить номер соседней автономной системы перед передачей ее объявления, чтобы соседняя автономная система, получив объявление и увидев в нем свой номер, отбросила его, решив, что объявление зациклилось;
- добавить номер своей автономной системы несколько раз, чтобы объявление стало непривлекательным для других провайдеров (из-за размера последовательности AS);
- составить ложную последовательность автономных систем, но поместить в качестве исходного правильный (но не свой) номер AS, чтобы вызвать доверие к маршруту.

Как видим, незащищенность маршрутного объявления дает большой простор для злонамеренных искажений и просто ошибок при его обработке. Вероятность ошибки усугубляется тем, что в отличие от внутренних протоколов маршрутизации, которые обрабатывают

сообщения с минимальным вмешательством администратора, работа протокола BGP обычно регулируется большим количеством правил фильтрации, которые задаются вручную администратором AS. Эти фильтры определяют *политику маршрутизации* той или иной автономной системы, отражающую взаимоотношения данного провайдера с каждым из провайдеров, с которым у него есть пиринговые соглашения о передаче трафика. Вместе с тем при правильном применении фильтры политики BGP являются мощным инструментом защиты BGP-маршрутизации от ошибок и атак.

## Инциденты с протоколом BGP

Первый широко известный масштабный инцидент с BGP-маршрутизацией ярко выявил уязвимости BGP, которые затем много раз проявляли себя в аналогичных ситуациях. Этот инцидент произошел 25 апреля 1997 года, когда немало провайдеров обнаружило, что в их маршрутизаторах исчезли маршруты, описывающие путь ко многим сетям Интернета. Причина этого неприятного обстоятельства довольно быстро была найдена: в объявлениях автономной системы **AS7007** указывалось, что путь к исчезнувшим сетям Интернета должен пролегать через ее сети, хотя на самом деле эта небольшая автономная система не являлась транзитной для этих маршрутов. Звонок провайдеру AS7007 помог устранить причину: оказалось, что виновником был единственный маршрутизатор, который после реконфигурирования начал генерировать некорректные маршрутные объявления. Некорректность состояла в том, что в объявлениях указывались адреса не собственных сетей AS7007, а адреса сетей, принадлежащих другим провайдерам Интернета. Кроме того, эти адреса оказались *более специфическими,* чем корректные адреса этих же сетей в маршрутизаторах провайдеров Интернета, поэтому весь трафик к этим сетям стал направляться в AS7007 и там теряться, так как всех этих сетей у названного провайдера не было. После отключения виновного маршрутизатора таблицы маршрутизации провайдеров быстро восстановились, однако шок от того, как просто оказалось вывести Интернет из строя, остался надолго.

Инцидент с AS7007 был первой масштабной демонстрацией уязвимости маршрутизации на основе протокола BGP — протокола, который, как и другие протоколы стека TCP/IP, был разработан в расчете на добрую волю всех пользователей Интернета и не имел никакой защиты от ошибок или злого умысла. В дальнейшем подобные инциденты повторялись достаточно регулярно, причем один из них привлек большое внимание, потому что привел к временной недоступности популярного сервиса Youtube — в 2008 году оператор Pakistan Telecom пытался заблокировать доступ к Youtube для пользователей Пакистана, а вместо этого распространил всем провайдерам Интернета объявления о том, что специфические маршруты к Youtube ведут через его сеть, что привело к направлению мирового трафика Youtube в сеть Pakistan Telecom в течение двух часов.

В представленном на рис. 29.7 примере диапазон адресов 12.24.0.0/16 принадлежит AS6. BGP-маршрутизатор этой автономной системы объявляет о достижимости данных адресов через свою сеть. Маршрутизаторы автономной системы AS5 принимают это объявление, помещают запись о достижимости адресов 12.24.0.0/16 в свои таблицы маршрутизации и передают его далее маршрутизаторам автономных систем AS4 и AS7. Теперь посмотрим, что произойдет, если маршрутизаторы автономной системы AS1 начнут по ошибке или умышленно распространять маршрутные объявления к адресам диапазона 12.24.128.0/17.

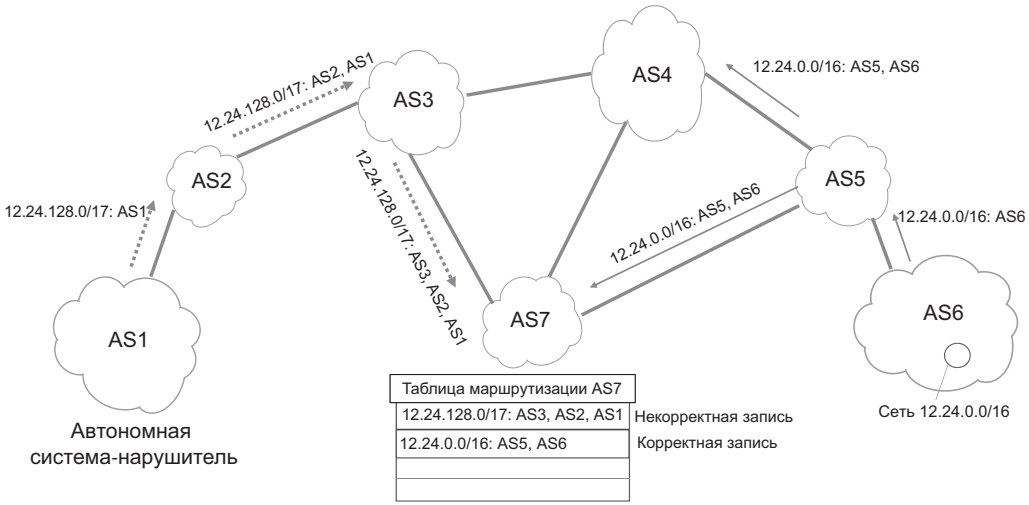



**Рис. 29.7.** Эффект захвата префикса адресов

Этот диапазон является поддиапазоном диапазона 12.24.0.0/16, но его префикс длиннее, то есть этот адрес является *более специфическим адресом*, чем адрес 12.24.0.0/16, поэтому маршрутизатор должен отдавать ему предпочтение перед более коротким адресом. В результате маршрутизаторы AS7 помещают запись о том, что трафик к хостам с адресами из диапазона 12.24.128.0/17 должен направляться в AS1, а не AS6. Эта информация распространится далее по всем автономным системам Интернета, в том числе по AS6, что приведет к потере связи с хостами 12.24.128.0/17, так как они находятся в AS6, а не в AS1. Если система AS1 будет распространять только объявление 12.24.128.0/17, то остальные хосты из диапазона 12.24.0.0/16 окажутся достижимыми в AS6, если же AS1 будет распространять подобные объявления для всех поддиапазонов этого диапазона с маской 17, такие как 12.24.0.0/17, 12.24.192.0/17, 12.24.224.0/17 и т. д., то все хосты сети 12.24.0.0/16 станут недостижимыми.

**(S)** *Защита BGP*

# Технологии защищенного канала

Известно, что задачу защиты данных можно разделить на две подзадачи: защиту данных внутри компьютера и защиту данных в процессе их передачи от одного компьютера к другому. Для обеспечения безопасности данных при их передаче по публичным сетям используются различные технологии защищенного канала.

**Технология защищенного канала** обеспечивает защиту трафика между двумя точками в открытой транспортной сети, например в Интернете. Защищенный канал подразумевает выполнение трех основных функций:

- взаимная аутентификация абонентов при установлении соединения, которая может быть выполнена, например, путем обмена паролями;

- защита передаваемых по каналу сообщений от несанкционированного доступа, например, путем шифрования;
- подтверждение целостности поступающих по каналу сообщений, например, путем передачи одновременно с сообщением его дайджеста.

## Способы образования защищенного канала

В зависимости от месторасположения программного обеспечения защищенного канала различают две схемы его образования:

- схема с конечными узлами, взаимодействующими через публичную сеть (рис. 29.8, *а*);
- схема с оборудованием поставщика услуг публичной сети, расположенным на границе между частной и публичной сетями (рис. 29.8, *б*).

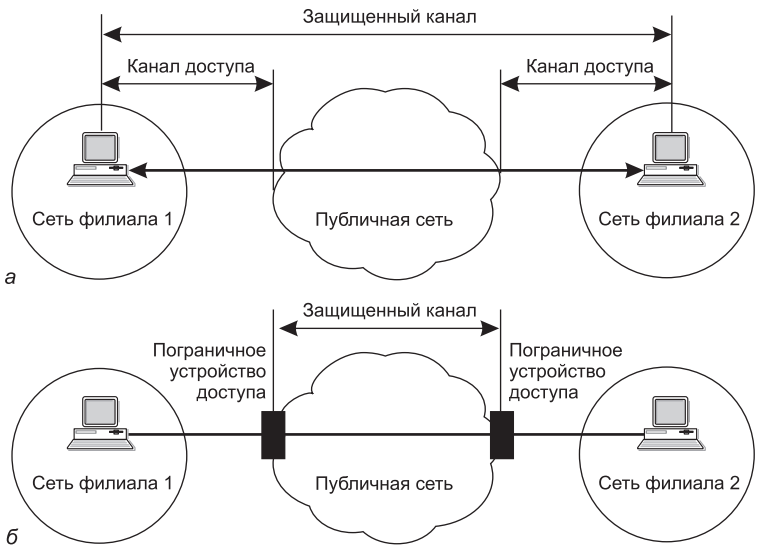



**Рис. 29.8.** Два подхода к образованию защищенного канала

В первом случае защищенный канал образуется программными средствами, установленными на двух удаленных компьютерах, принадлежащих двум разным локальным сетям одного предприятия и связанных между собой через публичную сеть. Преимуществом этого подхода является полная защищенность канала вдоль всего пути следования, а также возможность использования любых протоколов создания защищенных каналов, лишь бы на конечных точках канала поддерживался один и тот же протокол. Недостатки заключаются в избыточности и децентрализованности решения. Избыточность состоит в том, что вряд ли стоит создавать защищенный канал на всем пути следования данных: уязвимыми для злоумышленников обычно являются сети с коммутацией пакетов, а не каналы телефонной сети или выделенные каналы, через которые локальные сети подключены к территориальной сети. Поэтому защиту каналов доступа к публичной сети можно считать избыточной. Децентрализация заключается в том, что для каждого компьютера, которому требуется предоставить услуги защищенного канала, необходимо отдельно уста-

навливать, конфигурировать и администрировать программные средства защиты данных. Подключение каждого нового компьютера к защищенному каналу требует выполнять эти трудоемкие операции заново.

Во втором случае клиенты и серверы не участвуют в создании защищенного канала — он прокладывается только внутри публичной сети с коммутацией пакетов, например, внутри Интернета. Так, канал может быть проложен между сервером удаленного доступа поставщика услуг публичной сети и пограничным маршрутизатором корпоративной сети. Это — хорошо масштабируемое решение, управляемое централизованно администраторами как корпоративной сети, так и сети поставщика услуг. Для компьютеров корпоративной сети канал прозрачен — программное обеспечение этих конечных узлов остается без изменений. Такой гибкий подход позволяет легко образовывать новые каналы защищенного взаимодействия между компьютерами независимо от места их расположения. Реализация этого подхода сложнее, поскольку требуется и стандартный протокол образования защищенного канала, и установка у всех поставщиков услуг программного обеспечения, поддерживающего такой протокол, и поддержка протокола производителями пограничного коммуникационного оборудования. Кроме того, вариант, когда все заботы по поддержанию защищенного канала берет на себя поставщик услуг публичной сети, оставляет сомнения в надежности защиты: каналы доступа к публичной сети оказываются незащищенными, а потребитель услуг чувствует себя в полной зависимости от надежности их поставщика.

## Иерархия технологий защищенного канала

Защищенный канал можно построить с помощью системных средств, реализованных на разных уровнях модели OSI (рис. 29.9).


**Рис. 29.9.** Протоколы, формирующие защищенный канал на разных уровнях модели OSI

Если защита данных осуществляется средствами верхних уровней (прикладного, представления или сеансового), то такой способ защиты не зависит от технологий транспортировки

данных (IP или IPX, Ethernet или MPLS), что можно считать несомненным достоинством. В то же время приложения при этом становятся зависимыми от конкретного протокола защищенного канала, так как в них должны быть встроены явные вызовы функций этого протокола.

Защищенный канал, реализованный на самом высоком (*прикладном*) уровне, защищает только вполне определенную сетевую службу, например файловую, гипертекстовую или почтовую. Так, протокол **S/MIME** защищает исключительно сообщения электронной почты. При таком подходе для каждой службы необходимо разрабатывать собственную защищенную версию протокола.

Работа протокола защищенного канала на уровне *представления* делает его более универсальным средством, чем протокол безопасности прикладного уровня. Однако для того чтобы приложение смогло воспользоваться протоколом уровня представления, в него по-прежнему приходится вносить исправления, хотя и не столь существенные, как в случае протокола прикладного уровня. Модификация приложения в данном случае сводится к встраиванию явных обращений к API соответствующего протокола безопасности.

На уровне представления работает протокол безопасности транспортного уровня **TLS** (Transport Layer Security), заменивший протокол защищенных сокетов **SSL** (Secure Socket Layer), который в настоящее время признан уязвимым и не рекомендуется для использования. Протокол SSL был разработан компанией Netscape Communications для защиты данных, передаваемых между веб-сервером и веб-браузером, но он мог быть использован и любыми другими приложениями. Для установления защищенного канала оба протокола используют следующие технологии безопасности:

- взаимная аутентификация приложений на обоих концах защищенного канала выполняется путем обмена сертификатами (стандарт X.509);
- для контроля целостности передаваемых данных используются дайджесты;
- секретность обеспечивается шифрованием с использованием симметричных ключей сеанса.

Средства защищенного канала становятся прозрачными для приложений в тех случаях, когда безопасность обеспечивается на *сетевом* и *канальном* уровнях. Однако здесь мы сталкиваемся с другой проблемой — зависимостью сервиса защищенного канала от протокола нижнего уровня. Например, протокол PPTP, не являясь протоколом канального уровня, защищает кадры протокола PPP канального уровня, упаковывая их в IP-пакеты. При этом не имеет никакого значения, пакет какого протокола, в свою очередь, упакован в данном PPP-кадре: IP, IPX, SNA или NetBIOS. С одной стороны, это делает сервис PPTP достаточно универсальным, так как клиент сервиса защищенного канала может задействовать любые протоколы в своей сети. С другой стороны, такая схема предъявляет жесткие требования к типу протокола канального уровня, используемому на участке доступа клиента к защищенному каналу — для протокола PPTP таким протоколом может быть *только PPP*. Хотя протокол PPP очень распространен в линиях доступа, сегодня конкуренцию ему составляют протоколы Gigabit Ethernet и Fast Ethernet, которые все чаще работают не только в локальных, но и в глобальных сетях.

Работающий на сетевом уровне протокол IPSec является компромиссным вариантом. С одной стороны, он прозрачен для приложений, с другой — может работать практически во всех сетях, так как основан на широко распространенном протоколе IP и использует любую технологию канального уровня (PPP, Ethernet, MPLS и т. д.).

# Система IPSec

**Протокол IPSec** в стандартах Интернета называют *системой*. Действительно, IPSec — это согласованный набор открытых стандартов, имеющий сегодня вполне очерченное ядро, которое в то же время может быть достаточно просто дополнено новыми функциями и протоколами.

Ядро IPSec составляют три протокола:

- AH (Authentication Header — заголовок аутентификации), который гарантирует целостность и аутентичность данных;
- ESP (Encapsulating Security Payload — инкапсуляция зашифрованных данных), который шифрует передаваемые данные, обеспечивая конфиденциальность, может также поддерживать аутентификацию и целостность данных;
- IKE (Internet Key Exchange — обмен ключами Интернета), который решает вспомогательную задачу автоматического предоставления конечным точкам защищенного канала секретных ключей, необходимых для работы протоколов аутентификации и шифрования данных.

Как видно из краткого описания функций, возможности протоколов AH и ESP частично перекрываются (рис. 29.10). В то время как AH отвечает только за обеспечение целостности и аутентификации данных, ESP может шифровать данные и, кроме того, выполнять функции протокола AH (хотя, как увидим позднее, аутентификация и целостность обеспечиваются им в несколько урезанном виде). ESP может поддерживать функции шифрования и аутентификации/целостности в любых комбинациях, то есть либо всю совокупность функций, либо только аутентификацию/целостность, либо только шифрование.

<table>
<tr><th>Выполняемые функции</th><th colspan="2">Протокол</th></tr>
<tr><td>Обеспечение целостности</td><td rowspan="2">AH</td><td rowspan="3">ESP</td></tr>
<tr><td>Обеспечение аутентичности</td></tr>
<tr><td>Обеспечение конфиденциальности (шифрование)</td><td></td></tr>
<tr><td>Распределение секретных ключей</td><td colspan="2">IKE</td></tr>
</table>

**Рис. 29.10.** Распределение функций между протоколами IPSec

Частичное дублирование функций защиты протоколами AH и ESP связано с применяемой во многих странах практикой ограничения экспорта и/или импорта средств, обеспечивающих конфиденциальность данных путем шифрования. Каждый из этих протоколов может использоваться как самостоятельно, так и одновременно с другим, так что в тех случаях, когда шифрование из-за действующих ограничений применять нельзя, систему можно поставлять только с протоколом AH. Естественно, подобная защита данных во многих случаях оказывается недостаточной. Принимающая сторона получает лишь возможность проверить, что данные были отправлены именно тем узлом, от которого они ожидаются, и дошли в том виде, в котором были отправлены. Однако от несанкционированного просмотра данных на пути их следования по сети протокол AH защитить не может, так как не шифрует их — для шифрования данных необходим протокол ESP.

## Безопасная ассоциация

Чтобы протоколы AH и ESP могли выполнять свою работу по защите передаваемых данных, протокол IKE устанавливает между двумя конечными точками *логическое соединение*, которое в стандартах IPSec носит название **безопасной ассоциации** (Security Association, **SA**).

Стандарты IPSec позволяют конечным точкам защищенного канала использовать единственную безопасную ассоциацию для передачи трафика всех взаимодействующих через этот канал хостов или создавать для этой цели произвольное число безопасных ассоциаций, например, по одной на каждое TCP-соединение. Это дает возможность выбирать нужную степень детализации защиты — от одной общей ассоциации для трафика множества конечных узлов до индивидуально настроенных ассоциаций для защиты каждого приложения.

Безопасная ассоциация в протоколе IPSec представляет собой *однонаправленное* (симплексное) логическое соединение, поэтому если требуется обеспечить безопасный двусторонний обмен данными, то необходимо установить две безопасные ассоциации. Эти ассоциации в общем случае могут иметь разные характеристики, например, при передаче запросов к базе данных достаточно только аутентификации, а для ответных данных, несущих ценную информацию, дополнительно может потребоваться обеспечить и их конфиденциальность.

Установление безопасной ассоциации начинается с взаимной *аутентификации* сторон, потому что все меры безопасности теряют смысл, если данные передаются или принимаются не тем лицом или не от того лица. Выбираемые далее параметры SA определяют, какой из двух протоколов, AH или ESP, будет применяться для защиты данных, какие функции будет выполнять протокол (например, можно выполнять только аутентификацию и проверку целостности, а можно, кроме того, еще и обеспечивать конфиденциальность). Очень важными параметрами безопасной ассоциации являются также *секретные ключи*, используемые в работе протоколов AH и ESP.

Протокол IPSec допускает как *автоматическое*, так и *ручное* установление безопасной ассоциации. При ручном способе администратор конфигурирует конечные узлы так, чтобы они поддерживали согласованные параметры ассоциации, включая секретные ключи. При автоматической процедуре установления SA протоколы IKE, работающие по разные стороны канала, выбирают параметры в ходе переговорного процесса.

Для каждой задачи, решаемой протоколами AH и ESP, предлагается несколько схем аутентификации и шифрования (рис. 29.11). Это делает протокол IPSec очень гибким средством. Заметим, что выбор дайджест-функции для решения задач целостности и аутентификации никак не влияет на выбор функции шифрования, обеспечивающей конфиденциальность данных.

Для обеспечения совместимости в стандартной версии IPSec определен некоторый обязательный «инструментальный» набор, в частности, для аутентификации данных всегда может быть использована одна из стандартных дайджест-функций MD5 либо SHA-1, а в число алгоритмов шифрования непременно входит DES. При этом производители продуктов, в которых используется IPSec, вольны расширять протокол путем включения других алгоритмов аутентификации и симметричного шифрования, что они с успехом и делают. Например, многие реализации IPSec поддерживают популярный алгоритм шифрования Triple DES, а также сравнительно новые алгоритмы: Blowfish, Cast, CDMF, Idea, RC5.

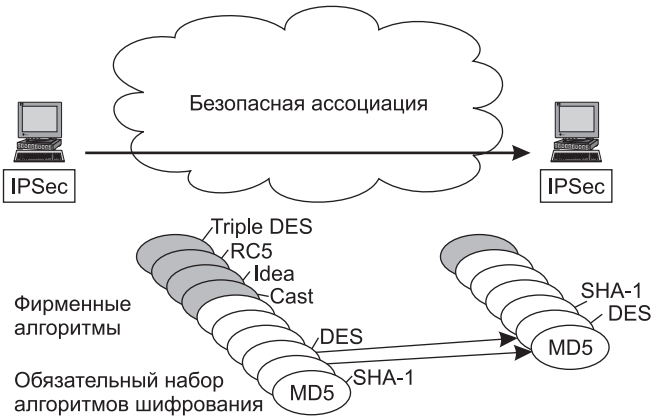



**Рис. 29.11.** Согласование параметров в протоколе ESP

## Транспортный и туннельный режимы

Протоколы AH и ESP могут защищать данные в двух режимах: транспортном и туннельном. В **транспортном режиме** передача IP-пакета через сеть выполняется с помощью оригинального заголовка этого пакета, а в **туннельном режиме** исходный пакет помещается в новый IP-пакет, и передача данных по сети выполняется на основании заголовка нового IP-пакета.

Применение того или иного режима зависит от требований, предъявляемых к защите данных, а также от роли, которую играет в сети узел, завершающий защищенный канал. Так, узел может быть хостом (конечным узлом) или шлюзом (промежуточным узлом). Соответственно, возможны три схемы применения протокола IPSec:

- хост-хост;
- шлюз-шлюз;
- хост-шлюз.

В схеме «хост-хост» безопасная ассоциация устанавливается между двумя конечными узлами сети. При этом протокол IPSec работает на каждом из этих узлов. Для схемы «хост-хост» чаще всего используется *транспортный* режим защиты.

В схеме «шлюз-шлюз» защищенный канал устанавливается между двумя промежуточными узлами, так называемыми **шлюзами безопасности** (Security Gateway, **SG**), на каждом из которых работает протокол IPSec (рис. 29.12). Защищенный обмен данными может происходить между любыми двумя конечными узлами, подключенными к сетям, которые расположены позади шлюзов безопасности. От конечных узлов поддержки протокола IPSec не требуется — они передают свой трафик в незащищенном виде через заслуживающие доверие внутренние сети предприятий. Трафик, направляемый в общедоступную сеть, проходит через шлюз безопасности, который и обеспечивает его защиту с помощью протокола IPSec. Шлюзам доступен только *туннельный* режим работы.

На рисунке пользователь компьютера с адресом IP1 посылает пакет по адресу IP2, используя туннельный режим протокола IPSec. Шлюз SG1 зашифровывает пакет вместе

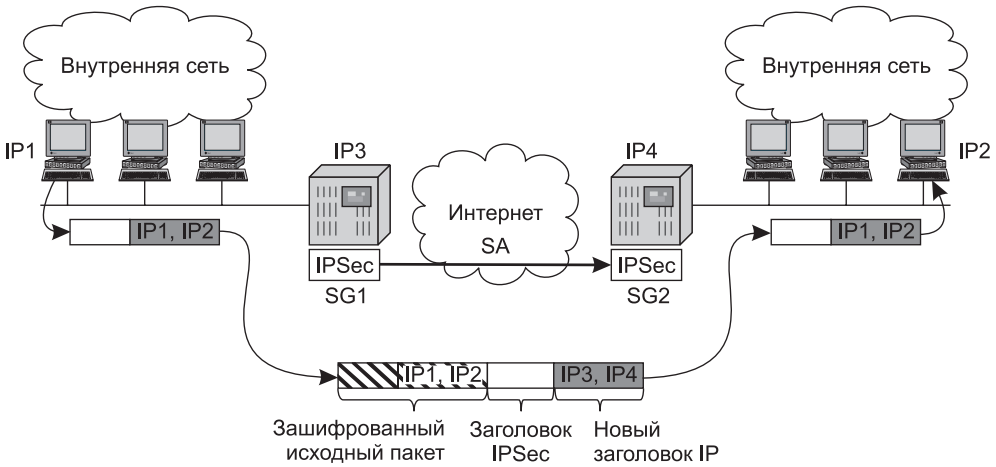



**Рис. 29.12.** Работа защищенного канала по схеме «шлюз-шлюз» в туннельном режиме

с заголовком, снабжая его новым IP-заголовком, в котором в качестве адреса отправителя указывает свой адрес — IP3, а в качестве адреса получателя — адрес IP4 шлюза SG2. Передача данных по составной IP-сети выполняется на основании заголовка внешнего пакета, а внутренний пакет становится при этом полем данных для внешнего пакета. На шлюзе SG2 протокол IPSec извлекает инкапсулированный пакет и расшифровывает его, приводя к исходному виду.

Схема «хост-шлюз» часто применяется при *удаленном доступе*. В этом случае защищенный канал прокладывается между удаленным хостом, на котором работает протокол IPSec, и шлюзом, защищающим трафик для всех хостов, входящих во внутреннюю сеть предприятия. Эту схему можно усложнить, создав параллельно еще один защищенный канал — между удаленным хостом и каким-либо хостом, принадлежащим внутренней сети, защищаемой шлюзом (рис. 29.13). Комбинированное использование двух безопасных ассоциаций позволяет надежно защитить трафик во внутренней сети.

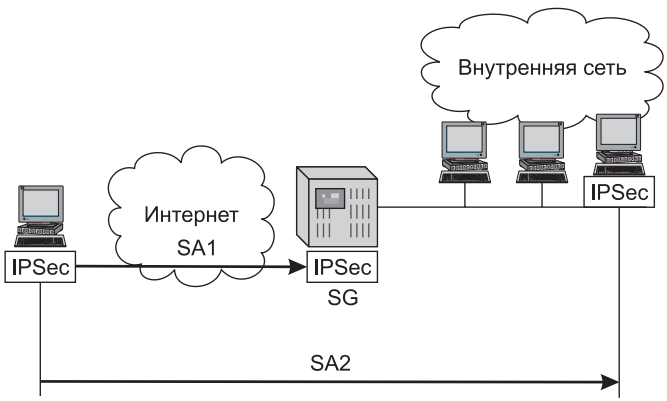



**Рис. 29.13.** Схема защищенного канала «хост-шлюз»

## Протокол AH

Протокол AH позволяет приемной стороне убедиться, что:

- пакет был отправлен стороной, с которой установлена безопасная ассоциация;
- содержимое пакета не было искажено в процессе его передачи по сети;
- пакет не является дубликатом уже полученного пакета.

Две первые функции обязательны для протокола AH, а последняя выбирается по желанию при установлении ассоциации. Для выполнения этих функций протокол AH использует специальный заголовок (рис. 29.14).

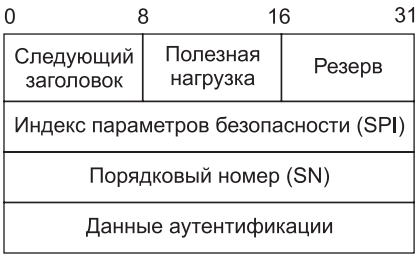



**Рис. 29.14.** Структура заголовка протокола AH

В поле *следующего заголовка* (next header) указывается код протокола более высокого уровня, то есть протокола, сообщение которого размещено в поле данных IP-пакета. Скорее всего, им будет один из протоколов транспортного уровня (TCP или UDP) или протокол ICMP, но может встретиться и протокол ESP, если он используется в комбинации с AH.

В поле *длины полезной нагрузки* (payload length) содержится длина заголовка AH.

*Индекс параметров безопасности* (Security Parameters Index, SPI) служит для связи пакета с предусмотренной для него безопасной ассоциацией (подробнее см. ниже).

Поле *порядкового номера* (Sequence Number, SN) пакета применяется для защиты от его ложного воспроизведения (когда третья сторона пытается повторно использовать перехваченные защищенные пакеты, отправленные реально аутентифицированным отправителем). Отправляющая сторона последовательно увеличивает значение этого поля в каждом новом пакете, передаваемом в рамках данной ассоциации, так что приход дубликата обнаружится принимающей стороной (если, конечно, в рамках ассоциации будет активирована функция защиты от ложного воспроизведения). Однако в любом случае в функции протокола AH не входит восстановление утерянных и упорядочивание прибывающих пакетов — он просто отбрасывает пакет, когда обнаруживает, что аналогичный пакет уже получен. Чтобы сократить требуемую для работы протокола буферную память, используется механизм скользящего окна — на предмет дублирования проверяются только те пакеты, чей номер находится в пределах окна, обычно 32 или 64 пакета.

Поле *данных аутентификации* (authentication data) содержит хеш-код (дайджест) исходного IP-пакета, вычисленный с помощью одной из двух обязательно поддерживаемых протоколом AH односторонних функций шифрования MD5 или SAH-1, но может использоваться и любая другая функция, о которой стороны договорились в ходе установления ассоциации. При вычислении дайджеста пакета в качестве параметра односторонней функ-

ции выступает симметричный секретный ключ, сгенерированный для данной ассоциации вручную или автоматически с помощью протокола IKE. Так как длина дайджеста зависит от выбранной функции, это поле имеет в общем случае переменный размер. Протокол AH старается охватить при вычислении дайджеста как можно большее число полей исходного IP-пакета, но некоторые из них в процессе передачи пакета по сети меняются непредсказуемым образом, поэтому не могут включаться в аутентифицируемую часть пакета. Например, целостность значения поля времени жизни (TTL) в приемной точке канала оценить нельзя, так как оно уменьшается на единицу каждым промежуточным маршрутизатором и никак не может совпадать с исходным.

Местоположение заголовка AH в пакете зависит от того, в каком режиме — транспортном или туннельном — сконфигурирован защищенный канал. Вид результирующего пакета в транспортном режиме представлен на рис. 29.15.

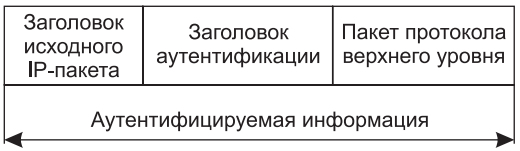



**Рис. 29.15.** Структура IP-пакета, обработанного протоколом AH в транспортном режиме

При использовании туннельного режима, когда шлюз IPSec принимает проходящий через него транзитом исходящий пакет и создает для него внешний IP-пакет, протокол AH защищает все поля исходного пакета, а также неизменяемые поля нового заголовка внешнего пакета (рис. 29.16).

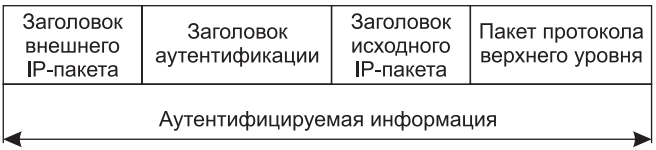



**Рис. 29.16.** Структура IP-пакета, обработанного протоколом AH в туннельном режиме

## Протокол ESP

Протокол ESP решает две группы задач. К первой относятся задачи обеспечения аутентификации и целостности данных на основе дайджеста, аналогичные задачам протокола AH, ко второй — защита передаваемых данных путем их шифрования от несанкционированного просмотра. Как видно на рис. 29.17, заголовок делится на две части, разделяемые полем данных. Первая часть, называемая собственно *заголовком ESP*, образуется полями SPI и SN, назначение которых аналогично одноименным полям протокола AH, и размещается перед полем данных. Остальные служебные поля протокола ESP, называемые *концевиком ESP*, расположены в конце пакета.

Два поля концевика — *следующего заголовка* и *данных аутентификации* — также аналогичны полям заголовка AH. Поле данных аутентификации отсутствует, если при установлении безопасной ассоциации принято решение не использовать средств протокола ESP,

касающихся обеспечения целостности. Помимо этих полей концевик содержит два дополнительных поля — *заполнителя* и *длины заполнителя*. Заполнитель может понадобиться в трех случаях. Во-первых, для нормальной работы некоторых алгоритмов шифрования необходимо, чтобы шифруемый текст содержал кратное число блоков определенного размера. Во-вторых, формат заголовка ESP требует, чтобы поле данных заканчивалось на границе четырех байтов. И наконец, заполнитель можно использовать, чтобы скрыть действительный размер пакета в целях обеспечения так называемой частичной конфиденциальности трафика. Правда, возможность маскировки ограничивается сравнительно небольшим объемом заполнителя — 255 байт, поскольку большой объем избыточных данных может снизить полезную пропускную способность канала связи.


**Рис. 29.17.** Структура IP-пакета, обработанного протоколом ESP в транспортном режиме

На рис. 29.17 показано размещение полей заголовка ESP в *транспортном режиме*. В этом режиме ESP не шифрует заголовок исходного IP-пакета, иначе маршрутизатор не сможет прочитать поля заголовка и корректно выполнить продвижение пакета между сетями. В число шифруемых полей не попадают также поля SPI и SN, которые должны передаваться в открытом виде, чтобы прибывший пакет можно было отнести к определенной ассоциации и предотвратить ложное воспроизведение пакета.

В *туннельном режиме* заголовок исходного IP-пакета помещается после заголовка ESP и полностью попадает в число защищаемых полей, а заголовок внешнего IP-пакета протоколом ESP не защищается (рис. 29.18).


**Рис. 29.18.** Структура IP-пакета, обработанного протоколом ESP в туннельном режиме

## Базы данных SAD и SPD

При установлении безопасной ассоциации, как и при любом другом логическом соединении, две стороны принимают ряд соглашений, регламентирующих процесс передачи

потока данных между ними и фиксируемых в виде набора параметров. Для безопасной ассоциации такими параметрами являются, в частности, тип и режим работы протокола защиты (AH или ESP), методы шифрования, секретные ключи, значение текущего номера пакета в ассоциации и другая существенная информация. Каким же образом протокол IPSec, работающий на хосте или шлюзе, определяет способ защиты, который он должен применить к трафику? Решение основано на использовании в каждом узле, поддерживающем IPSec, двух типов баз данных:

- *баз данных безопасных ассоциаций* (Security Associations Database, **SAD**), в которых хранятся наборы текущих параметров, определяющих все активные SA. Каждый узел IPSec поддерживает две базы SAD (для исходящих и входящих ассоциаций соответственно);

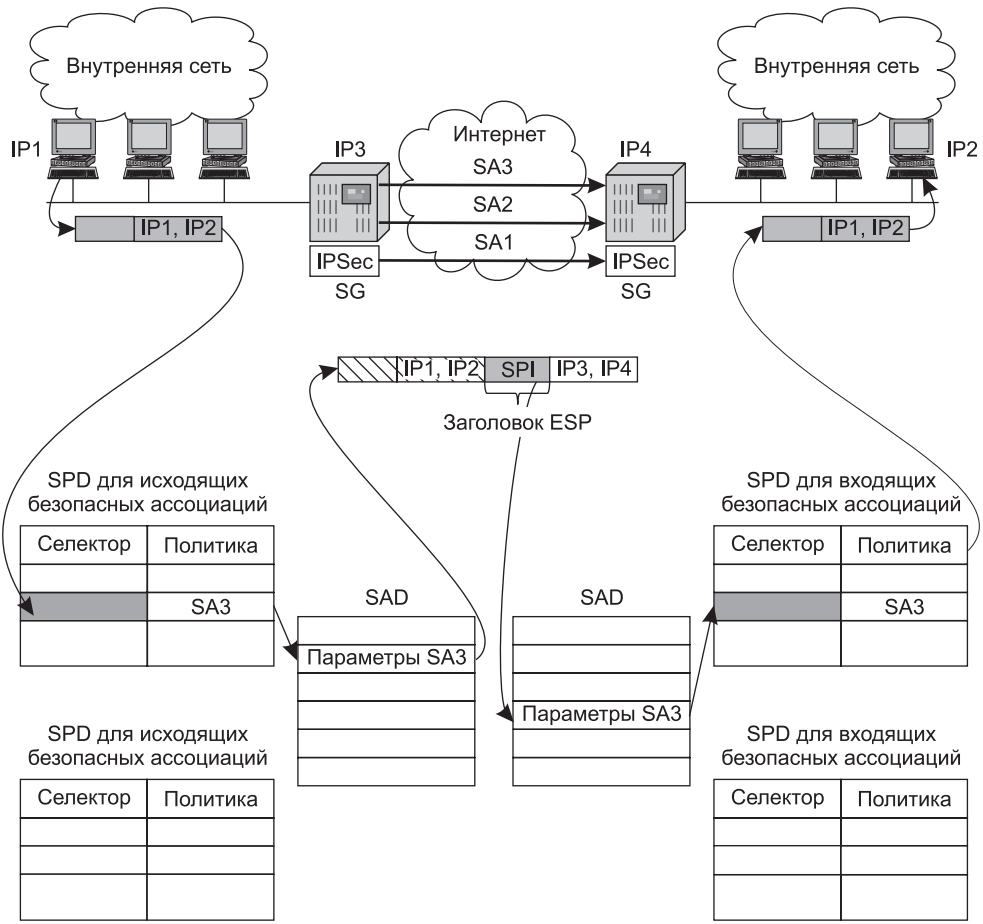



**Рис. 29.19.** Использование баз данных SPD и SAD

- *баз данных политики безопасности* (Security Policy Database, **SPD**), которые определяют соответствие между IP-пакетами и установленными для них правилами обработки. Записи SPD состоят из полей двух типов — полей селектора пакета и полей политики защиты для пакета с данным значением селектора.

Селектор в SPD (рис. 29.19) включает следующий набор признаков, на основании которых можно с большой степенью детализации выделить защищаемый поток:

- IP-адреса источника и приемника, которые могут быть представлены как в виде отдельных адресов (индивидуальных, групповых или широковещательных), так и диапазонами адресов, заданными с помощью верхней и нижней границ либо с помощью маски;
- порты источника и приемника (то есть TCP- или UDP-порты);
- тип протокола транспортного уровня (TCP, UDP);
- имя пользователя в формате DNS или X.500;
- имя системы (хоста, шлюза безопасности и т. п.) в формате DNS или X.500.

Для каждого нового пакета, поступающего в защищенный канал, IPSec просматривает все записи в базе SPD, сравнивая значение селекторов этих записей с соответствующими полями IP-пакета. Если значение полей совпадает с каким-либо селектором, то над пакетом выполняются действия, определенные в поле политики безопасности данной записи: передачу пакета без изменения, его отбрасывание либо обработку средствами IPSec. В последнем случае поле политики защиты должно содержать ссылку на запись в базе данных SAD, в которую помещен набор параметров безопасной ассоциации для данного пакета (на рисунке для исходящего пакета определена ассоциация SA3). На основании заданных параметров безопасной ассоциации к пакету применяются соответствующие протокол (на рисунке — ESP), функции шифрования и секретные ключи.

Если к исходящему пакету нужно применить некоторую политику защиты, но указатель записи SPD показывает, что в настоящее время нет активной безопасной ассоциации с требуемой политикой, то IPSec создает новую ассоциацию с помощью протокола IKE, помещая новые записи в базы данных SAD и SPD.

Базы данных политики безопасности создаются и администрируются либо пользователем (этот вариант больше подходит для хоста), либо системным администратором (вариант для шлюза), либо автоматически (приложением).

Ранее мы выяснили, что установление связи между исходящим IP-пакетом и заданной для него безопасной ассоциацией происходит путем селекции. Однако остается открытым другой вопрос: как *принимающий* узел IPSec определяет способ обработки прибывшего пакета, если при шифровании многие ключевые параметры пакета, отраженные в селекторе, оказываются недоступными, а значит, невозможно определить соответствующую запись в базах данных SAD и SPD и, следовательно, тип процедуры, которую надо применить к поступившему пакету? Для решения этой проблемы в заголовках AH и ESP и предусмотрено поле *SPI*. В это поле помещается указатель на ту строку базы данных SAD, в которой записаны параметры соответствующей безопасной ассоциации. Поле SPI заполняется протоколом AH или ESP во время обработки пакета в отправной точке

защищенного канала. Когда пакет приходит в конечный узел защищенного канала, из его внешнего заголовка ESP или AH (на рисунке — из заголовка ESP) извлекается значение SPI, после чего обработка пакета выполняется с учетом всех параметров заданной этим указателем ассоциации.

Таким образом, для распознавания пакетов, относящихся к разным безопасным ассоциациям, используются:

- на узле-отправителе — селектор;
- на узле-получателе — индекс параметров безопасности (SPI).

После дешифрования пакета приемный узел IPSec проверяет его признаки (ставшие теперь доступными) на предмет совпадения с селектором записи SPD для входящего трафика, чтобы убедиться, что ошибки не произошло и выполняемая обработка пакета соответствует политике защиты, заданной администратором.

Использование баз SPD и SAD для защиты трафика позволяет гибко сочетать механизм безопасных ассоциаций, который предусматривает установление логического соединения, с дейтаграммным характером трафика протокола IP.

### (S) *VPN на основе IPSec*

# ГЛАВА 30 Безопасность программного кода и сетевых служб

## Уязвимости программного кода и вредоносные программы

Программная система, состоящая из десятков тысяч строк кода, всегда имеет уязвимости, которые может использовать злоумышленник. Эти уязвимости могут быть результатом ошибок программистов — в соответствии с исследованием CyLab Университета Карнеги Мэллона в среднем каждые 1000 строк кода содержат 20–30 ошибок, из которых 5 % влияют на безопасность системы, а 1 % открывает возможности для взлома системы.

## Уязвимости, связанные с нарушением защиты оперативной памяти

Области оперативной памяти (адресные пространства) отдельных процессов защищены друг от друга. Защита памяти реализуется ОС в тесном взаимодействии с аппаратными механизмами процессора. Несмотря на это, некорректное использование областей памяти все же может происходить в пределах адресного пространства *отдельного процесса* или в области памяти *ядра* ОС. В последнем случае это особенно опасно, так как может вызвать крах всей системы, а не отдельного приложения, как в первом случае.

**Переполнение буфера памяти** является, по-видимому, наиболее часто используемой уязвимостью, связанной с нарушением защиты памяти. Мы уже знаем одну такую атаку, которая использует буфер, расположенный в памяти ядра, и приводит к краху всей системы — это атака «Пинг смерти». Точнее, приводила, так как ошибка в операционных системах, приводящая их к краху при превышении IP-пакетом размера в 65 535 байт, уже устранена. Тем не менее механизм, который эксплуатируется этой атакой, очень типичен — он использует отсутствие контроля над вводимой из внешнего мира информацией (в данном случае не контролируется длина помещаемого в буфер пакета).

**Переполнение стека** является частным случаем переполнения буфера памяти. Этот вид уязвимости часто используется злоумышленниками, чтобы заставить ОС выполнить код злоумышленника. Напомним, стеком является область памяти с реализацией стратегии записи LIFO (Last In First Out — последним пришел, первым вышел). Этот способ записи удобен при многократном вызове функций (подпрограмм), так как он обеспечивает экономичный возврат из вызванной функции в вызывающую.

Типичная структура стека, который растет в сторону меньших адресов (архитектура Intel x86), показана на рис. 30.1, *а*. Здесь мы видим стек, содержащий данные одной функции $f_1(A_1, A_2)$. В стек помещены аргументы этой функции, за которыми идет адрес возврата в функцию, ее вызвавшую — в данном случае это функция `main`, то есть основное тело программы, написанной на языке C. За адресом возврата идет локальная память функции $f_1$, используемая для хранения ее локальных переменных и массивов. Указатель стека содержит адрес первого слова свободной области стека.


**Рис. 30.1.** *а* — структура стека до вызова функции $f_2$ из функции $f_1$; *б* — структура стека после вызова функции $f_2$ из функции $f_1$

Если функция $f_1$ вызывает другую функцию, $f_2$, то ее аргументы, адрес возврата в функцию $f_1$ и ее локальная память будут размещаться над областью памяти, выделенной в стеке функции $f_1$ (рис. 30.1, *б*). При завершении функция $f_2$ должна выполнить специальную инструкцию `RETURN`, которая вернет управление по адресу возврата в функцию $f_1$ и очистит стек от данных функции $f_2$, вернув указатель стека на прежнее место. Переполнение стека может произойти, если в область локальной памяти функции помещаются данные, длина которых больше длины этой области. В таком случае эти данные могут наложиться на адрес возврата. В результате после завершения вызванной функции произойдет переход на некоторый адрес, который может быть как случайным, так и специально сформирован злоумышленником. Многие атаки основаны на том, что в область локальных данных стека помещается вредоносный код, которому затем передается управление за счет подмены содержимого поля адреса возврата адресом начала вредоносного кода.

Некоторые операционные системы помечают область стека как неисполняемую, что предотвращает выполнение вредоносного кода в случае его попадания в стек.

**(S)** *Пример использования техники переполнения стека для организации атаки*

**(S)** *Скрытые коммуникации и скрытые каналы*

Переполнение буфера является частным случаем уязвимостей, являющихся следствием слабого контроля вводимых данных. В более общем случае — *любая* непредвиденная создателем программы форма вводимых данных может вызвать совершено неожиданные последствия, и этот факт может быть использован злоумышленником. Как любят повторять специалисты по разработке безопасного кода, «любой ввод данных — это зло».

Тривиальным примером является веб-форма, в которой пользователю предлагается ввести номер статьи, выбранный из списка, включающего 10 статей. Если разработчик не предвидел, что вместо ожидаемого положительного числа из диапазона от 1 до 10 пользователь может ввести отрицательное, к примеру, –1, то приложение может повести себя совсем не так, как он планировал, например, выдать конфиденциальный документ вместо публично доступной статьи.

Единственный метод борьбы с внедрением вредоносного кода при вводе данных — подвергать любые данные, получаемые программой от источника, не вызывающего доверия, *тщательной проверке* перед их использованием. Этот подход аналогичен принципу защиты периметра сети, рассмотренному в главе 27: вся информация, поступающая извне доверенного периметра, должна тщательно фильтроваться. Фильтрацию вводимых данных могут выполнять программа или файервол, работающий на прикладном уровне, а также система обнаружения вторжений (ISD) хоста. В идеале фильтрацию выполняют все три компонента: программа лучше всего знает специфику вводимых данных и возможные угрозы, в то время как файервол и ISD могут выполнять более общие проверки для определенного типа угроз.

## Троянские программы

**Троянские программы**, или **трояны** (**trojan**), — это разновидность вредоносных программ, которые наносят ущерб системе, маскируясь под какие-либо полезные приложения.

Троянские программы могут быть отнесены к самому простому по реализации виду вредоносных программ, применяя в качестве прикрытия знакомые пользователю приложения, с которыми он работал и раньше, до появления в компьютере «троянского коня», либо принимая вид нового приложения, которое пытается заинтересовать пользователя-жертву какими-то своими якобы полезными функциями. Однако суть троянской программы в обоих случаях остается вредительской: она может уничтожать или искажать информацию на диске, передавать данные (например, пароли) с «зараженного» компьютера на удаленный компьютер хакера, приводить в неработоспособное состояние установленное на атакованном компьютере программное обеспечение, участвовать в проведении DoS-атак на другие удаленные компьютеры. Так, одна из известных троянских программ AIDS TROJAN DISK7, разосланная нескольким тысячам исследовательских организаций на дискете, при запуске перемешивала символы в именах всех файлов и заполняла все свободное пространство жесткого диска.

## Сетевые черви

**Сетевые черви** (**worm**) — это программы, способные к самостоятельному распространению своих копий среди узлов в пределах локальной сети, а также по глобальным связям, перемещаясь от одного компьютера к другому без всякого участия в этом процессе пользователей сети.

Поскольку большинство сетевых червей передаются в виде файлов, основным механизмом их распространения являются сетевые службы, основанные на файловом обмене. Так, червь может рассылать свои копии по сети в виде вложений в сообщения электронной почты или путем размещения ссылок на зараженный файл на каком-либо веб-сайте. Однако существуют и другие разновидности червей, которые для своей экспансии используют более сложные приемы, например, связанные с ошибками («дырами») в программном обеспечении.

Главная цель и результат деятельности червя состоит в том, чтобы передать свою копию на максимально возможное число компьютеров, — новых потенциальных жертв, для поиска которых черви задействуют встроенные в них средства.

> Типичная программа-червь не удаляет и не искажает пользовательские и системные файлы, не перехватывает электронную почту пользователей, не портит содержимое баз данных, а наносит вред атакованным компьютерам *потреблением их ресурсов*, например, для рассылки спама или проведения массированной атаки в составе ботнета.

При создании типичного сетевого червя хакер, прежде всего, определяет перечень сетевых уязвимостей, которые он собирается использовать для проведения атак средствами создаваемого червя. Такими уязвимостями могут быть как известные, но не исправленные на некоторых компьютерах ошибки в программном обеспечении, так и пока не известные никому ошибки, которые обнаружил сам хакер. Чем шире перечень уязвимостей и чем более они распространены, тем больше узлов может быть поражено червем. Червь состоит из двух основных функциональных компонентов:

- *Атакующий блок*, состоящий из нескольких модулей (векторов атаки), каждый из которых рассчитан на поражение конкретного типа уязвимости. Этот блок открывает «входную дверь» атакуемого хоста и передает через нее свою копию.
- *Блок поиска целей* (локатор), собирающий информацию об узлах сети, а затем на основании этой информации определяющий, какие из исследованных узлов обладают теми уязвимостями, для которых хакер имеет средства атаки.

Эти два функциональных блока являются обязательными и присутствуют в реализации любой программы-червя. Некоторые черви нагружены их создателями и другими вспомогательными функциями, о которых мы скажем позже.

Упрощенно жизненный цикл червя может быть описан рекурсивной процедурой, состоящей из циклического запуска локатора и атакующего блока на каждом из последующих заражаемых компьютеров (рис. 30.2).

В начале каждого нового цикла червь, базирующийся на захваченном в результате предыдущей атаки компьютере, запускает локатор для поиска и формирования списка узлов-целей, пригодных для проведения каждой из специфических атак, а затем, используя средства атакующего блока, пытается эксплуатировать уязвимости узлов из этого списка. В результате успешной атаки червь копирует все свои программы на «новую территорию» и активирует локатор. После этого начинается новый цикл. На рисунке показано, как червь лавинообразно распространяется по сети. Заражение тысяч компьютеров может занять всего несколько минут. Некоторые виды червей не нападают на уже зараженные и/или подвергающиеся атаке в данный момент узлы. Если же такая проверка не предусмотрена в алгоритме работы червя, то в сети случайным образом могут возникать очаги стихийных DoS-атак. Локатор идентифицирует цели по адресам электронной почты, IP-адресам, характеристикам установленных на хостах ОС, номерам портов, типам и версиям приложений.

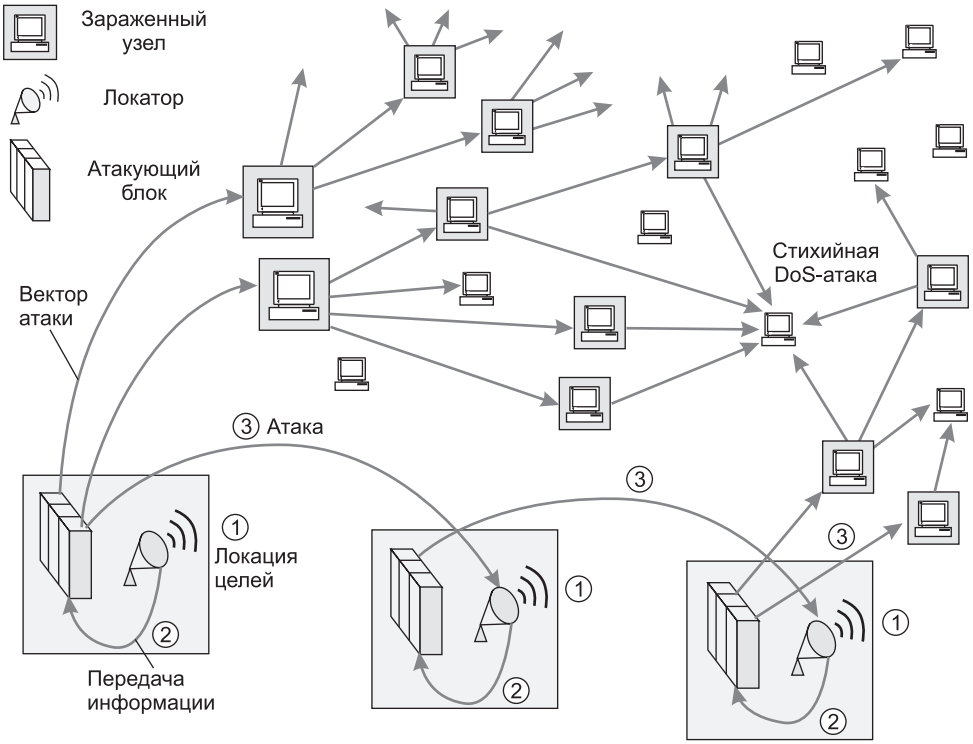



**Рис. 30.2.** Экспансия червя в сети

Для сбора информации локатор может предпринимать действия, связанные как с поисками интересующих данных на захваченном им в данный момент хосте, так и зондированием сетевого окружения. Простейший способ получить данные локально — прочитать файл, содержащий адресную книгу клиента электронной почты[1]. Помимо почтовых адресов локатор может найти на узле базирования другие источники информации: таблицы конфигурационных параметров сетевых интерфейсов, ARP-таблицы и таблицы маршрутизации. Зная IP-адреса хоста базирования и шлюзов, локатор достаточно просто может определить IP-адреса других узлов этой сети. Для идентификации узлов локатор может также использовать ICMP-сообщения или запросы ping, указывая в качестве адресов назначения все возможные IP-адреса. Для определения того, какие приложения работают на том или ином хосте, локатор сканирует различные хорошо известные номера TCP- и UDP-портов. Определив тип приложения, локатор пытается получить более детальные характеристики этого приложения. Например, пусть некая программа-червь имеет в своем арсенале средства для атаки на некоторые версии веб-сервера Apache. Для поиска потенциальных жертв локатор этого червя зондирует узлы сети, посылая умышленно ошибочные запросы к веб-серверу:

```
GET / HTTP/1.1\r\n\r\n
```

[1] Для коллекционирования почтовых адресов локатор может прибегать и к более интеллектуальным методам, использующим в работе спамеры (о спаме см. далее).

Узел, на котором установлен сервер Apache, отвечает на запрос, как и рассчитывал разработчик червя, то есть сообщением об ошибке, например, такого вида:

```
HTTP/1.1 400 Bad Request
Date: Mon, 23 Feb 2004 23:43:42 GMT
Server: Apache/1.3.19 (UNIX) (Red-Hat/Linux) mod_ssl/2.8.1
OpenSSL/0.9.6 DAV/1.0.2 PHP/4.0.4pl1 mod_perl/1.24_01
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=iso-8859-1
```

Из этого ответа локатор узнает о том, что на узле установлен веб-сервер Apache версии 1.3.19. Для червя этой информации может быть достаточно, чтобы внести данный узел в число целей.

Собрав данные об узлах сети, локатор анализирует их подобно тому, как это делает хакер при сетевой разведке. Для атаки выбираются узлы, удовлетворяющие некоторым условиям, которые говорят о том, что данный узел, возможно, обладает уязвимостями нужного типа (для них в атакующем блоке есть средства нападения). Понятно, что при таком «предположительном» способе отбора целей не всякая предпринятая атака обязательно приводит к успеху. Неудача рассматривается атакующим блоком червя как штатная ситуация, он просто сворачивает все свои действия, направленные на не поддавшийся атаке узел, и переходит к атаке на следующую цель из списка, подготовленного локатором. Для передачи своей копии на удаленный узел атакующий блок червя часто использует рассмотренную ранее уязвимость *переполнения буфера*. Помимо локатора и атакующего блока (см. выше) червь может включать некоторые дополнительные функциональные компоненты:

- *Блок удаленного управления и коммуникаций* служит для передачи сетевым червям команд от их создателя, а также для взаимодействия червей между собой. Такая возможность позволяет хакеру координировать работу червей для организации распределенных атак отказа в обслуживании. Сетевые черви могут быть использованы и для организации параллельных вычислений при решении таких требующих большого объема вычислений задач, как, например, подбор секретного ключа шифрования или пароля.
- *Блок управления жизненным циклом* может ограничивать работу червя определенным периодом времени.
- *Блок фиксации событий* используется автором червя для оценки эффективности атаки, реализации различных стратегий заражения сети или оповещения других пользователей о повреждениях, нанесенных их компьютерам. Результатом работы данного блока может быть, например, список IP-адресов успешно атакованных машин, посланный хакеру в виде файла или сообщения электронной почты.

## Вирусы

**Вирус** (**virus**) — это вредоносный программный фрагмент, который может внедряться в другие файлы.

Стремление злоумышленника сделать код вируса как можно более коротким часто ограничивает логику работы вируса очень простыми решениями, которые, однако, иногда приводят к весьма разрушительным последствиям. Так, например, один из реально су-

ществовавших вирусов, состоящий всего из 15 (!) байтов, записывал свою копию поверх других файлов в начало каждого сектора диска, в результате чего система быстро терпела крах. Некоторым утешением в подобных случаях является то, что одновременно с падением ОС прекращает свое существование и вирус.

Вирус может внедрять свои фрагменты в разные типы файлов, в том числе в файлы исполняемых программ (рис. 30.3). При этом возможны самые разные варианты: замещение кода, когда размер инфицированного файла не меняется, вставка вирусного кода целиком в начало или конец исходной программы, замена фрагментов программного кода фрагментами вируса с перестановкой замещенных фрагментов и без перестановки и т. д. и т. п. Более того, код вируса может быть зашифрован, чтобы затруднить его обнаружение антивирусными программами.

> В отличие от червей вирусы (так же, как и троянские программы) не содержат в себе встроенного механизма активного распространения по сети и способны размножаться *своими силами* только в пределах одного компьютера.

Как правило, передача копии вируса на другой компьютер происходит с участием пользователя. Например, пользователь может записать свой файл, зараженный вирусом, на сетевой файловый сервер, откуда тот может быть скопирован всеми пользователями, имеющими

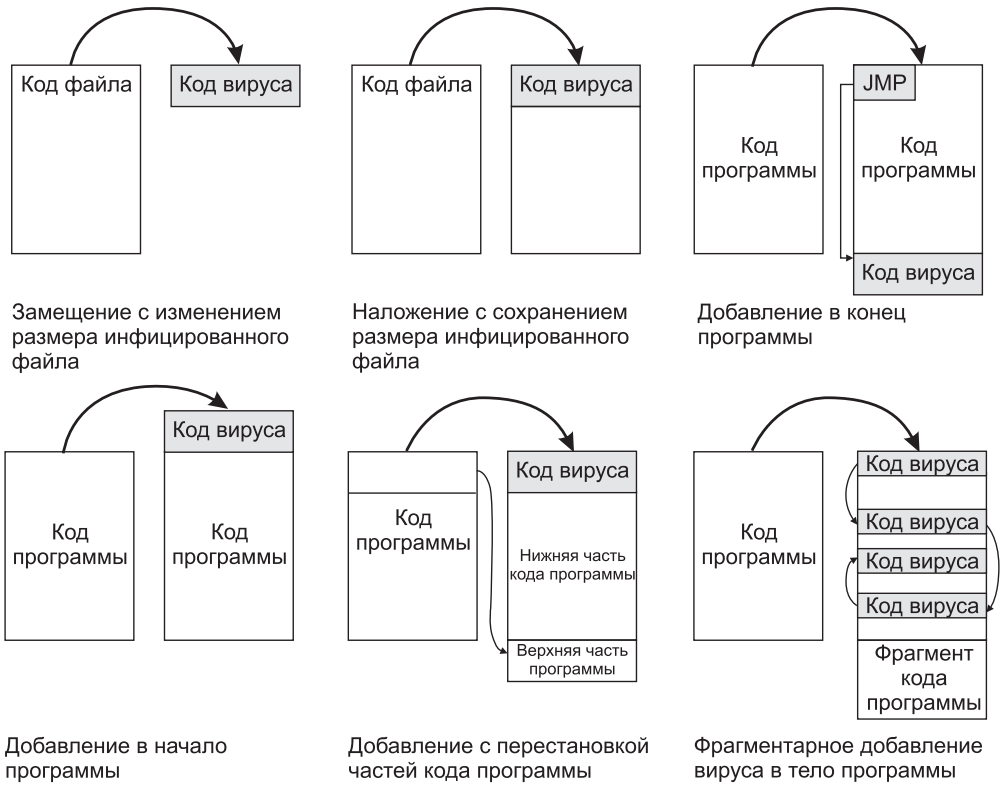



**Рис. 30.3.** Различные варианты расположения кода вируса в зараженных файлах

доступ к данному серверу. Пользователь может также передать другому пользователю съемный носитель с зараженным файлом или послать такой файл по электронной почте. То есть именно пользователь является главным звеном в цепочке распространения вируса за пределы своего компьютера. Тяжесть последствий вирусного заражения зависит от того, какие вредоносные действия были запрограммированы в вирусе злоумышленником. Это могут быть мелкие, но раздражающие неудобства (замедление работы компьютера, уменьшение размеров доступной памяти, трата рабочего времени на переустановку приложений) или серьезные нарушения безопасности: утечка конфиденциальных данных, разрушение системного программного обеспечения, частичная или полная потеря работоспособности компьютерной сети.

## Программные закладки

**Программная закладка** — это встроенный в программное обеспечение объект, который при определенных условиях (входных данных) инициирует выполнение не описанных в документации функций, позволяющих осуществлять несанкционированные воздействия на информацию. Функции, описание которых отсутствует в документации, называют **недекларированными возможностями**.

Программные закладки могут выполнять различную вредоносную работу, в частности:

- шпионить за действиями пользователя и передавать эту информацию на определенный сервер — это так называемые **шпионские программы** (spyware);
- получать доступ к конфиденциальной информации;
- искажать и разрушать данные.

В то же время недекларированные возможности программы не обязательно являются вредоносными. Так, они могут быть дополнительными функциями, включенными в программу для отладки, но не имеющими описания для рядовых пользователей. Это могут быть и забытые функции, особенно если речь идет о программной системе, в разработке которой участвовали десятки программистов. Существует также класс недекларированных возможностей программы, внедряемых в нее для развлечения пользователя (и самих программистов тоже), — так называемые «пасхальные яйца» (Easter Eggs). «Пасхальное яйцо» прерывает нормальную работу пользователя, который, возможно, устал рассматривать ячейки таблицы своего документа, и радостно приветствует его интересной картинкой, сообщением, а то и приглашением поиграть в игру. Авторы наблюдали однажды такое «пасхальное яйцо» в заставке экрана «Трубы» (3D Pipes) ОС Microsoft Windows — на экране на несколько минут среди труб появился очень симпатичный чайник. Появился, исчез, и больше никогда мы его не видели, хотя «Трубы» долго работали на наших компьютерах.

## Антивирусные программы

Антивирусные программы давно стали необходимым атрибутом жизни любого пользователя. На домашних компьютерах работают индивидуальные пакеты антивирусной защиты, на предприятиях — корпоративные пакеты, состоящие из клиентской программы и сервера, рассылающего обновления. Антивирусные программы используют различные методы для обнаружения вредоносных кодов в файлах, сообщениях электронной почты или HTML-страницах.

Вирус (будем так обобщенно называть далее любой вредоносный код) определенного типа имеет характерную последовательность программных кодов, которая его с какой-то степенью вероятности идентифицирует. Эта последовательность кодов называется **сигнатурой** (подписью) вируса. Чтобы обнаружить вирус, антивирусная программа должна иметь библиотеку сигнатур. Постоянное обновление этой библиотеки является одной из самых главных проблем любой компании, выпускающей антивирусное программное обеспечение. Сервер корпоративной антивирусной системы периодически рассылает обновленные версии такой библиотеки своим клиентам.

**Метод сигнатур** является основным методом обнаружения вирусов, но обладает принципиальным недостатком — *неспособностью обнаружить новый тип вируса*. Кроме того, разработчики вирусов прибегают к *маскировке* сигнатур, что приводит к нераспознаванию вируса. Для этого, например, злоумышленник может использовать *полиморфический код*, когда код изменяет сам себя во время выполнения, это, естественно, приводит к тому, что у него нет постоянной сигнатуры.

Антивирусные программы используют также **эвристические методы**, которые пытаются выявить вирус на основе структуры его кода или его поведения, не имея точной сигнатуры кода, но используя некоторые обобщенные признаки подозрительной структуры кода (*статический анализ*) или подозрительного поведения (*динамический анализ*).

Для безопасного анализа поведения анализируемой программы она помещается в изолированную виртуальную среду, например, в среду отдельной виртуальной машины или же созданной программной «песочницы»[1], ограждающей систему от опасных действий программы. В этом случае действия вируса не могут причинить вред основной операционной среде компьютера. Помещение анализируемой программы в специальную защищенную среду является затратным как по ресурсам, так и по времени. Существует более эффективный, хотя и более рискованный подход, когда анализируемой программе разрешают пробное выполнение в рабочей среде, при этом антивирусное программное обеспечение следит за всеми ее действиями и при необходимости блокирует их, не давая нанести ущерб рабочей среде. При обнаружении вируса антивирусная программа помещает зараженную программу в карантин и уведомляет об этом пользователя, который принимает решение об удалении зараженной программы или же, если это возможно, удалении из нее вируса.

**ПРИМЕЧАНИЕ**

Антивирусные программы работают в пространстве ядра, поэтому сами могут причинить ущерб операционной системе из-за своих ошибок. Зафиксированы случаи, когда под видом антивирусной программы пользователям предлагалось вредоносное программное обеспечение.

## Ботнет

**Бот** — это программа, которая выполняет некоторые автоматические (часто интеллектуальные) действия по командам удаленного центра управления.

[1] Песочница (sandbox) представляет собой механизм жесткого контроля набора ресурсов (оперативной памяти, места на диске и др.) и системных сервисов, доступных подозрительной программе.

Бот является *программным роботом*, способным реагировать на возникающую ситуацию и полученные извне команды некоторыми действиями — протоколированием сообщений (полезный бот ведет архив чатов), отправкой сообщений, например, поддержанием «разговора» с удаленным собеседником или же участием в DDoS-атаке на какой-то сайт или сеть. Бот может, например, распознавать определенный, заданный ему «хозяином» контекст в дискуссии пользователей социальных сетей Интернета (Livejournal, Facebook) и стать ее участником, выдавая те или иные сообщения. Бот обычно находится в следящем режиме, анализируя сообщения и ожидая команды из центра управления или возникновения заранее определенной ситуации.

Боты проникают в удаленные компьютеры нелегально как вирусы, черви или «троянские кони». Пользователь может не знать, что его компьютер заражен ботом, потому что компьютеру этого пользователя бот не причиняет вреда — его цели находятся где-то в Интернете. Обычно злоумышленник заражает кодом бота несколько компьютеров, используя различные известные уязвимости ОС и приложений, а затем уже код бота, подобно сетевому червю, пытается заразить как можно больше машин.

Зараженную ботом машину иногда называют **зомби**. Группа согласованно работающих ботов называется **ботнетом** (botnet) или **сетью ботов**. Боты часто управляются централизованно, из одного или нескольких центров, являющихся *серверами сети ботов*. Возможны и более сложные зависимости между ботами одной сети с иерархическими или одноранговыми схемами взаимодействия. Для управления ботами центр управления использует различные протоколы, одним из наиболее распространенных является протокол **IRC** (Internet Relay Chat), позволяющий передавать мгновенные сообщения (чат).

Так как «хозяин» ботнета точно не знает, какие именно машины оказались зараженными кодом бота, для распознавания компьютеров-зомби используются методы сетевого сканирования, например сканирование портов, если код бота слушает определенный порт TCP. Жертвы «зомбирования» могут составить внушительную армию, способную претворить в жизнь мощную DDoS-атаку, распространить огромное количество спама или осуществить массовый сбор персональных данных. Заметим, что ботнет может работать и как наемная армия — ее «командир» может предоставлять услуги своей сети третьим лицам.

Одним из инцидентов, связанных с пресечением вредоносной деятельности ботнета, была операция, проведенная компанией Microsoft совместно с ФБР в июне 2013 года по разрушению центров управления ботнетами, зараженными *вирусом Citadel*. Этот вирус фиксирует нажатия клавиш на компьютере и передает информацию в свой центр управления. В результате проведенной операции было выявлено и разрушено 1462 центра управления, каждый из которых контролировал свой ботнет. Сообщалось, что эти сети причинили ущерб почти 5 миллионам пользователей на сумму свыше полмиллиарда долларов.

## Безопасность веб-сервиса

Веб-браузер с его графическим интерфейсом является основным средством доступа пользователя к большинству сервисов Интернета: сайтам новостей, разнообразным справочникам, библиотекам, интернет-магазинам, онлайновым банкам, социальным сетям, таким как Facebook, Twitter, LiveJournal, ВКонтакте, облачным хранилищам информации и приложениям. Даже такие консервативные устройства, как сетевые маршрутизаторы и коммутато-

ры, стали поддерживать административный доступ посредством веб-интерфейса[1]. Поэтому справедливым будет сказать, что основная часть информации поступает в клиентский компьютер через веб-браузер, а, как отмечено, именно вводимые данные представляют собой главную угрозу для программного обеспечения компьютера. Через веб-браузер попадает в ваш компьютер большинство вредоносных кодов (вирусы, черви и троянские программы), а также назойливые программки, размещающие рекламные объявления на просматриваемой странице без вашего согласия.

## Безопасность веб-браузера

Особенностью защиты веб-службы является то, что такая защита требует решения двух достаточно независимых задач:

- обеспечение безопасности программных и аппаратных ресурсов компьютера, на котором эта служба выполняется;
- обеспечение приватности того лица, которое этой службой пользуется.

**Приватность** некоторого лица — это требование неприкосновенности частной жизни этого лица, включающая запрет на сбор, хранение, использование и распространение информации о его частной жизни без его согласия.

За время, прошедшее с момента появления первого браузера, разработчики браузеров накопили большой опыт в борьбе со злоумышленниками, и меню *приватности* и *безопасности* современного браузера включает много опций. Рассмотрим, что стоит за этими опциями, какие риски они стараются снизить и за счет каких средств.

## Приватность и куки

Популярность Интернета негативно повлияла на приватность его пользователей. Потенциально все действия пользователя в Интернете — посещенные сайты, просмотренные страницы, запросы поиска — могут быть зафиксированы и проанализированы, и антитеррористические службы, а также службы маркетинга торговых предприятий активно этим занимаются.

*Веб-серверы* ведут журналы посещений своих сайтов с запоминанием IP-адресов клиентов и предоставляют эти данные владельцам сайтов в удобной форме. Однако анонимность в этих журналах до какой-то степени сохраняется, особенно если адрес назначен провайдером динамически.

*Браузеры* также ведут журналы посещения сайтов и страниц. И если на веб-сервере данные о ваших посещениях, скорее всего, растворились бы в общей статистике, то на вашем компьютере (если он не используется вместе с другими сотрудниками или посетителями кафе или гостиницы) сохраняется история именно ваших посещений и интересов (последнее — в виде запросов к поисковым машинам). Поэтому теперь конфискация компьютера и просмотр журнала истории браузера — одно из первых действий следователя при рас-

[1] Это в основном относится к домашним маршрутизаторам, рассчитанным на администратора-неспециалиста, которому веб-интерфейс представляется гораздо более удобным, чем командная строка.

следовании дел в отношении подозреваемой личности. В то же время все современные браузеры позволяют пользователю достаточно детально управлять журналом истории посещений, который хранит как адреса посещенных сайтов и страниц, так и кэшированные страницы этих сайтов.

Угрозу приватности несут также куки. **Куки** (cookies — печенье) представляет собой небольшой фрагмент текстовых данных, которым обмениваются веб-сервер и браузер. Куки, относящийся к некоторому сеансу браузера с сервером, содержит информацию о текущем состоянии этого сеанса, аутентификационные данные и персональные настройки клиента, а также уникальный для сервера номер сеанса. В течение всего сеанса куки сохраняется *на стороне браузера*. При установлении соединения сервер генерирует содержимое куки и передает его браузеру. Веб-браузер, получив текст куки от веб-сервера, сохраняет его в виде файла. В течение всего сеанса пользователя, а возможно, и при всех повторных обращениях данного пользователя к данному сайту, браузер передает куки серверу в том же виде, в каком он его получил в последнем ответе сервера. Тем самым достигается эффект запоминания состояния сеанса, причем состояние запоминается на стороне клиента.

Веб-сервер обычно применяет данные куки пользователя для его же (пользователя) удобства, например, интернет-магазины обычно хранят в куки карту покупок пользователя, в них также может храниться история навигации пользователя по страницам сайта. Типичной информацией, помещаемой веб-сервером в куки, является идентификатор сеанса пользователя (SID), на основе которого связываются воедино отдельные запросы пользователя. Даже в случае работы по протоколу HTTP 1.1, который поддерживает длительные TCP-сеансы, эти сеансы могут прерываться из-за временной неактивности пользователя, так что объединение отдельных фрагментов сеанса (с тем, чтобы он представлялся пользователю единым) полезно для индивидуального обслуживания пользователя.

Куки бывают *постоянными* — они хранятся в файловой системе ОС и имеют длительные сроки действия — и *временными* — их браузер хранит в оперативной памяти и удаляет после своего закрытия.

Куки имеют не только срок, но и *область действия* — она задается доменным именем сайта, который создал куки. Браузер не передает куки сайту с другим доменным именем, но так как доменное имя может быть задано не для конкретного сайта, а для некоторого домена, то есть, например, не для www.cisco.com, а для cisco.com, то куки могут иметь более широкую область действия, чем один сайт.

Так как куки представляют собой текстовые файлы, то угрозы безопасности для пользователя они не представляют (за исключением случая, когда в них содержится аутентификационная информация пользователя — этот случай рассматривается в следующем разделе). Вирусы и другие вредоносные коды с помощью куки не распространяются, так что бытующее мнение, что куки могут заразить компьютер клиента, не соответствует действительности. В то же время куки могут повредить вашей *приватности*, особенно если в них помещается чувствительная личная информация — данные ваших карт покупок, формы запросов с вашими именем и фамилией, адресом и т. д. Некоторые сайты используют куки третьих сторон, например рекламных компаний. Таким образом, с помощью куки ваши предпочтения становятся известны большому количеству сайтов. Самым простым способом защиты своей приватности является полный запрет на прием куки от любых сайтов, но при этом вы можете лишиться некоторых удобств, основанных на использовании куки, например тех или иных дополнительных услуг интернет-магазина. Поэтому браузеры

оставляют пользователю возможность решать, от каких сайтов он запрещает принимать куки, а от каких, наоборот, разрешает.

## Протокол HTTPS

Веб-браузер для взаимодействия с веб-сервером по умолчанию использует протокол HTTP без дополнительных мер по обеспечению основных свойств безопасных коммуникаций, то есть аутентификации сторон, а также конфиденциальности, доступности и целостности данных. Естественно, это создает значительные риски безопасности при работе с сайтами Интернета.

Так, при перехвате злоумышленником незащищенных HTTP-пакетов, циркулирующих между веб-браузером и веб-сервером, вполне возможны атаки вида «*человек посередине*». Одной из разновидностей этой атаки является *захват сеанса*, при котором пользователь аутентифицируется на веб-сервере с помощью своего имени и пароля, а затем веб-сервер рассматривает куки, передаваемые в сообщениях браузера, как свидетельство того, что очередной запрос пришел от аутентифицированного пользователя, и продолжает сеанс без повторного запроса пароля. Понятно, что такой способ аутентификации пользователя в случае множественных сеансов протокола HTTP 1.0 или разрыва по какой-то причине длительного сеанса протокола HTTP 1.1 предоставляет злоумышленнику хорошую возможность для захвата сеанса. Для этого ему достаточно перехватить HTTP-запрос, содержащий куки, и затем посылать свои запросы от имени легального пользователя на соответствующий веб-сервер.

Другим вариантом атаки «человек посередине» является атака *повторения*, когда злоумышленник повторяет перехваченные запросы легального пользователя, возможно, несколько модифицируя их. Например, перехватив запросы сеанса пользователя с его банком, злоумышленник может инициировать повторный перевод денег, но теперь уже на свой счет.

В упомянутых примерах злоумышленник использовал уязвимости процесса *аутентификации* пользователя. Очевидно, что прослушивание открытого трафика между браузером и веб-сервером может также нарушить *конфиденциальность* данных и их *целостность*, если злоумышленник по какой-то причине внесет какие-то изменения в данные. Злоумышленник может также нарушить *доступность* данных, просто отбрасывая ответы веб-сайта.

Основным способом обеспечения перечисленных свойств безопасности данных, циркулирующих между веб-браузером и веб-сайтом, является использование **безопасного протокола передачи гипертекста** (Hypertext Transfer Protocol Secure, **HTTPS**) вместо HTTP. Словосочетание «протокол HTTPS» не вполне корректно, поскольку аббревиатура HTTPS подразумевает *совместно работающую пару протоколов: HTTP и SSL*. Тем не менее название HTTPS прижилось, и пользователь должен его употреблять, когда собирается инициировать защищенное соединение с веб-сервером, например, вводя адрес https://www.cisco.com. В HTTPS-соединении по умолчанию применяется порт 443 вместо порта 80 в HTTP-соединении.

*В HTTPS-соединении сам протокол HTTP, работающий поверх протокола SSL, остается неизменным. Все атрибуты безопасности коммуникаций — аутентификация, конфиденциальность и целостность — обеспечиваются протоколом защищенного канала SSL.*

Остановимся на некоторых особенностях аутентификации при работе веб-службы. Как вы помните, аутентификация в протоколе SSL основана на цифровых сертификатах. Поэтому при обращении веб-браузера к веб-серверу по протоколу HTTPS каждая из сторон должна иметь подписанный центром сертификации сертификат, достоверность которого можно проверить по цепочке доверия, ведущей к одному из доверенных корневых центров сертификации.

Производители с каждой копией своего браузера поставляют так называемый *встроенный цифровой сертификат*, который может применяться для аутентификации данного браузера. Этот сертификат не аутентифицирует пользователя, работающего с браузером, а служит только для создания защищенного канала при передаче данных между браузером и веб-сервером. В то же время пользователь может запросить *личный цифровой сертификат* у некоторого центра сертификации и установить его соответствующим образом в своей ОС, указав, что он должен применяться для логического входа. В таком случае вход в веб-сервер, требующий аутентификации, может происходить не на основе имени и пароля пользователя, а с помощью этого сертификата, который поставляется браузером серверу по запросу последнего.

Аутентификация сервера при установлении HTTPS-соединения всегда выполняется на основе *цифрового сертификата сервера*, получаемого владельцем сервера. Этот сертификат подтверждает, что данный веб-сервер имеет определенные (одно или несколько) доменные имена. Браузер обычно уведомляет пользователя о том, что сертификат сервера по какой-то причине является недействительным, оставляя на усмотрение пользователя окончательное решение — отказаться от соединения или все же установить его. Иногда сложный механизм проверки аутентичности сервера работает вхолостую, поскольку пользователи недооценивают угрозы со стороны «невыясненных» веб-серверов и предпочитают действовать на свой страх и риск.

**(S)** *Проверка действительности сертификата веб-сервера*

## Безопасность средств создания динамических страниц

Современные браузеры поддерживают разнообразные средства создания динамических страниц. Все они представляют собой программные коды, полученные извне, и, следовательно, несут риски, связанные с несанкционированным воздействием на клиентский компьютер, начиная с чтения конфиденциальных данных и удаления файлов пользователя до разрушения ОС. Из соображений безопасности пользователи должны очень серьезно относиться к любому предложению веб-сайта установить новую *надстройку* или *вставку*, чтобы, например, лучше проигрывать видеоклип определенного формата или же быстрее загружать файлы. Очень может быть, что, помимо своей основной функции, такая программа будет заниматься и какой-то побочной деятельностью, наносящей вред вычислительной среде пользователя, например, фиксировать нажатия клавиш клавиатуры и передавать их злоумышленнику.

Менее страшны вставки и надстройки, изменяющие параметры и внешний вид браузера так, чтобы заставить пользователя посещать определенные сайты, обращаться к определенным поисковым системам, ориентированным на рекламу, и пользоваться опре-

деленными программами. Этот вид вредоносного программного обеспечения получил название *рекламных вирусов* (Adversary Ware, *AdWare*). Избавиться от паразитов бывает непросто — они глубоко встраиваются в ОС и часто не удаляются обычными средствами браузера. Наибольшую опасность для браузера представляют *ActiveX-объекты*, действия которых не ограничены никакими рамками: они могут читать, создавать и удалять файлы, выполнять любые системные действия. Компания Microsoft и другие производители программного обеспечения снабжают свои ActiveX-объекты цифровой подписью, поэтому браузер должен принимать только ActiveX-объекты, подписанные вызывающим доверие разработчиком.

Разработчики *JavaScript-сценариев* и *Java-апплетов*, также применяющихся для создания динамических страниц, встроили в них средства безопасности, что значительно снижает риски, связанные с их использованием. Браузеры позволяют пользователям управлять процессом создания динамического содержания страницы. Так, пользователь может запретить выполнять ActiveX-объекты или Java-апплеты либо разрешить их выполнение только для доверенных сайтов, список которых он составляет сам.

# Безопасность электронной почты

Аналогично защите веб-службы, защита электронной почты также может осуществляться в двух направлениях — обеспечение приватности пользователя (например, конфиденциальности переписки) и обеспечение безопасности ресурсов компьютера (ОС, приложений).

## Угрозы приватности почтового сервиса

Пользователи, обменивающиеся сообщениями электронной почты через Интернет, должны принимать во внимание наличие следующих угроз:

- спуфинг имени отправителя — злоумышленник выдает себя за другого пользователя;
- спуфинг почтовых серверов — сервер предъявляет при передаче сообщения ложное имя домена;
- модификация сообщения — искажение или отбрасывание сообщения (то есть нарушение целостности или доступности сервиса);
- утечка информации — чтение сообщения злоумышленником (нарушение конфиденциальности);
- нарушение последовательности сообщений;
- нарушение свойства неотказуемости — отказ отправителя от факта отправки письма, отказ почтового сервера от факта приема письма, отказ получателя от факта получения письма;
- спам — засорение почтовых ящиков пользователей письмами, которые пользователи не просили или же не ожидали получить (обычно спам состоит из рекламных сообщений);
- фишинг — электронное письмо обычно является первым этапом фишинга (напомним, что целью такой атаки является завладение учетными данными пользователя

для последующего применения, например для снятия денег со счета, в электронных платежах и т. п.). Такое электронное письмо может выглядеть очень похожим на «настоящее», то есть иметь все атрибуты оформления письма некоторого банка или солидной организации и содержать просьбу обновить свой пароль по приводимой ссылке. Второй этап фишинга выполняет веб-сайт, на который попадает пользователь, перейдя по ссылке;

- нарушение приватности пользователя за счет сбора метаданных почтового сервиса.

Все перечисленные угрозы — следствие того, что изначально почтовая служба Интернета, основанная на протоколе SMTP, не поддерживала никаких механизмов защиты почтового обмена — текст сообщения в SMTP-пакетах передавался в открытом виде и его легко было прочитать и модифицировать. Спуфинг отправителя также являлся очень простым делом — почтовый клиент злоумышленника или же его почтовый сервер помещали туда любое имя, требуемое для обмана получателя. Факт такой подмены обнаружить трудно, так как имена пользователей не хранятся в DNS и проверить соответствие IP-адреса имени этим путем невозможно, а аутентификация отправителя в протоколе SMTP предусмотрена не была (получатель аутентифицировался паролем при получении сообщения). Отсутствие аутентификации отправителя делало сложным обеспечение *неотказуемости* — всегда можно было отказаться от факта отправки письма, сославшись на спуфинг отправителя, мол, это кто-то другой его написал, а указал меня в качестве отправителя. Квитанция о прочтении письма тоже не является в таких условиях достоверной — ее мог сгенерировать злоумышленник, преследуя какую-то свою цель. Отправителю спама также легко было отказаться от авторства рассылки.

К сожалению, применение прошедшего времени в описании такой грустной картины не совсем оправданно — сплошь и рядом электронная почта Интернета используется в своем первозданном виде, хотя за долгие годы существования сервиса разработаны различные стандарты безопасности электронной почты. Велика и инерция масштабной распределенной системы интернет-почты — существует огромное количество почтовых серверов, работающих под управлением старых версий программного обеспечения, не поддерживающего новые стандарты, или же под управлением новых версий, в которых новые функции защиты не активированы администраторами. Далее рассмотрены несколько стандартов безопасности почты Интернета, направленные на снижение рисков, связанных с ее работой.

## Аутентификация отправителя

Существует несколько методов аутентификации отправителя:

- ограничение отправителей провайдером услуг;
- аутентификация отправителя провайдером услуг;
- аутентификация отправителя на основе его личного сертификата.

*Ограничение отправителей провайдером услуг* не является в строгом смысле аутентификацией. Этот способ основан на том, что почтовый сервер провайдера принимает по протоколу SMTP только те письма, которые отправляются клиентами этого провайдера. Принадлежность отправителя к клиентам провайдера проверяется по его IP-адресу — адрес должен принадлежать пулу адресов, которым провайдер владеет и которые он выделяет своим клиентам. Некоторые провайдеры поступают еще строже — они не разрешают сво-

им клиентам пользоваться чужими почтовыми серверами для отправки писем, блокируя соединения на порт 25 от клиентских компьютеров, если они направлены не к почтовому серверу провайдера. То есть провайдер не только блокирует чужих пользователей, но и не разрешает своим пользователям обращаться к почтовым услугам других провайдеров. Тем самым осуществляется взаимная защита провайдеров от чужих пользователей (а также привязка пользователей к провайдеру, что преследует чисто коммерческие цели). Этот метод не гарантирует получателю аутентичности отправителя, но защищает провайдера от спама, отправляемого чужими пользователями.

*Аутентификация отправителя провайдером услуг.* Расширение протокола SMTP — **SMTP AUTH** — описывает процедуру аутентификации пользователя при отправке сообщения агентом пользователя серверу провайдера почтовых услуг. В соответствии с этим расширением почтовый сервер и агент пользователя в начале SMTP-сеанса договариваются о методе аутентификации. В список возможных методов, в частности, входят: открытый пароль (обычно передается по защищенному каналу SSL), аутентификация на основе слова-вызова и др.

Аутентификация пользователя первым сервером почтовой системы решает многие проблемы — защищает провайдера от спама, позволяет при необходимости решить проблему неотказуемости отправителя. Но при дальнейшей передаче информация об аутентичности пользователя теряется, поэтому отправитель должен полагаться на добросовестность провайдера, под чьим административным управлением находится почтовый сервер. Даже если провайдер достоин доверия, этот факт не исключает атаки «человек посередине», когда кто-то перехватывает сообщение по пути к почтовому серверу получателя и изменяет имя отправителя.

*Аутентификация отправителя на основе его личного сертификата.* Этот способ аутентификации работает «из конца в конец», так как сообщение подписывается цифровой подписью отправителя, чей открытый ключ находится в его личном сертификате. Возможность включения цифровой подписи в качестве части сообщения описана в расширении S/MIME и предусматривает использование различных стандартов цифровой подписи, например **PKCS-7** компании RSA или **PGP** (**Pretty Good Privacy**). Аутентификация на основе цифровой подписи отправителя решает несколько задач:

- получатель может проверить аутентичность отправителя и целостность сообщения;
- отправитель не может отказаться от факта отправки письма;
- подпись квитанции о получении/чтении письма делает невозможным отказ получателя от факта получения письма.

Цифровая подпись в расширении S/MIME занимает две части сообщения:

- в первой части описывается используемый стандарт цифровой подписи (протокол) и примененная хеш-функция;
- во второй части, которая является приложением, находится сама цифровая подпись, охватывающая все части сообщения вместе с их заголовками.

В варианте PKCS-7 частью цифровой подписи S/MIME является также цифровой сертификат, выданный одним из сертифицирующих центров, входящих в иерархию PKI, и удостоверяющий принадлежность открытого ключа отправителю, указанного в заголовке почтового сообщения. Вот пример сообщения, подписанного по стандарту PKCS-7 почтовым клиентом Microsoft Windows Mail 6.0 и принятого почтовым клиентом Apple Mail 6.6

(сообщение представлено в режиме Raw Source программы Apple Mail 6.6, показывающим все MIME-элементы сообщения):

```
Return-path: <natalia@olifer.co.uk>
Envelope-to: victor@olifer.co.uk
Message-ID: <5ED892093E784C5D9BFD602759D9A7C5@natashaPC>
From: <natalia@olifer.co.uk>
To: “victor” <victor@olifer.co.uk>
Subject: secure email
Date: Sat, 9 Nov 2013 11:05:18 -0000
MIME-Version: 1.0
Content-Type: multipart/signed;
    protocol=”application/x-pkcs7-signature”;
    micalg=SHA1;
    boundary=”----=_NextPart_000_0017_01CEDD3B.940A3930”
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Windows Mail 6.0.6002.18197
X-MimeOLE: Produced By Microsoft MimeOLE V6.0.6002.18463
This is a multi-part message in MIME format.
------=_NextPart_000_0017_01CEDD3B.940A3930
Content-Type: multipart/alternative;
    boundary=”----=_NextPart_001_0018_01CEDD3B.940A3930”
------=_NextPart_001_0018_01CEDD3B.940A3930
Content-Type: text/plain;
    charset=”iso-8859-1”
Content-Transfer-Encoding: quoted-printable
Hi,=20
It is much better to use a secure email correspondence.=20
Enjoy!

------=_NextPart_000_0017_01CEDD3B.940A3930
Content-Type: application/x-pkcs7-signature;
    name=”smime.p7s”
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
    filename=”smime.p7s”
MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIITJjCCBDYw
ggMeoAMCAQICAQEwDQYJKoZIhvcNAQEFBQAwbzELMAkGA1UEBhMCU0UxFDASBgNVBAoTC0FkZFRy
dXN0IEFCMSYwJAYDVQQLEx1BZGRUcnVzdCBFeHRlcm5hbCBUVFAgTmV0d29yazEiMCAGA1UEAxMZ
QWRkVHJ1c3QgRXh0ZXJuYWwgQ0EgUm9vdDAeFw0wMDA1MzAxMDQ4MzhaFw0yMDA1MzAxMDQ4Mzha
```

Здесь мы видим обе части цифровой подписи. Первая часть говорит о том, что это сообщение снабжено цифровой подписью:

```
Content-Type: multipart/signed;
protocol=”application/x-pkcs7-signature”;
micalg=SHA1;
```

Вторая часть (отделена пустой строкой) представляет собой собственно цифровую подпись, форматированную алгоритмом base64, который заменил 8-битные коды подписи ASCII-символами.

На клиенте отправителя был установлен личный сертификат, полученный от компании Comodo. Почтовые клиенты автоматически проверяют подлинность сертификата, с по-

мощью которого получена цифровая подпись PKCS-7. Для пользователя этот этап незаметен — в случае положительной проверки он видит только обычное сообщение (помеченное, как правило, особым значком и сообщением «подписано таким-то»). А вот в случае отрицательной проверки, когда сертификат отправителя по какой-то причине оказался недействительным, пользователю-получателю выводится на экран предупреждение, и решение о том, принять сообщение или нет, остается за ним.

Второй вариант цифровой подписи в стандарте S/MIME использует технологию PGP. Система **PGP** заслуживает особого внимания — именно она была первой системой цифровой подписи и шифрования почтовых сообщений Интернета, использующей технику публичных ключей. Одним из основных отличий подходов, применяемых в PGP и PKCS-7 к получению цифровой подписи, является то, что в PGP принадлежность открытого ключа некоторому отправителю должна быть подтверждена заранее, до получения письма от данного отправителя. Открытые ключи отправителей, с которыми получатель поддерживает защищенную переписку, должны храниться в некотором хранилище, доступном почтовому клиенту получателя. При приходе письма от доверенного отправителя клиентская почтовая программа получателя проверяет подлинность цифровой подписи с помощью открытого ключа отправителя, извлекая его из хранилища. Для поддержки операции проверки принадлежности открытого ключа некоторому пользователю в PGP вводится понятие **паутины доверия** (Web of Trust). Эта паутина похожа на публичную структуру PKI, так как использует цифровые сертификаты и подразумевает иерархию подписывающих их сущностей, то есть пользователей, которым вы прямо или косвенно (через иерархию доверительных отношений) доверяете. В принципе, пользователь системы PGP волен сам решать, каким образом проверять принадлежность открытого ключа другому пользователю PGP.

## Шифрование содержимого письма

Шифрование содержимого письма может происходить как «из конца в конец», так и на отдельных участках маршрута следования письма, например, между агентом пользователя и почтовым сервером, принимающим письма от пользователей.

- Шифрование содержимого письма *из конца в конец* предусмотрено спецификацией S/MIME. Она определяет способ шифрования определенной части составного сообщения, шифруемой вместе со своим заголовком.
- Шифрование *на отдельных участках* чаще всего осуществляется средствами защищенного канала, создаваемого между двумя непосредственно общающимися сторонами передачи сообщения. Этот канал может быть IPSec- или SSL-каналом, в зависимости от предпочтений администраторов сетей, в которых расположены эти стороны. Однако такой способ шифрования не гарантирует конфиденциальности сообщения на всем пути от отправителя до получателя, так как какой-то другой участок пути может не использовать защищенный канал, а протокола общей координации участников распределенной схемы передачи писем пока не существует.

Как и в случае цифровой подписи, для передачи шифрованного сообщения требуются две части — в первой части описывается факт шифрования и его способ, а вторая часть является приложением, в котором находится зашифрованная исходная часть сообщения. Спецификация S/MIME предусматривает использование PKCS-7 и PGP.

# Защита метаданных пользователя

**Метаданными электронной почты** называют некоторые характеристики сообщений, которые, не передавая самого содержимого сообщения, определяют адресатов переписки и некоторые другие обстоятельства этого процесса. Точнее, к метаданным электронной почты относят:

- имя отправителя, его почтовый адрес и его IP-адрес;
- имя получателя, его почтовый адрес и его IP-адрес;
- тип данных и их кодировки;
- уникальный идентификатор сообщения и связанных с ним сообщений;
- дату, время и временную зону отправки и получения сообщения;
- форматы заголовков сообщения;
- тему письма;
- статус сообщения;
- запрос на подтверждения получения и открытия письма.

Как видно из описания, сбор метаданных почтового сервиса может дать детальную картину о деятельности некоторого пользователя, даже если он шифрует свои сообщения. Собрать их достаточно просто. Во-первых, потому что метаданные телекоммуникационных сервисов — почты, мобильной связи, веб-сервиса и др. — законодательствами большинства стран либо совсем не защищаются, либо защищаются в намного меньшей степени, чем собственно данные сообщений сервиса. То есть в то время, как раскрытие содержимого переписки в Интернете требует решения суда, *сбор метаданных не считается атакой* и может проводиться беспрепятственно.

Во-вторых, метаданные именно электронной почты легче привязать к определенному пользователю. Метаданные электронной почты хранятся на компьютерах отправителя и получателя (как и сами сообщения), но, что опасно для приватности пользователей, еще и *в журналах почтовых серверов*, которые передавали эти сообщения. Метаданные пользователей почтового сервиса гораздо легче найти на серверах провайдеров, чем метаданные пользователей веб-сервиса, потому что пользователи почты «привязаны» к определенным почтовым серверам, например, они отправляют почту через сервер либо своего домашнего провайдера, либо корпоративный сервер, либо сервер провайдера гостиницы, вокзала или кафе, где они временно находятся, либо через сервер публичной почты, такой как Gmail. Пользователь получает почту также через вполне определенный сервер, на котором у него имеется учетная запись. Эта ситуация не похожа на веб-сервис, где пользователь может посетить любой сервер Интернета, так что найти следы его посещений путем проверки серверов практически невозможно, даже если пользователь регистрировался на некоторых из них.

## О ценности метаданных

Вынужденная отставка генерала Давида Петреуса из-за раскрытия любовной связи с его биографом Полой Бродвел подтверждает важность почтовых метаданных. Петреуса нельзя считать человеком, неосведомленным в вопросах информационной безопасности, — после многих лет блестящей военной карьеры, на пике которой он возглавлял штаб вооруженных сил США, командовал объединенной группировкой войск в Афганистане, Петреус был

назначен директором ЦРУ. Тем не менее главный шпион Америки понадеялся на то, что анонимный почтовый аккаунт в Gmail будет вполне безопасен для переписки с Полой, если они не будут отправлять с него писем, а только оставлять черновики писем в локальной папке сервера Gmail. Можно, конечно, сказать, что во всем была виновата Пола, которая начала отправлять с этого аккаунта угрожающие письма Джил Келли, другу семьи Петреусов. Джил заявила об этих письмах ФБР, и это инициировало расследование. Так как в деле было замешано имя директора ЦРУ, расследование было проведено тщательно, и выйти на след анонимного пользователя почтового аккаунта помогли почтовые метаданные. Хотя в содержании писем не было никаких «зацепок», позволяющих определить личность автора, агенты ФБР смогли его найти, сопоставив данные логических входов анонима с перемещениями лиц из круга знакомых Петреуса. Выяснилось, что IP-адреса анонима принадлежат нескольким гостиницам, в которых останавливалась Пола точно в те дни, когда аноним входил в свой аккаунт. Этого совпадения оказалось достаточно, чтобы основной подозреваемой стала Пола, а дальнейшие доказательства были уже добыты стандартными способами — обысками дома, личного компьютера и допросами.

Ценность метаданных хорошо понимают спецслужбы, недаром одна из программ NSA, о которых рассказал миру Сноуден, называется телефонной и связана с массовым сбором метаданных мобильных пользователей, благо что законы, охраняющие приватность в США, запрещают прослушивание телефонных разговоров, но не запрещают собирать метаданные мобильных клиентов.

## Спам

**Спамом** называют рассылку писем большому числу адресатов без их согласия или даже намерения вступить в переписку (по названию постоянно навязываемых посетителям кафе консервов из скетча Монти Пайтона). Является ли рассылка спама преступлением или нет, определяется законодательством конкретной страны. В начале 2000-х во многих странах были приняты акты, определяющие, что является спамом и какие наказания применять за его рассылку. Однако принятие этих актов только незначительно снизило процент спама, в общем потоке электронных писем он по-прежнему очень высок и достигает 80–85 %. Это связано с тем, что определение спама является достаточно безобидным. Например, массовая рассылка не считается спамом, если в письме ясно указана его рекламная цель, а получатель имеет возможность отписаться от рассылки. Кроме того, доказать на практике тот факт, что пользователь не давал согласие на получение письма, сложно.

Со спамом борются провайдеры Интернета. В «Терминах и условиях» их договоров с пользователями обычно есть пункт, запрещающий пользователю рассылать спам. В Интернете существуют так называемые «черные списки» (blacklists) IP-адресов электронной почты и/или доменных имен, с которых рассылался спам и письма с которых рекомендуется блокировать. Наиболее распространенной практикой является ведение черных списков в виде зон системы доменных имен (DNS). Такая практика получила название **DNSBL** (DNS Black Lists). Почтовый сервер провайдера может быть сконфигурирован так, что он автоматически опрашивает какой-либо файл DNS-зоны, содержащий черный список, и блокирует письмо, если адрес или имя его отправителя имеется в списке.

Одним из наиболее часто используемых черных списков спамеров является список некоммерческой компании Spamhouse. В главе 29 рассматривалась мощная DDoS-атака с суммарной интенсивностью трафика в 75 Гбит/с, которой в марте 2013 года был под-

вергнут веб-сервер этой компании. Атака на Spamhouse была местью одной из компаний, занимающейся рассылкой спама, за включение ее в черные списки. Spamhouse имеет очень мощную распределенную систему DNS-серверов, которые предоставляют по запросу почтовых серверов или клиентов черные списки. Spamhouse ведет несколько таких списков — для спамеров, для хостов, зараженных вирусами, для хостов, не выполняющих аутентификацию при передаче письма на почтовый сервер (это считается нарушением политики безопасности почтового сервиса). Владельцы адресов, попавших в черный список, могут оспорить решение Spamhouse, это нормальная процедура, так как при современном «незащищенном» состоянии почты Интернета ошибки при определении источника спама неизбежны.

## Атаки почтовых приложений

Текст почтового сообщения, на первый взгляд, не может причинить вред компьютеру пользователя. Однако гибкость современной почты позволяет злоумышленникам внедрять в сообщения разнообразную информацию, в том числе исполняемые коды, которые уже не являются столь безобидными. Проще всего поместить вредоносный код в *приложение* почтового сообщения. Почтовый клиент при открытии пользователем приложения передает его одной из программ клиентского компьютера для обработки. Если приложением является исполняемый файл, например файл с расширением .exe, то почтовый клиент передаст его на выполнение ОС.

Расширения выполняемых программ могут быть и другими, например, если это Java-программа или скрипт командного процессора. Исполняемый код может быть также выполнен в виде макроса или скрипта какого-либо документа, например документа MS Word или Excel. Такое приложение вызывает меньше подозрений (ведь это только текст или таблица), но скрипты документов также могут получить доступ к ресурсам компьютера и причинить ему вред. Наконец, вредоносный код может находиться и в *теле сообщения* (если оно написано на языке HTML) в виде JavaScript-скриптов или, что действительно опасно, ActiveX-объектов. Поэтому все почтовые сообщения должны проходить обязательную проверку антивирусной программой на наличие вредоносного кода в приложениях и самом сообщении.

# Безопасность облачных сервисов

## Облачные вычисления как источник угрозы

### Ограниченная подконтрольность провайдера

Модель облачных вычислений существенно отличается от традиционной модели вычислений, используемой сегодня в корпоративных ИС, и это отличие прежде всего сказывается на обеспечении их безопасности. Природа данного отличия довольно проста — вместо того, чтобы строить собственную ИС и управлять ею силами сотрудников предприятия, предприятие начинает пользоваться услугами системы, созданной посторонней организацией-провайдером. При этом организация-провайдер владеет всей инфраструктурой ИС и управляет ею, обеспечивая в том числе безопасность данных, принадлежащих предприятию-клиенту. Сотрудники предприятия-клиента используют свои компьютеры только как терминалы доступа к облаку, а все данные предприятия, включая личные данные

сотрудников, хранятся и обрабатываются «где-то там, в облаке». Предприятию-клиенту остается только следовать совету Рональда Рейгана: «доверяй, но проверяй».

Такой революционный переворот в модели вычислений не мог не обеспокоить специалистов по безопасности. Многие из них согласны в том, что облачные вычисления — вещь хорошая, но *отсутствие гарантий безопасности* облачных вычислений сводит на нет все преимущества облака.

Для этих опасений, безусловно, есть основания, однако многое зависит от типа организации-клиента и модели облачных вычислений, которую клиент собирается использовать. Понимание моделей облачных вычислений — необходимое условие для правильной оценки рисков предприятия при переходе на новый тип ИС.

Предприятие-клиент облачных сервисов имеет весьма *ограниченный контроль* над механизмами безопасности своих данных, обрабатываемых виртуальными машинами провайдера и хранящимися в виртуальных хранилищах. Особенно это справедливо для услуг модели SaaS, когда защита всех элементов ИС, включая прикладные программы пользователя, осуществляется провайдером. Предприятие-клиент в этом случае участвует лишь в обеспечении безопасности компьютеров своих сотрудников, которые применяются как терминалы облачной среды. При этом клиент IaaS-сервиса должен самостоятельно заботиться о безопасности своих приложений — следить за тем, чтобы обновления приложений периодически получались и устанавливались, устанавливать и обслуживать антивирусные программы и программы блокировки спама, выполнять все остальные действия в соответствии с политикой безопасности предприятия, которые относятся к приложениям. Обычно в этой модели конфигурирование средств безопасности ОС также является делом клиента.

В модели PaaS контроль над средствами безопасности верхних уровней разделяется между провайдером и клиентом. В таких условиях клиент должен стараться получить как можно более *полный доступ к средствам аудита провайдера* — к сообщениям и журналам средств безопасности, его виртуального файервола, системы IDS, антивирусных и антиспамовых программ и т. п. Кроме того, полезно также проводить *аудит работы самого провайдера*, привлекая для этого сторонние фирмы, пользующиеся устойчивой репутацией в этой области.

Получение информации от средств защиты провайдера в реальном времени (мониторинг) и доступ к историческим данным этих средств должен быть предусмотрен в *договоре о предоставлении услуг провайдером*. Этот важный документ должен оговаривать все детали взаимоотношений провайдера и клиента, а так как облачные услуги являются новым видом телекоммуникационных услуг, то внимание ко всем его деталям должно быть самое пристальное.

**(S)** *Соглашение об уровне обслуживания с провайдером облачных сервисов*

*Наличие сертификатов* на соответствие средств безопасности провайдера популярным программам сертификации также может частично компенсировать отсутствие полного контроля над этими средствами.

## Разделение сложной инфраструктуры провайдера

При использовании услуг облачного провайдера вы *разделяете его инфраструктуру* с другими арендаторами, которых не знаете. Разделение ресурсов провайдера осуществляется не на физическом уровне, а с помощью механизмов виртуализации. Значит, злоумышленник

может заключить договор с вашим провайдером и попытаться использовать бреши в механизмах виртуализации для получения несанкционированного доступа к вашим данным. Кроме того, нельзя исключать ошибок персонала провайдера, в результате которых виртуальные барьеры могут быть нарушены.

Инфраструктура облачного провайдера намного *сложнее* инфраструктуры стандартного центра данных, что представляет собой дополнительную угрозу безопасности облачных сервисов. Кроме таких стандартных элементов виртуализации, как гипервизоры, виртуальные машины, виртуальные маршрутизаторы и файерволы, подобная инфраструктура включает многочисленные дополнительные компоненты управления: средства самостоятельного динамического выделения ресурсов клиентами, измерители потребления ресурсов, средства управления квотами, нагрузкой, мониторинга качества, услуг и т. п. Кроме того, облачные сервисы могут быть реализованы в облачной среде другого провайдера, что еще более усложняет картину. Обычно администраторы корпоративных ОС и приложений получают доступ к ним через локальную сеть. При использовании облачных сервисов административный *доступ должен выполняться через Интернет*, что несет дополнительные угрозы. Необходимо убедиться, что облачный провайдер поддерживает только хорошо защищенные соединения для предоставления административного доступа своим клиентам.

## Угроза конфиденциальности персональным данным

При использовании услуг корпоративной сети персональные данные сотрудников предприятия хранятся в справочной службе предприятия (например, работающей на основе Microsoft Active Directory) и предприятие несет ответственность за их конфиденциальность в соответствии с законами и правовыми актами. В том случае, когда сотрудники пользуются услугами облачного провайдера, их персональные данные (имена и пароли) хранятся в справочной службе провайдера. Так как ответственность за их конфиденциальность, в конечном счете, все равно несет предприятие, необходимо убедиться, что провайдер надлежащим образом обеспечивает их конфиденциальность — как при передаче личных данных через Интернет, так и при хранении их в разделяемой между арендаторами справочной службе провайдера. Для повышения уровня защиты личных данных можно применять раздельные наборы данных для локальной аутентификации пользователей и их аутентификации у облачного провайдера. Но это довольно громоздкое решение, которое для большой организации может оказаться неработоспособным.

Другим решением является применение схемы федеративной аутентификации — личные данные пользователей хранятся в справочной службе предприятия, а служба аутентификации провайдера взаимодействует со службой аутентификации предприятия через защищенное соединение Интернета.

## Мультинациональность облачных услуг, законодательство и политика

В мире облачных вычислений существуют различные варианты реализации сервисов в отношении того:

- где данные физически размещены;
- где они обрабатываются;
- откуда происходит доступ к этим данным.

Зачастую эти три точки находятся в разных странах, в каждой из которых действует свое законодательство. В разных странах также могут быть зарегистрированы предприятие-клиент и облачный провайдер. Законодательные аспекты таких многонациональных услуг определены плохо — эта работа находится в начальной стадии. В результате появляется много неясностей во взаимоотношениях провайдеров и клиентов многонациональных облачных услуг, а значит, риски использования этих услуг весьма высоки. Снизить риски, связанные с многонациональными облачными сервисами, можно за счет непосредственного указания в договоре конкретной судебной инстанции определенной страны, которая будет разбирать любые споры между провайдером и клиентом, если они возникнут.

Чтобы облачные вычисления могли успешно развиваться как глобальная, не знающая границ услуга, они должны быть отделены от политики. К сожалению, сегодня законы, принимаемые разными правительствами, часто оказывают негативное влияние на развитие глобального облака. Так, одним из результатов принятия США Патриотического Акта 2004 стало то, что Канада решила не использовать серверы Интернета, расположенные на территории США, опасаясь за конфиденциальность данных, которые Канада хранит на своих компьютерах. Разоблачения Сноудена в отношении программы PRISM привели Бразилию к решению построить собственный сегмент Интернета с основными сервисами, поддерживаемыми серверами, находящимися на территории Бразилии.

## Облачные сервисы как средство повышения сетевой безопасности

Несмотря на то что облачные сервисы принято рассматривать как источник новых угроз безопасности, они могут существенно *улучшить информационную безопасность* предприятия — особенно если это небольшое предприятие, у которого нет специального подразделения, занимающегося безопасностью. Существует несколько преимуществ облачной модели над традиционными подходами к организации вычислений. Некоторые из них следует рассматривать в качестве потенциально применимых, поскольку они предполагают соответствующую корректную организацию процессов управления безопасностью провайдером услуг; другие же являются следствием самой парадигмы облачных вычислений.

### Избыточность и резервирование ресурсов

Высокая степень масштабирования ресурсов облачных провайдеров обеспечивает также *высокую доступность* этих ресурсов и, как следствие, данных, обрабатываемых этими ресурсами. Избыточность и резервирование ресурсов являются одними из основных принципов построения облачной среды. Обычно облачный провайдер располагает большим количеством центров данных в разных географических точках, а возможно и странах, при этом реплики данных одного и того же клиента хранятся в нескольких таких центрах. Это требуется как для повышения производительности за счет приближения данных к пользователям (если это данные публичные, предназначенные для всех пользователей Интернета), так и для обеспечения доступности данных в случае технических отказов или природных катастроф. Кроме того, избыточность ресурсов вызвана необходимостью обеспечивать эластичность сервисов, то есть возможность быстрого увеличения нагрузки по запросу клиента. В корпоративной сети такую высокую доступность ресурсов обеспечить трудно, и она чаще всего будет экономически неоправданна.

## Поглощение DDoS-атак

Распределенная избыточная инфраструктура центров данных облачного провайдера позволяет *эффективно бороться с DDoS-атаками*. Отражение мощной DDoS-атаки является, наверное, наиболее сложной задачей для администратора корпоративной сети, так как фильтрация потока пакетов интенсивностью в десятки, а то и сотни гигабайт в секунду, направленного на единственную копию сервера через канал связи, требует наличия очень производительного файервола. Такие файерволы, специально созданные для отражения DDoS-атак, существуют. Однако даже если предприятие может себе позволить приобрести и установить столь дорогостоящее устройство, остается проблема узкого места, которое представляет собой единственная копия атакуемого сервера (скорее всего, это веб-сервер, возможности которого публиковать открытые данные злоумышленник пытается подавить) и единственный канал связи сервера с Интернетом с фиксированной пропускной способностью. Поэтому в начальной стадии атаки, когда администратор еще не успел определить признаки, отличающие пакеты атаки от пакетов легальных пользователей, файервол принципиально не может защитить сервер от атаки, пропуская весь трафик к серверу и тем самым блокируя его работу.

В сети провайдера облачных услуг отразить DDoS-атаку на ресурсы клиента принципиально проще. На рис. 30.4 показана достаточно типичная структура сети облачного провайдера с четырьмя центрами данных, рассредоточенными по географическим регионам, при этом в каждом центре имеется сервер с копией данных некоторого клиента. Для баланса

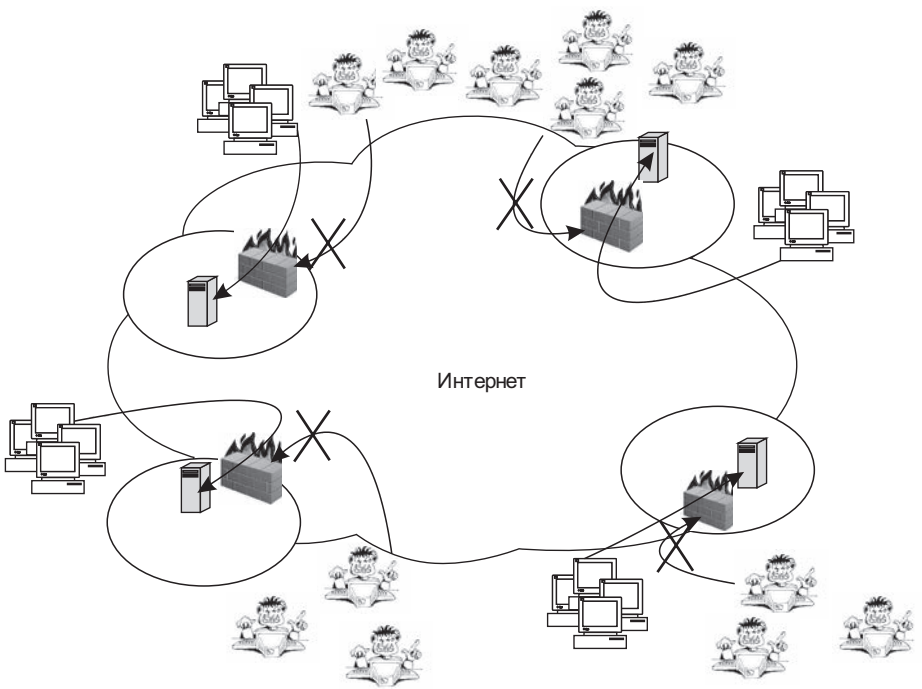



**Рис. 30.4.** Распределение и поглощение трафика DDoS-атаки инфраструктурой облачного провайдера

нагрузки провайдер применяет маршрутизацию с произвольной (anycast) рассылкой, в результате трафик запросов клиентов направляется ближайшему (относительно метрики маршрутизации) серверу.

При возникновении DDoS-атаки трафик ботов злоумышленника также распределяется между серверами провайдера, как и трафик клиентов, поэтому атака не может быть такой же эффективной, как в случае единственного сервера и единственной линии доступа к нему. К тому же провайдер, как правило, поддерживает значительный запас пропускной способности линий доступа для успешного обслуживания пиков потребностей клиентов, поэтому «забить» линии доступа облачного провайдера злоумышленнику труднее. Можно сказать, что в начальный период DDoS-атаки инфраструктура облачного провайдера «впитывает» трафик атаки как губка, без значительного ущерба для трафика легальных пользователей. А так как серверы провайдера защищены высокопроизводительными файерволами, то после обнаружения факта атаки и определения признаков, по которым можно отличить трафик атаки от трафика пользователей, администратор провайдера вносит соответствующие изменения в правила файерволов и они начинают блокировать трафик атаки. Внимательный читатель может заметить, что только что прочитанное пояснение вполне подходит к описанию DDoS-атак на корневые DNS-серверы — и это не удивительно, так как механизм поглощения трафика атаки в обоих случаях одинаков.

## Квалификация персонала и качество платформы вычислений

Провайдер облачных услуг является крупной организацией (иначе он не сможет обеспечить масштабируемость, эластичность и некоторые другие свойства этой модели) и, как всякая крупная организация, может себе позволить специализацию своих администраторов и операторов во всех областях обеспечения информационной безопасности. В результате специалисты провайдера получают большой опыт в распознавании всего спектра угроз, актуальных в настоящее время, а также в установке и конфигурировании средств отражения этих угроз, таких как файерволы, системы IDS/IPS, антивирусные системы и т. п., что, естественно, повышает безопасность данных клиентов облачных сервисов.

Важно и то, что платформа вычислений (сетевая и компьютерная) может оказаться более качественной у облачного провайдера, чем у предприятия-клиента. Причиной является ее более *высокая степень однородности*, которая определяется тем, что провайдер оказывает одни и те же услуги большому количеству клиентов. Поэтому центры данных провайдера, как правило, строятся на одних и тех же моделях маршрутизаторов, коммутаторов, файерволов, серверов и гипервизоров виртуальных машин. Однородность упрощает управление и сокращает количество ошибок конфигурирования, а значит, делает более простым и эффективным обеспечение безопасности такой платформы.

Другая причина заключается в *масштабности* центров данных и сети провайдера облачных сервисов, что позволяет ему покупать для платформы самые совершенные и производительные компоненты защиты данных, например высокопроизводительный файервол, способный отфильтровывать пакеты интенсивных DoS-атак. Провайдеры облачных услуг стараются сертифицировать свои платформы по различным программам сертификации безопасности, чтобы завоевать доверие клиентов, — это также повышает вероятность того, что используемая платформа обладает высоким качеством.

После выяснения новых видов угроз, связанных с применением публичных облачных сервисов, у корпоративных специалистов по безопасности может возникнуть вопрос: а стоит ли овчинка выделки? Ответ не очевиден, но при его выборе нужно принимать во внимание не только тактические, но и стратегические соображения. А стратегические соображения говорят, что у облачных сред большое будущее, так что если сегодня они, возможно, еще не созрели для немедленного применения крупным предприятием с большим количеством чувствительных данных, то их безусловно нужно изучать и, возможно, опробовать, имея в виду потенциальную значимость для развития информационных технологий (и не только их). Соображения перспективности и значимости облачных сервисов должны учитываться администраторами безопасности и руководством предприятия при выработке политики безопасности в отношении использования этих сервисов.

# Вопросы к части VIII

1. Отметьте в таблице, что из перечисленного может быть отнесено к субъектам, а что к объектам системы контроля доступа к ресурсам ИС:

| | **Объекты** | **Субъекты** |
|---|---|---|
| Пользователи | | |
| Устройства | | |
| Прикладные процессы | | |
| Файлы | | |
| Пропускная способность каналов связи | | |
| Сетевые сервисы | | |

2. Используя приведенный далее список терминов, вставьте пропущенные слова в следующее предложение: «Пользователи получают ... к ресурсам ИС в результате ..., однако прежде им необходимо успешно пройти ... и ...». (Аутентификация, авторизация, идентификация, права доступа).
3. Вставьте пропущенные слова из списка: «Стало известно, что в Интернете уже появились ..., направленные на использование ... новой версии браузера. Реализация данной ... может привести к ..., которая нанесет ... нашему предприятию». (Атака, уязвимость, эксплойт, ущерб, угроза).
4. Приведите примеры ситуаций, при которых обеспечивается конфиденциальность, но не гарантируется целостность данных.
5. Как называется свойство, которое характеризует систему, в которой документ всегда имеет достоверную информацию о его источнике (авторе)?
6. Сколько секретных ключей требуется для переписки 50 человек «каждый с каждым», использующих алгоритм DES?
7. Что определяет криптостойкость алгоритма шифрования:
   а) секретность алгоритма;
   б) секретность ключа;
   в) сложность алгоритма.
8. Можно ли передать секретный ключ по открытому каналу, не используя шифрование?
9. В главе 28 рассмотрен пример архитектуры сети с защитой периметра и разделением внутренних зон. Предложите альтернативный вариант.
10. Какие из перечисленных ниже свойств образуют триаду безопасности CIA:
    а) конфиденциальность;
    б) неотказуемость;
    в) аутентичность;
    г) владение;

д) целостность;
е) полезность;
ж) захват привилегий;
з) доступность.

11. Укажите, какие из перечисленных атак являются активными (выберите вариант ответа):
    а) прослушивание сетевого трафика;
    б) спуфинг;
    в) социальный инжиниринг;
    г) атака «отказ в обслуживании»;
    д) внедрение вируса.
12. Укажите, какие из следующих утверждений являются ошибочными:
    а) Нет смысла предусматривать подсистему аутентификации приложения, основанную на использовании многоразовых паролей, если многоразовые пароли уже используются и при логическом входе в систему.
    б) Надежная система авторизации может компенсировать недостаточную стойкость паролей пользователей, которые они использовали при входе.
    в) Стойкость системы защиты считается достаточной, если затраты на нее превысили запланированный уровень.
    г) Затраты на обеспечение безопасности информации, по крайней мере, не должны превышать величину потенциального ущерба от ее утраты.
    д) Иногда стойкость системы обеспечения безопасности считают достаточной, если стоимость ее преодоления злоумышленниками превосходит стоимость полученной ими выгоды.
13. Что из перечисленного может быть секретным ключом криптосистемы (выберите вариант ответа):
    а) геометрическая фигура;
    б) сборник детских сказок;
    в) картина;
    г) число 5.
14. В каких из перечисленных ниже средствах обеспечения безопасности используется шифрование (выберите вариант ответа):
    а) аутентификация,
    б) авторизация,
    в) протокол NAT;
    г) защищенный канал;
    д) сетевой экран прикладного уровня;
    е) фильтрующий маршрутизатор;
    ж) цифровая подпись.
15. Какую роль в обеспечении безопасности компьютерной сети играет протокол NAT?
16. Вставьте пропущенные слова («открытый» или «закрытый») в следующее предложение: «Сертификат может быть представлен в форме, состоящей из трех частей: во-первых, ... части; во-вторых, той же информации, зашифрованной ... ключом серти-

фицирующей организации, в-третьих, части, представляющей собой первые две части, зашифрованные... ключом владельца».

17. Установите соответствие следующих терминов на русском языке (ущерб; уязвимость; вредоносное ПО; доступность; целостность; подмена содержимого пакета; распределенная атака, «отказ в обслуживании») и английском языке (Denial of Service; availability; integrity; vulnerability; malware; impact; spoofing; DDoS).
18. Укажите, какие из нижеперечисленных свойств определяют одностороннюю функцию Y(x) (выберите вариант ответа):
    а) Y(x) чрезвычайно сложно вычислить для любого входного значения x;
    б) Y(x) просто вычисляется для любого входного значения x;
    в) аргумент x легко вычислить по известному Y;
    г) вычисление x по известному Y чрезвычайно затруднено;
    д) вычисление x по известному Y абсолютно невозможно;
    е) значения функции от близких значений аргументов не должны значительно отличаться.
19. Поясните назначение алгоритма Диффи—Хеллмана (выберите вариант ответа):
    а) решает проблему передачи секретного ключа по открытому каналу;
    б) это то же самое, что и алгоритм RSA;
    в) смягчает проблему масштабируемости распределения ключей.
20. Сколько ключей требуется для секретной переписки 50 человек «каждый с каждым», использующих алгоритм RSA?
21. Какие из следующих утверждений правильные (выберите вариант ответа):
    а) открытый ключ нужно защищать от подглядывания;
    б) открытый ключ нужно защищать от подмены;
    в) теоретически возможно зашифровать текст одним ключом, а расшифровать другим, который абсолютно никак не связан с первым;
    г) алгоритм RSA основан на использовании функции разложения произведения большого числа на сомножители.
22. В чем состоит ручная синхронизация паролей? Варианты ответов:
    а) администратор синхронизирует значения БД паролей на разных серверах сети;
    б) ручная процедура приведения всех паролей пользователя к единому значению;
    в) использование одного и того же пароля для доступа к разным по степени защищенности сервисам.
23. Поясните, что представляют собой аутентификаторы из разряда «что-то знаю» (выберите вариант ответа):
    а) многоразовый пароль;
    б) одноразовый пароль, сгенерированный устройством;
    в) преобразование пользователем в соответствии с некоторым правилом символов, выводимых системой на экран, и использование их в качестве пароля;
    г) информация о месте и времени встречи.
24. Отметьте недостатки аутентификации с использованием многоразовых паролей (выберите вариант ответа):
    а) возможность разгадывания пароля, если он слишком короткий;

б) необходимость передачи пароля по сети в открытом виде;
в) возможность подслушивания, подглядывания, «выманивания» пароля;
г) ручная синхронизация;
д) трудность запоминания надежных паролей;
е) сложность реализации.

25. Укажите, что из перечисленного содержится в сертификате (выберите вариант ответа):
а) информация о владельце сертификата;
б) открытый ключ владельца сертификата;
в) закрытый ключ владельца сертификата;
г) открытый ключ сертифицирующей организации;
д) закрытый ключ сертифицирующей организации;
е) информация о сертифицирующем центре, выпустившем данный сертификат.

26. Поясните, как убедиться в подлинности сертификата (выберите вариант ответа):
а) его надо расшифровать с помощью открытого ключа владельца сертификата и сравнить с открытой информацией сообщения;
б) его надо расшифровать с помощью закрытого ключа владельца сертификата и сравнить с открытой информацией сообщения;
в) его надо расшифровать с помощью открытого ключа сертифицирующей организации;
г) его надо послать для проверки в сертифицирующий центр.

27. Основная идея процедуры единого логического входа состоит в том, что (выберите вариант ответа):
а) пользователь выполняет логический вход в сеть только один раз при поступлении на работу, все остальное время система выполняет только его авторизацию;
б) все пользователи выполняют процедуру логического входа с одного и того же компьютера;
в) пользователь выполняет логический вход в сеть только один раз, затем результат этой аутентификации используется другими серверами или приложениями.

28. Укажите цели использования электронной подписи (выберите вариант ответа):
а) доказательство целостности сообщения;
б) обеспечение конфиденциальности сообщения;
в) доказательство авторства сообщения.

29. Поясните, что из перечисленного характеризует дискреционный метод управления доступом, а что — мандатный (выберите вариант ответа):
а) описание прав доступа дается в виде *списков ACL;*
б) *право назначать права* на доступ к объектам делегируются отдельным пользователям — владельцам объектов;
в) данная система управления доступом страдает от невозможности гарантированно проводить общую политику;
г) права доступа отдельного пользователя описываются ACE;
д) решение о предоставлении права доступа принимается операционной системой на *основе правила*;

е) субъект может по своему усмотрению передать часть своих полномочий другим субъектам;

ж) данная система управления доступом позволяет гарантированно проводить общую политику.

30. Основными функциями файервола являются (выберите вариант ответа):
    а) аутентификация;
    б) авторизация;
    в) аудит;
    г) фильтрация трафика.
31. Как может выглядеть запись в расширенном списке доступа, запрещающая передачу сообщений ICMP «Перенаправление маршрута» от любого хоста к хостам подсети 145.8.146.0/24?
32. Что означает данный список доступа:

    access-list 2 permit 93.8.25.2 0.0.0.0 access-list 2 deny 93.8.25.0 0.0.0.255?
33. Какие из следующих утверждений о технологии NAT справедливы? Варианты ответов:
    а) устройство, поддерживающее NAT, заменяет внешние IP-адреса пакетов, которые использовались при маршрутизации пакета через Интернет, внутренними;
    б) устройство, поддерживающее NAT, фильтрует входящий трафик, отбрасывая все пакеты с частными IP-адресами назначения;
    в) причина обращения к технологии NAT — дефицит адресов IPv4;
    г) причина обращения к технологии NAT — скрытие адресов хостов для повышения безопасности сети.
34. Какова должна быть защита общедоступного сервера (компьютера-бастиона)? Варианты ответов:
    а) поскольку доступ к этому компьютеру открыт, его помещают в демилитаризованную зону, поэтому он не требует защиты;
    б) уровень защиты компьютера-бастиона такой же, как у серверов внутренней сети;
    в) компьютер-бастион должен быть особенно тщательно защищен от внешних пользователей;
    г) правила, определенные для компьютера-бастиона по доступу к ресурсам внутренней сети, должны быть более строгими, чем правила, регламентирующие доступ к нему внешних пользователей.
35. В чем состоят главные отличия системы обнаружения вторжений (IDS) от файерволов с запоминанием состояния сеанса? Варианты ответов:
    а) IDS не блокирует подозрительный трафик;
    б) IDS анализирует не только события, связанные с прохождением трафика, но и другие подозрительные события в сети;
    в) IDS не выполняет журнализацию событий;
    г) в отличие от файерволов, системы IDS всегда являются исключительно программными.
36. Справедливо ли следующее утверждение: «Прокси-сервер всегда работает в режиме запоминания состояния сеанса»?

37. Какие функции системы безопасности из перечисленных направлены на обеспечение подотчетности (выберите вариант ответа):
    а) авторизация;
    б) аудит;
    в) протоколирование событий;
    г) мониторинг;
    д) журнализация событий.
38. В IDS для обнаружения вторжений применяются несколько типов правил:
    а) правила, основанные на сигнатуре атаки;
    б) правила доступа на основе меток безопасности объекта и субъекта;
    в) правила, основанные на анализе протоколов;
    г) правила, основанные на статистических аномалиях трафика.
39. В чем состоит главная уязвимость протокола IP? Варианты ответов:
    а) заголовок IP-пакета переносит адреса источника и назначения в незашифрованном виде;
    б) он использует широковещательные адреса назначения;
    в) он позволяет каждому узлу Интернета взаимодействовать с каждым без предварительного установления соединения;
    г) он поддерживает фрагментацию пакетов.
40. С помощью протокола ICMP злоумышленник может (выберите вариант ответа):
    а) определить, что некоторый хост находится в работоспособном состоянии;
    б) организовать DoS-атаку;
    в) перенаправить маршрут;
    г) узнать, через какие промежуточные маршрутизаторы проходит маршрут до некоторого конечного узла.
41. С какой целью злоумышленник должен подавить отправку ACK-сегментов на атакуемый сервер в ходе атаки SYN Flood? Варианты ответов:
    а) чтобы скрыть свой IP-адрес;
    б) чтобы открытые соединения оставались незавершенными;
    в) чтобы закрыть открытые соединения.
42. Чем атака DNS-спуфинга отличается от атаки отравления DNS-кэша? Варианты ответов:
    а) ничем, это разные названия одной и той же атаки;
    б) в результате атаки отравления DNS-кэша DNS-сервер терпит крах, в то время как атака DNS-спуфинга к краху сервера не приводит;
    в) в атаке DNS-спуфинга ложный ответ предается клиенту, а в атаке отравления DNS-кэша — DNS-серверу;
    г) в атаке DNS-спуфинга ложный ответ передается DNS-серверу, а в атаке отравления DNS-кэша — клиенту.
43. Каким образом можно «подделать» маршрутное объявление BGP, которое вы передаете вашему соседу, если ваша автономная система является транзитной для этого маршрута, а вы хотите, чтобы сосед не использовал этот маршрут для передачи трафика (выберите вариант ответа):

а) добавить в объявление номер автономной системы вашего соседа;
б) выбросить из последовательности определенную автономную систему;
в) добавить номер своей автономной системы несколько раз;
г) заменить своими адрес сети и номер исходной автономной системы.

44. С какой целью в семействе протоколов IPSec функции обеспечения целостности дублируются в двух протоколах — AH и ESP?
45. Какую цель обычно преследует злоумышленник, используя эффект переполнения стека? Варианты ответов:
    а) испортить локальные переменные функции ядра ОС;
    б) поместить в стек вредоносный код и передать ему управление;
    в) исчерпать оперативную память компьютера.
46. Антивирусная программа, работающая по методу сигнатур, может (выберите вариант ответа):
    а) выявлять только известные вредоносные коды;
    б) выявлять только вирусы, но не черви;
    в) выявлять «троянские кони»;
    г) выявлять только новые виды вредоносных кодов.
47. Могут ли куки, переданные веб-сервером вашему веб-браузеру, заразить ваш компьютер вирусом?
48. С какой целью веб-сервер передает куки веб-браузеру клиента? Варианты ответов:
    а) для создания динамических веб-страниц;
    б) для сохранения состояния сеанса пользователя;
    в) для повышения защищенности сеанса.
49. Какое утверждение относительно протокола HTTPS является правильным? Варианты ответов:
    а) протокол HTTPS использует защищенный канал SSL для предотвращения атак на сеанс между веб-браузером и веб-сервером;
    б) протокол HTTPS не является протоколом в строгом смысле этого термина;
    в) протокол HTTPS использует цифровые сертификаты веб-браузера и веб-сервера для образования защищенного канала.
50. Что делает облачные вычисления более безопасными, чем традиционные? Варианты ответов:
    а) высокая квалификация специалистов предприятий, предоставляющих облачные сервисы;
    б) высокое качество вычислительной платформы;
    в) способность поглощать трафик DDoS;
    г) централизация вычислительных ресурсов.

# Рекомендуемая и использованная литература


1. Самую полную и достоверную информацию можно почерпнуть из стандартов: IETF RFC — о протоколах стека TCP/IP, ITU-T — о технологиях первичных сетей, IEEE 802 — о технологии Ethernet и Wi-Fi, 3GPP — о технологиях мобильных сотовых сетей.
2. *Стивен Браун*. Виртуальные частные сети. М.: Лори, 2001.
3. *Шринивас Вегешна*. Качество обслуживания в сетях IP. Вильямс, 2003.
4. *Галатенко В*. Основы информационной безопасности. Бином. Лаборатория знаний, 2008.
5. *Аннабел З. Додд*. Мир телекоммуникаций. Обзор технологий и отрасли. М.: ЗАО «Олимп-Бизнес», 2002.
6. *Кейт Дж. Джонс*. Анти-хакер. Средства защиты компьютерных сетей. Справочник профессионала. 2003.
7. *Оливер Ибе*. Сети и удаленный доступ. Протоколы, проблемы, решения. ДМК Пресс, 2002.
8. *Дуглас Э. Камер*. Сети TCP/IP. Том 1. Принципы, протоколы и структура. Вильямс, 2003.
9. *Кеннеди Кларк, Кевин Гамильтон*. Принципы коммутации в локальных сетях Cisco. 2003.
10. *Куроуз Дж., Росс К*. Компьютерные сети. Нисходящий подход. Эксмо, 2016.
11. *Лапонина О. Р*. Основы сетевой безопасности: криптографические алгоритмы и протоколы взаимодействия: Курс лекций. Бином. Лаборатория знаний, 2009.
12. *Одом Уэнделл*. Официальное руководство Cisco по подготовке к сертификационным экзаменам CCNA ICND2 200-105. Маршрутизация и коммутация. Вильямс, 2018
13. *Олифер В*. Направления развития средств безопасности предприятия. Электроника, № 1, 2001.
14. *Олифер В. Г., Олифер Н. А*. Новые технологии и оборудование IP-сетей. СПб.: БХВ-Санкт-Петербург, 2000.
15. *Олифер В. Г., Олифер Н. А*. Сетевые операционные системы, 2-е изд. СПб.: Питер, 2008.
16. *Олифер В. Г., Олифер Н. А*. Основы компьютерных сетей. СПб.: Питер, 2009.
17. *Олифер В. Г., Олифер Н. А*. Безопасность компьютерных сетей. М.: Горячая Линия — Телеком, 2015.
18. *Олифер В., Петрусов Д*. Внедрение услуг IP-телефонии в сети оператора связи. Аналитический и информационный журнал Документальная Электросвязь, № 8, январь 2002.
19. *Фейт Сидни*. TCP/IP. Архитектура, протоколы, реализация. М.: Лори, 2000.

20. *Сингх Саймон*. Книга шифров. Тайная история шифров и их расшифровки. АСТ, Астрель, 2007.
21. *Слепов Н. Н.* Синхронные цифровые сети SDH. Эко-Трендз, 1998.
22. *Марк Спортак, Френк Паппас и др.* Компьютерные сети и сетевые технологии: ТИД «ДС», 2002.
23. *Степутин А. Н., Николаев А. Д.* Мобильная связь на пути к 6G. Инфра-Инженерия, 2018.
24. *Стивенс Ричард*. Протоколы TCP/IP. Практическое руководство. СПб.: БХВ, 2003.
25. *Столлингс В*. Передача данных, 4-е изд. СПб.: Питер, 2004.
26. *Столлингс В*. Современные компьютерные сети, 2-е изд. СПб.: Питер, 2003.
27. *Столлингс В*. Беспроводные линии связи и сети. Диалектика-Вильямс, 2003.
28. *Таненбаум Э., Уэзеролл Д*. Компьютерные сети, 5-е изд. СПб.: Питер, 2019.
29. *Уолрэнд Дж*. Телекоммуникационные и компьютерные сети. Вводный курс. М.: Постмаркет, 2001.
30. *Халсалл Фред*. Передача данных, сети компьютеров и взаимосвязь открытых систем. М.: Радио и связь, 1995.
31. *Чирилло Джон*. Защита от хакеров. СПб.: Питер, 2002.
32. *Харрис Ш*. CISSP. Руководство для подготовки к экзамену. 2011.
33. *Щербо В. К.* Стандарты вычислительных сетей. Взаимосвязи сетей. Справочник. М.: Кудиц-образ, 2000.
34. *Sauter Martin*. From GSM to LTE-Advanced, John Wiley & Sons, Ltd, 2014.
35. *Parker Donn B*. Fighting Computer Crime: A New Framework for Protecting Information, 1998.
36. *Davies Josef*. Understanding IPv6, 3E, Pearson, 2012.
37. *Peltier Thomas R*. Information Security Risk Analysis, Third Edition, 2010.
38. *Kabiri Peyman*. Privacy Intrusion Detection and Response, IGI Global, 2011.
39. *Solomon Michael G*. Security Strategies in Windows Platforms and Applications — Jones & Bartlett Learning, 2010.
40. *Pfleeger Charles P., Lawrence Shari*. Security in Computing, Fourth Edition — Prentice Hall, 2006.
41. *Noonan Wes, Dubrawsky Ido*. Firewall Fundamentals — Cisco Press, 2006.
42. *Pieprzyk Josef, Hardjono Thomas, Seberry Jennifer*. Fundamentals of Computer Security — Springer, 2010.
43. *Cole Eric*. Network Security Bible, 2nd Edition — John Wiley & Sons, 2009.
44. *Kocher Paul, Jaffe Joshua, Jun Benjamin*. Introduction to Differential Power Analysis and Related Attacks, 1998.

# Ответы

## Ответы на вопросы к части I

1. Интернет можно охарактеризовать, например, следующим образом: глобальная публичная сеть с коммутацией пакетов, имеющая смешанную топологию, использующая стек протоколов TCP/IP, включающая магистральные сети, сети доступа и агрегирования трафика. Интернет состоит из сетей операторов связи и их клиентов.
2. Вычислительные ресурсы многотерминальных систем централизованы, а в компьютерной сети они распределены.
3. Иерархическая звезда.
4. Варианты а), б), в), г).
5. Варианты б), в), г).
6. Практически невозможно.
7. То, в котором фиксируется маршрут.
8. Варианты а), г), д), ж).
9. Невозможность динамического перераспределения пропускной способности физического канала между абонентами, возможность отказа в соединении, необходимость предварительного установления соединения, неэффективность использования пропускной способности при передаче пульсирующего трафика.
10. Наличие очередей и связанный с этим случайный характер задержек.
11. Варианты б), г).
12. Пропускная способность должна быть кратной элементарному каналу.
13. Варианты а), б), в), г).
14. Варианты а), б), г).
15. Сетевые службы в основном относятся к прикладному уровню. Модель OSI не включает пользовательские приложения.
16. Да, конечно; например, стек TCP/IP состоит из 4 уровней.
17. Нет, так как протокол не обязательно является стандартным.
18. Да.
19. Нельзя.
20. Стандартное отклонение — 17,2, коэффициент вариации — 0, 27.
21. Скорость потока может быть уменьшена путем отбрасывания пакетов.
22. Скорость передачи отдельного пакета равна пропускной способности линии — 100 Гбит/с.

23. Дискретизация по времени соответствует частоте квантования амплитуды звуковых колебаний 1/25 мкс, или 40 000 Гц. Для кодирования 1024 градаций звука требуется 10 двоичных разрядов. Отсюда необходимая пропускная способность для передачи оцифрованного таким образом голоса равна 40 000 × 10 = 400 Кбит/с.
24. Взвешенные очереди.
25. Да, может.
26. Вытеснение низкоприоритетного трафика.
27. При активной схеме измерений на результат могут повлиять измерительные пакеты, при пассивной схеме — недостаточная синхронизация.
28. Маршрут.
29. Буфер в пакетных сетях необходим всегда.
30. 5К, 20К и 75К бит выбирается из каждой очереди за один цикл.

# Ответы к части II

1. Да.
2. Вариант в).
3. –
4. Вариант в).
5. 193,5 ТГц.
6. Да.
7. Варианты а), в) и г).
8. Вариант б).
9. –
10. Нет.
11. Влияние электромагнитного поля, создаваемого передатчиками, на соседние провода кабеля, к которым подключены входы приемников.
12. С большим по абсолютной величине значением NEXT (которое всегда отрицательно).
13. Вариант б).
14. 258,5 Мбит/с.
15. Да.
16. 400 бит/с.
17. Нет.
18. Вариант в).
19. 620.
20. Вариант б).
21. –
22. Варианты а) и б).
23. Варианты б) и в).

24. Асинхронный.
25. 25 МГц и 75 МГц.
26. В 3 раза.
27. Варианты а) и в).
28. Варианты а) и в).
29. Вариант б).
30. Варианты а) и б).
31. Вариант б).
32. Вариант б).
33. 60
34. Вариант в).
35. –
36. Нет.
37. 7-й.
38. Отдельный указатель для каждого из трех контейнеров VC-3.
39. –
40. –
41. Вариант б).
42. Вариант б).
43. Вариант в).
44. Варианты а), б) и в).
45. –
46. Вариант в).
47. –
48. Да.
49. Вариант б).
50. Варианты а), в).
51. –
52. Кадры SDH помещаются в кадры OTN как поток байтов, границы кадров игнорируются.
53. Варианты а), в).
54. Варианты б), в).

# Ответы к части III

1. –
2. Верно.
3. Индивидуальный локальный.
4. Вариант а).

5. Вариант б).
6. Вариант а).
7. Вариант б).
8. Варианты б), в) и г).
9. –
10. Вариант б).
11. Да.
12. Варианты в) и г).
13. Варианты а), в) и г).
14. Назначив ему наибольшее значение идентификатора.
15. Нет.
16. Вариант а), в) и г).
17. deny f8:f2:1e:0c:3a:b8 ac:1f:6b:64:c7:d6.
18. permit any any.
19. Нет.
20. Назначить ему отличное от группы значение административного ключа.
21. Вариант а).
22. Удаляет поле тега.
23. Варианты б) и в).
24. Варианты а) и в).
25. Нет.
26. Вариант в).

# Ответы к части IV

1. Варианты а), д), и), к), н).
2. Номер подсети 108.5.18.160. Для нумерации интерфейсов в данной сети может быть использовано 4 бита, то есть 16 значений. Так как двоичные значения, состоящие из одних нулей и одних единиц, зарезервированы, то в сети не может быть более 14 узлов.
3. 64 подсети, маска /30.
4. Вариант г).
5. Между DNS-именами и IP-адресами в общем случае нет связи, поэтому ничего определенного сказать нельзя.
6. Вариант в).
7. Да, для двухточечных соединений стандарт разрешает использовать адреса интерфейсов 0 и 1.
8. Варианты а), б), в).
9. Варианты б), д).
10. Варианты а), б), в), г).

11. Нет.
12. Вариант г).
13. Вариант а).
14. Вариант б).
15. Поскольку ARP-таблица строится для каждого интерфейса, то число таблиц для каждого из этих устройств равно количеству их сетевых интерфейсов с назначенными IP-адресами.
16. 20 адресов (при условии, что в сети установлен DHCP-сервер).
17. Вариант а).
18. Для агрегирования трафика.
19. Варианты а), б), г).
20. Вариант в).
21. Их может быть несколько. Выбор осуществляется на основе метрики или приоритета, а также для балансировки нагрузки.
22. Вариант б).
23. Нет. Для правильной маршрутизации пакетов в сети с использованием масок достаточно того, что маски передаются протоколами маршрутизации RIP-2, OSPF или устанавливаются вручную для каждой записи таблицы маршрутизации.
24. Вариант а).
25. Вариант б).
26. Да, если оно для обеспечения надежности возьмет на себя функции, подобные функциям TCP, например квитирование, тайм-аут и т. п.
27. Объем полученных данных — 145 005 байт.
28. Вариант г).
29. Варианты а), в).
30. Варианты а), б), в), е).
31. Варианты а), в), г).
32. Варианты а), б), в), д).
33. Вариант в).
34. 00-80-48-EB-7E-60.
35. Варианты в), г).
36. Варианты а), б), г), д).

# Ответы к части V

1. Варианты б) и в).
2. Вариант б).
3. –
4. –

5. 240 мс.
6. 1,124875 с.
7. –
8. Варианты а) и б).
9. Потому что активные компоненты сети дороже пассивных.
10. WDM.
11. С центральным арбитром.
12. –
13. Нет.
14. Вариант а).
15. Да.
16. Вариант в).
17. Вариант б) и в).
18. Вариант в).
19. Вариант б).
20. Вариант в).
21. Варианты б) и в).
22. Вариант б).

# Ответы к части VI

1. Да.
2. Видимого света.
3. Эффекта дифракции.
4. За счет распределения энергии полезного сигнала в широком диапазоне частот, так что узкополосные помехи не оказывают существенного влияния на сигнал в целом.
5. Как отношение выходной мощности, излучаемой данной антенной в определенном направлении, к выходной мощности, излучаемой идеальной изотропной антенной в любом направлении, при условии, что на вход обеих антенн поступают равные по мощности сигналы.
6. $\lambda/2$.
7. –
8. Быстрого.
9. За счет ортогональности кодов расширения, назначенных каналам.
10. 20 кГц.
11. Данные передаются медленнее, так как станция откладывает передачу кадра, считая, что среда занята, хотя она свободна.
12. Если подтверждение в получении отправленного кадра не пришло за заданное время.
13. Может.

14. 0.
15. За счет отсчета времени с момента окончания передачи кадра.
16. За счет более короткого межкадрового интервала.
17. Увеличение расстояния достигается за счет уменьшения битовой скорости передачи данных в два раза, до 1 Мбит/с, и применения кодов FEC.
18. Класса 1.
19. –
20. Уменьшить размер соты.
21. Потому что локализация телефона происходит с точностью до области локализации, в которую входит несколько сот.
22. Из-за небольших флуктуаций мощности сигнала телефон может слишком часто переходить из соты в соту — эффект пинг-понга.
23. Потому что номер телефона не хранится в SIM-карте, а только в домашней базе провайдера.
24. 160 Кбит/с.
25. Два ресурсных блока.
26. Для предотвращения перестройки таблиц маршрутизации в сети провайдера при передаче телефона между зонами действия разных шлюзов S-GW.
27. SDN, NFV, технология виртуальных машин.
28. Преимущества: более высокая скорость передачи данных за счет более широкой полосы частотных подканалов. Недостатки: меньший размер соты.

# Ответы к части VII

1.

| | |
|---|---|
| Используется почтовым клиентом для передачи письма на сервер | SMTP |
| Используется почтовым клиентом для получения письма с сервера | POP3, IMAP |
| При получении почты письмо перемещается с сервера на клиент | POP3 |
| При получении почты письмо копируется с сервера на клиент | IMAP |

2.

| | |
|---|---|
| Путь к объекту | /mobile/web/versions.shtml |
| DNS-имя сервера | www.bbc.co.uk |
| URL-имя | http://www.bbc.co.uk/mobile/web/versions.shtml |
| Тип протокола доступа | http:// |

3. Варианты б), в), е).
4. Варианты а), в), г).
5. Варианты а), г).

6. Вариант б).
7. Вариант б).
8. Варианты а), д).

# Ответы к части VIII

1.

| | **Объекты** | **Субъекты** |
|---|---|---|
| Пользователи | | + |
| Устройства | + | |
| Прикладные процессы | + | + |
| Файлы | + | |
| Пропускная способность каналов связи | + | |
| Сетевые сервисы | + | |

2. «Пользователи получают права доступа к ресурсам ИС в результате авторизации, однако прежде им необходимо успешно пройти идентификацию и аутентификацию».
3. «Стало известно, что в Интернете уже появились эксплойты, направленные на использование уязвимости новой версии браузера. Реализация данной угрозы может привести к атаке, которая нанесет ущерб нашему предприятию».
4. Передача зашифрованной информации по открытому каналу.
5. Аутентичность.
6. 1225.
7. Вариант б).
8. Да, с помощью алгоритма Диффи—Хеллмана.
9. В зависимости от решаемых задач некоторые зоны могут быть объединены, а другие — разделены.
10. Варианты а), д), з).
11. Варианты б), г), д).
12. Варианты а), б), в).
13. Все варианты — а), б), в), г).
14. Варианты а), б), г), д), ж).
15. Скрывает внутреннюю структуру сети.
16. «Сертификат может быть представлен в форме, состоящей из трех частей: во-первых, открытой части; во-вторых, той же информации, зашифрованной закрытым ключом сертифицирующей организации; в-третьих, части, представляющей собой первые две части, зашифрованные закрытым ключом владельца».

17. Ущерб — impact; уязвимость — vulnerability; вредоносное ПО — malware; доступность — availability; целостность — integrity; подмена содержимого пакета — spoofing; распределенная атака DDoS, отказ в обслуживании — Denial of Service.
18. Варианты б), г).
19. Варианты а), в).
20. 100, 50 открытых и 50 закрытых ключей.
21. Варианты б), г).
22. Вариант в).
23. Варианты а), в), г).
24. Варианты а), в), г), д).
25. Варианты а), б), е).
26. Сначала а), затем в).
27. Вариант в).
28. Варианты а), с).
29. Дискреционный — а), б), в), г), е), мандатный — д), ж).
30. Варианты в), г).
31. access-list 120 deny ICMP any 145.8.146.0 0.0.0.255 eq 5.
32. Разрешить прохождение через маршрутизатор пакетов хоста 93.8.25.2, запрещая передачу пакетов, отправляемых любым другим хостом подсети 93.8.25.0/24.
33. Варианты а), в), г).
34. Варианты в), г).
35. Варианты а), б).
36. Да.
37. Варианты б), в), г), д).
38. Варианты а), в), г).
39. Варианты в), г).
40. Варианты а), б), в), г).
41. Вариант б).
42. Вариант в).
43. Варианты а), в).
44. Вариант б).
45. Вариант а).
46. Нет.
47. Вариант б).
48. Варианты а), б), в).
49. Варианты а), б), в).

# Алфавитный указатель

## J



## L



## M



## N



## O

## P



## Q



## R



## S

## Б



## В

## Г

## Д

## Е



## З



## И

## К

## Л

## M

## Н

## О

## П

## Р

## С

## Т

## У

## Ф



## Х

## Ц



## Ч



## Ш



## Э



## Я

*Виктор Олифер, Наталья Олифер*

**Компьютерные сети. Принципы, технологии, протоколы: Юбилейное издание**


| | |
|---|---|
| Заведующая редакцией | *Ю. Сергиенко* |
| Ведущий редактор | *К. Тульцева* |
| Литературный редактор | *М. Рогожин* |
| Художественный редактор | *А. Михеева* |
| Корректоры | *С. Беляева, М. Молчанова, Н. Сидорова, Г. Шкатова* |
| Верстка | *Л. Егорова* |

Изготовлено в России. Изготовитель: ООО «Прогресс книга».
Место нахождения и фактический адрес: 194044, Россия, г. Санкт-Петербург,
Б. Сампсониевский пр., д. 29А, пом. 52. Тел.: +78127037373.

Дата изготовления: 09.2020. Наименование: книжная продукция. Срок годности: не ограничен.

Налоговая льгота — общероссийский классификатор продукции ОК 034-2014, 58.11.12.000 —
Книги печатные профессиональные, технические и научные.

Импортер в Беларусь: ООО «ПИТЕР М», 220020, РБ, г. Минск, ул. Тимирязева, д. 121/3, к. 214, тел./факс: 208 80 01.

Подписано в печать 08.09.20. Формат 70×100/16. Бумага писчая. Усл. п. л. 81,270. Доп. тираж 4000. Заказ 0000.