Обнаружение нарушений безопасности в сетях

Третье издание

Стивен Норткат
Джуди Новак

ВИЛЬЯМС
New Riders

Network Intrusion Detection

Third Edition

Stephen Northcutt
Judy Novak

201 West 103rd Street,
Indianapolis, Indiana 46290

Обнаружение нарушений безопасности в сетях

Третье издание

Стивен Норткат
Джуди Новак

Москва • Санкт-Петербург • Киев
2003

ББК 32.973.26-018.2.75
Н83
УДК 681.3.07

Издательский дом "Вильямс"
Зав. редакцией *С.Н. Тригуб*

Перевод с английского и редакция *В.С. Иващенко*

По общим вопросам обращайтесь в Издательский дом "Вильямс" по адресу:
info@williamspublishing.com, http://www.williamspublishing.com

Норткат, Стивен, **Новак**, Джуди.

Н83 Обнаружение нарушений безопасности в сетях, 3-е издание. : Пер. с англ. — М. : Издательский дом "Вильямс", 2003. — 448 с. : ил. — Парал. тит. англ.
ISBN 5-8459-0526-5 (рус.)

В этой книге описываются современные аппаратные и программные средства противодействия атакам хакеров, а также методы грамотного применения этих средств. Исключительно простое и наглядное изложение необходимых теоретических вопросов, с одновременным закреплением на реальных примерах, позволяет даже неподготовленному читателю организовать надежную защиту своей локальной сети или отдельного ее узла. При этом, в отличии от многих подобных изданий, здесь описывается полная последовательность действий, которые должны осуществляться при возникновении нестандартных ситуаций.

Книга рассчитана на системных администраторов, аналитиков сетевого трафика, а также всех желающих понять принципы безопасной работы в сети.

ББК 32.973.26-018.2.75

ISBN 5-8459-0526-5 (рус.)
ISBN 0-7357-1265-4 (англ.)

Оглавление

Содержание

Посвящается

светлой памяти доктора Ричарда Стивенса
(Richard Stevens)

Я и сейчас отчетливо помню его во время перерыва в комнате докладчиков на конференциях SANS — собранные сзади длинные светлые волосы, проницательный взгляд, которым он смотрел на всех своих студентов. Я помню все его комментарии на проверенных заданиях. Ричард Стивенс был лучшим преподавателем. Сейчас его уже нет с нами, но не проходит и нескольких дней, чтобы я не обратился к его *TCP/IP Illustrated, Volume 1.* Как правило, я смотрю на схемы заголовков пакетов, изображенные на первой странице обложки. Я очень рад, что у меня есть эта книга, она помогает мне не забывать механизм работы протоколов IP и TCP — тех самых протоколов обмена информацией, которые сейчас царствуют в мире компьютеров. Через несколько недель я начинаю преподавать основы TCP/IP приблизительно четырем сотням студентов. Мне так страшно, что я не смогу заменить Ричарда или хотя бы приблизиться к его уровню, но процесс передачи знаний не должен прерываться. У меня не хватает слов, чтобы донести всю важность проделанной им работы. В конце концов не существует универсального средства для обнаружения несанкционированного доступа, но благодаря доктору Стивенсу решение этой задачи стало намного проще. Кроме того, каждый специалист по безопасности работы в компьютерных сетях должен иметь базовые знания о работе протокола IP, чтобы вовремя определить нестандартную ситуацию. Именно этими способностями обладал доктор Стивенс, все мы учились у него, и эти уроки положены в основу материала книги *Обнаружение типовых нарушений безопасности в сетях*!

Стивен Норткат (Stephen Northcutt)

Весь наш вклад в дело обеспечения безопасности и в разработку методов анализа трафика нельзя сравнить с тем, что удалось сделать в этой области доктору Ричарду Стивенсу. Он был талантливым и очень плодовитым автором. Моей настольной книгой с потрепанными, замусоленными страницами и жирными пятнами является экземпляр *TCP/IP Illustrated, Volume 1.* Это просто шедевр для любой технической библиотеки, так как Ричард Стивенс стоял у истоков разработки TCP/IP и Unix, и к тому же он обладал редким даром делать сложный материал легко понятным. Я знакома с несколькими преподавателями SANS, которые считают эту книгу "Библией TCP/IP". Однажды мне повезло прослушать курс лекций, которые он читал в SANS, и я сидела, разинув рот, потрясенная глубиной его познаний. Прошлым летом он согласился отредактировать для издания курс лекций, которые я написала для SANS по основам работы стека протоколов TCP/IP. Результат его работы достоин сравнения с критическим просмотром Шекспиром какой-нибудь бухгалтерской книги. Это издание всегда со мной, и мне никогда не забыть этого человека.

Джуди Новак (Judy Novak)

Об авторах

Стивен Норткат (Stephen Northcutt) закончил колледж имени Мэри Вашингтон. До того как приступить к работе в области обеспечения безопасности работы в компьютерных сетях, он служил в вертолетном подразделении ВМС США и был одновременно спасателем, шеф-поваром, картографом, тренером по боевым искусствам и проектировщиком компьютерных сетей. Стивен является автором книг *Incident Handling Step by Step*, *Intrusion Signatures and Analysis*, *Inside Network Perimeter Security* и двух предыдущих редакций этой книги. Стивен Норткат – автор системы обнаружения вторжений Shadow. Он занимал пост начальника отдела защиты от несанкционированного доступа в сетях с помощью системы Shadow, а затем главы ведомства информационной безопасности департамента защиты от нанесения удара баллистическими ракетами Министерства обороны США. В настоящее время Стивен Норткат работает руководителем отдела обучения и аттестации института SANS.

Джуди Новак (Judy Novak) – старший специалист по вопросам безопасности балтиморской консалтинговой фирмы Jacob and Sundstrom, Inc. До этого она работала в лаборатории прикладной физики Университета Джона Хопкинса (Johns Hopkins University) и занималась обнаружением вторжений и отслеживанием трафика, а также исследованиями в области безопасности обмена информацией. Джуди является одним из основателей группы по реагированию на происшествия исследовательских лабораторий вооруженных сил США, в которой она состоит на протяжении трех лет. Она принимала участие в создании курса по TCP/IP и написала учебник по практическому курсу "Анализ сетевого трафика с помощью tcpdump". Обе эти книги используются на курсах сертификации института SANS. Выпускница университета штата Мэриленд, Джуди Новак – опытный специалист, она энергична, страстная велосипедистка, а героем современности считает Ланса Армстронга!

О технических рецензентах

Рецензенты внесли свою немалую лепту в создание книги *Обнаружение нарушений безопасности в сетях, 3-е издание.* Эти высококлассные профессионалы прошгудировали весь изложенный материал, проверяя его как с точки зрения технической правильности, так и со стороны удобства изложения и литературной грамотности. Их вклад, без сомнения, способствовал тому, чтобы данная книга стала качественным источником информации для наших читателей.

Карен Кент Фредерик (Karen Kent Frederick) занимает должность ведущего инженера по безопасности в группе быстрого реагирования проекта NFR. Степень магистра наук в области информационных технологий со специализацией в области компьютерной безопасности она получила в университете штата Айдахо по программе переподготовки инженеров. Карен уже 10 лет занимается технической поддержкой, системным администрированием и вопросами компьютерной безопасности. Ее квалификация подтверждена рядом сертификатов, среди которых SANS GSEC, GCIA, GCUX и GCIH. Карен Фредерик – соавтор книг *Анализ типовых нарушений безопасности в сетях (Intrusion Signatures and Analysis), Inside Network Perimeter Security.* Ее многочисленные статьи по теме обнаружения вторжений можно найти на сайте `SecurityFocus.com`.

Дэвид Хэйнбач (David Heinbuch) был зачислен в лабораторию прикладной физики университета Джона Хопкинса в 1998 году. Он имеет большой опыт в обнаружении несанкционированного доступа, моделировании и выполнении экспериментов, исследовании уязвимости компьютерных систем и разработке программного обеспечения. Являясь участником группы Information Operations, он работал с разнообразными программами, в том числе с системами выполнения безопасных вычислений, а также принимал участие в имитации, выявлении и анализе атак хакеров. Дэвид получил степень бакалавра в области информационных технологий в колледже штата Виржиния и степень магистра – в университете Джона Хопкинса.

Введение

Книга *Обнаружение нарушений безопасности в сетях, 3-е издание* призвана повысить уровень знаний специалистов в области анализа нарушений безопасности в сетях. Мы считаем, что, прочитав эту книгу "от корки до корки", параллельно закрепляя изложенный материал на практике, наши читатели смогут смело работать аналитиками в области выявления признаков несанкционированного доступа.

Многие люди уже прочли наши книги или прослушали курсы института SANS. Они многому научились и теперь готовы к борьбе со злоумышленниками. В книге мы раскрыли технические подробности работы стека протоколов TCP/IP и сделали все возможное для облегчения усвоения информации, предоставив десятки наглядных примеров.

Эта книга состоит из пяти частей. В первой части под названием "TCP/IP" представлены сведения, начиная с фундаментальных основ работы протоколов Internet и заканчивая обсуждением служб RPC (Remote Procedure Call — удаленный вызов процедур). Мы понимаем, что уже стало модным начинать книгу с информации по TCP/IP, но разработанная нами с Джуди программа обучения не только знакомит с протоколом IP, но и дает значительно больше сведений об этом протоколе — больше, чем любая другая существующая программа. Мы называем свою программу "истинным TCP/IP" потому, что излагаемый материал содержит реальную информацию о том, как именно пакеты передаются по сети, а не только одну сухую теорию. Даже если вы хорошо знакомы с работой протокола IP, все равно не помешает просмотреть первую часть книги. Мы уверены, что вы будете приятно удивлены. Возможно, самой важной главой первой части является глава 5, "Воздействия и реакции". При анализе сетевого трафика в первую очередь следует определить, является он исходным воздействием или же ответной реакцией. Это поможет принять верное решение. Не пожалейте времени на внимательное изучение представленного материала, и в дальнейшем вы сможете избежать многих опасных ошибок.

Совет

При изучении сетевого трафика прежде всего выясните, что он собой представляет — исходное воздействие или ответную реакцию.

Во второй части книги, "Анализ трафика", мы рассмотрим содержание полей заголовков протокола IP и протоколов более высоких уровней. Хотя шестнадцатеричные и ASCII-сигнатуры являются неотъемлемой частью выявления типовых нарушений безопасности, но это только инструменты из набора опытного аналитика сетевого трафика. Мы расскажем о значении каждого поля, о том, какую информацию они содержат как об отправителе, так и об адресате передаваемых данных. На последних страницах этой части книги наши читатели смогут ознакомиться с реальными примерами выявления необычного трафика. Эти истории помогут подготовиться к тому моменту, когда вы сами столкнетесь с нестандартной ситуацией.

Хотя бывают случаи, когда передаваемые данные полностью подходят под определенный шаблон, и понять происходящее не составляет труда, чаще прихо-

дится тщательно исследовать следы преступления. Написанное нарушителями программное обеспечение для атак отказа в обслуживании или других атак, как правило, оставляет характерные признаки, которые являются результатом получения созданного хакером вредоносного пакета. Это напоминает стандартный метод определения конкретного оружия по следам на пуле, появившимся после ее прохождения через ствол пистолета. В третьей части книги, "Фильтры и правила для контроля сетевого трафика", наши читатели научатся исследовать каждое поле в пакете и определять, что является нормальным, а что несет угрозу. Для этого мы будем использовать анализаторы пакетов TCPdump и Snort.

В четвертой части мы подробно изучим стандартные методы обнаружения типовых нарушений безопасности в сетях. В том числе будут рассмотрены вопросы выбора места для установки средств обнаружения атаки, необходимых для вывода информации о передаваемом трафике на монитор, а также описаны автоматизированные и выполняемые вручную действия в ответ на атаку хакера. Этот раздел также поможет узнать, какие преимущества несет создание системы предотвращения несанкционированного доступа в схему работы коммерческой организации и как реализовать эту систему.

И, наконец, завершают книгу три приложения, в которых описаны стандартные признаки деятельности хакеров на примерах атак отказа в обслуживании, других известных программ атаки, а также методы сканирования систем для предварительного определения возможности проведения этих атак. Мы надеемся, наши читатели не станут возражать против того, что в нашей книге нет бесполезных мест, которые можно было бы пропустить.

Авторов книги *Обнаружение нарушений безопасности в сетях, 3-е издание* нельзя назвать профессиональными творцами технической литературы. Джуди и я просто работали в области обеспечения безопасности компьютерных систем с 1996 года и лично устраняли многие ранее неизвестные проблемы. Мы благодарны за возможность поделиться накопленным опытом и надеемся, что для наших читателей эта книга станет надежным проводником в мире анализа и обнаружения нарушений безопасности в сетях.

I

TCP/IP

В этой части...

1

Фундамент Internet

Совершенно очевидно, что открывший эту книгу читатель принадлежит к одной из двух категорий людей – новичков или ветеранов изучения основ стратегии безопасной работы в сетях. Тема протокола IP (Internet Protocol – протокол Internet) чрезвычайно обширна и может отпугнуть начинающих, если не представить ее простым, понятным языком, постепенно объясняя неизвестные аббревиатуры, понятия и принципы работы. Поэтому основной целью первой главы является именно такое представление нового (для кого-то) материала. Для рассматриваемого здесь набора протоколов чаще используют аббревиатуру *TCP/IP* (Transmission Control Protocol/Internet Protocol – протокол управления передачей/протокол Internet). С помощью этого набора протоколов осуществляется взаимодействие между компьютерами по глобальной сети Internet. Существуют и другие протоколы для обмена данными по сети, например, протокол AppleTalk для компьютеров Apple. Как правило, такие протоколы применяются в корпоративных локальных сетях (intranet-сеть). Большинство соединений в Internet осуществляется с использованием стека протоколов TCP/IP, который признан стандартом для взаимодействия между компьютерами в глобальной сети.

У опытных специалистов, которые ежедневно работают с TCP/IP, может возникнуть желание пропустить эту главу, но мы все же рекомендуем хотя бы пролистать ее страницы. Тот, кому приходилось объяснять принципы работы IP (может быть, ваш непосредственный начальник), несомненно, оценит наш подход к изложению материала. Тот же, кто не уверен, в полноте своих знаний, наверняка, почерпнут много полезных сведений из этой вступительной главы.

Здесь, в одной главе, сконцентрирована общая информация о TCP/IP. Многие затронутые темы будут подробнее раскрыты в следующих главах, но сначала следует получить теоретическую базу для их уверенного восприятия. Ниже перечислены основные темы первой главы.

- **Базовая концепция TCP/IP**. Рассмотрены основы взаимодействия компьютеров по Internet с помощью стека протоколов TCP/IP.
- **Упаковка данных для их передачи по Internet**. В этом разделе изучаются методы инкапсуляции данных для их пересылки получателям, находящимся *в различных сегментах* сети.
- **Физические и логические адреса**. Освещена тема идентификации компьютера или хоста, подключенного к Internet.
- **Служба DNS (Domain Name System — система доменных имен)**. Основное внимание уделяется важности прямого и обратного преобразования символьных имен компьютеров и их IP-адресов.
- **Маршрутизация**. Изучаются методы выбора маршрута между двумя компьютерами при обмене данными по сети.

Модель TCP/IP

Существует множество мотивов, по которым пользователи очень часто получают информацию через Internet (взять хотя бы просмотр Web-страниц с удаленного Web-сервера). При этом кажется, что получение данных происходит практически мгновенно, но в процессе доставки информации незаметно для пользователя участвует множество устройств и выполняется масса отдельных задач.

Уровни

На рис. 1.1 изображена логическая схема взаимодействия двух удаленных компьютеров согласно модели TCP/IP. Итак, допустим, что нам нужно загрузить Web-страницу на компьютер, которому соответствует элемент блок-схемы, обозначенный как Web-броузер (см. рис. 1.1). Прежде чем отправить запрос на получение данных Web-страницы Web-серверу, на стороне отправителя этот запрос следует упаковать. Данные передаются на самый низкий уровень стека, проходя упаковку на каждом из промежуточных уровней. При этом на каждом уровне к сообщению добавляется определенная информация. Затем полученный пакет пересылается по Internet. На стороне компьютера-адресата сообщение распаковывается в обратном порядке, проходя через все уровни к самому верхнему. Каждый уровень использует предназначенную для него информацию. Оставшаяся часть сообщения передается на более высокий уровень вплоть до уровня приложений (элемент схемы, обозначенный как Web-сервер).

Кратко рассмотрим каждый из уровней модели TCP/IP.

Уровень приложений (application layer) является высшим уровнем модели TCP/IP. На этом уровне реализуется доступ приложений (в нашем примере Web-броузера и Web-сервера) к компьютерной сети.

Транспортный уровень (transport layer) расположен ниже уровня приложений. На этом уровне устанавливаются многие параметры взаимодействия двух компьютеров и обеспечивается надежная работа других, по своей сути ненадежных, уровней. Данный уровень служит посредником между уровнем приложений и нижними уровнями, ориентированными на передачу данных по сети.

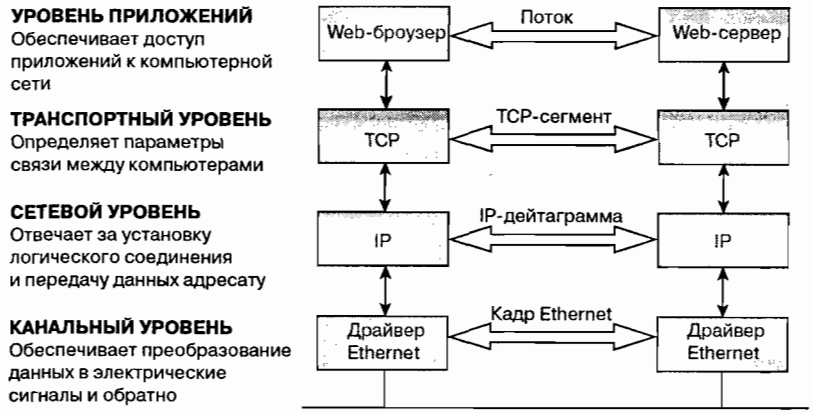


Рис. 1.1. Модель стека протоколов TCP/IP

Мы рассмотрим два протокола транспортного уровня: TCP, который гарантирует надежную доставку сообщений, и UDP (User Datagram Protocol – протокол доставки пользовательских дейтаграмм), который такой надежной доставки не гарантирует. В нашем примере требовалось использование TCP, так как потеря данных недопустима.

Сетевой уровень (network layer) отвечает за пересылку данных с одного компьютера на другой (в нашем случае запрос пересылается на Web-сервер) нередко только через один транзитный участок или переход (hop). *Переходом* мы будем называть участок сети между компьютером и маршрутизатором или между двумя маршрутизаторами на пути доставки пакета к адресату.

Канальный уровень (link layer) является самым низким в иерархии стека TCP/IP. На этом уровне обеспечивается взаимодействие с физической средой передачи данных. В нашем случае двоичные данные преобразовываются в электрические сигналы, так как физической средой передачи является Ethernet. Для получения и отправки данных используется определенный интерфейс.

Обмен данными

Еще раз обратимся к рис. 1.1. Теоретически процесс передачи данных описывается следующим образом. Запрос на получение Web-страницы проходит через уровни компьютера-отправителя (которые часто называют стеком TCP/IP) "сверху вниз". Сообщение направляется компьютеру-адресату, где оно проходит обратное преобразование по стеку TCP/IP "снизу вверх". На рис. 1.1 вертикальные стрелки между уровнями обозначают поток данных на локальном компьютере. Горизонтальные стрелки указывают на то, что каждый уровень передает определенную информацию (упаковывает сообщение) соответствующему уровню на удаленном компьютере. Несмотря на то что два компьютера не взаимодействуют непосредственно между собой, применение стека TCP/IP создает у пользователя такое впечатление.

Эта концепция крайне важна для правильного понимания материала этой главы и всей модели TCP/IP. Поэтому повторим основные моменты и закрепим терминологию. Термин *стек TCP/IP* используется для описания многоуровневой

модели обработки запросов и ответов. *Инкапсуляцией* называется упаковка сообщения одного протокола в сообщение другого и добавление к нему определенной информации (например, идентифицирующих заголовков), предназначенной для соответствующего уровня на удаленном компьютере. Каждый уровень на компьютере-отправителе добавляет к сообщению собственный заголовок, и на компьютере-получателе в первую очередь учитывается заголовок пакета для соответствующего уровня. Полученное сообщение проходит обратный процесс распаковки с удалением заголовков на каждом из уровней до тех пор, пока, наконец, не будет достигнут самый высокий уровень. При ответе на запрос процесс повторяется в обратном порядке, начиная с упаковки ответного сообщения на хосте Web-сервера и заканчивая его распаковкой и передачей уровню приложений, поддерживающему Web-броузер на компьютере-получателе.

Упаковка

Данные, обмен которыми осуществляется между двумя хостами, должны быть сохранены в каком-то формате, стандартном для каждого уровня стека TCP/IP. *Хост* (host) — это общий термин, которым можно назвать рабочую станцию, маршрутизатор, Web-сервер и т.д. Общей особенностью хостов является наличие соединения с сетью, по которой можно обмениваться данными. В общем случае все упакованные данные называются *пакетом* (package). Проблемы с терминологией возникают из-за того, что на каждом уровне стека TCP/IP при взаимодействии двух удаленных приложений под этим пакетом понимается различная информация (учитывая добавленные служебные заголовки). В этом разделе мы рассмотрим базовые понятия, касающиеся упаковки данных, а именно: бит, байт, пакет, инкапсуляция и интерпретация данных.

Биты, байты и пакеты

Наименьшей единицей информации принято считать *бит*. Значением бита может быть 0 или 1, поэтому бит часто называют двоичной цифрой (binary). Ясно, что в одном бите нельзя передать достаточный объем информации, поэтому их группируют по восемь. Восемь битов составляют один *байт*. Минимальный объем информации, пусть даже увеличенный в восемь раз, все равно остается недостаточно большим, но один байт способен хранить значение стандартного символа ASCII, например буквы или знака препинания, или целого числа до 255 (2^8–1).

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	1	1	1	1	1	1
128	64	32	16	8	4	2	1

Рис. 1.2. Схема байта

На рис. 1.2 показана схема байта. Нас больше всего интересуют биты, для которых используется двоичный код — набор нулей и единиц. Каждый бит можно представить некоторой степенью основы двоичной системы счисления — числа 2. Значение байта составляет диапазон степеней числа 2 от 2^0 до 2^7. Это пояснить довольно просто: если значением всех битов одного байта является 0, то и значение байта равно 0, если же значением всех битов

одного байта является 1, то, сложив все значения степеней битов, начиная с наименьшего (2^0=1), получим 1+2+4+8+16+32+64+128=255 — максимальное значение одного байта. Проанализируем смысл этого значения позже, при обсуждении IP-адресов.

Только что мы выполнили преобразование двоичного значения в десятичное. Для преобразования байта данных из двоичного вида в десятичный достаточно представить его в виде степеней числа 2 и простым сложением полученных значений каждого бита получить искомое десятичное значение. Вот и весь секрет. Это не так сложно, как запуск ракеты в космос.

Для передачи по сети несколько байтов объединяются в один пакет. На рис. 1.3 показана истинная ситуация при передаче данных по сети – передача любого количества полезных данных обеспечивается за счет добавления определенного объема служебной информации. Требуются некоторые действия для упаковки данных перед отправкой по сети и их последующей распаковки на стороне адресата (и, конечно, для подтверждения достоверности сообщения). Для проверки целостности переданного пакета предназначено специальное поле CRC (Cyclic Redundancy Check – циклическая проверка четности с избыточностью), значение которого часто называют *контрольной суммой*.

Заголовок содержит информацию об адресате и отправителе, а также о типе передаваемой информации, что напоминает обычный почтовый конверт

Рис. 1.3. Пакет данных

Как и почтовый конверт, IP-пакет должен нести информацию об адресате и отправителе (см. рис. 1.3). В сетях, по крайней мере в сетях Ethernet, аналогом домашнего адреса можно считать MAC-адрес (Media Access Controller – контроллер доступа к среде) вашего сетевого адаптера. Этот аппаратный адрес присваивается производителем сетевого оборудования. MAC-адрес представляет собой 48-битовое число, т.е. может быть достаточно большим (2^{48}–1). О том, чем различаются IP- и MAC-адреса, рассказано в разделе "Адреса" этой главы.

Для создания *кадра* (frame) к нему должна быть добавлена информация заголовков каждого из уровней TCP/IP. В последнюю очередь к кадру добавляется информация физического уровня, и он передается в линию связи с помощью сетевого адаптера (NIC – network interface card). Заголовок кадра имеет размер 14 байт и содержит поля для хранения MAC-адресов отправителя и адресата, служебную информацию кадра (ее размер может изменяться) и 4-байтовую завершающую часть (окончание) для передачи кода CRC.

Инкапсуляция

Рассмотрим схему многоуровневой упаковки сообщения (рис. 1.4). Теоретически различные уровни стека протоколов одного компьютера "обращаются" к соответствующим уровням на другом компьютере. Уровни расположены один над другим, что позволяет говорить о "стеке протоколов TCP/IP". На каждом уровне пакет состоит из

собственного заголовка и самих данных, которые часто называют *полезной нагрузкой* (payload). Смысл всего процесса инкапсуляции заключается в передаче на другой компьютер какой-то информации, но на пути к адресату на каждом из уровней заголовки добавляются к уже существующей информации. Заголовки вышележащего уровня считаются данными для более низкого уровня.

При прохождении пакета вниз по стеку на каждом уровне добавляются свои заголовки

Рис. 1.4. Добавление заголовков

Предположим, что нам нужно послать сообщение или передать какую-либо информацию на удаленный компьютер. Сначала эта информация формируется в единое целое с помощью программы типа `telnet` или программы для работы с электронной почтой (более подробно эти программы рассмотрены ниже, в разделе "Протоколы стека TCP/IP"). TCP-пакет, который называют TCP-сегментом (TCP segment), состоит из TCP-заголовка и данных. UDP-пакет называют дейтаграммой, что часто проводит к путанице, так как дейтаграммой называется и пакет, формирующийся на сетевом уровне (протокол IP).

TCP-сегмент передается вниз по стеку протоколов на уровень протокола IP. На сетевом уровне в начало TCP-сегмента добавляется информация заголовка IP, и, таким образом, формируется IP-дейтаграмма. В действительности и заголовок, и данные TCP на уровне протокола IP рассматриваются как единый блок данных. IP-дейтаграмма передается на канальный уровень стека TCP/IP, где превращается в кадр – в начало IP-дейтаграммы добавляется заголовок кадра, содержащий служебную информацию для доставки этого кадра по физической среде передачи, например, по сети Ethernet.

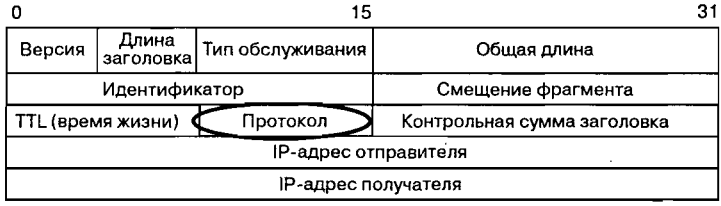
Когда пакет достигает компьютера-адресата, весь описанный процесс повторяется с точностью до наоборот – при прохождении сообщения снизу вверх по стеку TCP/IP на каждом из уровней удаляется соответствующий заголовок. С помощью заголовков осуществляется обмен информацией между аналогичными уровнями стека TCP/IP взаимодействующих компьютеров.

Интерпретация полученных данных

При прохождении сообщения по стеку протоколов его информация представляет собой набор нулей и единиц. Как же понять, что скрывается за этой последовательностью? Представим себе, например, заголовок IP-дейтаграммы. Как узнать, какой протокол был использован на более высоком уровне? Безусловно, эти сведения позволяет понять принципы работы протокола. В данном случае под термином *протокол* понимается набор согласованных правил или форматов об-

мена сообщениями. В каждом протоколе (например, IP, TCP, UDP или ICMP) используется собственный вариант схемы обмена данными и соответствующий формат этих данных.

Рассмотрим заголовок IP-дейтаграммы (рис. 1.5). Для каждого поля заголовка зарезервировано определенное количество битов. В поле "Протокол" сохраняется информация о протоколе более высокого уровня. Каждая строка IP-заголовка содержит 32 бит (с 0 по 31 включительно), что равняется четырем байтам. Счет с нуля несколько усложняет задачу определения места нужного байта или бита, но к этому придется привыкнуть. В первой строке отображены байты с 0 по 3, во второй — с 4 по 7, а в третьей — с 8 по 11. Обратите внимание на то, что выделенное поле "Протокол" находится в третьей строке. Длина предшествующего поля TTL составляет 1 байт. Этот байт является восьмым. Следовательно, поле "Протокол", длина которого тоже 1 байт, является 9-м байтом. Это значит, что для определения протокола, использованного на более высоком уровне, необходимо исследовать этот 9-й (по сути, 10-й, ведь счет начинается с 0) байт. То есть весь секрет в том, чтобы знать, в каком месте пакета определенного уровня следует искать необходимую информацию; расположение конкретного поля можно найти по известному смещению.



Заголовок IP-дейтаграммы без параметров общей длиной 20 байт

Рис. 1.5. Поля IP-дейтаграммы

Итак, мы определили место расположения поля "Протокол". Что же оно собой представляет и для каких целей служит? Значение этого поля указывает на то, какой протокол использовался для инкапсуляции данных на более высоком уровне. Предположим, что значение байта этого поля равно 17. На самом деле в поле хранится шестнадцатеричное значение, например 11, которое соответствует десятичному значению 17. Это означает, что на транспортном уровне был использован протокол UDP. Значение 6 свидетельствует, что встроенным пакетом является TCP-пакет, а значение 1 соответствует ICMP-пакету.

Шестнадцатеричная система счисления

Основой двоичной системы счисления является число 2, а двоичный код состоит из нулей и единиц. Именно двоичный код используется при работе всех компьютеров. Так зачем же нужно создавать дополнительные сложности и вводить новую систему счисления, основой которой является число 16? Проблема заключается в том, что при использовании двоичной системы счисления для записи большого числа приходится использовать значительное количество битов, и числа быстро становятся слишком громоздкими. Шестнадцатеричная система позволяет представить двоичные числа в более кратком виде. Один шестнадцатеричный символ позволяет заменить четыре двоичных разряда (2^4=16).

Рассмотрим, например, поле заголовка протокола IP размером в 8 бит. Его значение можно преобразовать в два шестнадцатеричных символа. Как уже указывалось, двоичное число 17 соответствует использованному протоколу UDP. Как же получить из десятичного 17 шестнадцатеричное 11?

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	0	1	0	0	0	1

Здесь показаны степени числа 2, составляющие десятичное число 17 (для получения этого числа нужно сложить $2^4=16$ и $2^0=1$). Эти биты разделены на два шестнадцатеричных числа, по 4 бит каждый. Крайние слева четыре бита (или шестнадцатеричный символ), которые называют еще битами старшего разряда (most significant), имеют значение 0001. Аналогично четыре крайних справа бита, называемых битами младшего разряда (least significant), тоже имеют значение 0001. Каждый шестнадцатеричный символ может принимать значения от 0 до 15. Каждый значащий разряд шестнадцатеричного числа в 16 раз больше предыдущего разряда. Поэтому при делении десятичного числа (17) на 16 число, получившееся в остатке (1), является младшей шестнадцатеричной цифрой. Продолжим деление, записывая остаток слева от первой цифры, и получим шестнадцатеричное число 11. Для того чтобы шестнадцатеричные значения можно было отличить от десятичных, к ним обычно добавляют приставку 0x, например 0x11.

Адреса

Скорее всего, вам знаком термин *IP-адрес*. Но что он действительно означает и для чего предназначен? Как именно один хост направляет информацию другому? Ответы на эти и другие вопросы можно найти в этом разделе.

Физические адреса

Можно искать физические MAC-адреса отправителя и получателя в заголовке IP-пакета до посинения, но так ничего и не обнаружить. MAC-адреса не имеют никакого отношения к протоколу IP, в котором используются логические адреса. В некоторых случаях MAC-адресов может вообще не существовать.

MAC-адреса используются для идентификации сетевого адаптера Ethernet в сети. Сетевой адаптер сам по себе никак не зависит от существования IP, заголовков IP-пакетов или логических IP-адресов. Итак, мы столкнулись с проблемой несоответствия двух понятий. Очевидно, что для реализации взаимодействия двух компьютеров должен существовать способ установки соответствия между логическими IP-адресами и физическими MAC-адресами.

Знаете ли вы IP-адрес вашего настольного компьютера? Если нет, то в этом нет ничего страшного или необычного. Все дело в том, что на сегодняшний день большинство наших компьютеров не имеют постоянного IP-адреса. Получить и зарезервировать определенный IP-адрес (или диапазон адресов) очень дорого. Большинству компьютеров IP-адреса выделяются на время одного сеанса работы в сети, для чего провайдер услуг Internet (ISP) использует специальное программное обеспечение, например протокол DHCP (Dynamic Host Configuration Protocol – протокол динамической конфигурации хостов).

Выделение IP-адреса

Протокол DHCP позволяет осуществлять динамическое назначение компьютерам IP-адресов и отказаться от трудоемкого процесса управления закрепленными за каждым хостом статическими IP-адресами. С помощью DHCP осуществляется централизованное и автоматическое назначение временных IP-адресов, что значительно повышает эффективность управления крупными сетями. Это очень удобно для администратора сети, но усложняет работу администратора системы безопасности (например, при поиске компьютеров с временными IP-адресами, которые являются источником потенциально опасных действий).

Каково же общее количество возможных IP-адресов? Их точное число – 2^{32} (так как адрес состоит из 32 бит), т.е. более 4 млрд. Однако не все IP-адреса доступны, существуют зарезервированные диапазоны адресов. Во время стремительного роста популярности Internet пришлось с грустью констатировать, что количество свободных IP-адресов быстро уменьшается. Встал вопрос: как же выйти из создавшегося положения?

Первый способ заключается в использовании DHCP отдельными сетевыми узлами, которые выделяют IP-адреса во временное пользование. Это позволяет снизить общее количество необходимых одновременно IP-адресов. Альтернативным способом является применение *зарезервированных адресов для частных сетей.* Организация IANA (Internet Assigned Numbers Authority – полномочный комитет по надзору за присвоением номеров Internet) зарезервировала некоторые диапазоны адресов для использования только во внутренних локальных сетях. Например, IP-адреса, начинающиеся с 192.168 и 172.16, могут использоваться только для обмена информацией между хостами в рамках локальной сети. Такой трафик не должен проходить через внешний шлюз конкретного узла. Этот метод позволяет сэкономить IP-адреса для отдельного узла Internet и использовать для его внутренних целей указанные адреса сетей класса B.

Продолжим. Некоторые из наших читателей знают IP-адрес своего компьютера. Уже хорошо. Но знаете ли вы на память свой MAC-адрес? Скорее всего, нет. Не так-то просто запомнить 48-битовый адрес, да это и не нужно.

Для преобразования физических MAC-адресов в логические IP-адреса предназначен протокол ARP (Address Resolution Protocol – протокол преобразования адресов). ARP как таковой не является протоколом Internet. Он служит для отправки кадра Ethernet (запроса) всем компьютерам данного сегмента сети. Такой запрос называют *широковещательным.* Широковещательное сообщение отправляется всем компьютерам сегмента или всей сети. Стоит особо подчеркнуть, что протокол ARP предназначен для опроса только хостов только локальной сети и не может применяться для обмена данными между хостами разных сетей.

Компьютер-отправитель посылает широковещательный ARP-запрос всем компьютерам локальной сети о наличии у них MAC-адреса, соответствующего определенному IP-адресу. Компьютер, которому предназначено сообщение, в ответ на запрос отправляет свой MAC-адрес. В результате такого обмена информацией и запрашивающий, и ответивший, а также все другие компьютеры, подключенные к сети, заносят новую запись в свои ARP-таблицы соответствий физических и логических адресов. Создание такой таблицы позволяет сократить количество новых ARP-запросов. В конечном итоге, все компьютеры сегмента локальной сети будут взаимодействовать с помощью MAC-, а не IP-адресов. При транзакции по стеку TCP/IP обмен информацией может осуществляться между соответствующими уровнями стека, но при действительной доставке данных используются именно MAC-адреса двух хостов.

Почему же MAC-адреса столь объемны? Ведь 48 бит дают огромное число возможных значений. Такой размер был выбран для получения абсолютно уникальных и "вечных" адресов. Эта фраза хороша только на слух, и уже сейчас планируется расширить длину MAC-адреса до 128 бит с целью удовлетворения существующих требований указания кода производителя в MAC-адресах сетевых адаптеров.

Логические адреса

IP-адрес состоит из 32 бит, уникально идентифицирующих хост. Это число записывается в виде четырех десятичных чисел, разделенных точками (например 192.168.5.5). Каждое из четырех чисел выбирается не случайно. Первая часть IP-адреса указывает на сеть, к которой подключен данный хост, а оставшиеся числа определяют место размещения хоста в этой сети. Различают несколько классов адресов. Класс определяет предельное число хостов, которые могут быть подключены к данной сети, или количество битов в уникальном IP-адресе хоста сети (табл. 1.1). Так, в адресах класса A предназначено 8 бит для сетевой части IP-адреса, а оставшиеся 24 бит – для определения конкретного хоста в этой сети. Таким образом, в сетях класса A могут работать более 16 млн. хостов (2^{24}–1). В качестве примера сети класса A может послужить сеть Массачусетского технологического института с диапазоном IP-адресов от 18.0.0.0 до 18.255.255.255.

Таблица 1.1. Классы IP-сетей

Класс	Сетевая часть адреса, бит	Узловая часть адреса, бит	Число хостов сети
A	8	24	более 16 млн.
B	16	16	более 65 тыс.
C	24	8	255

IP-адреса также классифицируют, начиная с класса A и заканчивая классом E. Классы A, B и C используются для хранения уникальных адресов. Адрес такого класса выделяется для одного конкретного компьютера. Адреса класса D являются широковещательными и предназначены для отправки пакетов указанной группе хостов. Адреса класса E зарезервированы для исследовательских целей. В табл. 1.2 перечислены классы IP-адресов и зарезервированные для них диапазоны.

Таблица. 1.2. Классы адресов

Класс	Начальный IP-адрес	Конечный IP-адрес
A	0.0.0.0	127.255.255.255
B	128.0.0.0	191.255.255.255
C	192.0.0.0	223.255.255.255
D	224.0.0.0	239.255.255.255
E	240.0.0.0	247.255.255.255

Использование CIDR

Вы могли уже встречать новый термин, касающийся назначения IP-адресов, — CIDR (Classless Inter-Domain Routing — бесклассовая междоменная маршрутизация). На протяжении длительного периода все IP-адреса должны были принадлежать к определенному классу, т.е. одна сеть могла объединять или чуть более 16 млн. хостов, или до 65534 хостов, или до 255 хостов. Однако чаще всего возникала потребность в сетях класса B, в которых работало от 255 до 65534 хостов. При этом многие IP-адреса таких сетей оставались свободными. С учетом того, что количество IP-адресов ограничено, необходимо было устранить проблему выделения лишних IP-адресов для сетей определенного класса.

CIDR представляет собой метод увеличения адресного пространства в Internet. С помощью CIDR устраняется 8-битовое ограничение на минимальную длину сетевой части адреса. В случае применения метода CIDR разделительная линия между сетевой и узловой частями

адреса может отделять любое число разрядов 32-битового адреса. При этом выделенный узлу диапазон сетевых адресов указывается уникальным образом. Например, запись всех адресов сети 192.168 по методу CIDR будет выглядеть следующим образом: 192.168/16. Первая часть этой записи является десятичной записью набора двоичных разрядов всей сети. После косой черты указывается число битов, представляющих сетевую часть IP-адреса. В данном случае CIDR-адрес аналогичен адресу сети класса B, но его легко изменить для обозначения менее крупной сети.

Маска подсети

Еще одним важным понятием является термин *маска подсети*. С помощью этой маски указывается, сколько битов в IP-адресе относится к сетевой части, а сколько — к хосту. Каждый бит сетевой части "маскируется" с помощью 1. Например, адрес сети класса A имеет 8-битовую сетевую часть и 24-битовую часть для представления хоста. Восемь последовательных единиц в двоичном коде соответствуют десятичному значению 255. Значит, маской подсети в данном случае будет 255.0.0.0. Соответственно, для сетей класса B маской подсети будет 255.255.0.0, а для сетей класса C — 255.255.255.0. Зачем же они нужны, если можно определить класс и число битов, зарезервированных для сети, проанализировав IP-адрес? Дело в том, что сетевые администраторы могут разбивать свои сети на несколько подсетей. Например, сеть класса C можно разбить на несколько отдельно адресуемых подсетей с помощью определенной маски подсети.

Порты

В соответствующих полях заголовков своих пакетов протоколы TCP и UDP сохраняют 16-разрядные номера портов. Это означает, что в этих полях может содержаться 65536 вариантов номера порта или службы: от 0 до 65535. Очень важно запомнить, что хотя каждой службе назначается стандартный номер порта, нет никакой гарантии, что так и будет в действительности. За службой `telnet`, например, почти повсеместно закреплен TCP-порт 23. Но не существует никаких препятствий для изменения этого номера на, скажем, 31337. А для хакера, взломавшего компьютер, лучшим способом скрыть свое присутствие будет выполнение своих действий через нестандартный порт. Если хакер запустит `telnet` на каком-нибудь порту с большим номером, то обнаружить несанкционированный доступ будет значительно сложнее. Любая служба может связаться с любым портом. Но если вы хотите обмениваться информацией с другими компьютерами, то все же лучше использовать стандартные порты. Для хостов под управлением UNIX для закрепления номеров TCP- и UDP-портов за стандартными, популярными службами используется информация файла `/etc/services`.

Рассмотрим пример типичной информации файла `/etc/services`. В данном фрагменте файла указаны названия служб и выделенные этим службам порты. Обратите внимание на то, что служба `domain` (это служба DNS) может работать и по протоколу TCP, и по UDP. Это кажется необычным, но вполне нормально. Большинство служб используют или тот, или другой протокол, но есть и исключения, как, например, служба DNS.

```
ftp.........21/tcp
telnet......23/tcp
smtp........25/tcp
```

```
domain.....53/udp
domain.....53/tcp
```

Посмотрим, как служба указывается в пакете (рис. 1.6). В заголовке UDP-пакета содержится 16-битовое поле под названием "Порт получателя". В этом поле определяется порт на удаленном компьютере, которому передается сообщение. В нашем примере значением этого поля является 53, т.е. дейтаграмма предназначена для службы DNS.

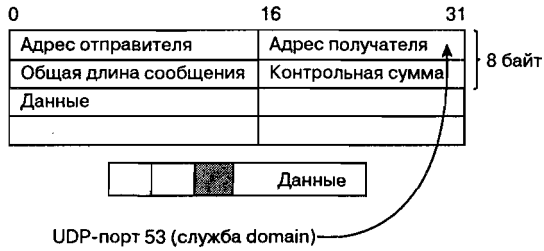


Рис. 1.6. Порт получателя в заголовке пакета UDP

Ранее особое значение придавалось портам с номерами до 1024, которые также называли *доверенными* (смешно звучит?) или *привилегированными*, так как эти порты были выделены для системных процессов. В этом был определенный смысл, пока сеть Internet была достаточно безопасной. На сегодняшний день все большее значение приобретают порты с номерами больше 1024, которые еще называют *временными портами* (ephemeral port) из-за того, что они могут использоваться любой службой по любой причине.

Протоколы стека TCP/IP

Еще раз вернемся к четырем базовым уровням модели TCP/IP (см. рис. 1.1). Взаимодействие со стеком TCP/IP осуществляется посредством какого-либо приложения. Для обмена файлами используется программа наподобие FTP, в качестве эмулятора терминала — `telnet`, а для пересылки текстовых сообщений (например, последнего анекдота сразу 50 друзьям) — приложение электронной почты. Такое приложение принимает сообщение пользователя и упаковывает его для передачи вниз по стеку TCP/IP.

Теперь рассмотрим методы передачи данных на транспортном уровне. Они реализуются на основе протоколов, ориентированных (TCP) или не ориентированных (UDP) на установление соединений. "Ориентированный на установление соединений" означает, что данный протокол выполняет максимальный объем действий для надежной доставки сообщений и полного согласования параметров соединения (handshake). Протокол, не ориентированный на установление соединений, отправляет сообщения "на удачу" и не гарантирует их доставку адресату. При этом надежность должна обеспечиваться на уровне приложений. В табл. 1.3 приведены некоторые свойства TCP и UDP.

UDP является одним из простейших протоколов, он только собирает пакеты и выдает их в сеть. На стороне получателя эти пакеты принимаются, разбираются

(последовательно удаляются заголовки сообщения при его передаче вверх по стеку), и пользователю выдается исходное сообщение. Конечно, на маршруте следования несколько пакетов могут оказаться утерянными, но часто это не вызывает серьезных последствий и даже допустимо. Например, при широковещательном распространении аудиоинформации потеря одного слова не приведет к утрате общего смысла текста. При передаче видеоданных на утерянный пакет укажет пустое место в окне изображения. В большинстве случаев такие потери не выходят за рамки допустимых. Передача данных с помощью UDP не означает, что они обязательно будут утеряны, просто сам протокол UDP не гарантирует их надежной доставки. Приложение может игнорировать утерянные пакеты, а может выдать запрос на их повторную передачу.

Таблица 1.3. Сравнение TCP и UDP

TCP	UDP
Надежный	Ненадежный
Ориентирован на установление соединений	Не ориентирован на установление соединений
Медленный	Быстрый

А что делать, если даже минимальная потеря данных недопустима? В этом случае нужно использовать протокол TCP, который гарантирует надежную доставку данных по назначению. Для проверки доставки пакетов в правильном порядке в TCP предусмотрено несколько механизмов. Одним из них является метод подтверждения получения.

Сообщение о подтверждении получения (ACK — acknowledgement) является важным элементом работы протокола TCP. За счет уведомления о получении каждого пакета и обеспечивается надежность TCP. Если пакет не был доставлен (то есть подтверждения не получено), то он отправляется повторно. Таким образом, TCP гарантирует доставку всех пакетов и поэтому считается надежным. Он работает чуть медленнее, чем UDP, но с помощью правильных настроек можно немного ускорить процесс обмена данными.

Последним из рассматриваемых протоколов будет ICMP (Internet Control Message Protocol — протокол управляющих сообщений Internet). Этот удобный простой набор приложений был первоначально разработан для поиска неисправностей в сетях и уведомления об ошибках. Самой известной программой из набора ICMP является программа проверки доступности адресата с помощью эхо-запроса (известная как `ping`). Благодаря ICMP также осуществляется управление потоком, изменение маршрута сообщений и сбор информации о сети (и это только некоторые из доступных функций). Более подробно о ICMP и его возможностях рассказано в главе 4, "Протокол ICMP".

Служба DNS

Назвать что-то еще не означает понять, чем это является на самом деле, но часто правильное название — половина успеха. Я помню, когда впервые услышал о DNS (Domain Name System — система доменных имен). В то время основные поставщики программного обеспечения для работы с базами данных анонсировали

свои продукты для работы с распределенными базами данных, и вскоре я понял, что имею дело именно с таким продуктом. DNS – это распределенная база данных, так как единая таблица адресов не хранится на каком-то одном хосте; информация распределена по многим серверам.

Когда-то все IP-адреса и имена компьютеров хранились в таблицах, информация которых обновлялась по ночам. С увеличением глобальной сети такой подход стал непрактичным по целому ряду причин, среди которых можно выделить размер таблиц и проблему надежности работы при сбое только в одном сервере. Рассмотрим фрагмент файла `/etc/hosts` сервера UNIX.

```
127.0.0.1       loopback
127.20.1.41....relay relay.sans.org
172.20.31.19    goo goo.sans.org
```

На хостах UNIX и Windows 2000 по-прежнему используются небольшие локальные файлы `hosts`. В этих файлах хранится информация о IP-адресах и именах локальных хостов и хостов, вызов которых осуществляется чаще всего. Информация этих файлов была дополнена возможностями DNS. Конфигурация большинства серверов UNIX и Windows 2000 предусматривает поиск адреса интересующего хоста, начиная с файла `/etc/hosts`, и только если этот поиск будет неудачным, происходит обращение к службе DNS. Таким образом, бо́льшая часть обязанностей системных администраторов теперь перекладывается на администраторов DNS-серверов.

Перед изучением самой службы DNS следует разобраться в понятии доменов DNS. *Домен* представляет собой логический элемент базы данных DNS. Первоначально было только семь "общих" доменов первого уровня `.com`, `.org`, `.edu`, `.net`, и менее известные `.int`, `.gov` и `.mil`. Не так давно к ним были добавлены `.aero`, `.biz`, `.coop`, `.info`, `.museum`, `.name` и `.pro`. Существуют и двухбуквенные домены первого уровня, используемые для обозначения принадлежности определенному государству (например, `.us`, `.fr` и `.uk` для США, Франции и Великобритании соответственно). Эти домены состоят из более мелких доменов, к которым ежедневно обращаются сотни пользователей (например, `yahoo.com` и `sans.org`). И каждый из этих доменов является фрагментом единой системы DNS.

Теперь давайте разберемся, как в DNS осуществляется преобразование символьного имени хоста в IP-адрес и наоборот. Если не вдаваться в детали, то вообще для решения этой задачи предназначены две команды: `gethostbyaddr` и `gethostbyname`. Это преобразование необходимо, так как пользователи вызывают удаленный компьютер по его символьному имени, а компьютер может взаимодействовать с другим компьютером, только обладая его IP-адресом. Да и в самой IP-дейтаграмме нет поля для хранения имен хостов, а только для их IP-адресов.

Используя вызов `gethostbyaddr`, хост отправляет IP-адрес DNS-серверу. DNS-сервер должен найти соответствующее имя хоста в базе данных и вернуть его источнику запроса. Этот процесс только на первый взгляд кажется элементарным (более подробно он рассматривается в главе 6, "DNS"). Для обратного преобразования имени хоста в IP-адрес DNS-серверу отправляется запрос `gethostbyname`. Это весьма упрощенная схема работы службы DNS, приведенная здесь только для начального ознакомления с технологией.

Маршрутизация

Вы еще не забыли о том, что стек TCP/IP состоит из четырех уровней: приложений, транспортного, сетевого и канального?

Мы прервали наш рассказ на сетевом уровне. Сетевой уровень неразрывно связан с понятием маршрутизации, то есть метода доставки сообщений с одного хоста на другой независимо от физической схемы линий связи. Возможно, этот уровень стоило назвать "IP-уровень", так как именно здесь для маршрутизации используются IP-адреса. Важно помнить, что сам протокол IP не зависит от физической среды передачи данных.

Мы уже рассказали о методе доставки трафика хосту сети с тем же идентификатором сети и маской подсети, что и хост-отправитель. Для отправки широковещательного запроса всем компьютерам локальной сети используется протокол ARP. На этот запрос отвечает только один компьютер – тот, который обладает указанным IP-адресом. В ответе содержится MAC-адрес этого хоста. Как же тогда трафик доставляется в другие сети, если широковещательные ARP-запросы возможны только в локальных сетях? Для этой цели и существует маршрутизация.

На каждом хосте хранится таблица маршрутизации, в которой указан адрес маршрутизатора, используемого по умолчанию. Если интересующий нас хост находится за пределами локальной сети, трафик должен быть отправлен этому маршрутизатору по умолчанию. Маршрутизатор отвечает за передачу трафика на один переход ближе к адресу назначения. Конечной точкой этого перехода может быть как другой маршрутизатор, так и хост-получатель, если он работает в сети, непосредственно подключенной к интерфейсу маршрутизатора. Возникает вопрос: как маршрутизатор узнает, куда следует передать данные, и как он обновляет информацию о доступных маршрутах? Ведь это должен быть постоянный процесс, связанный с возможным отключением одних сетей (например, из-за неисправностей) и подключением других.

Для обновления таблиц маршрутизации, хранящихся на каждом маршрутизаторе, используются специальные протоколы динамической маршрутизации.

Протоколы маршрутизации можно разделить на две категории: протоколы типа IGP (Interior Gaterway Protocol – протокол внутреннего шлюза) и протоколы типа EGP (External Gaterway Protocol – протокол внешнего шлюза). Протокол IGP предназначен для маршрутизации трафика между компьютерами *автономной системы* (Autonomous System – AS). Автономной системой называют совокупность маршрутизаторов, объединенных по иерархическому принципу, и находящихся под единым управлением. Среди протоколов категории IGP широкое распространение получил протокол RIP (Routing Information Protocol – протокол маршрутной информации). Это простой протокол, который не требует значительных усилий при настройке работы, поддерживаемый практически всеми сетевыми устройствами. Протокол OSPF (Open Shortest Path First – открытый протокол маршрутизации по кратчайшему пути) – это еще один протокол категории IGP. Протоколы RIP и OSPF различаются методом получения обновлений о доступных маршрутах и критериями поиска наилучшего маршрута.

Протоколы категории EGP требуются для передачи пакетов между автономными системами. Эти протоколы позволяют связать отдельные AS в единую сеть, в которой все компьютеры могут беспрепятственно обмениваться информацией

друг с другом. Наиболее известным EGP-протоколом можно назвать протокол BGP (Border Gaterway Protocol – протокол граничного шлюза). На текущий момент протокол BGP обеспечивает обмен данными по основным магистралям Internet. BGP-серверы на магистральных линиях Internet должны поддерживать таблицы маршрутизации, в которых хранятся адреса всех внешних маршрутизаторов Internet. Это, действительно, сложная задача.

Резюме

В этой вводной главе был сконцентрирован материал по множеству самых разнообразных вопросов. Не вдаваясь в детали, мы постарались познакомить наших читателей с базовыми концепциями, необходимыми для понимания следующих глав этой книги.

Прежде всего, необходимо запомнить, что обмен информацией между двумя подключенными к сети компьютерами осуществляется посредством нескольких уровней обработки данных. На стороне отправителя сообщение передается вниз по стеку и на каждом уровне к нему добавляется определенный заголовок. На стороне получателя осуществляется обратный процесс – заголовки последовательно удаляются при обработке пакета на определенном уровне стека. Таким образом, осуществляется взаимодействие между соответствующими уровнями хоста-отправителя и хоста-получателя. Передающийся пакет на каждом уровне немного отличается, поэтому и имеет другое название.

На разных уровнях стека TCP/IP для достижения пакетом пункта назначения используются и IP-, и MAC-адреса. Номера портов позволяют обратиться к нужному приложению на удаленном хосте, например к `sendmail` или `telnet`. Протокол TCP ориентирован на установку соединений и гарантирует их надежную доставку, тогда как UDP не обеспечивает такой гарантии и считается ненадежным. Служба DNS позволяет выполнять преобразование имен хостов в их IP-адреса и наоборот. Передача дейтаграмм по указанному адресу осуществляется благодаря процессу маршрутизации. Различные аспекты работы стека протоколов TCP/IP будут подробнее рассмотрены в следующих разделах этой книги.

2

TCPdump и TCP

После того как мы вкратце ознакомились со стеком протоколов TCP/IP, можно приступать к более глубокому изучению принципов его работы. Для этого воспользуемся практическим средством сетевого аналитика — анализатором пакетов TCPdump. Подобное средство так же необходимо для выявления атак хакеров или выполнения анализа трафика, как и знание основ TCP/IP. Анализатор пакетов TCPdump (или его Windows-собрат Windump) является весьма популярным и широко используемым средством, позволяющим исследовать трафик, передающийся в конкретной сети. В этой главе будет рассказано, как использовать TCPdump для достижения собственных целей и как интерпретировать полученные результаты. Затем мы перейдем к подробному рассмотрению TCP — одного из важнейших и популярнейших сетевых протоколов. Основное внимание будет уделено изучению правил его работы с помощью TCPdump.

Еще одним прекрасным и к тому же бесплатным средством перехвата и анализа пакетов является TCPdump Ethereal, доступный для использования и под Windows, и под UNIX. Вывод информации обо всех заголовках пакета и объеме его полезной информации выполняется в графическом пользовательском интерфейсе. Ethereal поддерживает анализ пакетов различных протоколов и даже способен расшифровать закодированный DNS-запрос. Резонно спросить: почему же тогда анализатор Ethereal не стал основным средством анализа пакетов, используемым для примеров данной книги? Тому есть несколько причин. Во-первых, результаты работы Ethereal сложнее отобразить в доступном для печати формате. TCPdump более лаконичен. Во-вторых, TCPdump — более простое средство и не требует от пользователя глубоких знаний для интерпретации предоставляемых результатов. Анализатор Ethereal заставляет больше размышлять над предоставляемыми сведениями, чем дает готовые ответы.

Вторая часть этой главы начинается с изучения протокола TCP. Во всех главах этой книги изучение различных протоколов ведется по одному принципу. Сначала исследуется нормальная работа протокола в обычных условиях. Но, так как

современная сеть Internet стала местом непредсказуемых событий, то, скорее всего, возникнут и нестандартные ситуации, описанию которых посвящается завершающая часть материала о каждом протоколе.

TCPdump

Анализатор пакетов TCPdump работает под управлением UNIX и позволяет перехватывать сетевой трафик, расшифровывать двоичную информацию и выводить на экран отчет. Смысл выдаваемого отчета становится понятным после непродолжительного изучения данного средства. Например, когда я только начал работать в лаборатории BMC Далгрина, то провел первую неделю, наблюдая за отчетами анализатора пакетов. Мой начальник, Боб Хотт, заходил ко мне каждые несколько часов, чтобы задать несколько вопросов и спросить, что еще послать по сети. В конце недели мы уже разобрались в правилах работы TCP/IP и свойствах своей локальной сети. Я настоятельно рекомендую не жалеть времени на изучение передающейся по сети информации. Поверьте, этот опыт не раз пригодится любому аналитику сетевого трафика.

Хотя результаты работы коммерческих программ могут немного отличаться или быть более удобными, чем отчеты TCPdump, но TCPdump все же ближе к реальности и помогает понять принцип действия и других подобных средств. В этой части демонстрируется использование TCPdump и поясняется смысл, выдаваемых им отчетов.

Где достать TCPdump или нечто подобное

TCPdump доступен для загрузки по адресу `ftp://ftp.ee.lbl.gov/tcpdump.tar.Z`.

Кроме того, нужно скачать программу `lippcap` для создания переносимой оболочки, позволяющей перехватывать низкоуровневый сетевой трафик. Для этого обратитесь по адресу `ftp://ftp.ee.lbl.gov/libpcap.tar.Z`.

Это так называемая "официальная" версия TCPdump, проверенная лабораторией Беркли. Не так давно совместными усилиями программный код TCPdump был отлажен и дополнен новыми возможностями. Последние, более функциональные версии можно найти по адресу `www.tcpdump.org`.

Windump — это версия TCPdump для Windows, которая доступна для загрузки по адресу `http://netgroup-serv.polito.it/windump`. Для его работы нужна программа winpcap, загрузить которую можно с этого же сайта.

Использование TCPdump

После установки TCPdump для запуска этого средства в большинстве операционных систем требуются права корневого пользователя (root). Ведь чтение пакетов требует доступа к сетевым устройствам, который возможен только при наличии привилегий суперпользователя. Запуск TCPdump выполняется с помощью одноименной команды **`tcpdump`**. По умолчанию перехватывается весь трафик указанного сетевого интерфейса, а все результаты выводятся на экран монитора. Такой режим работы не всегда удобен, так как при прослушивании загруженной сети новые строки будут появляться непрерывно. Поэтому предусмотрено несколько параметров командной строки, которые позволяют изменить режим работы по умолчанию.

Фильтры

Предположим, что нас интересуют только пакеты TCP. Анализатор TCPdump позволяет выбрать интересующие пакеты из общего трафика и выводить отчет только по этим пакетам. В TCPdump встроен "язык" задания фильтров, с помощью которого можно указать одно или несколько полей, исследуемых в IP-дейтаграмме. Если пакет удовлетворяет указанным условиям, то запись о нем будет сохранена. Для составления отчета только по TCP-пакетам применяется команда **`tcpdump 'tcp'`**. В этой команде параметр **`'tcp'`** обозначает используемый фильтр.

Это самый простой фильтр. Фильтры могут быть значительно сложнее, если исследовать несколько полей пакетов на предмет соответствия различным условиям. Для ограничения собираемой информации можно проверять характеристики любого поля IP-дейтаграммы, в том числе и объем полезных данных. Логично предположить, что в TCPdump пригодилась бы возможность указывать файл, в котором хранятся фильтры. Тогда пользователю не пришлось бы каждый раз вручную указывать этот файл в командной строке. И такая возможность, действительно, существует. Параметр **`-F`** ***`имя_файла`*** устанавливает, что фильтр сохранен в указанном файле.

Сбор двоичных данных

TCPdump выводит все полученные данные на экран. Такой режим приемлем при поиске определенной записи. Однако в большинстве случаев TCPdump работает в автоматическом режиме, собирая информацию для последующего анализа. Для последующего анализа пригодятся данные в двоичном формате, т.е. в необработанном (raw) виде. В отчете, который выводится на экран, данные уже обработаны и представлены в удобном для восприятия виде. Но для последующего анализа предпочтительнее сохранять все перехваченные данные в двоичном формате. Для сбора информации в режиме без обработки используется команда `tcpdump -w` *`имя_файла`*, где *`имя_файла`* — имя файла для хранения перехваченного трафика в двоичном формате.

Для чтения файла с двоичными данными необходим другой параметр командной строки: **`-r`**. Команда **`tcpdump -r`** ***`имя_файла`*** позволяет выполнить чтение информации из заданного файла, а не из сетевого интерфейса по умолчанию. Прочесть данный файл другим способом невозможно. Если вы применяли утилиту UNIX `tar`, то знаете, что созданный с ее помощью файл архива можно прочесть, только используя эту же утилиту. Подобный принцип применяется и в TCPdump.

Управление объемом сохраняемой информации

Рассмотрим еще один важный вопрос — определение необходимого количества информации, сохраняемой TCPdump. Этот анализатор не стремится сохранить всю передаваемую по сети дейтаграмму. Прич[illegible]ной тому — сбор лишних сведений. Очень часто интерес представляют только заголовки дейтаграммы, для сохранения которых достаточно установить определенный размер. *Фиксируемая длина* (snapshot length) — это параметр, который задает точное число сохраняемых TCPdump байтов пакета. Стандартным значением этого параметра является 68 байт.

Что же именно содержится в этих 68 байт? Рассмотрим структуру пакета (рис. 2.1). Длина полей заголовка может отличаться от длины, указанной на схе-

ме. Это зависит от используемого протокола и наличия параметров. Первым идет заголовок пакета канального уровня. Для Ethernet заголовок кадра равен 14 байт и содержит поля для хранения MAC-адресов отправителя и получателя. Далее следует заголовок IP-дейтаграммы, минимальная длина которого (при отсутствии IP-параметров) составляет 20 байт, а за ним — заголовок инкапсулированного протокола (TCP, UDP, ICMP и т.д.) размером от 8 и до более чем 20 байт (максимальное значение будет при наличии всех параметров в заголовке TCP-пакета). Из-за установленного размера фиксируемой длины дейтаграммы часть самих данных (полезной нагрузки), сохраняемых анализатором, не может быть очень большой. Для изменения фиксируемой длины используется команда **`tcpdump -s длина`**, где ***длина*** — заданное число сохраняемых байтов дейтаграммы. Если нужно сохранить полный Ethernet-кадр (не включая 4 байт трейлера), воспользуйтесь командой `tcpdump -s 1514`. Это позволит сохранить 14-байтовый заголовок кадра Ethernet плюс 1500 байт — максимальный размер единицы информации, передаваемой по сети Ethernet.

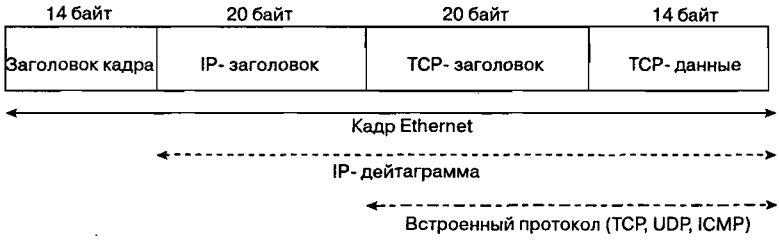


Рис. 2.1. Пример структуры пакета

Для TCPdump существует много других параметров командной строки. Команда **`man tcpdump`** позволяет вывести их полный список и описание. Приготовьтесь к тому, что информация будет довольно объемной (лучше заменить картридж в принтере и загрузить побольше бумаги для распечатки), но очень полезной для тех, кому это действительно интересно.

Результаты работы TCPdump

В этой книге приведено множество результатов работы (или, как их еще называют, трассировок) TCPdump. Правильная интерпретация выдаваемых TCPdump данных является одной из сложнейших задач для начинающего аналитика сетевого трафика. Несмотря на то что при исследовании пакетов различных протоколов (например, TCP, UDP, ICMP) TCPdump выдает однотипную информацию, существуют и характерные особенности. Первым шагом анализа является определение использованного протокола. Рассмотрим стандартный формат отчета TCPdump на примере записи, создаваемой этим анализатором при прохождении по сети TCP-пакета:

```
09:32:43:910000 nmap.edu.1173 > dns.net.21: S 62697789:62697789(0) win 512
```

- `09:32:43:910000` — отметка времени, в которой первые три пары цифр используются для указания соответственно часа, минуты и секунды записи информации, а последние шесть — для хранения дробной части секунды.

- `nmap.edu` — имя хоста-отправителя. Если невозможно получить имя хоста по IP-адресу или такое преобразование отменено (с помощью параметра `-n` команды `tcpdump`), то вместо имени хоста будет выведен его IP-адрес.
- `1173` — номер порта отправителя.
- `dns.net` — имя хоста-получателя.
- `21` — номер порта получателя (например, порт 21 считается портом FTP).
- `S` — обозначение флага TCP. `S` обозначает флаг SYN, используемый при установке TCP-соединения.
- `62697789:62697789(0)` — запись в формате *начальный порядковый номер TCP-сегмента:конечный порядковый номер TCP-сегмента*. Числа обозначают байты данных. Порядковые номера в TCP используются для сбора полученных данных в правильной последовательности. *Начальный порядковый номер* (ISN — initial sequence number) устанавливается при каждом TCP-соединении и обозначает номер первого байта передаваемых данных. Конечный порядковый номер *номер* определяется сложением начального порядкового номера с числом байтов, отправленных в TCP-сегменте. В данном случае число байтов, переданных в виде запроса на установление соединения, равно 0 (это стандартное значение). Поэтому значения начального и конечного порядковых номеров совпадают. В процессе нормального установления соединения никаких данных не передается.
- `win 512` — размер окна хоста (в байтах), заданное для передающего хоста `nmap.edu`. Окно — это буфер для хранения отправляемых данных.

Таблица 2.1. Флаги TCP в записях TCPdump

Флаг	Обозначение	Описание
SYN	`S`	Используется для запроса на установление TCP-соединения
ACK	`ack`	Используется для подтверждения получения данных. Может использоваться в комбинации с другими флагами
FIN	`F`	Указывает, что хост-отправитель начинает процедуру стандартного завершения соединения
RESET	`R`	Используется для указания немедленного разрыва установленного соединения
PUSH	`P`	Применяется для немедленной отправки данных с передающего хоста без ожидания заполнения всего буфера. В данном случае основной целью является ускорить получение ответа, а не повысить эффективность обмена данными. Для многих интерактивных приложений, например Telnet, скорость ответа является важнейшим параметром, о чем и говорит наличие флага PUSH
URGENT	`urg`	Этот флаг указывает на существование "экстренной" информации, которая должна отправляться в первую очередь. Примером такой информации являются сведения о нажатии комбинации клавиш <Ctrl+C> для немедленного прекращения загрузки данных по FTP
Заполнитель	. (точка)	Если для пакета не установлено флагов SYN, FIN, RESET или PUSH, то после порта получателя указывается заполнитель (точка)

Флаги TCP

В пакетах обычного TCP-соединения устанавливается один или несколько флагов. Флаги используются для указания предназначения пакета. В табл. 2.1 приведены флаги TCP, их обозначение в TCPdump и функция каждого флага.

Узнать TCP-пакет в отчете TCPdump можно по наличию порядковых номеров и обозначений флагов. В записях о UDP-пакетах обычно присутствует слово *udp*. В большинстве случаев этот метод выявления UDP-протокола работает, но все же не всегда. Дело в том, что TCPdump выполняет анализ пакетов некоторых UDP-служб, например пакетов службы DNS и SNMP-пакетов, не только на уровне транспортного протокола, но и на уровне приложений. Как и анализатор пакетов Ethereal, TCPdump способен различать протоколы и дешифровать полезные данные некоторых из них. В первое время выводимые отчеты могут показаться необычными, так как в них не будет присутствовать ни слово *udp*, ни характерные признаки протокола TCP (флаги и порядковые номера). Как правило, это просто более подробный отчет о UDP-пакетах. Определить ICMP-пакет довольно просто: в строке отчета обязательно должно присутствовать слово *icmp*.

Абсолютные и относительные порядковые номера

Хотелось бы обойтись без излишних сложностей при обсуждении отчетов TCPdump, но порядковые номера протокола TCP требуют более внимательного изучения. Как уже указывалось, они присутствуют только в строках отчета о TCP-пакетах и предназначены для сбора отправленной информации в исходном порядке на хосте-получателе. Напомним, что TCP, в отличие от UDP, гарантирует сохранение порядка передаваемой информации. Порядковые номера – это десятичное представление 32-разрядного поля, поэтому они могут иметь довольно большие значения, которые трудно воспринимать на глаз. TCPdump облегчает эту задачу с помощью преобразования абсолютных ISN-номеров (начальных порядковых номеров) в относительные порядковые номера. Это преобразование выполняется после обмена двумя хостами своими ISN-номерами. Рассмотрим следующий фрагмент отчета TCPdump (для простоты была опущена метка времени).

```
client.com.38060 > telnet.com.telnet: S 3774957990:3774957990(0) win 8760
↳<mss 1460> (DF)
telnet.com.telnet > client.com.38060: S 2009600000:2009600000 ack
↳3774957991 win 1024 <mss 1460>
client.com.38060 > telnet.com.telnet: .ack 1 win 8760 (DF)
client.com.38060 > telnet.com.telnet: P 1:28(27) ack 1 win 8760 (DF)
```

В разделе "Установление TCP-соединения" этот отчет рассматривается с теоретической точки зрения. А сейчас обратим внимание только на числа, выделенные жирным шрифтом. Первые два из них представляют собой ISN-номера в абсолютном формате, которыми обмениваются хосты `client.com` и `telnet.com` соответственно. В третьей строке фрагмента содержится выделенный жирным шрифтом относительный порядковый номер – `1`. Это означает, что хост `client.com` подтвердил получение предыдущего SYN-пакета от `client.com`, ISN-номер которого `2009600000`. Единица используется для подтверждения того, что следующим байтом, который будет принят хостом `client.com`, будет байт с относительным порядковым номером 1.

Следовательно, если бы отображался абсолютный порядковый номер, то он бы составил 2009600001. Если изложенное кажется непонятным, то обратитесь к более подробной информации о порядковых номерах при подтверждении получения сообщения, которая содержится в разделе "Введение в TCP".

В последней строке выделенные жирным шрифтом числа 1 и 28 обозначают, что хостом client.com хосту telnet.com были отправлены 27 байт с относительными порядковыми номерами, начиная с 1 и по (не включительно) 28 байт по отношению к абсолютному порядковому номеру 3774957990. Значение порядкового номера подтверждения ack 1 не изменится, пока хост telnet.com не подтвердит получение новых данных.

С помощью команды tcpdump -S можно запретить выполняемое по умолчанию преобразование абсолютных порядковых номеров в относительные после обмена хостами ISN-номерами.

Смена интерфейса, исследуемого TCPdump

TCPdump позволяет исследовать трафик, проходящий как через один заданный интерфейс, так и трафик сразу нескольких интерфейсов. При этом в отчете используемый по умолчанию интерфейс будет иметь наименьший номер (не учитывая интерфейса петли обратной связи). Сервер под управлением Linux может иметь два сетевых адаптера, интерфейсы которых будут, например, eth0 и eth1. Для изменения интерфейса, исследуемого по умолчанию, используется параметр -i команды tcpdump. Следующая команда позволяет назначить для прослушивания интерфейс ppp0.

```
tcpdump -i ppp0
```

Вывод данных в шестнадцатеричном формате

В отчетах TCPdump не выводится информация всех полей сохраненных данных. Например, в заголовке IP-дейтаграммы содержится поле, в котором сохраняется информация о размере заголовка. Как узнать его содержимое, если оно не выводится в стандартном отчете TCPdump? С помощью команды tcpdump -x на экран выводится вся сохраненная информация дейтаграммы (согласно заданному параметру фиксируемой длины) в шестнадцатеричном формате. Хотя читать и интерпретировать данные в таком виде значительно сложнее, зато предоставляется возможность изучить всю сохраненную часть дейтаграммы.

Чтобы разобраться в выводимых в шестнадцатеричном формате результатах TCPdump, необходимо обратиться к дополнительному справочному материалу, в котором описан формат заголовков IP-дейтаграмм и указано значение каждого поля (мы рекомендуем книгу Ричарда Стивенса *TCP/IP Illustrated, Volume 1*). После изучения соответствующего материала нужно преобразовать шестнадцатеричные значения в десятичные для числовых полей и числовые значения — в код ASCII для символьных полей. Судя по всему, лучшим средством для преобразования записей TCPdump, сохраненных в двоичном формате, является анализатор пакетов Ethereal, в котором для этой цели предназначена команда -w tcpdump.

Введение в TCP

TCP — это надежный, ориентированный на установление соединений протокол. Он используется многими приложениями, в том числе telnet и SMTP. При-

ложения, подобные `telnet`, не допускают потери дейтаграмм или их доставки в отличном от исходного порядке. Поэтому они нуждаются в протоколе, обеспечивающем упорядоченную и надежную доставку сообщений. Для обеспечения этих функций в TCP используются следующие механизмы.

- **Уникальное TCP-соединение.** Организация TCP-сеанса означает создание уникального (одиночного) соединения между двумя хостами. После установки соединения оба участвующих в сеансе компьютера получают возможность взаимодействовать между собой в процессе обмена данными.
- **Применение порядковых номеров.** С помощью порядковых номеров обеспечивается сохранение порядка отправляемых и получаемых пакетов. Команда `telnet` позволяет передать все данные, упакованные в несколько TCP-пакетов. Пересылаемым данным присваиваются порядковые номера для указания начала полезных данных в каждом пакете. Пакеты могут приходить в порядке, отличном от порядка их отправления, но порядковые номера позволяют восстановить правильную последовательность данных.
- **Подтверждения**. Подтверждения используются для уведомления отправителя о получении данных. В поле подтверждения (ACK) передается значение порядкового номера плюс 1, чтобы обозначить получение конкретных данных. Если отправитель не получает подтверждения о получении конкретных данных за определенный период времени, то данные считаются утерянными, и они отправляются повторно.

Установление TCP-соединения

Процесс установления TCP-соединения полностью регламентирован и заключается в выполнении специальной процедуры согласования параметров (handshake). Эта процедура выполняется до пересылки каких-либо данных между хостами. На рис. 2.2 схематически показано, как клиент (хост-отправитель) инициирует установку соединения с сервером (хост-получатель). Термин *клиент* означает, что данный хост обращается к какой-то службе на другом компьютере. Сервер — это хост, который ожидает запросов к своему порту, выделенному для работы определенной службы. Протокол TCP требует, чтобы в отправляемом пакете был указан порт получателя. Например, стандартными портами сервера являются порты с номерами 23 (`telnet`), 25 (`sendmail`) и 80 (HTTP или порт Web-сервера).

Рис. 2.2. Полная процедура согласования параметров соединения

Полная процедура согласования параметров соединения осуществляется следующим образом.

1. Клиент отправляет SYN-пакет в качестве запроса на установку TCP-соединения.
2. Если сервер подключен к сети, ожидает запросов к порту запрашиваемой службы и способен в данный момент принять входящий вызов, то он отправляет клиенту ответный пакет. В этом пакете установлены флаг SYN в качестве запроса на соединение и флаг ACK для подтверждения полученного запроса.
3. Если клиент получает пакет с установленными флагами SYN и ACK и хочет продолжить процедуру установки соединения, то он отправляет серверу ACK-пакет. Таким образом он подтверждает получение запроса на соединение.

После завершения полной процедуры согласования параметров соединение считается установленным. С этого момента два хоста могут обмениваться данными. Если рассмотреть процедуру установки соединения более внимательно, то можно прийти к выводу, что на самом деле устанавливаются два односторонних соединения: первое от клиента к серверу и второе от сервера к клиенту. Так и должно быть, ведь протокол TCP является *полнодуплексным*, т.е. данные могут передаваться в двух направлениях по независимым каналам.

В следующем примере показан отчет TCPdump при исследовании трафика, передаваемого при выполнении полной процедуры согласования параметров соединения.

```
tclient.net.39904 > telnet.com.23: S 733381829:733381829(0) win 8760
↳<mss 1460> (DF)
telnet.com.23 > tclient.net.39904: S 1192930639: 1192930639(0) ack 733381830
↳win 1024 <mss 1460> (DF)
tclient.net.39904 > telnet.com.23: . ack 1 win 8760 (DF)
```

В первой записи клиент `tclient.net` отправляет запрос на соединение с портом 23 хоста `telnet.com`. При этом в пакете установлен флаг SYN, за которым указан ISN-номер `733381829` и совпадающий с ним конечный порядковый номер. В скобках указана полезная нагрузка пакета (0 байт). Следующее число — это размер окна (`8760`) и максимальный размер TCP-сегмента (maximum segment size — mss). Значение размера окна свидетельствует о том, что клиент использует буфер размером в 8760 байт для хранения принимаемых данных. Значение mss информирует получателя о том, что сеть, в которой работает хост, не способна пропускать более 1460 байт полезной нагрузки в одном TCP-сегменте (20-байтовый заголовок IP-дейтаграммы + 20-байтовый заголовок TCP-сегмента + 1460 байт полезных данных TCP = 1500 байт, что составляет максимальную единицу передачи данных в сети Ethernet). В данном случае, хотя клиент и способен принимать до 8760 байт данных, но сеть, в которой он работает (скорее всего Ethernet), не позволяет доставить более 1460 байт полезной нагрузки в одном TCP-сегменте.

Во второй записи зафиксирована отправка хостом `telnet.com` пакета с установленными флагами SYN и ACK хосту `tclient.net`. Этот пакет информирует клиента о доступности сервера и его готовности установить собственное соединение. Кроме того, передается ISN-номер сервера (`1192930639`), который одновременно является и его конечным порядковым номером, так как никаких данных не передается (стандарт для пакетов с флагами SYN и ACK). Следующим чис-

лом является номер подтверждения, в данном случае 733381830. Обратите внимание на то, что это значение равно сумме ISN-номера первой записи (733381829) плюс 1. Таким образом, хост telnet.com подтвердил, что он ожидает в качестве следующего порядкового номера передаваемых данных от tclient.net абсолютный порядковый номер 733381830. Установленным размером окна для telnet.com является значение в 1024 байт, а максимальным размером сегмента — 1460 байт.

В последней строке содержится запись об отправке клиентом tclient.net ACK-пакета, которая подтверждает получение от сервера пакета с флагами SYN и ACK. Число 1 — это относительный номер подтверждения, указывающий на то, что ожидается получение первого байта информации от telnet.com. Обратите внимание на то, что с этой записи абсолютные порядковые номера меняются на относительные. Непосредственно после двоеточия, отделяющего адрес получателя, установлена точка. Напомним, что точка используется в качестве заполнителя, если в пакете не установлен ни один из флагов PUSH, RESET, SYN или FIN.

Порты клиента и сервера

Ранее было принято, чтобы для работы служб сервера выделялись порты, номера которых находятся в диапазоне от 1 до 1023. В UNIX-системах доступ к этим портам возможен только для процессов, запущенных с привилегиями суперпользователя. Эти порты закрепляются за определенными службами. Другими словами, если однажды на хосте для службы telnet был выделен порт 23, то эта служба и далее будет использовать этот порт по умолчанию. На привилегированных портах работают многие старые, хорошо известные службы. Новые службы отошли от этой традиции, и стандартным портом для AOL Instant Messenger, например, является TCP-порт 5190. Во многом это объясняется тем, что уже существует больше служб, чем номеров в данном диапазоне.

Порты клиента, которые часто называют *временными портами* (ephemeral ports), выделяются только для конкретного соединения и доступны для использования сразу после завершения этого соединения. Их номера, как правило, больше числа 1023. При организации клиентом доступа к серверу выбирается один из временных портов. Для большинства служб порты, используемые клиентом и сервером, не меняются до завершения сеанса. Порты двух хостов, между которыми установлено соединение, называются *парой сокетов* (socket pair). Такое соединение является уникальным — в сети Internet не существует аналогичной комбинации IP-адреса и порта отправителя и IP-адреса и порта получателя.

С одного компьютера к определенной службе на удаленном компьютере могут подключиться несколько пользователей. Но при этом для соединений, организованных различными пользователями, будут выделены разные временные порты. Пусть, например, два пользователя обращаются к информации одного Web-сервера. Хотя при этих двух соединениях используются одинаковые IP-адреса получателя и отправителя и тот же порт Web-сервера (порт 80), но Web-сервер отправляет каждому из получателей ответ именно на его запрос, основываясь на различных номерах временных портов.

Еще раз вернемся к процедуре согласования параметров соединения, но в этот раз обратим основное внимание на порты клиента и сервера.

```
tclient.net.39904 > telnet.com.23: S 733381829:733381829(0) win 8760
↳<mss 1460> (DF)
telnet.com.23 > tclient.net.39904: S 1192930639: 1192930639(0) ack 733381830
↳win 1024 <mss 1460> (DF)
tclient.net.39904 > telnet.com.23: . ack 1 win 8760 (DF)
```

Хост `tclient.net` для подключения к порту 23 (`telnet.com`) сервера использует временный порт 39904. После установки соединения все операции обмена данными выполняются посредством этих двух портов. Спустя определенный промежуток времени после завершения соединения хост освобождает порт 39904 и тот становится доступным для использования при другом соединении. Порт 23 сервера `telnet.com` остается в состоянии ожидания новых запросов к службе `telnet`.

Завершение соединения

Любой сеанс может быть завершен либо по стандартной процедуре (постепенный метод), либо внезапно, без выполнения этой процедуры. Постепенный метод завершения соединения можно сравнить с разговором по телефону с дилером, когда клиент отвечает продавцу: "Нет, спасибо. Нас это не интересует", после чего клиент вешает трубку. Таким образом, продавца уведомили о том, что разговор закончен, что ему следует повесить трубку и надоедать звонками другим потенциальным покупателям. Внезапное прекращение соединения выглядит аналогично неучтивому бросанию телефонной трубки без каких-либо лишних разговоров.

Постепенный метод

При использовании стандартного (постепенного) метода завершения TCP-соединения один из взаимодействующих хостов уведомляет другой хост о желании завершить соединение с помощью FIN-пакета. Второй хост отвечает пакетом с установленным флагом ACK (для подтверждения получения запроса). Этим закрывается первый канал двухстороннего соединения. Затем второй хост также отправляет FIN-пакет и ждет подтверждения (ACK) от первого хоста. Такая процедура необходима, так как TCP является полнодуплексным протоколом. И клиент, и сервер ведут асинхронный обмен данными, поэтому и требуется завершение соединения на обеих сторонах. С помощью TCPdump рассмотрим две операции обмена данными.

1. Клиент инициирует завершение соединения с помощью FIN-пакета, а сервер подтверждает его получение ACK-пакетом.

   ```
   tclient.net.39904 > telnet.com.23: F 14:14(0) ack 186 win 8760 (DF)
   telnet.com.23 > tclient.net.39904: . ack 15 win 1024 (DF)
   ```

2. С помощью FIN-пакета сервер инициирует завершение соединения со своей стороны, а клиент отвечает ACK-пакетом подтверждения.

   ```
   telnet.com.23 > tclient.net.39904: F 186:186(0) ack 15 win 1024 (DF)
   tclient.net.39904 > telnet.com.23: . ack 187 win 8760 (DF)
   ```

С этого момента можно считать, что был выполнен разрыв соединения между хостами `tclient.net` и `telnet.com`.

Метод внезапного разрыва

Другим методом завершения соединения является его внезапный разрыв. Для этого один хост отправляет другому пакет с установленным флагом RESET, сообщая этим о необходимости немедленного разрыва соединения.

```
tclient.net.39904 > telnet.com.23: R 28:28(0) ack 1 win 8760 (DF)
```

В отчете TCPdump показано, как хост `tclient.net` обрывает соединение с `telnet.com` с помощью отправки ему пакета с флагом RESET. Дальнейший обмен информацией между хостами в рамках установленного сеанса невозможен.

Обмен данными

После изучения правил установки и завершения TCP-соединений пришло время обсудить, что же происходит в промежутке между двумя этими событиями. Основной причиной организации сеанса связи является необходимость обмена какими-то данными. Рассмотрим отчет об обмене данными между клиентом и сервером после выполнения полной процедуры согласования параметров соединения.

```
tclient.net.39904 > telnet.com.23: P 1:28(27) ack 1 win 8760 (DF)
telnet.com.23 > tclient.net.39904: P 1: 14(13) ack 1 win 1024
telnet.com.23 > tclient.net.39904: P 14: 23(9) ack 28 win 1024
```

Первая строка информирует о том, что хост `tclient.net` отправил 27 байт данных (число в круглых скобках) хосту `telnet.com`. Здесь мы впервые встречаемся с использованием символа `P`, которым обозначается флаг PUSH. Telnet является интерактивным приложением, для работы которого требуется максимальная скорость обратного ответа. Поэтому использование флага PUSH позволяет немедленно передавать данные приложению после их поступления в буфер получателя. В этой строке также подтверждается, что следующим байтом данных, который ожидается от `telnet.com`, является байт с относительным порядковым номером 1.

Во второй строке содержится запись об отправке хостом `telnet.com` 13 байт данных хосту `tclient.net` и подтверждение получения 1 байт данных от этого хоста. Позднее он должен будет еще подтвердить получение 27 новых байтов, только что отправленных клиентом `tclient.net`.

В последней строке зафиксирована информация об отправке сервером еще 9 байт данных. Обратите внимание на то, что относительные номера байтов начинаются с 14 (`14:23`), так как предыдущие 13 байт (`1:14`) были отправлены в прошлом пакете. В этой строке также подтверждается получение 27 байт данных от клиента. Подтверждение (ACK) имеет номер `28` и называется *ожидаемым подтверждением* (expectational acknowledgement). Этот номер — номер следующего ожидаемого к получению байта данных. Флаг ACK используется во всех операциях обмена данными по протоколу TCP, а также в качестве подтверждения организации сеанса.

Практические результаты

Представим, что вам нужно проанализировать сетевой трафик на предмет несанкционированных действий. Действительно ли изложенные теоретические сведения помогут сделать выводы о корректности или вредоносном содержании

передаваемого трафика? Элементарный анализ можно выполнить без исследования отдельных битов. Ниже указано несколько ключевых моментов, на основании которых можно сделать предварительное заключение.

- **Проведена ли полная процедура согласования параметров соединения?** Если это так, то сервер ожидал запросов к запрошенному клиентом порту и готов установить соединение. Все нормально, если на сервере запущена служба, которая и должна была ожидать запросов на этот порт. Но что, если указан порт, который не должен был использоваться? Это может означать, что работает какая-то служба, о которой знает только системный администратор. Кроме того, есть вероятность, что кто-то без вашего ведома установил приложение и создал потайной ход, который и используется в данном случае.
- **Передаются ли данные?** В отчетах TCPdump после порядковых номеров в скобках указывается количество переданных байтов данных. Изучение количества переданных байтов данных может пригодиться при ретроспективном анализе необычной активности во время взаимодействия двух хостов и для оценки масштабов происшествия. Иногда нельзя просмотреть сами байты данных, но объем переданной информации фиксируется всегда. Чрезмерно длительный сеанс или большое количество отдельных операций обмена данными могут свидетельствовать об опасности вероятной атаки злоумышленника.
- **Кто инициировал и кто завершил соединение?** Определив, какой хост инициировал и завершил сеанс связи, можно сделать вывод о том, кто управляет соединением. Как правило, клиент организует запрос, на который отвечает сервер. Каждый из этих двух хостов может разорвать соединение с помощью пакетов с флагами FIN или RESET, поэтому следует выяснить, какой хост был инициатором завершения сеанса.

Оценка нанесенного ущерба

Использование TCPdump для оценки нанесенного злоумышленником ущерба напоминает изучение обстоятельств и оценку убытков при реальной краже. Вначале следует определить, удалось ли нарушителю действительно проникнуть в компьютерную систему. Отчет о получении повторяющихся SYN-пакетов, на которые не последовало ответа, можно сравнить с безуспешными попытками взлома входной двери. Завершение полной процедуры согласования параметров соединения аналогично входу грабителя в чужой дом (возможно, преступнику придется взламывать еще одни двери, но все же вторжение уже произошло). Если обнаруживается, что злоумышленником была проведена полная процедура согласования параметров соединения, то это эквивалентно обнаружению открытой входной двери, которая была заперта.

По указанному в полученном пакете номеру порта сервера можно определить предмет интереса нарушителя. Если используется порт стандартной службы, например `telnet`, то, вероятно, атака будет серьезной и нацеленной на серьезный ущерб (получение файлов паролей, информации о доверительных отношениях с другими хостами и т.д.), что равнозначно интересу грабителя к ювелирным украшениям. А что если для взлома были использованы порты, которые не закреплены за стандартными службами? Это признак того, что нарушитель просто проник в вашу систему, доказав возможность взлома. Представьте себе, что, вернувшись домой, вы обнаружили, что кто-то взял ваше молоко из холодильника, выпил его и бросил пустой пакет на пол. Где гарантия, что он больше ничего не взял или не сделал?

Ущерб от квартирной кражи можно оценить по отсутствию хорошо известных ценных вещей. Но что делать, если произошел взлом большого склада, битком набитого разными товарами, которые хранились без надлежащего учета? Как узнать, что именно было украдено? С помощью изучения отчетов TCPdump о количестве переданных байтов данных можно определить средства, с помощью которых нарушитель извлекал украденную информацию. Найденные незначительные улики позволяют добиться серьезных успехов в расследовании преступления.

Вредоносное использование TCP

В следующих главах приведено множество примеров атак с использованием протокола TCP. В приложениях A, "Программы атаки и методы сканирования", и Приложение B , "Выявление разведывательных действий", рассматриваются методы сканирования, при которых применяются различные и иногда совершенно неожиданные комбинации TCP-флагов для исследования сетей в обход средств обнаружения и фильтрации. Здесь же мы остановимся на других методах вредоносного использования TCP, например, сканировании с помощью ACK-пакетов, сканировании `telnet` и перехвате TCP-сеансов.

Сканирование с помощью ACK-пакетов

Сканирование портов выполняется с различными целями, но в большинстве случаев предназначено для выявления хоста (или хостов), на котором запущена определенная служба. Если на удаленном хосте будет обнаружена потенциально уязвимая служба, то хакер, используя ее уязвимые места, может попытаться проникнуть в систему. Часто попытки сканирования являются прямолинейными и очевидными, когда нарушитель не пытается скрыть своих намерений (иногда скрывается только действительный адрес хоста, с которого выполняется сканирование). Хакер предполагает, что сканирование пройдет незамеченным или (при использовании скомпрометированного чужого хоста), что его не удастся выследить.

Однако иногда нарушители стараются скрыть попытки сканирования и выполнить их максимально незаметно. Рассмотрим приведенный ниже фрагмент отчета TCPdump о нескольких попытках установки соединения. Зондируя сеть, нарушитель пытается найти компьютер, который бы ответил на его ACK-пакет. Для упрощения листинга были удалены отметки времени.

```
ack.com.23 > 192.168.2.112.23: . ack 778483003 win 1028
ack.com.23 > 192.168.31.4.23: . ack 778483003 win 1028
ack.com.143 > 192.168.2.112.143: . ack 778483003 win 1028
ack.com.143 > 192.168.31.4.143: . ack 778483003 win 1028
ack.com.110 > 192.168.2.112.110: . ack 778483003 win 1028
ack.com.110 > 192.168.31.4.110: . ack 778483003 win 1028
ack.com.23 > 192.168.14.19.23: . ack 778483003 win 1028
ack.com.143 > 192.168.14.19.143: . ack 778483003 win 1028
ack.com.110 > 192.168.14.19.110: . ack 778483003 win 1028
ack.com.23 > 192.168.33.53.23: . ack 778483003 win 1028
ack.com.23 > 192.168.37.3.23: . ack 778483003 win 1028
ack.com.23 > 192.168.14.49.23: . ack 778483003 win 1028
```

В данном случае с помощью ACK-пакетов выполняется зондирование сети `192.168`. Источником зондирования является хост `ack.com`. Пакет, в котором установлен только один флаг ACK, отправляется на последнем этапе полной процедуры согласования параметров соединения при подтверждении получения данных FIN-пакета или при передаче данных, когда не отправляются все данные из буфера. Все это не соответствует данному случаю, так как больше никаких данных (которые бы указывали на нормальную реакцию в ответ на полученные пакеты) от `ack.com` не поступает.

Такая попытка найти подключенные компьютеры напоминает `ping`-зондирование с помощью эхо-запросов. Если работающий компьютер получает

ACK-пакет, отправленный или на закрытый, или на открытый порт, то он обязан ответить пакетом с флагом RESET. Кроме того, фильтрующие маршрутизаторы, которые разрешают обмен данными по сети только для "полностью установленных" соединений (т.е. только пакетами с установленным флагом ACK), пропустят этот вид зондирования. По мере того как на узлах Internet повышаются требования безопасности и добавляются фильтры для блокирования потенциально опасного трафика, нарушителям приходится придумывать все более хитроумные и незаметные способы сканирования, подобные тому, что использовался в данном примере.

Обратите внимание на то, что в полученных пакетах порты отправителя совпадают с портами получателя. Для клиента необычно выбирать для отправки данных временный порт, номер которого больше 1023. Это еще один признак выполняемого зондирования. При наличии в пакете только одного установленного флага ACK и одинаковых номеров портов отправителя и получателя можно предположить, что такие пакеты созданы специально. Кто-то написал программу для выполнения такого зондирования – данные пакеты не являются стандартным TCP-трафиком.

Частные сети

В тексте нашей книги для использования в примерах было решено использовать IP-адреса, которые относятся к классам 192.168.x.y или 172.16.x.y. Сети таких классов запрещены для использования в Internet (согласно документу RFC 1918) и зарезервированы для локальных сетей. Поэтому все IP-адреса примеров книги будут недействительными для Internet. Это сделано с целью скрыть настоящие IP-адреса компьютеров, которые подвергались зондированию или сканированию, и не допустить использования наших примеров для настоящих атак.

Сканирование Telnet

Внимательно рассмотрим следующий листинг, в котором представлен еще один пример сканирования. Вы видите что-то необычное?

```
scanner.se.45820 > 192.168.209.5.23: S 4195942931:4195942935(4) win 4096
scanner.se.45820 > 192.168.216.5.23: S 4195944723:4195944727(4) win 4096
scanner.se.52526 > 172.16.68.5.23: S 357331986:357331990(4) win 4096
scanner.se.45820 > 192.168.183.5.23: S 4196001810:4196001814(4) win 4096
scanner.se.52526 > 172.16.248.5.23: S 357312531:357312535(4) win 4096
scanner.se.45820 > 192.168.205.5.23: S 4196007442:4196007446(4) win 4096
scanner.se.52526 > 172.16.250.5.23: S 357313043: 357313047(4) win 4096
scanner.se.52526 > 172.16.198.5.23: S 357365266: 357365270(4) win 4096
scanner.se.52526 > 172.16.161.5.23: S 357355794: 357355798(4) win 4096
```

С хоста `scanner.se` выполняется сканирование компьютеров сетей `192.168` и `172.16`. При этом запрос отправляется на порт 23 (служба `telnet`). Можно сделать вывод, что осуществляется попытка выявить все хосты этих сетей, на которых запущена служба `telnet`. Но незначительная деталь дает основание подозревать нечто более хитрое. SYN-запросы обычно не содержат данных, а в данном случае передается по 4 байт данных (число в скобках).

Может показаться, что 4 байт данных, полученных до завершения полной процедуры согласования параметров соединения, будут просто отброшены. Но это не так. Эти четыре байта будут добавлены к информации, переданной после установки соединения. Согласно спецификации RFC 793 любые байты данных, которые были переданы во время процедуры согласования параметров соединения, добавляются к данным, передаваемым по завершению этой процедуры. Та-

ким образом хакер может обойти *системы обнаружения вторжений* (intrusion detection system – IDS), которые проверяют данные, получаемые только уже после завершения процедуры установки соединения.

Если в первом SYN-пакете, предназначенном для DNS-порта 53 вашего DNS-сервера, содержится 64 байт данных, то это не обязательно означает попытку атаки. Существует программное средство под названием 3DNS, которое предназначено для получения максимально быстрых ответов на запросы к Web-серверам. Одним из методов достижения этой цели является попытка измерить время ответа вашего DNS-сервера на запросы одного или нескольких Web-серверов, которые в свою очередь могут использоваться для ответа на запрос пользователя. При этом используется типичный размер запроса к Web-серверу – 64 байт. При обнаружении передачи таких пакетов не следует сразу подозревать проведение атаки, эти пакеты могут показаться надоедливыми или даже бесполезными, поскольку на многих узлах блокируются запросы к TCP-порту 53, но они не являются вредоносными.

Перехват TCP-сеанса

Хотя TCP производит впечатление достаточно защищенного протокола (ведь при организации сеанса должна быть выполнена полная процедура согласования параметров соединения, а обмен данными должен осуществляться по строгим правилам), но все же обольщаться не стоит. С помощью анализаторов пакетов злоумышленник может перехватывать данные TCP-сеансов или сеансов, организованных с помощью других протоколов. Анализаторы протоколов, установленные на некоммутируемых сетях, способны перехватывать любую информацию, которая передается в незашифрованном текстовом формате, например идентификаторы пользователей и пароли.

В программном обеспечении для перехвата сеансов, например Hunt, используется другой принцип несанкционированного получения данных TCP-сеансов. Такие средства пытаются вмешаться в установленный TCP-сеанс и организовать передачу данных этого сеанса через хост злоумышленника. Дело в том, что при обмене данными по протоколу TCP не выполняется никакой аутентификации или проверки подлинности взаимодействующих хостов. Все эти проверки выполняются на этапе установления соединения. После этого для доставки данных конкретному хосту используется только следующая информация.

- **IP-адрес**. Использованные при установлении соединения IP-адреса не могут меняться в ходе сеанса.
- **Номера портов**. Большинство протоколов не позволяют изменять номера портов отправителя и получателя в ходе сеанса.
- **Порядковые номера**. Порядковые номера должны изменяться в соответствии с ISN-номером и общим количеством байтов данных, переданных с одного хоста на другой.
- **Номера подтверждения**. Изменение номеров подтверждения должно осуществляться в соответствии с полученными порядковыми номерами и общим количеством байтов данных, переданных с одного хоста на другой.

Если злоумышленник способен наблюдать за передающимися данными и вмешается в установленное соединение, учитывая все параметры подтверждения, которые используется при обмене пакетами, то он осуществит перехват сеанса. Вообразите весь возможный ущерб, если будет перехвачен сеанс, обладающий привилегиями суперпользователя (root). Задача перехвата сеанса довольно трудна и требует особых навыков, но применение Hunt и других подобных средств позволяет ее упростить.

Резюме

Существует огромное и постоянно увеличивающееся количество средств обеспечения безопасности, которые можно применить для контроля за работой сети. При выборе конкретного средства лучше остановится на том, которое обеспечивает объем выдаваемой информации, не меньший, чем у TCPdump. Безусловно, отчеты TCPdump не отличаются особенной красотой, но предоставляют достаточный объем сведений для квалифицированной оценки передающегося трафика. Если выбрать более удобное, но менее информативное средство, можно пропустить что-то важное.

Протокол TCP используется для приложений, в которых необходима надежная доставка данных. Операции обмена данными по TCP выполняются по стандартной процедуре организации сеанса, возможной передачи данных и завершения соединения с использованием механизмов для подтверждения доставки и получения данных. Анализируя TCP-трафик с помощью TCPdump, при желании или необходимости можно ознакомится с более подробной информацией, что позволит подтвердить или опровергнуть предварительные выводы о характере осуществляющихся операций.

TCP чрезвычайно сложный и одновременно надежный протокол, и для его вредоносного использования тоже применяются не менее сложные методы. Поэтому при анализе TCP-трафика требуется тщательная проверка всех выполняемых операций, которые кажутся хоть немного необычными. Одновременно с усовершенствованием систем обнаружения вторжений (IDS) и брандмауэров хакеры совершенствуют свои методы незаметного проникновения в чужие системы. Для контроля за несанкционированным доступом очень важно хорошо разбираться в протоколе TCP, а TCPdump – прекрасное средство для изучения работы этого протокола.

3

Фрагментация пакетов

Нарушители используют фрагментацию на разных этапах своих атак с целью сокрытия своих действий и зондирования сетей, а также для запуска вредоносных программ. Некоторые системы обнаружения вторжений и фильтры пакетов не поддерживают проверки фрагментов пакетов или не выполняют их правильной сборки и поэтому не в состоянии выявить признаков атаки (а значит, и заблокировать опасный трафик), распределенных среди нескольких дейтаграмм. Атаки отказа в обслуживании основаны на использовании большого количества фрагментированных пакетов, что приводит к чрезмерной трате ресурсов атакованной системы. И это только несколько причин, по которым следует изучать фрагментацию пакетов — основную тему данной главы.

Четкое представление об этом аспекте работы протокола IP позволяет выполнять анализ трафика фрагментированных пакетов и выявлять фрагментацию пакетов, организованную с вредоносными целями. В нормальных условиях фрагментация часто используется для обмена данными между сетями с разными размерами максимальных единиц передачи данных (MTU). В начале этой главы рассмотрена теория создания и передачи по сети фрагментированных пакетов.

Основы фрагментации

Фрагментация используется при необходимости передачи IP-дейтаграммы через сеть, в которой максимально допустимая единица передачи данных (maximum transmission unit — MTU) меньше размера этой дейтаграммы. Например, MTU, или максимальный размер IP-дейтаграммы, для сетей Ethernet составляет 1500 байт. Если размер дейтаграммы превышает это значение, то на маршрутизаторе, который передает данную дейтаграмму в сеть Ethernet, должна быть выполнена ее фрагментация. Фрагментация также осуществляется при отправке локальным хостом дейтаграммы в сеть, если размер этой дейтаграммы превышает MTU сети.

Фрагменты передаются по адресу назначения, где хост-получатель должен выполнить их повторную сборку. На пути доставки фрагментированных пакетов они могут подвергаться дальнейшей фрагментации, если необходима их передача через сеть с еще меньшим MTU. Сама по себе фрагментация является абсолютно нормальным явлением, но существует возможность создания фрагментов пакетов, которые при их сборке на хосте-получателе позволят провести атакующие действия. Злоумышленники используют фрагментацию для того, чтобы избежать выявления атак системами обнаружения вторжений и фильтрующими маршрутизаторами, которые не способны определить вредоносный эффект отдельных фрагментов.

Какая информация должна быть сохранена во фрагментах пакетов для их успешной сборки в первоначальное состояние на хосте-получателе? Ответ на этот вопрос можно представить в виде набора следующих условий.

- Каждый фрагмент единой дейтаграммы должен быть связан с другим фрагментом с помощью общего для всех фрагментов идентификационного номера. Этот номер копируется из поля идентификации заголовка IP-дейтаграммы, и, если пакет фрагментирован, его называют идентификатором фрагмента (fragment ID).
- Каждый фрагмент должен сохранять информацию о своем месте, т.е. о смещении относительно исходного (нефрагментированного) пакета.
- В каждом фрагменте должен быть указан размер передаваемых в нем данных.
- Каждый фрагмент должен уведомлять о наличии следующих за ним фрагментов. Для этой цели служит флаг MF (More Fragments – следующий фрагмент).

Идентификационный номер IP (идентификатор фрагмента)

В заголовке всех IP-дейтаграмм содержится 16-битовое поле, которое служит в качестве уникального идентификатора отправленной хостом дейтаграммы. Обычно значение этого поля увеличивается на единицу для каждой новой дейтаграммы.

При фрагментации дейтаграммы все созданные фрагменты получают тот же идентификационный номер (идентификатор фрагмента). В следующей строке отчета TCPdump указан идентификатор нефрагментированного пакета, который имеет значение 202.

```
ping.com > 192.168.244.2: icmp: echo request (ttl 240, id 202)
```

Если эта дейтаграмма на пути следования к адресату подвергнется фрагментации, то идентификатор всех фрагментов будет иметь значение 202. Данная строка отчета TCPdump получена с помощью указания в командной строке параметра -vv. Этот параметр используется для получения расширенного отчета, в котором дополнительно показываются значения полей идентификатора и TTL (время жизни).

Все эти сведения содержатся в заголовке IP-дейтаграммы. После заголовка следует инкапсулированный фрагмент. Как уже указывалось, весь TCP/IP-трафик должен быть упакован в IP-дейтаграммы, так как именно протокол IP отвечает за доставку пакетов.

Фрагментация наглядно

При наглядной демонстрации процесса фрагментации (как говориться, "лучше один раз увидеть...") мы будем считать, что в качестве физической среды передачи данных используется сеть Ethernet. Рассмотрим сначала нефрагментированную дей-

таграмму (рис. 3.1). Как мы уже упоминали, максимальная единица передачи данных (MTU) в сети Ethernet равна 1500 байт. Каждая дейтаграмма должна иметь заголовок, стандартный размер которого составляет 20 байт, но может быть и больше при наличии параметров IP, например, маршрутизации от источника.

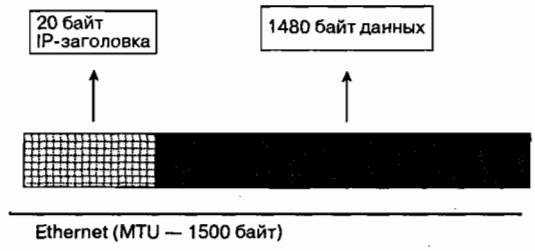


Рис. 3.1. Дейтаграмма для сети Ethernet

Напомним, что заголовок IP-дейтаграммы содержит сведения об IP-адресах отправителя и получателя. Такая информация считается "сетевой" составляющей дейтаграммы, так как маршрутизаторы используют эти сведения для передачи дейтаграммы по адресу назначения. Инкапсулированные данные сохраняются после IP-заголовка. Для их формирования могут использоваться различные протоколы транспортного уровня, такие как TCP, UDP или ICMP. Если, например, был использован протокол TCP, в 1480 байт данных содержится TCP-заголовок и TCP-данные.

На рис. 3.2 представлена схема дейтаграммы, размер которой составляет 4028 байт. Это эхо-запрос ICMP, который должен быть передан в сеть Ethernet (MTU равно 1500 байт). Данный эхо-запрос слишком велик и не характерен для нормального трафика. Здесь он используется только для иллюстрации фрагментации. Итак, дейтаграмму размером в 4028 байт необходимо разделить на несколько фрагментированных пакетов длиной до 1500 байт. При этом в каждом пакете нужно по 20 байт для IP-заголовка, поэтому для данных в каждом пакете остается по 1480 байт. На рис. 3.3 показана та же исходная дейтаграмма, но уже с указанием количества байтов, выделенных для каждого пакета при ее фрагментации. В следующих разделах подробно рассматривается каждый из трех созданных фрагментов.

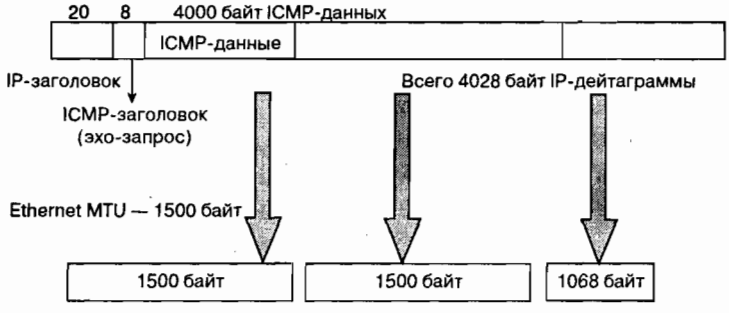


Рис. 3.2. Фрагментация исходной 4028-байтовой дейтаграммы

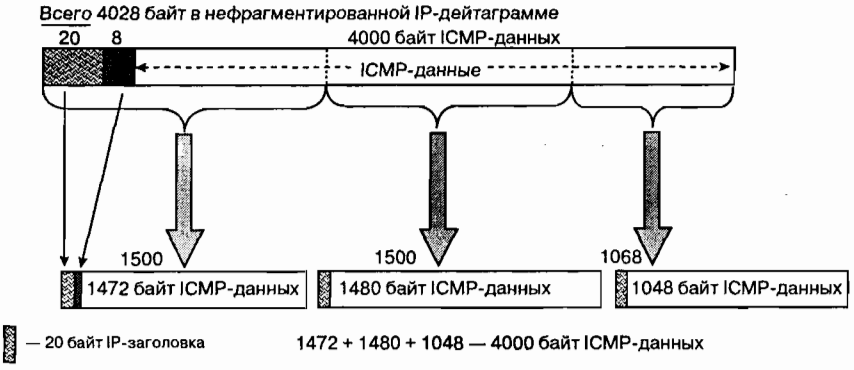


Рис. 3.3. Распределение байтов данных по фрагментам

Последовательность фрагментов

Рассмотрим первый фрагмент созданной последовательности (рис. 3.4). В качестве идентификатора и первого, и всех остальных фрагментов копируется "исходный" идентификатор IP-дейтаграммы.

Только в первом фрагменте будет содержаться заголовок ICMP-пакета. Этот заголовок не копируется во все следующие фрагменты, что очень важно. Смещение первого фрагмента равно 0, его длина – 1480 байт, он имеет 1472 байт данных и 8 байт заголовка ICMP-пакета. Установленный флаг MF (More Fragments) означает, что этот фрагмент не является конечным в последовательности фрагментов.

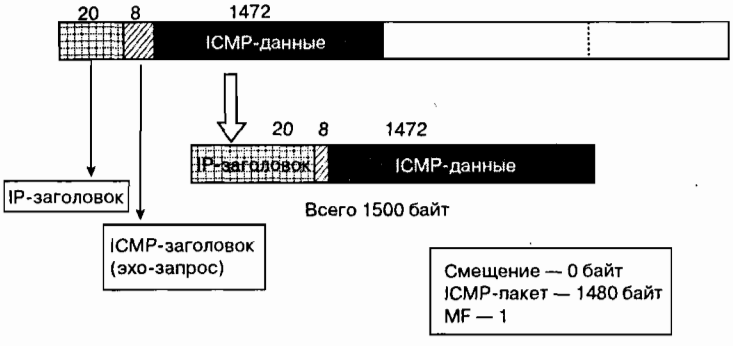


Рис. 3.4. Начальный фрагмент

На рис. 3.5 представлена структура начального фрагмента. Его первые 20 байт из общих 1500 байт являются IP-заголовком. Следующие 8 байт – заголовок ICMP-пакета. Не забывайте, что это эхо-запрос, и что в исходном пакете присутствовали 8 байт ICMP-заголовка. Оставшиеся 1472 байт начального фрагмента отведены для ICMP-данных.

Во всех фрагментах, кроме таких обычных полей заголовка IP-дейтаграммы, как IP-адреса отправителя и получателя, а также поля "Протокол" (в данном случае его значение определяет использование ICMP), имеются специальные поля, используемые при фрагментации. Идентификатор фрагмента (в данном случае 21223) свя-

зывает все фрагменты в единую последовательность. Поле More Fragments (MF) указывает на наличие или отсутствие следующих фрагментов. В начальном фрагменте этот флаг имеет значение 1. Кроме того, должно быть сохранено значение смещения данных этого фрагмента по отношению к расположению в исходной, нефрагментированной, дейтаграмме (для начального пакета это значение равно 0). И, наконец, длиной фрагмента считается длина передающихся в нем данных. В данном случае длина фрагмента составляет 1480 байт: 8-байтовый ICMP-заголовок плюс 1472 байт ICMP-данных.

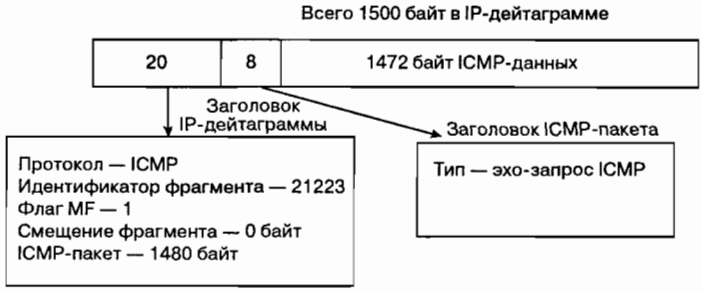


Рис. 3.5. Содержимое начального пакета

Средний фрагмент

Рассмотрим следующий фрагмент нашей последовательности (рис. 3.6). Его IP-заголовок скопирован из заголовка "исходной" дейтаграммы. В этот новый IP-заголовок также копируется бо́льшая часть других исходных данных (например, IP-адреса отправителя и получателя). После заголовка размещаются 1480 байт ICMP-данных. Смещение второго фрагмента равно 1480 байт, а его длина имеет такой же размер. Поскольку этот фрагмент не является последним, установлен флаг MF.

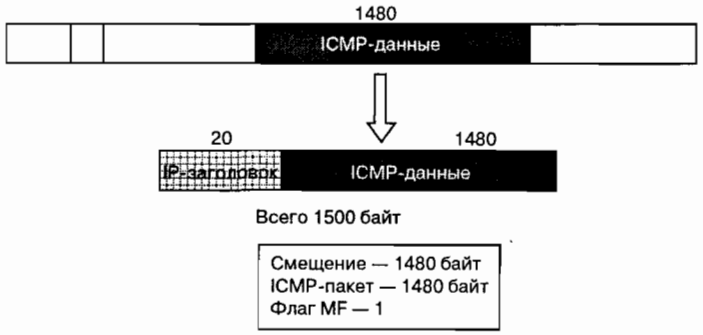


Рис. 3.6. Средний фрагмент

Продолжим изучение фрагментации и рассмотрим IP-дейтаграмму, использующуюся для передачи второго фрагмента (рис. 3.7). Как и для всех фрагментов единой последовательности в этой дейтаграмме используется 20-байтовый заголовок. В заголовке указано использование протокола ICMP, идентификатор

фрагмента по-прежнему равен 21223, а установленный флаг MF указывает на наличие следующего пакета. Смещение равно 1480 байт по отношению к первому байту данных в нефрагментированной дейтаграмме (первые 1480 байт данных были отправлены в начальном фрагменте). Длина фрагмента также составляет 1480 байт, которые целиком являются ICMP-данными.

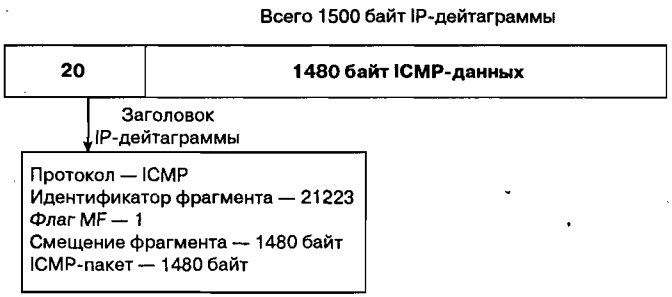


Рис. 3.7. Содержимое среднего фрагмента

Стоит повторить, что ICMP-заголовок из первого фрагмента не копируется во все остальные фрагменты с ICMP-данными. Следовательно, по этому одному отдельному фрагменту нельзя сделать вывод о типе ICMP-сообщения (в данном случае это эхо-запрос). Данный аспект имеет особое значение для работы фильтров пакетов.

Последний фрагмент

На рис. 3.8 изображен последний фрагмент нашей последовательности. Как и в других фрагментах, в заголовок его IP-дейтаграммы копируются значения полей исходной, нефрагментированной, дейтаграммы, например идентификатор фрагмента. После заголовка добавляются последние 1048 байт ICMP-данных. В третьем фрагменте смещение имеет значение 2960 байт, а длина фрагмента равна 1048 байт. Так как этот фрагмент является последним, то для флага MF установлено значение 0.

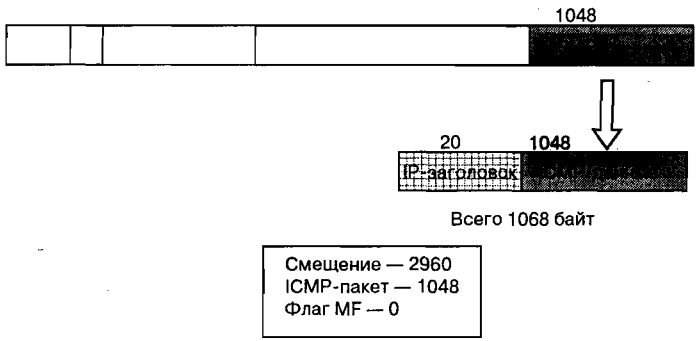


Рис. 3.8. Последний фрагмент

Рассмотрим IP-дейтаграмму последнего фрагмента (рис. 3.9). Для IP-заголовка снова зарезервировано 20 байт. В оставшейся части фрагмента содержится последняя часть ICMP-данных. Идентификатор фрагмента имеет значение 21223, а

флаг MF не установлен, так как этот фрагмент – последний. Смещение составляет 2960 байт (сумма длин двух предыдущих 1480-байтовых фрагментов). В этом фрагменте передается только 1048 байт данных, причем все эти данные – остаток данных ICMP-сообщения. В последнем фрагменте, как и во втором, не содержится ICMP-заголовок.

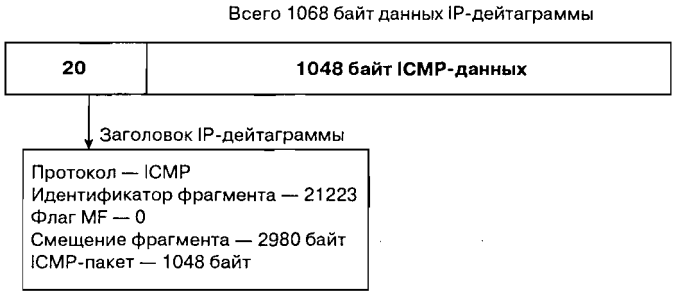


Рис. 3.9. Содержимое последнего фрагмента

Фрагментация в отчетах TCPdump

Посмотрим на следующий отчет TCPdump. Три сохраненные записи соответствуют передаче по сети трех фрагментов, рассмотренных выше. Это означает, что на хосте, на котором запущен TCPdump, проведена сборка фрагментированного эхо-запроса.

```
ping.com > myhost.com: icmp: echo request (frag 21223:1480@0+)
ping.com > myhost.com: icmp: (frag 21223:1480@1480+)
ping.com > myhost.com: icmp: (frag 21223:1480@2960)
```

Первая строка содержит запись об отправке хостом `ping.com` эхо-запроса хосту `myhost.com`. Анализатор TCPdump смог распознать в этом фрагменте эхо-запрос, так как тот содержал 8-байтовый ICMP-заголовок. Теперь обратимся к обозначению фрагментации, содержащемуся в правой части строки. В TCPdump для указания фрагментированного трафика используется слово *frag*, после которого установлены идентификатор фрагмента (в данном случае `21223`) и двоеточие. Далее указана длина данного фрагмента (`1480`), символ `@` и значение смещения данных (для первого пакета равно 0). Знак плюс (+) означает, что установлен флаг MF. Таким образом, по этому фрагменту можно определить предназначение "исходной" дейтаграммы, а также, что существуют следующие фрагменты, но не известно их количество и содержимое.

Вторая запись незначительно отличается от первой. Обратите внимание на то, что в ней отсутствует указание на эхо-запрос. Это объясняется тем, что в данном фрагменте нет ICMP-заголовка, и анализатор пакетов не в состоянии определить тип ICMP-трафика. В поле "Протокол" заголовка IP-дейтаграммы по-прежнему содержится значение, указывающее на протокол ICMP, но это вся информация, предоставляемая данным фрагментом. Во второй строке отчета TCPdump также указан идентификатор фрагмента (`21223`), длина его данных (`1480`) и смещение (`1480`). Знак плюс указывает на установку во фрагменте флага MF. Этот фрагмент участвует

в общем процессе передачи данных, но сам ничего не знает о предназначении этих данных (чем-то напоминает первокурсника, правда?).

Последняя строка практически не отличается от второй записи. Тот же идентификатор фрагмента (21223), затем длина фрагмента (1048) и смещение (2960). Единственным отличием является отсутствие установленного флага MF, но это вполне ожидаемо для последнего фрагмента. Этот фрагмент также не предоставляет никаких сведений о предназначении его данных.

Как хранится значение смещения данных

Хотя TCPdump вычисляет и выдает готовое значение смещения данных фрагмента, но в самом пакете хранится другое значение. Если необходимо узнать значение смещения данных в байтах, например, при изучении отчета TCPdump в шестнадцатеричном формате, то для получения конечного результата придется выполнить математическое действие.

Значение смещения данных фрагмента занимает часть шестого байта и весь седьмой байт заголовка IP-дейтаграммы. Для его хранения используется 13-битовое поле, максимальное значение которого составляет 8191 ($2^{13}-1$). Теоретически, хотя и в исключительно редких случаях нормальной фрагментации пакетов, это значение может превышать число 8191, так как максимальный размер дейтаграммы составляет 65535 ($2^{16}-1$) байт. Для получения десятичного значения смещения данных в байтах хранящееся в этом поле число следует умножить на 8. Математическое толкование данного действия (тем, кому это интересно) выглядит следующим образом: 65536 (2^{16}) разделить на 8192 (2^{13}) равняется 8.

Фрагментация и фильтры пакетов

В этом разделе описывается влияние фрагментации на работу устройств, обеспечивающих фильтрацию пакетов, например маршрутизаторов или брандмауэров. Основная проблема возникает при необходимости блокирования фрагментированного трафика.

Поскольку заголовок любого из пакетов транспортного уровня (TCP, UDP или ICMP) содержится только в первом фрагменте, то и на устройствах, которые не отслеживают содержимое заголовка IP-пакета, будет заблокирован только этот первый пакет последовательности фрагментов. В данном случае, казалось бы, должны блокироваться все фрагменты, идентификаторы которых совпадают с заблокированным фрагментом. На самом деле некоторые устройства фильтрации пакетов этого не выполняют. Они рассматривают каждый пакет отдельно и никак не сопоставляют его с предыдущими или последующими пакетами. Если сразу видно, что такой поход имеет очевидные недостатки, то зачем он вообще используется? Здесь стоит задуматься о накладных расходах, связанных с хранением информации о содержимом пакета. Ведь каждый фрагмент должен быть рассмотрен и сохранен, это требует значительных затрат времени, ресурсов процессора и памяти. К тому же требуется принятие решения о разрешении или запрещении прохождения пакетов, что также весьма накладно. Поэтому гораздо проще рассматривать каждый пакет отдельно.

Если конкретный пакет не удовлетворяет условиям блокирования, например, из-за отсутствия заголовка вложенного протокола, то он пропускается фильтром пакетов. Фрагментированные TCP- и UDP-пакеты содержат заголовки только в первом фрагменте созданной последовательности. Часто решение о блокировании пакета принимается только на основании информации заголовка о порте получателя. Это означает, что фрагментированные TCP- и UDP-пакеты могут использоваться для обмана несовершенных устройств фильтрации пакетов.

Тут важен еще один момент. Протокол IP не является надежным, и вполне вероятна ситуация потери первого фрагмента, в котором содержится информация заголовка вложенного протокола. В этом случае задача фильтра пакетов по блокированию или разрешению прохождения трафика усложняется. Ведь если один фрагмент утерян, вся последовательность фрагментов должна быть отправлена повторно.

Флаг DF

В некоторых строках отчетов TCPdump можно увидеть буквы DF, заключенные в круглые скобки. Это означает, что для данного пакета установлен флаг DF (Don't Fragment – "Без фрагментации"), запрещающий его фрагментацию. Если дейтаграмма, для которой установлен флаг DF, должна пройти по сети, требующей фрагментации, то маршрутизатор отбрасывает эту дейтаграмму и возвращает отправителю ICMP-сообщение об ошибке.

В ICMP-сообщении об ошибке указывается значение MTU сети, в которой требуется фрагментация дейтаграммы. Некоторые хосты намеренно отправляют пробную дейтаграмму с установленным флагом DF, чтобы узнать минимальный MTU на пути к получателю. Если будет получено сообщение об ошибке с указанием MTU, то хост-отправитель уменьшит размер отправляемых пакетов до значения, которое позволит избежать фрагментации. Этот метод часто используется для TCP-трафика, так как применение протокола TCP требует значительной траты ресурсов. Одной из причин, по которым применение фрагментации нежелательно, является возможное снижение эффективности передачи данных – при потере одного фрагмента повторно должна быть отправлена вся последовательность. Можно догадаться, что злоумышленник тоже может применить данный метод для определения MTU сегмента сети. Это поможет ему впоследствии провести атаку с использованием фрагментированных пакетов. Нарушитель может создавать пакеты разной длины с установленным флагом DF и смотреть, когда появляется ICMP-сообщение об ошибке. Здесь предполагается, что атакуемая сеть не блокирует возвращение ICMP-сообщений об ошибках. В следующей строке отчета TCPdump содержится ICMP-сообщение о необходимости фрагментации пакета, для которого установлен флаг DF.

```
router.ru > mail.mysite.com: icmp: Host.ru unreachable - need to frag
↳(mtu 308) (DF)
```

Причиной появления этого сообщения явилась попытка отправки хостом `mail.mysite.com` хосту `host.ru` дейтаграммы с установленным флагом DF, размер которой превышает 308 байт. При прохождении этой дейтаграммы по сети к получателю хост `router.ru` обнаружил, что для следующего сегмента сети значение MTU составляет 308 байт, и без фрагментации дальнейшая передача дейтаграммы невозможна. Хост `router.ru`, определив, что для дейтаграммы установлен флаг DF, возвращает отправителю ICMP-сообщение, уведомляющее о проблеме. Таким образом, хост `mail.mysite.com` должен или создать дейтаграммы с размером меньше 308 байт (и фрагментация не потребуется) или сбросить флаг DF (и тогда будет проведена фрагментация на пути следования), а затем повторно отправить свою дейтаграмму.

Вредоносная фрагментация

Аналитику, отвечающему за безопасность сетей, не придется отдыхать, если при атаке на его сети будут использованы фрагментированные пакеты. Фрагментация предоставила обширное поле деятельности для хакеров, что позволило им придумать огромное количество довольно опасных атак.

Этот совет еще не раз повторяется в других местах книги при изучении разных протоколов: будьте особо внимательны и соблюдайте максимальную осторожность при анализе фрагментированного трафика. Я знаю многих известных специалистов в области безопасности, которых осмеяли и обозвали параноиками, подозревающими каждого человека в намерении тем или иным способом атаковать их сети. Что ж, я предлагаю присоединиться к этой группе "параноиков", когда дело касается фрагментации. Если система обнаружения вторжений не позволяет выполнять тщательное исследование фрагментированных пакетов, то можно пропустить атакующие действия злоумышленника. Если же эта система способна определять состояние фрагментов, осуществлять их повторную сборку, а затем делать выводы об их безопасности, то, значит, защищаемая сеть находится под надежной охраной.

В приложении Б, "Отказ в обслуживании", описана одна из наиболее опасных атак отказа в обслуживании, при которых используется фрагментация, — атака Ping of Death. В следующих разделах рассмотрены несколько других атак с применением фрагментации.

Фрагментация заголовков TCP-пакетов

Программа nmap является прекрасным средством сканирования. Она работает на многих платформах UNIX и доступна для загрузки по адресу `www.insecure.org/nmap`. Программа nmap осуществляет сканирование портов интересующего компьютера с целью выявления открытых портов, а также позволяет осуществлять скрытое (stealth) сканирование, усложняющее задачу выявления сканирования системами обнаружения вторжений. С помощью специального параметра командной строки (`-f`) nmap разбивает 20-байтовые заголовки TCP-пакетов на несколько фрагментов с целью избежать их обнаружения. Например, пусть будет выполнена следующая команда.

```
nmap -f -sS -p 53  target.com
```

Эта команда приведет к фрагментации SYN-пакета, служащего для установки соединения с портом 53 хоста `target.com`. Рассмотрим отчет TCPdump о результатах сканирования с помощью фрагментации TCP-заголовка.

```
truncated-tcp 16 (frag 25096:16@0+)
fragger.org > target.com: (frag 25096:4@16)
truncated-tcp 16 (frag 4265:16@0+)
fragger.org > target.com: (frag 4265:4@16)
truncated-tcp 16 (frag 34927:16@0+)
fragger.org > target.com: (frag 34927:4@16)
```

В данном случае хостом `fragger.org` *с помощью стандартного* SYN-*запроса выполнялось сканирование порта* 53 *хоста* `target.com`. Но выявить сканирование не очень просто из-за того, что передаются небольшие фрагменты SYN-пакета.

В первой строке отчета содержится сообщение о получении 16 байт усеченного (truncated) TCP-пакета. Минимальный заголовок TCP-пакета равен 20 байт, поэтому TCPdump сообщает о получении 16-байтового пакета с помощью выражения `truncated-tcp`. В следующем фрагменте содержатся оставшиеся 4 байт TCP-заголовка. Вполне вероятно, что система обнаружения вторжений ничего не выявит и не сообщит о подобном скрытом сканировании.

Teardrop

Ознакомившись с теоретическими основами процесса фрагментации, рассмотрим следующий отчет TCPdump. Можете ли вы самостоятельно обнаружить опасность фрагментации, связанную с использованием атакующей программы Teardrop?

```
evilfrag.com.139 > target.net.139: udp 28 (frag 242:36@0+)
evilfrag.com > target.net: (frag 242:4@24)
```

Первым полученным фрагментом является UDP-пакет, идентификатор которого 242, длина – 36 байт и смещение – 0 байт. На рис. 3.10 этот фрагмент представлен в виде заштрихованного прямоугольника. Первый фрагмент начинается с байта 0 и заканчивается 35-м байтом включительно.

Теперь обратимся ко второму фрагменту. Он связан с первым, так как его идентификатор тоже имеет значение 242. Длина второго фрагмента – 4 байт, а смещение составляет 24 байт. Второй фрагмент выделен на рис. 3.10 сплошной заливкой. Таким образом, происходит замещение 4 байт (с 24 по 27 включительно) первого фрагмента.

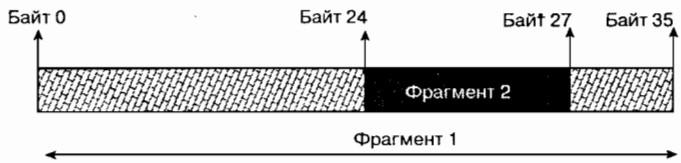


Рис. 3.10. Вредоносные фрагменты Teardrop

Атака Teardrop использует слабое место процесса сборки фрагментов на хосте-получателе. Эта программа создает фрагменты, в которых указанное во фрагментах смещение приводит к затиранию доставленных ранее данных. При сборке таких фрагментов на хосте-получателе некоторые системы могут выйти из строя, зависнуть или начать перезагрузку. Эта атака была впервые применена в 1997 году и ярко продемонстрировала, какой ущерб могут наносить специально подготовленные фрагменты.

Некорректная или неполная последовательность фрагментов по-прежнему опасна для некоторых хостов. Совсем недавно появилась программа Jolt2 (более подробно она рассматривается в главе 5, "Воздействия и реакции"), которая с помощью отправки фрагментов с ненулевым смещением на хосты под управлением Windows (вплоть до Windows 2000) может вызывать отказ в обслуживании из-за возникшей нехватки ресурсов.

Такое большое количество проблем объясняется тем, что хосты, маршрутизаторы и системы обнаружения вторжений должны учитывать множество аспектов

фрагментации. Во-первых, следует проверить, все ли фрагменты последовательности получены. Во-вторых, необходимо проконтролировать, чтобы все фрагменты были правильно сформатированы (не было замещения одних данных другими), и в совокупности размер данных всех фрагментов не превышал максимально допустимый размер дейтаграммы — 65535 байт. И последнее, заголовки пакетов не должны передаваться по частям в различных фрагментах. Выполнить все перечисленные проверки нелегко, и для решения этой задачи требуется сборка пакетов и обнаружение вредоносных изменений, что в свою очередь связано с выделением памяти и ресурсов процессора. Если же к данной задаче не подойти с максимальной ответственностью, то вероятно успешное проведение атак отказа в обслуживании или возникновение других проблем.

Анализ фрагментированного трафика

Хотите верьте, хотите — нет, но процесс фрагментации не такой уж сложный, если знать его теоретические основы и разбираться в обозначениях, используемых при его регистрации. Работая специалистом по вопросам безопасности и анализируя отчеты TCPdump, я много раз в уме решал задачу "что же не так с этим фрагментированным трафиком?". Здесь требуется нечто большее, чем обычные академические знания; теоретическая база для анализа трафика, проходящего по конкретной сети, должна обязательно сочетаться с мерами предосторожности против возможных атак, в которых используется фрагментация.

Обнаружение какого-то нового неизвестного способа вредоносной фрагментации вызывает естественное и вполне заслуженное чувство триумфа. Но нужно понять, что обнаружение это только первый этап разгадки фокуса. Следующим шагом является понимание истинной цели атаки, осуществляемой с помощью фрагментации. Наиболее распространенным случаем является атака отказа в обслуживании, например, блокирование работы определенной службы или полный вывод из строя атакуемого хоста. Среди других причин проведения подобных атак можно назвать попытку избежать обнаружения и обойти несовершенные устройства фильтрации пакетов и системы обнаружения вторжений. Для того чтобы сделать какие-либо выводы, придется проверить всю сеть в целом и конкретно атакованный хост.

И в заключение, если вы считаете свой узел надежно защищенным на внешнем периметре и при этом не используете полнофункциональный брандмауэр или фильтрующий маршрутизатор, то я рекомендую быть осторожнее! При такой дыре в линии обороны не будет ничего необычного, если даже неопытный злоумышленник сможет взломать столь слабую защиту.

Резюме

Процесс стандартной фрагментации заключается в разбиении и упаковке исходной дейтаграммы в новые пакеты, размер которых будет меньше или равен значению MTU "узкого" сегмента сети. Каждый новый фрагмент представляет собой отдельный пакет со своим собственным IP-заголовком. Значения полей этого заголовка в основном копируются из исходной нефрагментированной дейтаграммы (например, IP-адреса, идентификатора и т.д.). Но каждый новый фрагмент также должен содержать и определенную уникальную информацию, например смещение данных, количество байтов данных во фрагменте и сведения о наличии следующих фрагментов.

Существует множество видов использования фрагментации в неблаговидных целях. Целью злоумышленников могут быть проведение атаки отказа в

обслуживании или передача в сеть такого трафика, который в нормальном, нефрагментированном виде был бы заблокирован. Некоторые несовершенные устройства фильтрации пакетов не в состоянии обнаружить криминала в отдельных фрагментах и разрешают прохождение вредоносных фрагментов. Обладая общим представлением о процессе фрагментации, значительно легче обнаруживать ее некорректное использование и отличать его от нормальной фрагментации.

4

Протокол ICMP

Протокол ICMP (Internet Control Message Protocol – протокол управляющих сообщений Internet) был задуман как безопасное средство для уведомления об ошибках, а также для выдачи несложных запросов и ответа на них. Возможно, именно из-за простоты его реализации современные способы использования ICMP в собственных корыстных целях кажутся особенно возмутительными. В своем естественном виде ICMP является сравнительно простым и строгим протоколом, но с помощью некоторых модификаций может служить для осуществления коварных планов злоумышленников. Поэтому особенно важно уметь различать нормальное и нестандартное использование этого протокола.

В этой главе рассмотрено несколько аспектов работы протокола ICMP. После изложения теоретических основ будет рассказано о том, как ICMP применяется для поиска активных компьютеров интересующей сети. Затем читатели узнают, каким образом можно различить нормальное и нестандартное использование ICMP-сообщений. Полученные теоретические сведения закрепляются практическими примерами анализа необычного ICMP-трафика. В заключение главы изложены методы блокирования входящих ICMP-сообщений и рассматриваются преимущества и недостатки этого блокирования.

Теория ICMP

До того как изучать примеры ICMP-сообщений, рассмотрим несколько общих принципов работы протокола ICMP. Те, кто считают, что достаточно осведомлены о теории ICMP, могут сразу же переходить к разделу "Методы составления схемы сети".

Зачем нужен ICMP

Как говорилось в главе 2, "TCPdump и TCP", TCP является протоколом, ориентированным на установление соединений, и для надежной доставки TCP-

сегментов требуются значительные ресурсы вычислительных систем. Протокол UDP не ориентирован на установление соединений и не гарантирует надежной доставки данных. Для работы обоих протоколов необходим свободный порт сервера, с которым может взаимодействовать клиент.

Для простого запроса, например, для проверки активности определенного компьютера, которую обычно называют *эхо-запросом* (или ping-запросом), не требуется наличие свободных портов, а надежность доставки данных не обязательна. Для отправки подобных несложных запросов и получения ответов на них как раз и применяется протокол ICMP.

Кроме того, как маршрутизаторы или хосты должны уведомлять отправителя о возникновении той или иной ошибки при доставке сообщения? Достаточно надежный протокол TCP способен указывать причины некоторых ошибок, например, отправку данных на закрытый порт с помощью возвращения TCP-пакетов с установленными флагами RESET и ACK. Если TCP-клиент или сервер получает слишком много информации, то предусмотрен механизм закрытия буфера входящих сообщений с помощью уменьшения размера окна до значения 0. Это будет указывать на неспособность хоста-получателя принимать следующие порции данных, пока не будут обработаны данные из буфера.

Однако протоколы UDP и IP не обладают подобными возможностями уведомления об аномальных ситуациях. Если UDP-порт получателя не ожидает запросов или на этот порт передается слишком много данных, то в UDP не предусмотрено метода уведомления об ошибке. Вот здесь и приходит на помощь ICMP. Он обеспечивает простой метод обмена служебной информацией между двумя хостами или хостом и маршрутизатором при возникновении каких-либо ошибок.

Область применения ICMP

Рассмотренная в главе 1, "Фундамент Internet", многоуровневая модель стека протоколов TCP/IP определяет обмен данными между двумя хостами на различных уровнях (рис. 4.1).

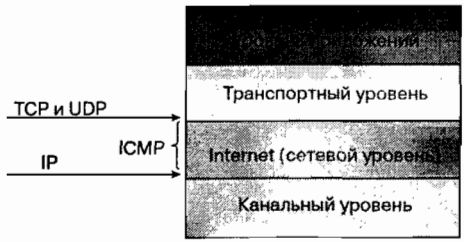


Рис. 4.1. Многоуровневая модель стека TCP/IP

На самом высоком уровне работают TCP/IP-приложения, например `telnet`. На следующем транспортном уровне с помощью протоколов TCP и UDP осуществляется взаимодействие между хостами. Ниже расположен уровень Internet (сетевой), на котором обеспечивается доставка дейтаграмм по адресу назначения. Последним уровнем стека протоколов TCP/IP является канальный уровень, ответственный за физическую доставку дейтаграммы по сети.

Как мы видим, протокол ICMP работает на том же сетевом уровне, что и IP. При этом ICMP-сообщение инкапсулируется в IP-дейтаграмму и записывается после IP-заголовка.

Свойства ICMP

Протокол ICMP отличается от TCP и UDP рядом свойств. Во-первых, в ICMP не используются номера портов, как это принято в протоколах транспортного уровня (TCP или UDP). Для того чтобы различать службы, в ICMP указывается только тип передаваемого сообщения и код, содержащийся в первых двух байтах заголовка ICMP-пакета. По этим байтам можно определить предназначение конкретного ICMP-сообщения.

Типы ICMP-сообщений

Перечисление и изучение всех возможных ICMP-сообщений выходит за рамки данной книги. Тем, кто хочет получить более подробную информацию по данной теме, рекомендуем обратится по адресу `www.iana.org/assignments/icmp-parameters`.

Для ICMP практически не существует понятий "клиент" или "сервер". Действительно, при доставке ICMP-сообщений об ошибке хост-получатель может учитывать это уведомление, но ничего не сообщать об этом отправителю. Кроме того, ICMP не предоставляет никаких гарантий доставки сообщений.

Одна из необычных особенностей ICMP заключается в том, что нет необходимости в работе определенных служб и ожидании запросов к соответствующим портам. На эхо-запрос ICMP способны отвечать практически все операционные системы, и отключить используемый по умолчанию режим обязательного ответа на эхо-запрос совсем не просто.

Еще одной уникальной возможностью ICMP является поддержка широковещательного трафика. Протокол TCP требует установления единственного соединения клиент-сервер, а ICMP позволяет отправлять данные сразу нескольким адресатам. Как будет показано в разделе "Атака Smurf", эта возможность может быть успешно использована злоумышленниками.

Два взаимодействующих хоста используют ICMP для выдачи несложных запросов и получения ответов, а также для уведомления друг друга о каких-либо ошибках. Например, возможна ситуация, когда хост-получатель не в состоянии принимать предназначенный ему трафик на определенной скорости. С помощью ICMP-сообщения этот хост может уведомить отправителя о необходимости снижения скорости отправки данных.

Протокол ICMP используется маршрутизаторами в качестве механизма уведомления отправителя о возникновении проблем при доставке сообщений. Например, маршрутизатор способен выдать ICMP-сообщение "запрещено администратором". Это сообщение уведомляет отправителя, что отправленный им трафик запрещен для пересылки через интерфейс маршрутизатора согласно правилу, присутствующему в списке контроля доступа.

В данном случае очевидно, что сообщение отправляет маршрутизатор, так как именно он запрещает выполнение операции. Но маршрутизаторы также уведомляют отправителя об ошибках в случае невозможности доставить сообщение указанному адресату. Например, если хост-получатель недостижим, то, очевидно, он не может ответить на запрос. Вместо него отвечает маршрутизатор.

Хотя уведомления маршрутизаторов предназначены для информирования отправителя о проблемах и устранения ошибок, но эти же сообщения могут применяться в разведывательных операциях нарушителей. Отправитель может собрать сведения, касающиеся того, о каких действиях уведомляет маршрутизатор. Одним из хороших правил безопасности является запрет отправки ICMP-уведомлений маршрутизатора о недостижимости хостов, что позволяет предотвратить нежелательную рассылку информации (этот вопрос рассматривается подробнее в разделе "Хост недостижим".

Резюме теоретических сведений

Подведем итог изложенной информации. Протокол ICMP применяется как средство обмена сообщениями об ошибках. ICMP-сообщение встраивается в заголовок IP-дейтаграммы, но принадлежит тому же уровню стека протоколов, что и IP.

ICMP является уникальным протоколом, так как для доставки сообщений не использует портов. Допускается потеря ICMP-сообщений. ICMP-сообщения могут быть широковещательными, так как единственное соединение с определенным хостом не устанавливается.

Отправителями ICMP-сообщений могут быть как хосты, так и маршрутизаторы. Хосты готовы к приему ICMP-сообщений и большинство из них возвращают ответы, если только такой режим не был специально отключен.

Методы составления схемы сети

Составление схемы сети является одним из важнейших стратегических этапов любой тщательно спланированной атаки. Это – разведка боем с целью выявить работающие хосты в интересующей сети. После составления схемы сети хакер может сосредоточиться на сканировании или атаке только активных хостов.

Атака злоумышленника, который не составил схемы сети, приведет к резкому увеличению сетевого трафика, т.е. не будет действительно эффективной. В последнем квартале 1999 года была проведена подобная атака сканирования, которая напомнила поговорку "как слон в посудной лавке". Троянская программа RingZero, поражавшая компьютеры под управлением Windows, выполняла сканирование хостов на предмет открытых портов proxy-серверов. Основным недостатком ее работы было сканирование случайных хостов, работающих в различных сетях. Поэтому выполнялось сканирование как по действительным, так и по несуществующим IP-адресам. Такие действия быстро выявлялись системами обнаружения вторжений. Кроме того, для извлечения полезной информации о проведенном сканировании приходилось выполнять большой объем лишней работы. Если бы атака была более направленной, и все сканируемые IP-адреса принадлежали бы активным компьютерам, то нарушитель, несомненно, добился бы бо́льших результатов.

Троянская программа RingZero

В зарегистрированной атаке RingZero на защищенную сеть было обнаружено, что сканирование проводилось по множеству IP-адресов, причем сканировались большей частью неактивные TCP-порты: 3128 (proxy-сервер Squid), 80 (стандартный HTTP-порт) и 8080 (альтернативный HTTP-порт). В наблюдаемой сети класса B выполнялось около шести таких сканирований каждый час. Сообщения о выявлении данной атаки поступили от множества других узлов Internet.

Первым предположением было, что при данной атаке были использованы подложные IP-адреса, а цель ее неизвестна. Но, Рон Маркум (Ron Marcum), системный администратор Университета Вандербильта, обнаружил в своей сети компьютер под управлением Windows, выполняющий данное сканирование. Процессом сканирования управляла программа под именем RingZero. Программа RingZero была подробно рассмотрена на конференции SANS (System Administration, Networking and Security — системное администрирование, работа в сети и безопасность), проходившей в октябре 1999 года.

После запуска в сети, выбранной для эксперимента, хост, на котором была инсталлирована программа RingZero, начал сканировать другие компьютеры с целью обнаружения открытых портов proxy-серверов. При этом IP-адреса выбирались случайно. После обнаружения интересующих портов вся полученная информация возвращалась назад на FTP-сервер, который собирал ее в единое целое для злоумышленника. По всей видимости, собранные сведения предназначались для какой-то будущей атаки. До настоящего времени отмечаются случаи проведения сканирования с помощью RingZero, и по-прежнему остается неясным способ заражения этим троянцем и принцип, по которому выбираются IP-адреса сканируемых хостов.

Одним из наиболее популярных методов составления схемы сети является применение эхо-запросов ICMP. Хост, который ответил на эхо-запрос, явным образом сигнализирует о своей активности. На этом и построен принцип использования команды `ping`, с ее помощью отправляется эхо-запрос, на который должен быть возвращен эхо-ответ. Многие системные администраторы защищаются от этой функции протокола ICMP, блокируя получение эхо-запросов. Это неплохой и даже нужный метод защиты, но он не является универсальным решением, а только усложняет задачу настойчивого злоумышленника. Блокирование эхо-запросов стимулировало хакеров на изобретение других методов сканирования, реализуемых с помощью других протоколов.

В разделе "Сканирование с помощью ACK-пакетов" главы 2, "TCPdump и TCP", показано, как с помощью TCP-пакетов с установленным флагом ACK можно выявить работающие хосты. Этот метод – достойная альтернатива сканированию с помощью эхо-запросов. В следующих разделах рассмотрен ряд обычных и нестандартных методов сканирования.

Неутомимый составитель схем

Ниже представлен отчет о классическом сканировании для составления схем сетей с помощью отправления отдельных эхо-запросов ICMP всем хостам определенной подсети. В данном случае сканируется сеть класса C `192.168.117`. Выявить такое сканирование не составляет труда.

```
00:05:58.560000 scanner.net > 192.168.117.233: icmp: echo request
00:06:01.880000 scanner.net > 192.168.117.139: icmp: echo request
00:12:45.830000 scanner.net > 192.168.117.63: icmp: echo request
00:15:36.210000 scanner.net > 192.168.117.242: icmp: echo request
00:15:58.600000 scanner.net > 192.168.117.129: icmp: echo request
00:18:51.650000 scanner.net > 192.168.117.98: icmp: echo request
00:20:42.750000 scanner.net > 192.168.117.177: icmp: echo request
00:26:36.680000 scanner.net > 192.168.117.218: icmp: echo request
00:05:58.560000 scanner.net > 192.168.117.233: icmp: echo request
```

Тем не менее даже такое сканирование может остаться незамеченным, если на узле не контролируется ICMP-трафик. Тут возникает риторический вопрос: если хакер сканирует всю сеть, но никто за этим не следит, то будет ли поднята тревога? Если считать подозрительным каждый одиночный эхо-запрос, то системы обнаружения вторжений поднимали бы тревогу слишком часто, поэтому одиночные эхо-запросы,

как правило, игнорируются. Тем не менее система обнаружения вторжений, которая отслеживает все попытки сканирования, способна выявить сканирование при отправке большого числа одиночных запросов с одного хоста. Другими словами, система обнаружения вторжений контролирует количество попыток подключения с одного IP-адреса за установленный период времени (например, не более семи соединений в час). Поэтому приведенное выше сканирование будет обнаружено.

Эффективное составление схем

Рассмотренное в предыдущем примере сканирование, скорее всего, выполнялось автоматически с помощью простой программы. Но зачем нужны лишние действия, если протокол ICMP позволяет организовать широковещательную отправку запросов и проверить активность многих хостов с помощью нескольких команд? Именно такую возможность использует программа сканирования, отчет о действиях которой представлен ниже.

```
13:51:16.210000 scanner.net > 192.168.65.255: icmp: echo request
13:51:17.300000 scanner.net > 192.168.65.0: icmp: echo request
13:51:18.200000 scanner.net > 192.168.66.255: icmp: echo request
13:51:18.310000 scanner.net > 192.168.66.0: icmp: echo request
13:51:19.210000 scanner.net > 192.168.67.255: icmp: echo request
13:53:09.110000 scanner.net > 192.168.67.0: icmp: echo request
13:53:09.940000 scanner.net > 192.168.68.255: icmp: echo request
13:53:10.110000 scanner.net > 192.168.68.0: icmp: echo request
13:53:10.960000 scanner.net > 192.168.69.255: icmp: echo request
13:53:10.980000 scanner.net > 192.168.69.0: icmp: echo request
```

Как видим, составляется схема подсети `192.168`. Здесь представлен только фрагмент полного сканирования, в котором значения третьего байта IP-адреса изменяются от `65` до `69`. В качестве значения последнего байта используется или `0`, или `255`. Значение 255 — стандартный адрес для широковещательного пакета, а 0 — широковещательный адрес для хостов, в которых стек протоколов TCP/IP работает под управлением операционной системы BSD UNIX (Berkeley Software Distribution). При использовании в ICMP-запросах обоих этих широковещательных адресов должны ответить все активные хосты исследуемой сети.

Можно сделать вывод о том, что в своей сети лучше запретить любые действия с использованием широковещательных адресов. Лично мне не известны другие законные операции при рассылке широковещательных запросов, кроме как проверки работы компьютеров. Кроме того, запрещение широковещательного трафика предотвратит распространение атаки Smurf за счет ресурсов вашей сети (см. раздел "Атака Smurf").

Искусное составление схем

Следующий листинг дает представление о новом варианте уже рассмотренного метода составления схемы сети.

```
06:34:31.150000 scanner.net > 192.168.21.0: icmp: echo request
06:34:31.150000 scanner.net > 192.168.21.63: icmp: echo request
06:34:31.150000 scanner.net > 192.168.21.64: icmp: echo request
06:34:31.150000 scanner.net > 192.168.21.127: icmp: echo request
06:34:31.150000 scanner.net > 192.168.21.128: icmp: echo request
06:34:31.150000 scanner.net > 192.168.21.191: icmp: echo request
```

```
06:34:31.150000 scanner.net > 192.168.21.192: icmp: echo request
06:34:31.150000 scanner.net > 192.168.21.255: icmp: echo request
```

Рассмотрим данный метод сканирования. Эхо-запросы ICMP были отправлены в подсеть класса C `192.168.21`. Обратите внимание на последний байт использованных IP-адресов. Как видим, первый запрос был отправлен по широковещательному адресу 0, а последний — по широковещательному адресу 255. Эти адреса уже применялись в предыдущем методе сканирования.

Но в данном случае запросы отправляются еще и по другим IP-адресам, при этом используются первое и последнее значение промежутка в 64 IP-адреса. Например, первый IP-адрес заканчивается байтом со значением 0, а следующий за ним — байтом со значением 63. Почему используется именно такой интервал в 64 адреса?

Согласно стандарту сеть класса C имеет 256 IP-адресов в диапазоне от 0 до 255. С помощью маски подсети эту сеть можно разделить на несколько подсетей, и одним из вариантов является создание четырех независимых подсетей по 64 IP-адреса каждая. При этом соответственно изменяются широковещательные адреса для каждой подсети, которые и были использованы в данном примере. Таким образом, при составлении схемы учитывается возможность существования "нестандартной" схемы сети класса C. Если в исследуемой сети действительно было использовано указанное разделение, то ответили бы все активные хосты. То же самое произойдет и при сканировании стандартной сети класса C (в которой не запрещено использование широковещательных адресов), так как применяются широковещательные адреса 0 и 255.

Интеллектуальное составление схемы

В этом последнем отчете о сканировании с целью составления схемы сети показано использование другого типа ICMP-запросов. ICMP-запрос маски адреса (`address mask request`) предназначен для получения маски подсети, в которой установлен запрашиваемый хост. Помните, как в прошлом примере основной целью сканирования было угадать схему адресации сети? Данный метод позволяет устранить лишние проблемы.

```
20:39:38.120000 scanner.edu > router.com: icmp: address mask request (DF)
20:39:38.170000 router.com > scanner.edu: icmp: address mask is 0xffffff00
(DF)
20:39:39.090000 scanner.edu > router2.com: icmp: address mask request (DF)
20:39:39.230000 router2.com > scanner.edu: icmp: address mask is 0xffffff00
(DF)
20:39:38.120000 scanner.edu > routerx.com: icmp: address mask request (DF)
20:39:38.170000 routerx.com > scanner.edu: icmp: address mask is 0xffffff00
(DF)
```

Этот метод нельзя назвать классическим методом составления схемы сети, но он позволяет провести предварительную разведку и определить маски подсетей различных маршрутизаторов. Как правило, на запросы о масках подсетей отвечают только маршрутизаторы, поэтому нарушитель получает адреса наиболее интересных для исследования хостов. Как говорилось в главе 1, "Фундамент Internet", с помощью маски подсети можно узнать, сколько битов IP-адреса отведено для указания сети, а сколько — для хоста.

Если нарушитель определит маску подсети, он будет точно знать, сколько хостов нужно сканировать. Хотя маску подсети хоста обычно можно узнать по первому байту

IP-адреса, использованный в данном случае ICMP-запрос позволяет выявить сети, в которых применяются нестандартные маски. Такую информацию невозможно получить при изучении только IP-адреса. В этом примере сканируемые маршрутизаторы отвечают на запрос шестнадцатеричным числом `0xffffff00`. Десятичное значение этого числа составляет маску подсети `255.255.255.0`, т.е. все хосты принадлежат сети класса C. Для обеспечения безопасности ответы маршрутизаторов на ICMP-запросы о масках подсети следует запрещать.

Резюме о составлении схемы сети

Итак, существуют следующие методы составления схемы сети:

- отправка эхо-запросов ICMP отдельным хостам сети;
- отправка эхо-запросов ICMP по широковещательным адресам сети;
- отправка эхо-запросов ICMP по широковещательным адресам сети и вероятных подсетей;
- отправка некоторым хостам сети ICMP-запросов о маске подсети с целью выявления наиболее интересных целей сканирования.

Стандартные ICMP-сообщения

В этом разделе будет рассмотрена нормальная работа протокола ICMP, а именно, несколько ICMP-сообщений об ошибках, с помощью которых отправитель уведомляется о проблемах при доставке пакета. Конечно, рассматривать вредоносные действия с помощью ICMP-сообщений интереснее, но до выявления нестандартных ситуаций следует изучить нормальную работу протокола.

Хост недостижим

Следующий отчет содержит информацию о получении хостом `sending.host` ICMP-сообщения об ошибке при доставке отправленного им пакета хосту `target.host`.

```
router > sending.host: icmp: host target.host unreachable
```

По какой-то причине хост `target.host` недостижим. Возможно, вообще не существует хоста с указанным IP-адресом, или он в данный момент не в состоянии ответить на запрос, или же этот хост не отвечает в результате неправильной настройки.

В любом случае, хост не в состоянии самостоятельно отправить сообщение об ошибке. Поэтому этим придется заняться маршрутизатору, ответственному за доставку сообщений в указанной сети. Маршрутизатор уведомляет хост-отправитель о возникшей проблеме с помощью сообщения `host unreachable` (хост недостижим). Несложно догадаться, что предоставляемые таким образом сведения будут весьма полезными при зондировании сети. Если злоумышленник собирает информацию об активных хостах сети для их последующего сканирования, то выявленные в качестве недостижимых IP-адреса больше проверяться не будут и сканирование станет более направленным.

Ценная разведывательная информация, добытая с помощью получения множества ICMP-сообщений о недостижимости хостов, снижает безопасность исследуемой сети. В списках контроля доступа маршрутизаторов Cisco присутствует правило `no ip unreacheables`, которое запрещает возвращение ICMP-сообщений о недостижимости хостов сети.

Порт недостижим

Рассмотрим случай, когда хост-получатель с помощью ICMP-сообщения уведомляет отправителя о том, что на указанный UDP-порт запросов не ожидается. В данном случае пакет был отправлен на NTP-порт (Network Time Protocol – протокол синхронизации сетевого времени) получателя.

```
target.host > sending.host: icmp: target.host udp port ntp unreachable (DF)
```

Для доставки сообщения об ошибке используется протокол ICMP. Не забывайте, что протокол TCP использует другой способ уведомления отправителя о том, что запрошенный порт закрыт. С этой целью отправителю возвращается TCP-сегмент с установленным флагом RESET. Протокол UDP не обладает подобной возможностью, поэтому на помощь привлекается ICMP.

И снова мы видим, что данное ICMP-сообщение об ошибке предоставляет ценную информацию. Все сканируемые UDP-порты, которые не ответили на запрос стандартным ICMP-сообщением `port unreachable`, возможно, являются открытыми. Но есть вероятность, что ответа не последовало из-за банальной потери пакета. Кроме того, сообщения о недостижимости порта могут быть заблокированы для передачи за пределы локальной сети. Таким образом, отсутствие ICMP-сообщения о недостижимости указанного UDP-порта еще не дает гарантии, что на этом порту работает какая-либо служба.

Запрещено администратором

В следующем примере содержится отчет о еще одном ICMP-уведомлении об ошибке.

```
router > sending.host: icmp: target.host unreachable - admin prohibited
```

В данном случае хост-отправитель пытался передать пакет хосту `target.host`. Маршрутизатор (`router`) работает как шлюз для сети получателя.

В списке контроля доступа маршрутизатора установлены ограничения на передачу в локальную сеть определенного трафика. Этот трафик, например, предназначен заблокированному порту, а может быть заблокирован прием пакетов от хоста с определенным IP-адресом или от компьютеров какой-то сети, или же установлена защита на доступ к определенному компьютеру или подсети. В любом случае маршрутизатор может ответить ICMP-сообщением "`unreachable - admin prohibited`" (недостижимо – запрещено администратором). Хотя в этом сообщении и не указан объект блокирования (например, порт получателя или IP-адрес отправителя), но проницательный нарушитель может использовать различные варианты установки соединения, чтобы выявить истинную причину блокирования. Затем он постарается найти другие незащищенные пути проникновения в сеть.

Необходима фрагментация

Еще одно ICMP-сообщение позволяет уведомить отправителя о невозможности доставки дейтаграммы по указанному адресу без ее фрагментации.

```
router > sending.host: icmp: target.host unreachable - need to frag (mtu 1500)
```

Выражение DF в отчете TCPdump означает, что был установлен флаг Don't Fragment ("не фрагментировать"), указывающий на запрещение фрагментации данной дейтаграммы. Если дейтаграмма с этим флагом должна пройти через сеть, в которой требуется ее фрагментация, то маршрутизатор выявляет проблему, отбрасывает дейтаграмму и возвращает отправителю ICMP-сообщение об ошибке (`need to frag`).

Данное ICMP-сообщение содержит значение максимальной единицы передачи данных (MTU) сети, в которой требуется фрагментация дейтаграммы. Некоторые хосты намеренно отправляют пробную дейтаграмму с установленным флагом DF, чтобы узнать минимальное значение MTU на пути к получателю. Если возвращаемое ICMP-сообщение указывает наименьшее значение MTU на всем пути, то отправитель может установить размер дейтаграммы, который бы позволил обойтись без фрагментации. Избегать фрагментации следует в связи с большими расходами ресурсов при потере хотя бы одного фрагмента из последовательности.

Превышение лимита времени

ICMP-сообщение `time exceeded in-transit` информирует отправителя о чрезмерно долгом пребывании его дейтаграммы в Internet.

```
routerx > sending host: icmp: time exceeded in-transit
```

Протокол IP должен иметь возможность удалить из Internet потерянную дейтаграмму, которая, возможно, бесцельно и бесконечно передается по замкнутому маршруту между несколькими маршрутизаторами. Необходимость отбрасывания потерянных дейтаграмм определяет значение поля TTL (time-to-live – время жизни) в заголовке IP-дейтаграммы.

При отправке дейтаграмм в различных операционных системах задаются различные значения поля TTL. Узнать эти стандартные значения можно, обратившись по адресу `http://project.honeynet.org/papers/finger/traces.txt`.

При прохождении дейтаграммы через маршрутизаторы на пути к получателю каждый маршрутизатор уменьшает значение поля TTL на 1. Когда значение этого поля станет равным 0, маршрутизатор отбрасывает дейтаграмму и возвращает ICMP-сообщение `time exceeded in-transit` хосту-отправителю. В главе 5, "Воздействия и реакции", рассказано, как с помощью этих ICMP-сообщений и утилиты Traceroute выявляются промежуточные маршрутизаторы на пути прохождения пакетов по сети до места назначения.

Информация ICMP-сообщений об ошибках

Важно понять, что в возвращаемой дейтаграмме с ICMP-сообщением об ошибке предоставляется дополнительная информация. В частности, после самого ICMP-сообщения возвращается заголовок IP-дейтаграммы и 8 байт заголовка вло-

женного пакета и данных оригинальной дейтаграммы, которая вызвала ошибку (рис. 4.2). Эта информация предназначена для того, чтобы хост-отправитель смог идентифицировать недошедшую дейтаграмму и внести коррективы в процесс ее отправки. Согласно документу RFC 1122 ответа на полученное ICMP-сообщение об ошибке не требуется.

Кроме того, не следует ожидать, что все операционные системы будут точно возвращать IP-заголовок и следующие за ним 8 байт информации первоначальной дейтаграммы. Логично предположить, что в возвращаемую дейтаграмму после ICMP-сообщения об ошибке всегда скопируются те же первые 28 байт, что не удалось доставить в исходной дейтаграмме. На самом деле, с помощью программы `nmap` возможно определение операционной системы удаленного хоста по результатам его ответов на нормальные и нестандартные пакеты. Все операционные системы отвечают по-разному, что позволяет провести классификацию ответов. Одним из тестов в серии пакетов, предназначенных сканируемому хосту, является попытка отправить дейтаграмму на закрытый UDP-порт. При этом целью является возврат ICMP-сообщения "`port unreachable`". Программа `nmap` исследует заголовок IP-дейтаграммы этого сообщения и следующие 8 байт данных начального пакета. С помощью сравнения полей в полученном сообщении с полями исходной дейтаграммы `nmap` идентифицирует операционную систему удаленного хоста.

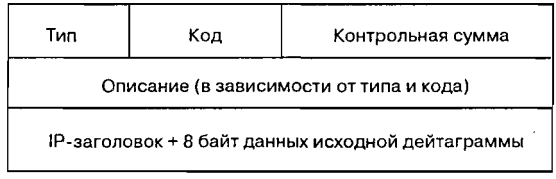


Рис. 4.2. Формат ICMP-сообщения об ошибке

Резюме о стандартном использовании ICMP

Мы рассмотрели только несколько ICMP-сообщений, которые могут быть зарегистрированы при анализе сетевого трафика. Существует множество других IPCM-сообщений об ошибках, также предоставляющих полезные сведения. Их могут отправлять как хосты, так и маршрутизаторы.

Также указывалось, что лучше блокировать возврат отправителю некоторых ICMP-сообщений о невозможности доставки, что не позволит разглашать важную информацию о схеме сети.

Использование ICMP для проведения атак

Нет ничего удивительного в том, что ICMP, как другие протоколы, стал использоваться с дурными намерениями. В настоящее время ICMP применяется в различных типах атак отказа в обслуживании, а также в качестве самого надежного средства для проведения скрытого сканирования. В этом разделе мы рассмотрим эти и другие примеры вредоносного использования ICMP.

Гололедица

Одним зимним утром после ночного снегопада, по дороге на работу я подумал, что эта поездка похожа на мою работу в области обеспечения безопасности сетей. Я осторожно ехал на своем автомобиле по длинной, извилистой, занесенной снегом дороге, снижая скорость, переключаясь на более низкую передачу, съезжая вниз, и внимательно объезжая оставленные у обочины машины на подъеме. Я старался соблюдать максимальную осторожность по отношению к любому препятствию, но были и невидимые мне угрозы, например, гололедица.

Очень часто, анализируя трафик, предназначенный нашим узлам, у меня возникало это ощущение невидимой угрозы. На личном опыте я убедился в настойчивости, хитрости и сообразительности, с которыми Internet-пираты добиваются своих целей. Для каждого специалиста в области сетевой безопасности основным вопросом должен стать вопрос "Чего я не заметил?". Если проявить беспечность по отношению к защите своего Web-узла, то однажды он может выйти из-под контроля по неизвестной причине.

Атака Smurf

Атака Smurf (рис. 4.3) основана на использовании возможности протокола ICMP рассылать дейтаграммы по нескольким адресам. Ответить на один широковещательный эхо-запрос ICMP может большое количество хостов. Эта возможность используется для проведения атаки отказа в обслуживании на избранный хост или сеть.

Сначала злоумышленник должен сформировать широковещательный эхо-запрос ICMP к хостам атакуемой сети, подменив при этом действительный IP-адрес своего компьютера.

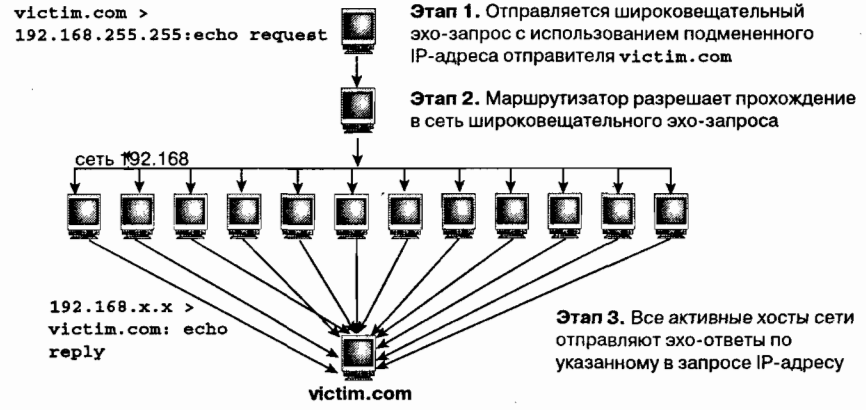


Рис. 4.3. Схема атаки Smurf

В качестве этого IP-адреса используется адрес атакуемого хоста (или сети). Чтобы запрос попал ко всем хостам сети, его должен пропустить внешний маршрутизатор. Успешным завершением атаки является отправка всеми работающими хостами эхо-ответов на адрес атакуемого хоста. Атакованный хост (или сеть, в которой он находится) может пострадать от такой внезапной активности и перестанет выполнять возложенные на него задачи при следующих условиях:

- нарушитель отправляет большое количество широковещательных эхо-запросов;

- внешний узел (маршрутизатор) позволяет прохождение входящего трафика с указанием широковещательного адреса;
- машрутизатор работает в крупной сети, хосты которой одновременно отправляют большое количество эхо-ответов (с другой стороны, того же результата можно добиться, использовав несколько машрутизаторов более мелких сетей);
- канал, с помощью которого атакуемый узел соединен с Internet, имеет низкую пропускную способность. Точнее, количество одновременно отправленных пакетов должно превысить максимальную пропускную способность этого канала. Хотя можно "затопить" пакетами *любое* Internet-соединение при наличии достаточного трафика, но для соединений с меньшей пропускной способностью сделать это будет проще.

Вот еще одна причина, по которой следует запретить вхождение извне пакетов с широковещательным адресом — это не позволит использовать вашу сеть для распространения атаки Smurf.

Атака Tribe Flood Network

Атака Tribe Flood Network (TFN) является еще одной атакой отказа в обслуживании, в которой используются ICMP-сообщения (рис. 4.4). В отличие от атаки Smurf, организуемой с одного компьютера с применением одной сети для ее распространения, атака TFN использует большое количество распределенных хостов. Эти хосты часто называют *хостами-демонами*. Поэтому термин *распределенная атака отказа в обслуживании* (DDoS) наиболее точно определяет использование нескольких рассредоточенных в Internet хостов для совместного осуществления атаки.

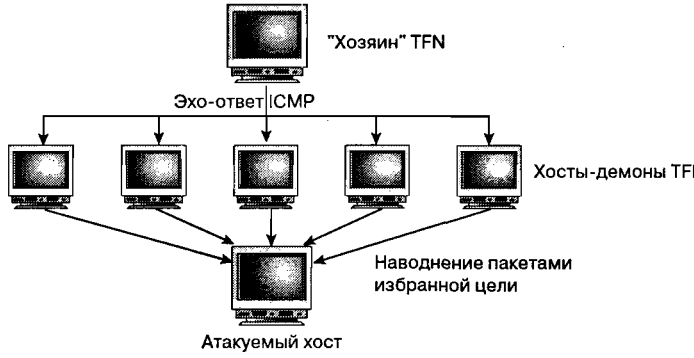


Рис. 4.4. Схема атаки Tribe Flood Network

Для проведения этой атаки требуется установка программы на ведущем компьютере — "мастере" TFN и на нескольких агентах — хостах-демонах TFN. Как правило, в качестве хостов-демонов используются скомпрометированные компьютеры. Мастер TFN дает хостам-демонам команду на атаку (часто одновременную) избранной цели. Взаимодействие между мастером и хостами-демонами осуществляется с помощью эхо-ответов ICMP. Демоны TFN могут организовать UDP-наводнение, SYN-наводнение, наводнение эхо-запросами ICMP или атаку Smurf.

Мастер информирует хосты-демоны о начале атаки с помощью эхо-ответов ICMP. При этом тип атаки определяется по значению поля идентификации в ICMP-заголовке эхо-ответа. В области данных такого эхо-ответа передаются необходимые аргументы.

Почему для организации атаки вместо эхо-запросов применяются эхо-ответы? Дело в том, что на многих узлах в целях обеспечения безопасности блокируются внешние эхо-запросы ICMP. В то же время, прохождение эхо-ответов часто разрешается. Это дает возможность локальным пользователям узнать о доступности внешних хостов, но весьма опасно с точки зрения рассылки неконтролируемых эхо-запросов ICMP.

Как вы уже, наверно, догадались, одновременное использование нескольких распределенных хостов для наводнения пакетами избранной цели позволит провести успешную атаку отказа в обслуживании против хоста или сети. Чтобы получить более подробную информацию об атаке TFN, зайдите на сайт `www.cert.org` и обратитесь к отчету об инциденте IN-99-07.

Самостоятельно организованный отказ в обслуживании

Это было 29 декабря 1999 года. Приступая к работе в центре Y2K (проблема 2000 года), расположенном в резиденции министра обороны, я размышлял над слухами о грядущем компьютерном апокалипсисе. Большинство полагало, что будут проведены массированные атаки отказа в обслуживании, направленные против транспортных и энергетических служб. Несмотря на заверения хакеров о том, что они будут праздновать Новый год вместе со всеми, преобладало мнение, что появление средств для распределенных атак отказа в обслуживании приурочено именно к наступлению 2000 года.

Реакцией на эту угрозу стало закрытие многих сайтов и ограничение доступа ко многим сетям. Парадоксальность этой ситуации подчеркнула фраза одного из наших сотрудников: "Забавно, что, защищаясь от атаки отказа в обслуживании, они сами отключают собственные службы".

Атака WinFreeze

Атака WinFreeze, по существу, заставляет избранный компьютер атаковать самого себя.

```
router > victim.com: icmp: redirect 243.148.16.61 to host victim.com
router > victim.com: icmp: redirect 110.161.152.156 to host victim.com
router > victim.com: icmp: redirect 245.211.87.115 to host victim.com
router > victim.com: icmp: redirect 49.130.233.15 to host victim.com
router > victim.com: icmp: redirect 149.161.236.104 to host victim.com
router > victim.com: icmp: redirect 48.35.126.189 to host victim.com
router > victim.com: icmp: redirect 207.172.122.197 to host victim.com
router > victim.com: icmp: redirect 113.27.175.38 to host victim.com
router > victim.com: icmp: redirect 114.102.175.168 to host victim.com
```

С помощью ICMP-сообщения `redirect` хост-отправитель уведомляется о выборе для доставки сообщения неоптимального маршрутизатора и о необходимости добавить адрес оптимального маршрутизатора в таблицу маршрутизации. При наводнении этими ICMP-сообщениями о перенаправлении атака WinFreeze может вызвать отказ в обслуживании уязвимого хоста, работающего под управлением Windows NT. Атака выполняется в сети атакуемого компьютера, а ICMP-сообщения приходят от имени маршрутизатора этой сети. При получении массы сообщений `redirect` атакованный хост пытается внести изменения в таблицу маршрутизации, и ресурсы центрального процессора в основном тратятся на обработку поступающих пакетов.

В этом примере маршрутизатор `router` заставляет хост `victim.com` перенаправить отправляемые им (хостом) пакеты на самого себя. В результате при попытке внести многочисленные изменения в таблицу маршрутизации хост `victim.com` может не справиться с другими возложенными на него задачами.

Программа Loki

К настоящему времени программу Loki можно ставить в пример как наиболее опасное и вредоносное средство применения возможностей протокола ICMP. В скандинавской мифологии Локи — бог обмана и зла. Программа взлома Loki тоже обладает незаурядными возможностями в этой области. Протокол ICMP был создан для уведомления об ошибках и выдачи простых запросов. Поэтому аналитики сетевого трафика считали его довольно безобидным средством, за исключением разве что генерируемых с его помощью атак отказа в обслуживании и сбора информации о сети при отсутствии блокирования ICMP-сообщений. Так было до появления программы Loki.

Loki использует ICMP в качестве туннельного протокола для создания скрытого канала связи. *Скрытым* каналом (covert channel) называют канал, использующий метод передачи данных или поле дейтаграммы нестандартным способом для осуществления незаконных действий. Другими словами, ICMP выступает как транспортный протокол, а программа Loki работает по стратегии клиент-сервер. Если на скомпрометированном хосте установлен сервер Loki, то он будет отвечать на запросы Loki-клиента. Например, клиент Loki может послать запрос `cat/etc/passwd`, чтобы получить файл паролей. Пользователю компьютера, на котором работает клиент Loki выводится на дисплей содержимое запрошенного файла, он может его сохранить и попытаться взломать. Более подробную информацию о программе Loki можно получить по адресу `www.phrack.com` (выпуск 49, статья 6).

Самым неприятным является то, что кажущийся безвредным протокол применяется для проведения очень сложной и опасной атаки. ICMP никогда не предназначался для поддержки работы подобных приложений. Поэтому советую специалистам в области безопасности относиться к ICMP-трафику с максимальной возможным вниманием.

Незатребованные эхо-ответы

Попробуем провести небольшой анализ и применить изложенные выше теоретические сведения на практике с помощью следующего примера.

```
reply.com > 192.168.127.41: icmp: echo reply
reply.com > 192.168.127.41: icmp: echo reply
reply.com > 192.168.127.41: icmp: echo reply
reply.com > 192.168.127.41: icmp: echo reply
reply.com > 192.168.127.41: icmp: echo reply
reply.com > 192.168.127.41: icmp: echo reply
```

Что мы видим? Хост `reply.com` отправляет хосту `192.168.127.41` поток эхо-ответов ICMP. Все было бы нормально, если бы хост `192.168.127.41` отправлял эхо-запросы для возвращения данных ответов. Но это не так, никаких эхо-запросов с хоста `192.168.127.41` не отправлялось. Зачем это кому-то нужно? Возможные причины таких действий изложены в трех следующих разделах.

Не забывайте, что для выявления подобных операций должна быть установлена система обнаружения вторжений или программное обеспечение, которое способно сохранять содержимое пакетов (т.е. должна существовать возможность определить большое количество одинаковых эхо-запросов). Многие системы обнаружения вторжений не сохраняют сведений о доставленных ранее пакетах и не в состоянии выявить действия нарушителя. Теперь рассмотрим несколько возможных причин для отправки незатребованных эхо-ответов.

Предположение 1: подмена IP-адреса

Первое предположение при получении отчета о подобном трафике – кто-то подменил IP-адрес своего хоста IP-адресом `192.168.127.41` и отправил серию эхо-запросов на хост `reply.com`. В результате хост `reply.com` возвращает эхо-ответы указанному отправителю. Если эхо-ответы приходят от многих компьютеров, работающих в той же сети, что и `reply.com`, это означает, что компьютер `reply.com` стал целью атаки Smurf.

В последнее время объем действий с использованием подмененных IP-адресов резко увеличился, поэтому наше первое предположение наиболее вероятно. Как правило, если получение незатребованных эхо-ответов ICMP связано с подменой IP-адреса вашего хоста (в этом примере `192.168.127.41`), то от этого же хоста (в данном случае `reply.com`) поступает и другая незапрошенная информация. Обычно эти эхо-ответы отправляются нескольким хостам локальной сети (в нашем случае один и тот же эхо-ответ повторяется много раз).

Предположение 2: атака TFN

Предположим, что проводится атака TFN. Как известно, "хозяин" TFN взаимодействует с демонами TFN с помощью эхо-ответов ICMP.

Следовательно, причиной получения незапрошенных эхо-ответов могло стать использование хоста `192.168.127.41` в качестве демона TFN. Хотя значение поля идентификационного номера ICMP используется, чтобы указать хосту-демону начало атаки TFN определенного типа, все же точно описать смысл этого значения невозможно, так как хакер мог изменить исходный код программы атаки. Определить, не стал ли данный хост демоном TFN, проще по исходящему от этого хоста трафику после получения незапрошенных эхо-ответов ICMP. Если он отправляет большой объем информации непонятного предназначения, значит, скорее всего, он участвует в атаке TFN.

Предположение 3: Loki

Последнее из вероятных предположений заключается в вероятном обмене информацией между клиентом и сервером Loki. При этом вполне возможно, что на каждый эхо-запрос ICMP генерируется несколько эхо-ответов.

В первых версиях программы Loki значения шестого и седьмого байта ICMP-сообщений оставались неизменными. По этому признаку и можно было выявить работу программы Loki – достаточно было сохранить отчет о проходящем трафике с помощью TCPdump в шестнадцатеричном формате и убедиться в неизменности значения в поле порядкового номера ICMP-сообщения. Согласно стандарту значение этого поля должно быть уникальным для каждого эхо-запроса

(аналогично значению идентификатора в заголовке IP-дейтаграммы) и увеличиваться на 1 или на 256 для каждого последующего эхо-запроса. В более поздних версиях Loki значение этого поля может быть зашифровано, и указанным методом программу Loki выявить не удастся.

Таким образом, становится очевидно, что ICMP-трафик (будь то эхо-запросы или эхо-ответы) может быть использован для проведения вредоносных операций. Следовательно, его лучше блокировать с помощью устройства фильтрации пакетов.

Резюме о незаконном использовании ICMP

Подводя итоги этого раздела, можно сказать, что протокол ICMP может применяться не только для нормальных, но и для вредоносных операций, например для атак отказа в обслуживании (Smurf и WinFreeze). В атаке TFN этот протокол служит, скорее, в качестве транспортного средства, предоставляющего возможность для проведения еще одного типа атак отказа в обслуживании. Программа Loki вообще превращает ICMP в туннельный протокол для проведения атакующих действий.

Блокировать или не блокировать

Возможных вариантов вредоносного использования ICMP довольно много. Только в разведывательных целях, например для определения активности конкретного хоста, достаточно получить от него одно из следующих ICMP-сообщений:

- "`protocol unreachable`";
- "`port unreachable`";
- "`IP reassembly time exceeded`";
- "`parameter problem`";
- "`echo replay`";
- "`timestamp replay`";
- "`address mask replay`".

Кроме того, если маршрутизатор сети будет уведомлять отправителя об ошибках недостижимости некоторых хостов (`host unreachable`), то, предположив активность всех остальных хостов, можно составить схему сети.

И это еще не все. Некоторые ICMP-сообщения отправляют только маршрутизаторы. Поэтому получение одного из следующих сообщений позволяет определить маршрутизатор сети:

- "`fragmentation needed but don't-fragment bit set`";
- "`admin prohibited`";
- "`time exceeded in transit`";
- "`network unreachable`";
- "`host unreachable`".

И, наконец, дополнительную информацию можно узнать из перечисленных ниже ICMP-сообщений:

- "`admin prohibited`" – позволяет узнать о типе блокируемого трафика;

- address mask replay" — предоставляет значение маски подсети, в которой установлен запрошенный хост;
- "time exceeded in transit" используется утилитой Traceroute для определения IP-адресов маршрутизаторов и топологии сети;
- "protocol unreachable" может использоваться для полного сканирования хоста на предмет запущенных служб;
- "port unreachable" может применяться для выявления активных хостов с открытыми UDP-портами;
- "fragmentation needed but don't-fragment bit set" позволяет определить значение MTU сетей с целью проведения атаки при использовании фрагментированного трафика.

Так почему же полностью не блокировать весь входящий и исходящий ICMP-трафик? На некоторых узлах так и делается, но давайте рассмотрим последствия блокирования всего входящего ICMP-трафика.

Эхо-запросы без ответа

Очевидно, что при блокировании входящих эхо-запросов и эхо-ответов ICMP невозможно провести диагностику удаленного хоста с помощью утилиты ping. С другой стороны, эти ICMP-сообщения не будут использованы для проведения незаконных операций. Таким образом, приносимое неудобство компенсируется повышением уровня безопасности и устранением еще одного пути проникновения в сеть.

Иногда блокируют только входящие эхо-запросы, что позволяет диагностировать удаленные компьютеры и получать результаты из разрешенных к прохождению эхо-ответов. Однако хакеры тоже знают об этом, доказательством чему служат программы TFN и Loki, которые для доставки информации в основном используют эхо-ответы ICMP.

Отказ от возможностей Traceroute

Для определения маршрутизаторов, через которые проходит дейтаграмма на пути к получателю, в UNIX используется команда **traceroute**, а в Windows — **tracert**. Блокирование входящего ICMP-трафика не позволит запускать обе эти команды из вашей сети, так как для них требуется получить входящие ICMP-сообщения об истечении времени жизни пакета ("time exceeded in transit").

Для команды **tracert**, используемой в системах под управлением Windows, требуется получение внешних эхо-запросов, поэтому удаленный пользователь не сможет применить эту команду для компьютеров в вашей сети. В свою очередь, UNIX-команда **traceroute** в качестве транспортного средства использует протокол UDP, поэтому блокирование входящего ICMP-трафика не повлияет на возможности удаленных пользователей применять ее для компьютеров вашей локальной сети.

Тишина в локальной сети

Протокол ICMP позволяет информировать о возникновении проблем при доставке сообщения к определенному хосту или на определенный порт. При блокиро-

вании всех входящих ICMP-сообщений хосты или маршрутизаторы локальной сети не смогут получать эти уведомления. Это, конечно, не приведет к катастрофическим последствиям, но вызовет некоторые затруднения. Пусть например, хост локальной сети пытается установить TCP-соединение с другим хостом, который в данное время не работает. Удаленный маршрутизатор отправляет ICMP-уведомление о недостижимости хоста, но оно блокируется на входе в локальную сеть. Хост-отправитель будет предпринимать попытки установить соединение до истечения установленного времени, засоряя сеть бесполезным трафиком.

Неизвестное значение MTU

Как уже указывалось, по возможности хост-отправитель TCP-сообщений старается избежать фрагментации дейтаграмм на пути к адресату. Для этого определяется значение MTU всего пути – отправляется пробный пакет с установленным флагом DF. Этот пакет либо будет доставлен получателю, либо отправителю должно вернуться ICMP-сообщение "`need to frag`" с указанием наименьшего значения MTU.

Блокирование всех входящих ICMP-сообщений не дает работать этому механизму, что может привести к довольно серьезным проблемам. Хост-отправитель будет ожидать уведомления при необходимости фрагментации. Поскольку в результате блокирования он его не получит, будет продолжена отправка чрезмерно больших дейтаграмм с установленным флагом DF. Все они будут отброшены, но отправитель ничего об этом не узнает. Следовательно, пакет попадет адресату, только если его размер будет меньше значения MTU.

Поэтому, если принято решение о блокировании ICMP-трафика, то сделайте исключение для входящих ICMP-сообщений "`host unreachable - need to frag`".

Резюме

Протокол ICMP предназначен для уведомления хостов о проблемах при доставке дейтаграмм и обмена простыми сообщениями. Информация может передаваться как одному хосту, так и сразу нескольким с помощью широковещательного адреса.

Относитесь к ICMP-трафику с осторожностью. В этой главе было описано несколько современных способов вредоносного использования возможностей ICMP. Нет никакого сомнения в том, что появятся и новые с неизвестным доныне предназначением.

В целях защиты блокируйте входящий ICMP-трафик, но делайте это разумно и избирательно. Закрыв путь хакерам, проверьте, не приводит ли установленное блокирование к каким-либо тяжелым последствиям.

5

Воздействия и реакции

До этой главы в основном рассматривались операции, являющиеся первоначальными воздействиями. Ответные реакции на каждый вид воздействия описывались очень кратко. Этот метод весьма удобен при изучении какого-либо нового материала. Но теперь, мы надеемся, наши читатели уже усвоили базовые теоретические сведения и готовы углубить свои знания.

Большинство современных систем обнаружения вторжений слишком часто извещают о ложных тревогах. Другими словами, они не способны принять обоснованные решения по вопросу опасности или безвредности проходящего по сети трафика. Поэтому сетевые системы обнаружения вторжений (Network Intrusion Detection System – NIDS) часто перестраховываются и сообщают о несуществующих опасностях. Тому есть много причин. Кратко объяснить это можно так, что в большинстве случаев сигнатуры и наборы правил, которые используют системы NIDS для выявления опасного трафика, слишком неконкретны. Если эти сигнатуры не могут быть определены более четко, или этого просто не сделано, то система обнаружения вторжений будет сообщать о ложных тревогах.

Поэтому специалист, отвечающий за безопасность, должен уметь отличать ложную тревогу от "боевой". Он должен уметь исследовать вызвавший тревогу трафик и оценить степень его опасности. Для принятия подобных решений необходимо точно знать, как выглядит нормальный трафик, и как определить вредоносные изменения. Понятно, что все варианты возможных воздействий и ответных реакций нельзя описать в одной главе. Основной целью представленного ниже материала является изложение некоторых общих сведений, которые помогут принять правильное решение при анализе подозрительного сетевого трафика.

Сначала мы рассмотрим нормальную работу стандартных приложений и протоколов. Затем перейдем к изучению трафика, который немного отличается от обычного и может послужить причиной тревоги. В заключение будут описаны абсолютно патологические события – явные атаки хакеров.

Это чем-то напоминает отношения в некоторых семьях. Сначала партнеры ухаживают друг за другом и ведут себя максимально "прилично". После начала со-

вместной жизни хорошие манеры постепенно остаются в прошлом, никто не соблюдает этикет, и можно уже себе позволить пить чай, причмокивая от удовольствия. И чем дальше — тем больше может портиться поведение супругов.

Личный пример ложной тревоги

Несколько месяцев назад по дороге на работу на приборной панели моего автомобиля одновременно замигали красные лампочки аккумулятора и тормозов. Мигание сразу же прекратилось, но повторилось еще несколько раз до окончания пути, что меня сильно обеспокоило.

Я должен признать, что ничего не понимаю в автомобилях. Но мне все равно показалось странным, что одновременно поступило два внешне абсолютно не связанных сигнала. Кроме того, так как мои тормоза совершенно не нуждались в электричестве, то у меня не было никаких предположений о взаимосвязи этих сигналов. Я попытался объяснить это какими-то другими причинами, например, что это ложная тревога, что где-то отходит проводок и нет никаких серьезных механических неполадок.

Прошло немного времени и мои проблемы стали серьезнее и мне пришлось звонить в мастерскую по ремонту автомобилей. Я описал работнику мастерской симптомы, на что в ответ тот почти закричал: "Ты идиот!". Несмотря на умение общаться с клиентами, ему не удалось сдержать свое возмущение моей глупостью. Он объяснил мне, что у меня сломался генератор тока, и что моя машина могла взорваться. Думаю, не нужно пояснять, почему я больше не стал искушать судьбу и отвез машину в ремонт.

Позже я задумался над случившимся и попытался понять, почему это произошло, ведь я всегда был довольно предусмотрительным и осторожным человеком, и обращался в мастерскую при первом признаке неисправности. Мое единственное объяснение заключалось в том, что, просматривая результаты работы систем обнаружения вторжений, я настолько привык к постоянным ложным тревогам, что попытался поступить так же и в жизни. Другими словами, я просто перестал доверять чему-либо.

Ожидаемый трафик

Так что же такое "нормальный" трафик? Попытка продемонстрировать все возможные варианты нормального трафика будет бесполезной и, несомненно, очень скучной задачей. Чтобы сделать ее более реальной и интересной, в этом разделе представлены различные стандартные ситуации и образцы трафика, который передается чаще всего. Особое внимание уделяется ответной реакции хостов и маршрутизаторов при получении различной информации при разнообразных обстоятельствах и при использовании разных протоколов.

При изложении данного материала очень трудно пояснить, что значит "нормальный" трафик. Невозможно рассмотреть все бесчисленные варианты нормального трафика. Как ни странно, но, возможно, лучшей характеристикой нормальности можно признать отсутствие ненормальности. По этой причине в нашей книге рассмотрено большое количество примеров трафика, отличающегося от нормы.

Запросы на комментарии

Базовая документация по работе в Internet содержится в документах RFC (Requests for Comments — запросы на комментарии). В них содержится описание стандартов отдельных протоколов. С логической точки зрения сеть Internet можно рассматривать как набор различных протоколов, задокументированных в одном или нескольких RFC. Опубликованные документы RFC никогда не изменяются, а усовершенствование протокола описывается в новом RFC. К наиболее важным для данного раздела документам RFC можно отнести следующие.

- **RFC 793**. В этом RFC стандартизируется протокол TCP, описываются функции, реализуемые этим протоколом, использующие его программы и интерфейс взаимодействия протокола с программами или пользователями, которым требуются его возможности.
- **RFC 768**. В этом RFC рассматривается использование протокола UDP, который не гарантирует доставки сообщений и не ориентирован на установление соединений.
- **RFC 791** определяет работу протокола IP – протокола, который позволяет организовать передачу данных между хостами в виде дейтаграмм.
- **RFC 792** описывает стандарт протокола ICMP, предназначенного для уведомления об ошибках в процессе доставки сообщений.

Более подробную информацию о документах RFC можно получить по адресу `www.rfc-editor.org`.

Воздействие-реакция при TCP-трафике

В этом разделе изучаются ответы при попытке установить `telnet`-соединения в различных условиях, например, при закрытом порте службы `telnet` или при блокировании `telnet`-соединений маршрутизатором. Службу `telnet` можно считать стандартным TCP-приложением. Показаны различные варианты ответов на одинаковые запросы. Конечно, здесь не приведен весь утомительный перечень различных обстоятельств, возможных при установке TCP-соединения. Приведенные здесь ситуации позволяют проиллюстрировать наиболее типичные случаи.

Хост-получатель ожидает запросов на указанный порт

Пусть хост `tel_client.com` пытается установить `telnet`-соединение с хостом `myhost.com`, который ожидает запросов на порт службы `telnet` (TCP-порт 23).

Воздействие:

```
tel_client.com.38060 > myhost.com.telnet: S 3774957990:3774957990(0) win 8760
↳ <mss 1460> (DF)
```

На хосте `myhost.com` запущена служба `telnet` и он разрешает `telnet`-соединение.

Реакция:

```
myhost.com.telnet > tel_client.com.38060: S 2009600000:2009600000(0) ack
↳ 3774957991 win 1024 <mss 1460>
```

В этом отчете TCPdump представлена ожидаемая реакция на попытку клиентского хоста `tel_client.com` установить соединение с портом службы `telnet` хоста-получателя `myhost.com`. Мы уже рассказывали о полной процедуре установления TCP-соединения. Как вы помните, клиент инициирует это соединение с помощью SYN-пакета. В данном случае SYN-пакет отправляется на порт службы `telnet` хоста `myhost.com`.

После этого, если данная служба запущена, установка соединения разрешается, о чем хост `myhost.com` уведомляет клиента пакетом, в котором установлены флаги SYN и ACK. Этот пакет свидетельствует о готовности хоста установить TCP-соединение через запрошенный порт. Последним этапом процедуры установки со-

единения (здесь он не показан) является отправка хостом tel_client.com серверу пакета с установленным флагом ACK.

Хост-получатель не прослушивает указанный порт

Рассмотрим следующий отчет TCPdump, полученный при аналогичной попытке установить telnet-соединение. На этот раз хост myhost.com не ожидает запросов на TCP-порт 23. Ожидаемой реакцией является возврат сервером пакета с установленными флагами RESET и ACK с целью разрыва соединения.

Воздействие:

```
tel_client.com.38060 > myhost.com.telnet: S 3774957990:3774957990(0) win 8760
↳ <mss 1460> (DF)
```

Хост myhost.com не ожидает запросов на порт службы telnet.

Реакция:

```
myhost.com.telnet > tel_client.com.38060: R 0:0(0) ack 3774957991 win 0
```

В этом ответе хоста myhost.com номер подтверждения равен 3774957991, что на единицу больше значения последовательного номера из SYN-пакета. Таким образом, сервер подтвердил получение запроса и сообщил номер следующего ожидаемого байта данных. Однако символ R указывает на установленный флаг RESET – флаг разрыва соединения, так как myhost.com не прослушивает порт службы telnet. От клиента tel_client.com никакого ответа не ожидается.

Хост-получатель не существует

Что произойдет, если попытаться установить telnet-соединение с несуществующим хостом? Рассмотрим следующий пример. Зачастую, если хосты не могут ответить самостоятельно, за них отвечают маршрутизаторы. В данном случае маршрутизатор подсети (router.com), в которой должен работать хост myhost.com, с помощью ICMP-сообщения уведомляет клиента о недостижимости запрошенного хоста.

Воздействие:

```
tel_client.com.38060 > myhost.com.telnet: S 3774957990:3774957990(0) win 8760
↳ <mss 1460> (DF)
```

Хост myhost.com не существует.

Реакция:

```
router.com > tel_client.com: icmp: host myhost.com unreachable
```

Предполагается, что IP-адрес хоста myhost.com зарегистрирован в системе DNS, но он больше недействителен, или хост временно не работает в силу неизвестных причин, или он неверно настроен и не может ответить на запрос. Маршрутизатор router.com уведомляет отправителя о невозможности доставить сообщение указанному хосту.

Порт получателя заблокирован

Рассмотрим еще возможный вариант. Что если фильтрующий маршрутизатор блокирует запросы к порту службы telnet? Каким будет ответ? В этом случае маршрутизатор router.com снова уведомляет о недостижимости хоста и с помощью

ICMP-сообщения `admin prohibited filter` указывает причину – блокирование доступа. И в этом, и в предыдущем случае, указывая причину проблем, маршрутизатор оказывает информационные услуги, но при этом предоставляется ценная информация, особенно полезная хакеру при зондировании сети. Как уже говорилось в главе 4, "Протокол ICMP", маршрутизатору Cisco можно запретить выдавать подобные уведомления, активизировав правило `no ip unreachable` в его списке контроля доступа для нужного интерфейса. Это позволит ограничить объем информации, разглашаемой маршрутизатором.

Воздействие:

```
tel_client.com.38060 > myhost.com.telnet: S 3774957990:3774957990(0) win 8760
↳ <mss 1460> (DF)
```

На заблокированный запрос к службе `telnet` отвечает маршрутизатор.

Реакция:

```
router.com > tel_client.com: icmp: myhost.com unreachable - admin prohibited
↳ filter
```

Порт получателя заблокирован, маршрутизатор не отвечает

Следующий отчет TCPdump иллюстрирует ситуацию, в которой маршрутизатор блокирует входящий трафик, но при этом и не выдает сообщения о недостижимости хоста. Так как хост-отправитель никак не уведомляется о существующей проблеме, он упорно продолжает посылать запросы на соединение. Количество повторных запросов и интервал времени между их отправкой определяется видом стека TCP/IP той операционной системы, под управлением которой работает хост-отправитель. В конце концов после отправки максимально допустимого количества запросов на соединение хост `tel_client.com` отказывается от дальнейших попыток.

Воздействие:

```
17:14:18.726864 tel_client.com.38060 > myhost.com.telnet: S
↳3774957990:3774957990(0) win 8760 <mss 1460> (DF)
```

Маршрутизатор не отвечает на заблокированный запрос к службе `telnet`.

Реакция:

```
17:14:21.781140 tel_client.com.38060 > myhost.com.telnet: S
↳3774957990:3774957990(0) win 8760 <mss 1460> (DF)
17:14:27.776662 tel_client.com.38060 > myhost.com.telnet: S
↳3774957990:3774957990(0) win 8760 <mss 1460> (DF)
17:14:39.775929 tel_client.com.38060 > myhost.com.telnet: S
↳3774957990:3774957990(0) win 8760 <mss 1460> (DF)
```

Тема повторных запросов более подробно рассмотрена в главе 9, "Анализ заголовков вложенных пакетов".

Воздействие-реакция при UDP-трафике

В этом разделе для иллюстрации различных ответов на UDP-запрос используется запрос к службе DNS. Рассмотрены варианты открытого и закрытого UDP-портов. Так как ответы на другие воздействия (например, запрос к несуществующему хосту или к порту, заблокированному маршрутизатором) практически аналогичны ответам, представленным для TCP-трафика, то повторять их не имеет смысла.

Хост-получатель ожидает запросов к указанному порту

В следующем примере отчета TCPdump хост `nslookip.com` организует DNS-запрос к порту `domain` хоста `myhost.com`. В главе 6, "DNS", подобные отчеты о DNS-запросах рассмотрены более подробно. Идентификационный номер DNS-сообщения (`51007`) используется для установления соответствий запросов ответам. Хост `myhost.com` получает запрос и отвечает на него, возвращая на порт `domain` (53) хоста `nslookip.com` пакет, идентификационный номер которого тоже равен `51007`. Обозначение `1/0/0` в отчете TCPdump указывает, что: возвращается одна запись из базы данных DNS без единой записи о подтверждении полномочий, и нет никаких других записей. Как и в случае с протоколом TCP, здесь клиентом используется временный порт `45070`, а также стандартный порт службы DNS-сервера. При ответе хоста `myhost.com` используются именно эти порты.

Воздействие:

```
nslookup.com.45070 > myhost.com.domain: 51007+ (31) (DF)
```

На хосте `myhost.com` запущена служба DNS, поэтому он отвечает на запрос.

Реакция:

```
myhost.com.domain > nslookup.com.45070: 51007 1/0/0 (193) (DF)
```

Хост-получатель не прослушивает запрашиваемый порт

В следующем отчете TCPdump `myhost.com` отвечает ICMP-сообщением о недостижимости UDP-порта `domain`. Еще раз напомним, что предоставление таких ответов позволяет выявлять запущенные на компьютере службы. В данном случае лишнюю информацию предоставляет сам хост, а не маршрутизатор.

Воздействие:

```
nslookup.com.45070 > myhost.com.domain: 51007+ (31) (DF)
```

На хосте `myhost.com` не запущена служба `domain`, поэтому он отвечает на запрос ICMP-уведомением.

Реакция:

```
myhost.com > nslookup.com: icmp: myhost.com udp port domain unreachable
```

В главе 9, "Анализ заголовков вложенных пакетов", рассказано, как программа `nmap` выполняет сканирование открытых UDP-портов. Вывод о работе службы на том или ином UDP-порту делается на основе отсутствия ICMP-сообщений "`port unreachable`". Такой метод иногда называют "сканированием от обратного", так как нет никаких прямых доказательств работы служб на этих портах.

В отличие от TCP-служб, которые отвечают на полученный запрос пакетом с установленными флагами SYN и ACK, большинство UDP-служб не предоставят никакого ответа на простой запрос. Например, на предыдущий DNS-запрос к UDP-порту 53 ответ был получен только из-за того, что взаимодействие осуществлялось выше уровня протокола — на уровне приложений. Стандартный DNS-запрос можно определить по полезной нагрузке дейтаграммы. При сканировании UDP-портов с помощью `nmap` полезная нагрузка равна 0 байт, и поэтому взаимодействие на уровне приложений невозможно.

Воздействие-реакция при ICMP-трафике

ICMP отличается от UDP и TCP, поэтому и набор ответных реакций при взаимодействии по этому протоколу будет другим. Очень кратко перечислим характерные особенности ICMP:

- ICMP не использует портов для взаимодействия;
- ICMP позволяет одностороннюю передачу сообщений об ошибках, при этом ответов не требуется;
- ICMP-сообщение может быть запросом, на который требуется ответ.

ICMP-сообщения об ошибках в основном касаются вопросов работоспособности, например, они используются для уведомления о недоступности хостов или о запрете доступа. Эти вопросы уже были рассмотрены в разделе о TCP-трафике. Поэтому не станем повторяться и воспользуемся командой **tracert** для демонстрации нормального ICMP-ответа на запрос о выявлении промежуточных хостов на пути доставки сообщения к получателю.

Windows-команда tracert

В команде `tracert` пары эхо-запросов и эхо ответов ICMP (ping) используются для определения IP-адресов маршрутизаторов, участвующих в процессе доставки дейтаграммы на пути к получателю. Рассмотрим пример выполнения этой команды.

```
tracert target.my.com
Tracing route to target.my.com [1.2.3.1]
over a maximum of 30 hops:
  1   129 ms   126 ms   130 ms   router.my.com [1.2.3.1]
  1   229 ms   124 ms   118 ms   target.my.com [1.2.3.1]
trace complete
```

С помощью команды `tracert` можно узнать промежуточные маршрутизаторы, через которые передается эхо-запрос ICMP. В этом примере есть только один такой маршрутизатор (`router.my.com`). Затем сообщение достигает хоста-получателя `target.my.com`.

Каждый маршрутизатор и хост-получатель получают по три отдельных эхо-запроса. Результат выполнения команды содержит время кругового обращения (туда и обратно) каждой из этих дейтаграмм. Например, на получение эхо-ответов для трех первых эхо-запросов, отправленных маршрутизатору `router.my.com`, потребовалось 129, 126 и 130 мс. Трехкратное повторение эхо-запросов к маршрутизатору или хосту выполняется на случай возможной потери эхо-запросов. Затем эхо-запросы отправляются на хост `target.my.com`.

Отчет TCPdump о работе tracert

Ниже представлен отчет TCPdump о выполнении предыдущей команды `tracert`.

```
tracer.net > target.my.com: icmp: echo request [ttl 1]
router.my.com > tracer.net: icmp: time exceeded in-transit
tracer.net > target.my.com: icmp: echo request [ttl 1]
router.my.com > tracer.net: icmp: time exceeded in-transit
tracer.net > target.my.com: icmp: echo request [ttl 1]
router.my.com > tracer.net: icmp: time exceeded in-transit
```

```
tracer.net > target.my.com: icmp: echo request
target.my.com > tracer.net: icmp: echo reply (DF)
tracer.net > target.my.com: icmp: echo request
target.my.com > tracer.net: icmp: echo reply (DF)
tracer.net > target.my.com: icmp: echo request
target.my.com > tracer.net: icmp: echo reply (DF)
```

Первый отправленный с помощью `tracert` эхо-запрос передается в IP-дейтаграмме, значение поля TTL которой равняется 1. Значение поля TTL уменьшается на единицу при прохождении дейтаграммой каждого сетевого устройства, благодаря чему пакеты не могут вечно передаваться по Internet и будут отброшены, когда это значение станет равным 0. Маршрутизатор, который отбрасывает пакет, уведомляет об этом отправителя с помощью сообщения "`time exceeded in-transit`".

В нашем примере, получив дейтаграмму со значением поля TTL, равным 1, и уменьшив его до 0, маршрутизатор `router.my.com` уведомляет хост-отправитель об уничтожении пакета.

При использовании `tracert` и получении ICMP-сообщения об ошибке отправитель сохраняет IP-адрес маршрутизатора, который послал это уведомление. Затем `tracert` увеличивает значение поля TTL на 1 и получает таким образом IP-адрес следующего маршрутизатора. Так будет продолжаться до тех пор, пока не будет достигнут хост-получатель, и получен обратный эхо-ответ.

По умолчанию команда `tracert` отправляет по три эхо-запроса для каждого хоста на пути к получателю с целью повышения надежности при возможных потерях ICMP-сообщений. Посмотрим на отчет: хост `tracer.net` отправляет эхо-запрос хосту `target.my.com`. Мгновенно хост `router.my.com` уведомляет об истечении времени жизни этого пакета. Это повторяется для всех трех первых эхо-запросов ICMP. Затем хост `tracer.net` увеличивает значение поля TTL до 2, что позволяет передать пакет адресату — `target.my.com`. По умолчанию в отчете TCPdump значение поля TTL выводится только в том случае, когда для передающегося пакета оно равно 1. Хост `target.my.com` на полученные эхо-запросы возвращает эхо-ответы. Для того чтобы в отчетах TCPdump всегда указывалось значение поля TTL, в командной строке запуска этой программы следует указать параметр `-vv`.

Отклонения в работе протоколов

Промежуточную позицию между стандартным и аномальным использованием возможностей протоколов занимают приложения, которые работают вполне нормально, хотя и немного необычно. Причинами отклонений от стандартного шаблона при использовании таких приложений являются уникальные характеристики самих этих приложений. В этом разделе изложены методы выявления работы этих приложений, по которым можно отличить их действия от вредоносного трафика.

В частности, будут рассмотрены протокол FTP и программа Traceroute. Необычность работы FTP заключается в том, что этот протокол во время соединения использует один временный и один стандартный порт сервера. Программа Traceroute отличается объединением возможностей протоколов ICMP и UDP при выявлении IP-адресов промежуточных маршрутизаторов на пути доставки сообщения.

FTP

При использовании протокола TCP для установки соединения клиентом и сервером используются два порта. Клиент обычно использует временный порт, номер которого больше 1023, а сервер прослушивает стандартный привилегированный порт. Оставшееся время сеанса после установки TCP-соединения клиент и сервер взаимодействуют только через эти выбранные порты. Службы FTP отличаются от TCP-служб, так как при соединении используются два различных порта сервера. Первый из них – это порт 21, стандартный порт для канала команд FTP. Второй порт используется для передачи FTP-данных между клиентом и сервером. Номер этого порта зависит от того, какой режим (активный или пассивный) используется для канала данных FTP.

Активный режим канала данных FTP

При активном режиме канала данных порт для передачи данных открывает сам FTP-сервер (для передачи команд в любом случае используется порт 21, например для команд получения или сохранения файла). В активном режиме канала данных на сервере для обмена данными открывается порт 20. В качестве данных клиенту может передаваться файл или список файлов определенного каталога.

Рассмотрим следующий отчет TCPdump о сеансе с использованием активного режима канала данных FTP, в котором осуществляется необычная, но нормальная для FTP, смена портов.

Установка FTP-сеанса:

```
ftp.client.com.35955 > ftp.server.com.21: S 1884312222:1884312222(0)
ftp.server.com.21 > ftp.client.com.35955: S 3113925437:3113925437(0)
↳ ack 1884312223
ftp.client.com.35955 > ftp.server.com.21: . ack 1
ftp.server.com.21 > ftp.client.com.35955: P 1:24(23) ack 1
ftp.client.com.35955 > ftp.server.com.21: . ack 24
```

Пользователь применяет команду **dir**:

```
ftp.server.com.20 > ftp.client.com.35956: S 3558632705: 3558632705(0)
ftp.client.com.35956 > ftp.server.com.20: S 1901007864:1901007864(0)
↳ ack 3558632706
ftp.server.com.20 > ftp.client.com.35956: . ack 1
```

В приведенном выше примере FTP-соединение устанавливается между хостом `ftp.client.com` (временный порт `35955`) и сервером (порт `21`). Проводится полная процедура установки соединения, а затем передаются определенные данные (как правило, приветственное сообщение). Это стандартный механизм работы TCP-службы.

Затем пользователь вызывает FTP-команду `dir`, запрашивая с ее помощью список каталогов сервера. С порта `20` сервера устанавливается новое соединение к временному порту `35956` клиента. Хотя этого и не показано в отчете, но клиент с помощью команды **port** предварительно проинформировал сервер, что он будет ожидать получения запросов на временный порт 35956. После завершения этой новой процедуры установки соединения `ftp.server.com` может отправлять данные клиенту `ftp.client.com`. Следующие операции обмена данными потребуют установки новых соединений и привязки к новым временным портам. Этот режим называется *ак-*

тивным, так как передачу данных клиенту инициирует FTP-сервер. Несложно понять, что такой метод вызывает проблемы для устройств фильтрации пакетов, которые должны пропускать весь трафик, поступающий из порта отправителя 20. Пассивный режим канала данных позволяет избежать этих проблем за счет инициирования процесса передачи данных со стороны клиента.

Пассивный режим канала данных FTP

Пассивный и активный режимы канала данных FTP отличаются методом установки соединения для передачи данных. В пассивном режиме для канала команд устанавливается подключение к тому же 21 порту FTP-сервера. Но при активном режиме для возможности передачи данных устройство фильтрации пакетов должно пропускать внешние SYN-пакеты с указанным портом отправителя 20. Эти SYN-пакеты отправляются на временный порт системы (с номером выше 1023), которую защищает фильтр пакетов. Как закрыть этот канал доступа для хакера? В конце концов, фильтр пакетов не в состоянии исследовать содержимое каждого пакета, передаваемого по этому каналу, и невозможно гарантировать, что это действительно FTP-трафик.

При пассивном режиме канала данных эта проблема решается благодаря установке клиентом второго соединения с сервером. Рассмотрим следующий отчет о проведении сеанса установки соединения и обмена данными в пассивном режиме.

```
ftp.client.com.44890 > ftp.server2.com.21: S 4276284026:4276284026(0)
↳ win 8760 <mss 1380> (DF)
ftp.server2.com.21 > ftp.client.com.44890: S 1669630260:1669630260(0)
↳ ack 4276284027 win 8280 <mss 1460> (DF)
ftp.client.com.44890 > ftp.server2.com.21: . ack 1 win 9660 (DF)
```

Пользователь выдает команду **dir**:

```
ftp.client.com.44891 > ftp.server2.com.3967: S 4282611109:4282611109(0)
↳ win 8760 <mss 1380> (DF)
ftp.server2.com.3967 > ftp.client.com.44891: S 1669768808:1669768808(0)
↳ ack 4282611110 win 8280 <mss 1460> (DF)
ftp.client.com.44891 > ftp.server2.com.3967: . ack 1 win 9660 (DF)
```

Когда на хосте `ftp.client.com` выполняется команда `dir`, должен быть установлен канал передачи данных. Хоть этого и не показано в отчете TCPdump, но сервер `ftp.server2.com` с помощью команды `port` информирует клиента о том, что он ожидает запросов на порт `3967`. Клиент отправляет на этот порт SYN-пакет, а сервер отвечает пакетом с установленными флагами SYN и ACK. По этому каналу передается запрошенный список каталогов сервера. Так как клиент сам организовал исходящее соединение к серверу, то устройство фильтрации пакетов может разрешить прием ответной информации от сервера со сравнительно большой уверенностью в "безопасности" соединения. Такой метод менее рискован, чем разрешение поступления любых данных с порта получателя 20.

Traceroute

UNIX-программа Traceroute использует возможности протоколов UDP и ICMP для определения IP-адресов хостов, которые участвуют в процессе передачи дейтаграммы заданному адресату. Промежуточные маршрутизаторы, как и в Tracert, выявляются с

помощью отправки дейтаграмм с увеличивающимся на 1 значением поля TTL (когда TTL становится равным 0, они возвращают ICMP-сообщение "time-exceeded in transit"), пока не будет достигнут хост назначения. UDP-порт получателя обычно выбирается из диапазона 33000–33999 – один из тех портов, которые почти наверняка не используются никакими службами. Это делается с целью получения ICMP-сообщения о недостижимости порта, что служит для Traceroute сигналом о доставке пакета на хост-получатель. В Traceroute по умолчанию также отправляется по три пакета каждому промежуточному маршрутизатору и конечному хосту. В следующем примере для упрощения этот режим заменен отправкой только одного сообщения.

```
tracer.com.62615 > target.com.33456: udp 12 (DF) [ttl1]
router.com > tracer.com: icmp: time exceeded in-transit
[tos 0xc0]
tracer.com.62615 > target.com.33457: udp 12 (DF)
target.com > tracer.com: icmp: target.com udp port 33457 unreachable (DF)
```

В приведенном выше отчете хост `tracer.com` отправляет UDP-пакет на порт `33456` хоста `target.com`. Значение поля TTL первой отправленной дейтаграммы равно 1, маршрутизатор `router.com` уменьшает его до 0 и возвращает отправителю сообщение `time exceeded in-transit`. Затем хост `tracer.com` отправляет следующую дейтаграмму, при этом порт получателя меняется на `33457`, а значение поля TTL становится равным 2. Новая дейтаграмма проходит через первый маршрутизатор и достигает получателя `target.com`, который возвращает ICMP-сообщение о недостижимости своего порта 33457.

Следует заметить, что обе программы – и Traceroute, и Tracert – будут работать только при разрешении возвращения входящих ICMP-сообщений в сеть хоста-отправителя. Возникает вопрос: следует ли разрешать поступление в локальную сеть ICMP-сообщений, используемых при работе этих программ? Это зависит от принятой политики безопасности. На самых защищенных узлах, доступ к которым должен быть максимально ограничен, такое решение не приветствуется. Вероятная опасность превосходит возможные преимущества, так как ICMP-сообщения могут использоваться и хакерами для проведения атак (см. главу 4, "Протокол ICMP", раздел "Программа Loki").

Однако, если безопасность узла менее критична, и можно себе позволить небольшой риск, то разрешение на поступление в сеть этих ICMP-сообщений предоставляет очевидные преимущества для хостов локальной сети: определение маршрута доставки сообщений и канал обратной связи.

Аномальные действия

В этом разделе будут рассмотрены вредоносные варианты использования возможностей протоколов. Каждое из описанных действий предназначено для достижения определенной цели, которые можно разделить на несколько категорий. Здесь дан далеко не полный список всех вариантов атак, их существует гораздо больше.

Незаметное воздействие, отсутствие ответа

В следующем отчете TCPdump зарегистрирован процесс сканирования портов хоста `victim.org`, выполняемый с помощью пакетов с установленным флагом

FIN. Это скрытое сканирование, которое проводится с хоста `stealthy.com`, позволяет определить работающие службы удаленного хоста. Согласно RFC 793 при получении такого пакета открытый порт (на который ожидает запросов запущенная служба) отвечать не должен, а закрытый обязан вернуть пакет с установленными флагами RESET и ACK.

```
stealthy.com.50141 > victim.org.5: F 0:0(0) win 4096 (DF)
stealthy.com.50141 > victim.org.3: F 0:0(0) win 4096 (DF)
stealthy.com.50141 > victim.org.26: F 0:0(0) win 4096 (DF)
stealthy.com.50141 > victim.org.45: F 0:0(0) win 4096 (DF)
stealthy.com.50141 > victim.org.17: F 0:0(0) win 4096 (DF)
stealthy.com.50141 > victim.org.7: F 0:0(0) win 4096 (DF)
stealthy.com.50141 > victim.org.51: F 0:0(0) win 4096 (DF)
stealthy.com.50141 > victim.org.52: F 0:0(0) win 4096 (DF)
stealthy.com.50141 > victim.org.30: F 0:0(0) win 4096 (DF)
stealthy.com.50141 > victim.org.53: F 0:0(0) win 4096 (DF)
stealthy.com.50141 > victim.org.20: F 0:0(0) win 4096 (DF)
```

Некоторые системы обнаружения вторжений не реагируют на FIN-пакеты, поэтому данное сканирование можно назвать более замаскированным, чем сканирование с помощью SYN-пакетов для установки соединения. Раньше для выявления открытых портов использовались только SYN-пакеты, и в разработанных в то время системах обнаружения вторжений контролируются именно такие попытки сканирования. Хакеры быстро поняли, как их обнаруживают, и придумали новый способ сканирования. Для выполнения этого скрытого сканирования с помощью FIN-пакетов была использована команда **`nmap -sF victim.org`**.

Вредоносное воздействие, фатальная реакция

Атаки отказа в обслуживании (DoS) приводят к чрезмерным тратам ресурсов атакованного компьютера, в результате чего он не в состоянии выполнять свои функции. Существует множество различных вариантов DoS-атак. Программа Jolt2 позволяет заблокировать работу удаленного компьютера за счет непроизводительного расходования ресурсов оперативной памяти. Рассмотрим отчет о проведении этой атаки.

```
10:48:56.848099 verbo.com > win98.com: (frag 1109:9@65520)
10:48:56.848099 verbo.com > win98.com: (frag 1109:9@65520)
10:48:56.848295 verbo.com > win98.com: (frag 1109:9@65520)
10:48:56.848295 verbo.com > win98.com: (frag 1109:9@65520)
10:48:56.848351 verbo.com > win98.com: (frag 1109:9@65520)
10:48:56.848351 verbo.com > win98.com: (frag 1109:9@65520)
10:48:56.848420 verbo.com > win98.com: (frag 1109:9@65520)
10:48:56.848420 verbo.com > win98.com: (frag 1109:9@65520)
10:48:56.848584 verbo.com > win98.com: (frag 1109:9@65520)
```

Программа Jolt2 отправляет бесконечный поток эхо-запросов ICMP (ICMP используется по умолчанию, могут применяться пакеты других протоколов) на атакуемый хост под управлением Windows. Эти запросы отправляются как фрагменты с одинаковым идентификатором фрагмента с повторяющимся ненулевым смещением фрагмента.

Ни один из фрагментов не содержит заголовка вложенного протокола, поэтому в данном случае хост-получатель осведомлен только о том, что это фрагменты ICMP-сообщения. Для определенных компьютеров под управлением Windows 98,

Windows NT и Windows 2000 при попытке обработать последовательность фрагментов возникает проблема при отсутствии первого фрагмента со смещением 0. Атакованный хост не в состоянии выполнить сборку пакета, растет потребление оперативной памяти, что приводит к отказу в обслуживании.

Из отчета TCPdump можно узнать, что хост `verbo.com` отправляет какие-то пакеты хосту `win98.com`. При этом идентификатор фрагментов остается постоянным, их длина составляет 9 байт, а смещение равно `65520`. Последнее значение установлено в исходном коде программы Jolt2. Это значение очень близко к предельному значению 65535 байт, поэтому вначале думали, что именно оно позволяет провести успешную атаку. Однако после уменьшения этого значения в исходном коде программы Jolt2 и ее перекомпиляции отказ в обслуживании все равно наблюдался.

Чтобы проверять возможность ответов хоста `win98.com` до и во время атаки Jolt2, с хоста хакера `verbo.com` время от времени отправляются `ping`-запросы. В нашем случае после запуска Jolt2 отказ в обслуживании произошел практически мгновенно. Хост `win98.com` не отвечал ни на внешние запросы, ни на команды с клавиатуры. После прекращения атаки он возобновил свою работу без перезагрузки.

Причины сканирования

Одним из первых этапов в процессе взлома чужого компьютера или сети является проведение разведывательных операций. Может быть, у нарушителя появилась новая программа, которая позволяет ему получить доступ с правами суперпользователя, если ему удастся найти уязвимый хост. Может быть, хакер ищет любой путь проникновения в чужую систему или сеть. У каждого злоумышленника своя собственная цель. Кому-то нужны хосты для проведения распределенной атаки отказа в обслуживании, а кому-то желательно скомпрометировать компьютер, чтобы проводить свои атаки от чужого имени.

Для выявления работающих хостов нарушители используют различные программы зондирования сетей, а затем уже сканируют порты наиболее интересных хостов. Например, злоумышленник раздобыл какую-то программу, которая позволит ему доступ с правами суперпользователя (root) на тех хостах, где запущена уязвимая версия службы DNS. Если при сканировании сети он обнаружит, что на всех ее хостах порт службы DNS находится в состоянии ожидания запросов, то у хакера хорошие шансы на успех атаки. После удачного сканирования можно проверить программу атаки на обнаруженных хостах.

Сканирование может выполняться по ночам, когда контроль за работой сети менее вероятен. Чтобы избежать выявления, хакер может проводить сканирование от имени ранее скомпрометированного хоста или с помощью замаскированных методов, известных как скрытое сканирование. В последнем случае применяются нестандартные методы, обычно не фиксируемые системами обнаружения вторжений. Целью сканирования может оказаться определение типа операционной системы, под управлением которой работает интересующий хост. Это позволит хакеру увеличить вероятность успеха своей атаки.

Воздействие отсутствует, реагируют все

В центре внимания этого раздела находится подмена IP-адресов (более подробная информация содержится в приложении А, "Программы атаки и методы сканирования"). В следующем отчете TCPdump хосты сети `1.2` получают ICMP-сообщения "`time exceeded in-transit`", т.е. уведомления об истечении срока жизни отправленных этими хостами дейтаграмм. Таким образом, хосты сети `1.2` должны были отправить какой-то тарфик, который вызвал ответную реакцию. Но это не так. Никакого исходящего трафика не зафиксировано.

```
router.com > 1.2.10.72: icmp: time exceeded in-transit
router.com > 1.2.18.13: icmp: time exceeded in-transit
router.com > 1.2.11.67: icmp: time exceeded in-transit
```

```
router.com > 1.2.16.13: icmp: time exceeded in-transit
router.com > 1.2.19.1: icmp: time exceeded in-transit
router.com > 1.2.1.252: icmp: time exceeded in-transit
router.com > 1.2.13.56: icmp: time exceeded in-transit
router.com > 1.2.143.6: icmp: time exceeded in-transit
router.com > 1.2.13.15: icmp: time exceeded in-transit
```

Как можно объяснить появление этого трафика? Все очень просто: кто-то подменил IP-адрес хостов своей сети на IP-адреса хостов сети `1.2`. О мотиве создания оригинальных запросов можно только догадываться, но, скорее всего, осуществляется наводнение пакетами или проводится разведка чужой сети.

А не выполняет ли сам хост `router.com` разведывательных действий по отношению к хостам сети `1.2`? Может ли полученный трафик вызывать какие-либо ответы если не локальных хостов, то хотя бы маршрутизатора? Дело в том, что в данном случае передаются ICMP-сообщения об ошибке, а согласно RFC 1122 ICMP-сообщение об ошибке не может послужить причиной отправки другого ICMP-сообщения, так как это может привести к созданию замкнутого цикла обмена сообщениями об ошибке. Поскольку ни один другой протокол не ответит на ICMP-трафик, то предположение о подмене IP-адреса кажется наиболее логичным.

Обратное рассеяние атак

О случаях, подобных приведенному в этом разделе, было сделано очень интересное исследование. Авторы назвали подобные случаи примерами *обратного рассеяния атак* (backscatter). На протяжении длительного периода времени исследовалась работа в Internet сети класса A. Атаки обратного рассеяния определялись по получению незапрошенных ответных пакетов различных протоколов. При этом считалось, что данная ситуация возникла в результате подмены IP-адресов хостов контролируемой сети. На основе собранных данных были определены количество и типы атак, проведенных в Internet за ограниченный период времени. Высокая частота и разнообразие этих атак оказались весьма впечатляющими. Статью "Inferring Internet Denial-of-Service Activity" ("Исследование о проводимых в Internet атаках отказа в обслуживании") можно прочесть по адресу `www.cs.ucsd.edu/~savage/papers/UsenixSec01.pdf`.

Нестандартное воздействие, идентифицирующий ответ

В этом разделе обсуждаются приемы определения операционных систем удаленных компьютеров. Для этого интересующему хосту отправляются нестандартные пакеты, и на основании свойств его ответа делается вывод об используемой операционной системе. Одним из самых популярных средств для данной цели является программа `nmap`.

Знание операционной системы удаленного хоста позволяет хакерам подобрать соответствующие программы атаки, поэтому раскрытие этой информации несет потенциальную угрозу. Некоторые узлы настолько открыты, что о типе и версии используемой на них операционной системы можно узнать по идентификационным маркерам (banner) службы `telnet` или FTP. Конечно, эта информация предоставляется не всегда, а иногда она не соответствует действительности. В каждой операционной системе реализация стека протоколов TCP/IP имеет некоторые отличия. Если хакер отправляет специально подготовленные пакеты и знает, какие характерные признаки следует искать в полученных ответах, то он сможет, например, отличить Linux от Solaris, иногда даже без какой-либо дополнительной разведки.

Ниже перечислены некоторые нестандартные запросы, которые могут использоваться программой nmap для определения типа операционной системы удаленного хоста.

- **Отправка FIN-пакета на открытый порт**. Согласно стандарту RFC 793 для всех открытых портов ответ должен отсутствовать, но некоторые хосты отправляют пакет с установленным флагом RESET. Этот метод также позволяет провести скрытое сканирование портов.
- ***Фальсификация значений "зарезервированных" TCP-флагов***. Программа nmap проверяет, сбросит ли (установит равными 0) операционная система удаленного хоста значения битов несуществующих флагов.
- **Некорректные комбинации TCP-флагов**. Большинство хостов не отвечают при получении пакетов с нестандартной комбинацией TCP-флагов, но по ответам немногих отвечающих хостов можно точно определить операционную систему.
- **Отправка NULL-пакета (без установленных флагов).**

Фиктивные TCP-флаги

Одним из способов получения данных для идентификации операционной системы является отправка пакета с фиктивными значениями TCP-флагов. Байт TCP-флагов содержит информацию обо всех установленных в пакете TCP-флагах (рис. 5.1). По этим флагам определяется назначение конкретного TCP-сегмента. Так как TCP-флагов существует всего шесть, то в байте TCP-флагов остается 2 свободных бита. До изобретения явного уведомления о перегрузке (Explicit Congestion Notification – ECN) значение этих старших битов предполагалось равным 0.

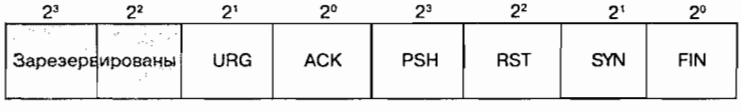


Рис. 5.1. Байт TCP-флагов

Для проверки установки значений всех битов в байте TCP-флагов нужно запустить программу TCPdump с параметром командной строки -x для вывода сохраняемой информации в шестнадцатеричном формате. В стандартных отчетах TCPdump значения двух старших битов байта TCP-флагов не указываются.

Байт состоит из двух шестнадцатеричных символов, или полубайтов (nibbles). Младший полубайт содержит биты, определяющие установку флагов PUSH, RESET, SYN и FIN. Теперь рассмотрим значения зарезервированных битов старшего полубайта. С помощью присвоения значений этим битам несуществующих флагов программа nmap осуществляет тестирование удаленных хостов. Если значение старшего полубайта превышает 3, значит, значение присвоено одному или обоим зарезервированным битам. Значение 3 получается следующим образом: установка флага ACK дает значение 2^0=1, а флага URG – значение 2^1=2. Сумма этих значений равняется 3. Поэтому любое значение старшего полубайта байта TCP-флагов свидетельствует о нестандартной ситуации.

В приведенном ниже отчете TCPdump представлена попытка узнать с помощью nmap характерные особенности стека TCP/IP на удаленном хосте `target.com` и таким образом определить его операционную систему. При попытке организовать рассматриваемое соединение установлен один из несуществующих TCP-флагов (конкретно тот, что на схеме слева от бита URG). Вначале показана строка стандартного отчета TCPdump, из которой нельзя ничего узнать об установке несуществующих флагов. Следующий отчет в шестнадцатеричном формате позволяет получить значения всех полей, включая и значение поля байта TCP-флагов.

```
scanner.com.44388 > target.com.domain: S 403915838:403915838(0) win 4096
↳<wscale 10,nop,mss 265, timestamp 1061109567 0, eol> (DF)

[4500 003c 7542 4000 3b06 15bd 0102 0304
0102 0305] ad64 0035 1813 443e 0000 0000
a042 1000 fa4c 0000 0303 0a01 0204 0109
080a 3f3f 3f3f 0000 0000 0000
```

В последнем отчете первые 20 байт заголовка IP-дейтаграммы заключены в квадратные скобки. Затем следует TCP-заголовок и любые данные. Байт TCP-флагов — это 13-й байт в TCP-заголовке. В шестнадцатеричном формате его значение составляет **42**, при этом значение полубайта высшего порядка больше 3 (равно 4), что означает установку значения для зарезервированного бита. Нарушитель надеется, что ответ на такой пакет с фиктивным TCP-флагом позволит идентифицировать операционную систему удаленного хоста.

Рассмотрим этот ответ хоста `target.com`. Прежде всего и нас, и хакера интересует ответ на фиктивный TCP-флаг. Стандартный TCP-отчет снова оказывается бесполезным, поэтому нужно вывести информацию в шестнадцатеричном формате.

```
target.com.domain > scanner.com.44388: S 4154976859:4154976859(0)
↳ ack 403915839 win 8855 < nop, nop,timestamp 16912287
1061109567,nop,wscale 0,
↳mss 265> (DF)

[4500 003c e04e 4000 ff06 e6af 83da d684
83da d683] 0035 ad64 f7a7 ea5b 1813 443f
a012 2297 fd3f 0000 0101 080a 0102 0f9f
3f3f 3f3f 0103 0300 0204 0109
```

Таким образом, хост `target.com` отвечает на полученный пакет с фиктивным TCP-флагом пакетом с установленными флагами SYN/ACK. Это нормально, и кажется, что исследуемый хост никак не прореагировал на присутствие фиктивного флага. Как это проверить? Отчет о проведенной транзакции в шестнадцатеричном формате показывает, что в байте TCP-флагов установлены флаги SYN и ACK (выделенное жирным шрифтом шестнадцатеричное значение **12**). Бит ACK — это младший бит старшего полубайта, поэтому его значение равно 1. Значение 2 младшего полубайта указывает на присвоение значения второму справа биту (флаг SYN). Следовательно, в ответе фиктивный флаг сброшен. Другая операционная система может оставить без изменений значение бита фиктивного флага.

Некорректные комбинации TCP-флагов

В стандарте RFC 793 установка и возможные комбинации TCP-флагов определены довольно жестко. Стеки протоколов TCP/IP большинства операционных

систем выполняют требования этой спецификации. Но существует и несколько исключений из правила, благодаря чему можно выявить такие операционные системы. Рассмотрим следующий отчет TCPdump о работе программы nmap в режиме определения операционной системы удаленного хоста (запускается с помощью параметра командной строки -O).

```
Nmap -o win 98
20:33:16.409759 verbo.47322 > win98.netbios-ssn: SFP
861966446:861966446(0)
↳win 3072 urg 0 <wscale 10,nop,mss 265, timestamp 1061109567[|tcp]
20:33:16.410387 win98.netbios-ssn > verbo.47322: S 49904150: 49904150(0)
↳ack 861966447 win 8215 <mss 1460> (DF)
```

Сканирующий хост отправляет пакет с установленным набором флагов SYN, FIN и PUSH. Очевидно, что такой набор флагов является некорректным, так как флаг SYN служит для установки соединения, флаг FIN – для его разрыва, а флаг PUSH – для отправки данных при установленном соединении. Естественным ответом на получение подобного бессмысленного пакета кажется его игнорирование или возврат пакета с флагом RESET. Однако в данном случае хост win98 под управлением операционной системы Windows 98 интерпретирует полученный пакет как запрос на соединение и возвращает пакет с установленными флагами SYN и ACK. Этот уникальный ответ позволяет точно определить тип операционной системы хоста win98.

TCP-сегмент без установленных флагов

В следующем отчете TCPdump представлен еще один пример определения операционной системы удаленного хоста. В данном случае используется TCP-сегмент без установленных флагов (так называемый NULL-пакет).

```
scanner.com.44389 > target.com.domain: . win 4096 <wscale 10,nop,mss 265,
↳ timestamp 1061109567 0, eol> (DF)
[4500 003c 7543 4000 3b06 15bc 0102 0304
0102 0305] ad65 0035 1813 443e 0000 0000
a000 1000 fa8d 0000 0303 0a01 0204 0109
080a 3f3f 3f3f 0000 0000 0000
```

Рассмотрим отчет в шестнадцатеричном формате. Значение поля байта TCP-флагов выделено жирным шрифтом (00). Следовательно, не установлено ни одного флага. Большинство хостов не отвечают на NULL-пакет, но находятся отдельные экземпляры, для выявления которых и проводится сканирование данного типа.

В нормальном TCP-пакете установлен по крайней мере один флаг. Хост не отвечает на переданный ему NULL-пакет. И даже отсутствие ответа дает подсказку о типе операционной системы этого хоста, потому что хосты под управлением других операционных систем могут реагировать иначе, например возвратом пакета с установленным флагом RESET.

Использование параметров TCP

При анализе приведенного ниже отчета TCPdump о сканировании с помощью программы nmap сконцентрируем внимание на выделенных жирным шрифтом параметрах TCP.

```
scanner.com.44388 > target.com.domain: S 403915838:403915838(0) win 4096
↳ <wscale 10,nop,mss 265,timestamp 1061109567 0, eol> (DF)
```

```
target.com.domain > scanner.com.44388: S 4154976859:4154976859(0)
↳ ack 403915839 win 8855 < nop, nop,timestamp 16912287
1061109567,nop,wscale 0,
↳mss 265> (DF)
```

Программа nmap обладает еще одним методом определения операционной системы удаленного хоста, который заключается в отправке TCP-пакетов с различными параметрами. Некоторые операционные системы не поддерживают работу со всеми этими параметрами и в своих ответах отбрасывают их. Кроме того, некоторые операционные системы устанавливают собственные значения TCP-параметров, чем еще больше упрощают свое определение.

К тому же различные операционные системы устанавливают эти параметры в заголовке TCP-сегмента в различном порядке. Вся эта информация может быть использована для определения операционных систем удаленных хостов. Как видно из предыдущего примера, в ответе изменились порядок и значения некоторых параметров (например, значение wscale изменилось с 10 на 0). Обратите внимание на то, что параметры nop и eol поменялись местами или вообще пропали. Эти поля используются для дополнения размера TCP-параметров до 4-байтового значения и в ответе могут не потребоваться.

Более подробно значение TCP-параметров описано в RFC 1323. Здесь рассмотрены только некоторые из них.

- -wscale. Позволяет увеличить размер окна до значения выше 65535 байт. Как правило, этот параметр применяется для увеличения производительности работы протокола TCP в сетях с высокой пропускной способностью и большими задержками.
- -timestamp. Этот параметр позволяет сохранить время кругового обращения пакетов, что зачастую необходимо для оптимизации производительности при изменениях в сети.
- -nop. Используется для добавления 1 байт к TCP-параметрам, размер которых должен составлять не менее 4 байт.
- -eol. Параметр конца списка (end-of-list) применяется для дополнения последнего байта до минимального размера в 4 байт.

Мы рассмотрели различные варианты потенциально опасных действий. Каждое из них имеет свою собственную цель. Одни из них предназначены для маскировки от бдительного ока системы обнаружения вторжений и фильтрующих устройств. Другие применяются абсолютно открыто при организации атак отказа в обслуживании.

Нужно учитывать, что иногда за враждебные действия можно принять ответные реакции хоста, при сканировании которого был использован IP-адрес вашего компьютера. Также существуют специальные программы, позволяющие с помощью определенных запросов, выявить операционную систему удаленного хоста.

Резюме

Нельзя с абсолютной точностью предсказать правильный ответ на определенное воздействие. Реализация стека протоколов TCP/IP в различных операционных системах не всегда соответствует требованиям стандартам RFC. Поэтому нестандартный ответ не всегда означает действия злоумышленника.

В теории "воздействие–реакция" есть один очень важный момент. При появлении любого подозрительного трафика делается вывод об атаке со стороны компьютера-отправителя. Я рекомендую немного подумать перед автоматическим принятием этого решения. Нет сомнений, очень часто такое предположение будет правильным. Но не исключено, что этот трафик является следствием другого постороннего воздействия, и его появление связано, например, с подменой хакером вашего IP-адреса.

И, наоборот, при получении каких-либо ответов, например незатребованных эхо-ответов ICMP, весьма вероятно, что хост-отправитель этого трафика выполняет атакующие действия. Как изложено в главе 4, "Протокол ICMP", при атаке TFN эхо-ответы ICMP используются в качестве средства связи между хозяином и демоном для запуска или управления распределенной атакой отказа в обслуживании. Если возникают какие-то сомнения о предназначении получаемого трафика, то лучше всего тщательно исследовать всю сохраненную дейтаграмму и проанализировать содержимое отдельных полей заголовка и данные. При анализе сетевого трафика никогда не стоит действовать на основе одних только предположений.

6

DNS

Зачем посвящать целую главу изучению DNS? Разве не правда, что эта служба используется только для преобразования имен хостов в IP-адреса и наоборот? Да, это большая и важная часть функций DNS, но далеко не все.

DNS-серверы, вероятно, являются одними из самых популярных целей для разведки и атаки хакеров. Скомпрометировать DNS-сервер — значит добиться серьезного успеха, поэтому хакеры упорно ищут их уязвимые места. Основными причинами для атак на DNS-сервер являются следующие.

- На DNS-серверах хранится большой объем полезной информации о хостах локальной сети. Эта информация пригодится хакеру при подготовке к проведению атаки избранной сети.
- Служба DNS управляет преобразованием имен хостов в IP-адреса и наоборот. Поэтому, если хакер будет дублировать работу DNS-сервера или полностью возьмет его под свой контроль, то он сможет манипулировать процессом преобразования имен и адресов в своих целях. Нередко при ненадежных методах аутентификации доступ предоставляется по имени хоста или его IP-адресу. Когда процессом преобразования имен руководит хакер, то такая проверка будет бесполезной.
- DNS-серверы предоставляют свои услуги всем желающим и хранят совместно используемую информацию. Устройства фильтрации пакетов часто пропускают любой трафик на стандартный порт службы DNS (UDP-порт 53) с целью обеспечения нормальной работы внутренних серверов имен.

В этой главе будут изложены теоретические основы и практические методы работы службы DNS. Читатели узнают, как отправляются ответы на DNS-запросы, как взаимодействуют DNS-серверы, как с помощью DNS-сервера можно получить сведения о сетевом узле и об использовании возможностей DNS хакерами. Короче говоря, этот материал поможет больше узнать о безопасной работе в сетях и методах анализа DNS-трафика.

Теория DNS

Для объяснения и иллюстрации различных типов транзакций, проводимых в рамках сеансов службы DNS, снова воспользуемся программой TCPdump. В частности в этом разделе будет описан процесс отправки и получения ответа на DNS-запрос. Служба DNS отличается от обычных приложений типа клиент-сервер (например `telnet`), в которых клиент запрашивает информацию у сервера, и в дальнейшем происходит взаимодействие между этими двумя хостами. Когда DNS-сервер получает запрос от клиента, он может обратиться за требующейся информацией к другим DNS-серверам и только затем возвратить клиенту ответ.

В этом разделе исследуется структура службы DNS как распределенной системы и процесс преобразования имен хостов в их IP-адреса. Кроме того, обсуждаются роли первичных и вторичных DNS-серверов и методы взаимодействия между ними. Вы узнаете, что, в отличие от других служб, DNS может использовать протокол TCP или UDP в зависимости от типа выполняемой операции.

Структура DNS

DNS представляет собой глобальную распределенную систему, которая основана на согласованном взаимодействии большого количества DNS-серверов. Эти DNS-серверы хранят записи о своих доменах и информацию для взаимодействия друг с другом. Домен – это подмножество записей DNS, хранящих информацию о логической группе хостов. На рис. 6.1 представлена иерархическая модель DNS.

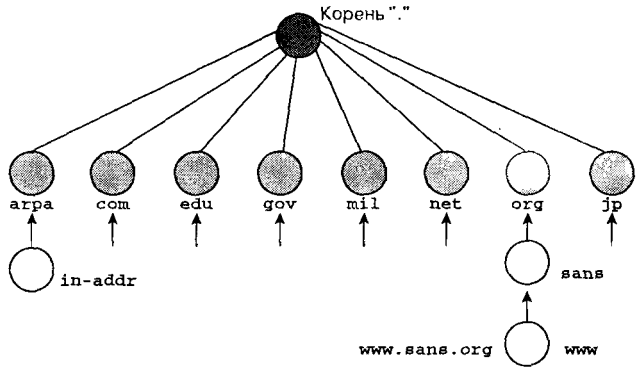


Рис. 6.1. Иерархия DNS

Логически высшей точкой иерархии DNS является корень (root), который обозначается символом "." (точка). С функциональной точки зрения на вершине иерархии службы DNS работают корневые серверы (root servers), которые служат исходным пунктом для определения соответствий между именами хостов и IP-адресами. Эти серверы только указывают на другие DNS-серверы, которые могут хранить ответы на получаемые DNS-запросы. Домены верхнего уровня, расположенные непосредственно ниже корневых серверов, наверняка знакомы нашим читателям (давно известные `.edu`, `.org`, `.com`, `.net`, `.mil` и `.gov` и недавно добавленные `.aero`, `.biz`, `.coop`, `.info`, `.museum`, `.name` и `.pro`). Существуют

также дополнительные домены верхнего уровня, указывающие на принадлежность какому-либо государству, например домен .jp для Японии.

Путешествие по Internet

Предположим, что вы хотите зайти на сайт www.sans.org – домашнюю страничку института SANS. Для этого в окне Address (Адрес) броузера нужно ввести http://www.sans.org, и секундой позже должна открыться запрошенная Web-страничка.

Теперь вспомним, что в IP-дейтаграммах в качестве адресов получателя и отправителя используются IP-адреса. Протокол IP не работает с доменными именами хостов. Человеку проще запомнить, что столицей штата Флорида является город Таллахасси, чем значение числа π с девятью знаками после запятой (3,141592654), хотя оба эти выражения содержат по 10 символов. Имена и названия более благозвучны и менее случайны, чем числа, поэтому и запомнить их проще. Вот почему для обращения к Web-страничке используются имена хостов, а не их IP-адреса. Совершенно ясно, что необходим механизм для преобразования указанных имен хостов в используемые в стеке TCP/IP соответствующие IP-адреса этих хостов.

Как же происходит это загадочное преобразование имени www.sans.org в IP-адрес? Еще до отправки информационного запроса на сайт www.sans.org клиент должен знать его IP-адрес. Этот адрес нужен для поля IP-дейтаграммы, которая используется в качестве запроса на соединение. В следующем разделе этот "туманный" процесс рассмотрен подробнее.

Рекурсивные и итеративные запросы

Различают два типа DNS-запросов: рекурсивные и итеративные. При рекурсивном запросе сервер имен находит ответ самостоятельно. Другими словами, сервер имен может попросить помощи у других DNS-серверов, например, корневых серверов, которые хотя сами и не знают ответов, но хранят ссылки на другие DNS-серверы. Сервер имен будет проверять все предоставленные ему ссылки, пока не обнаружит необходимую ему информацию.

При итеративном запросе сервер имен должен сразу предоставить ответ, не обращаясь к другим DNS-серверам. Если этот сервер не может предоставить запрошенную информацию, он возвращает ссылку на другой сервер имен, который, вероятно, может дать ответ. Задача поиска ответа на запрос перекладывается на первый запрашивающий сервер.

Преобразование имени хоста в IP-адрес

Рассмотрим процесс определения IP-адреса для доменного имени хоста (www.sans.org) с самого начала (рис. 6.2).

Хост host.my.com, на котором работает броузер, должен определить IP-адрес хоста www.sans.org. Если первый хост не является сервером имен, то в этом процессе он не принимает активного участия. Он только выдает запрос на преобразование и после получения IP-адреса продолжает процесс установки соединения со страницей www.sans.org. Всю работу по преобразованию доменного имени в IP-адрес выполняет запрошенный DNS-сервер (в данном случае dns.my.com). Обычно имя DNS-сервера по умолчанию выбирается при установке операционной системы на конкретном хосте. В UNIX-системах эта информация хранится в файле /etc/resolv.conf. В Windows-системах DNS-сервер по умолчанию определяется в окне свойств протокола TCP/IP в разделе Сеть (Network) окна Панель

управления (Control Panel). Как правило, этот DSN-сервер по умолчанию управляется локально и расположен в пределах внутренней сети организации. В данном случае хост dns.my.com – это локальный DNS-сервер сетевого узла.

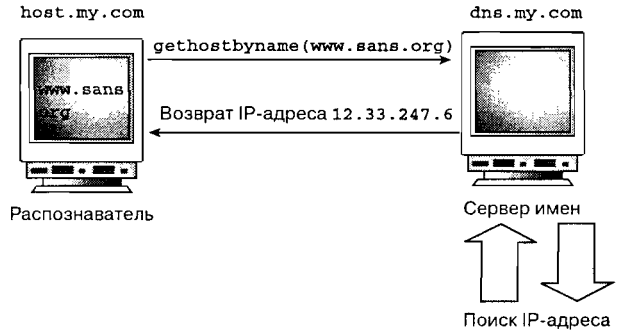


Рис. 6.2. Запрос клиента к DNS-серверу

Такие приложения TCP/IP, как telnet, FTP, Netscape или Internet Explorer, на стороне клиента для преобразования доменных имен в IP-адреса вызывают библиотечные подпрограммы "распознавателя" (resolver). В данном случае поиск IP-адреса для имени хоста выполняется при запросе приложением Web-страницы www.sans.org. Запрос gethostbyname на преобразование имени www.sans.org в соответствующий IP-адрес передается от хоста host.my.com на DNS-сервер. DNS-сервер получает этот запрос, обрабатывает его и возвращает ответ клиенту.

На рис. 6.3 показана вторая часть процесса определения IP-адреса. Ранее DNS-сервер dns.my.com получил задачу узнать IP-адрес хоста по имени www.sans.org. Для сокращения теоретического материала (хотя такое упрощение может усложнить действительный процесс определения IP-адреса) предположим, что dns.my.com не обладает сведениями ни о www.sans.org, ни о каком-либо еще хосте домена .org. Поэтому dns.my.com начинает поиск IP-адреса с запроса к корневому серверу DNS.

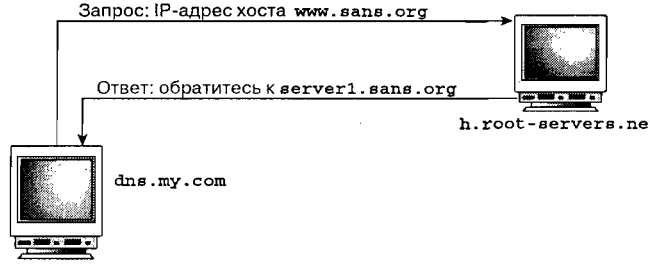


Рис. 6.3. DNS-сервер запрашивает неизвестный IP-адрес

Если DNS-сервер должен определить неизвестный IP-адрес по имени хоста, и он не обладает сведениями о домене интересующего хоста, он должен обратиться к корневому серверу имен. На корневых серверах имен хранятся соответствия между доменными именами (например sans.org) и адресами уполномоченных

серверов имен — DNS-серверов, которые хранят записи DNS для этих доменов. Когда локальный сервер имен (dns.my.com) запрашивает корневой сервер имен IP-адрес хоста www.sans.org, последний возвращает ссылку на сервер имен домена sans.org.

А откуда локальный сервер имен знает имена и IP-адреса корневых серверов имен, к которым он может обращаться? Очевидно, что список этих корневых серверов имен должен быть предварительно сохранен в памяти локального DNS-сервера. Эта информация предоставляется центром InterNIC и ее можно получить по адресу ftp://ftp.rs.internic.net/domain/named.ca.

Итак, корневой сервер имен предоставил локальному DNS-серверу ссылку на сервер server1.sans.org как на уполномоченный сервер имен домена www.sans.org. На последнем этапе определения искомого IP-адреса dns.my.com отправляет запрос серверу server1.sans.org и получает ответ, что IP-адрес интересующего хоста 12.33.247.6 (рис. 6.4).

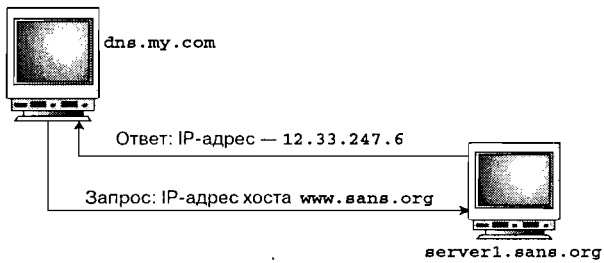


Рис. 6.4. Определение искомого IP-адреса, информация из первых рук

Отчет TCPdump о процессе преобразования

С помощью TCPdump можно исследовать трафик, вызванный исходным DNS-запросом.

```
host.my.com.1716 > dns.my.com.53: 1+ (35)
dns.my.com.53 > h.root-servers.net.53: 12420 (30) (DF)
h.root-servers.net.53 > dns.my.com.53: 12420- 0/3/3 (153) (DF)
dns.my.com.53 > server1.sans.org.53: 12421+ (30) (DF)
server1.sans.org.53 > dns.my.com.53: 12421* 1/3/3 (172)
dns.my.com.53 > host.my.com.1716: 1* 1/3/3 (197) (DF)
```

Сначала host.my.com (операции обмена данными с host.my.com выделены жирным шрифтом) отправляет DNS-серверу dns.my.com запрос на определение IP-адреса хоста www.sans.org. Программа TCPdump анализирует DNS-трафик на уровне приложений, поэтому слово udp, указывающее на применение протокола UDP, не появляется в записях отчета. Протокол UDP используется в подавляющем большинстве операций службы DNS, так как запросы и ответы этой службы обычно коротки, и допускается потеря данных. Если запрошенные данные не поступают, выдается новый DNS-запрос.

Затем dns.my.com устанавливает соединение с портом 53 сервера h.root-servers.net (обратите внимание на то, что и отправитель и получатель используют порты 53). Корневой сервер имен отвечает на тот же 53 порт dns.my.com.

Подробное описание номеров и обозначений, используемых в конце каждой записи TCPdump приведено в разделе "Незнакомые обозначения TCPdump". Сервер не может самостоятельно ответить на запрос. Он возвращает ссылку на другой DNS-сервер, который либо сможет ответить сам, либо предоставит ссылку на следующий DNS-сервер. Поиск IP-адреса для `www.sans.org` представляет собой итеративный процесс, при котором происходит обращение к различным DNS-серверам. Этот процесс продолжается до установки соединения с сервером имен, хранящим нужный IP-адрес.

Согласно полученной ссылке `dns.my.com` обращается с запросом к другому DNS-серверу — `server1.sans.org`. Этот DNS-сервер хранит нужный IP-адрес, соответствующий имени хоста `www.sans.org`, который он и возвращает серверу `dns.my.com`. И, наконец, локальный DNS-сервер `dns.my.com` передает полученный ответ клиенту `host.my.com`.

Результаты работы TCPdump выводятся в уникальном формате, который позволяет проанализировать действия при DNS-соединениях. Следующий раздел поможет понять все подробности этих отчетов TCPdump.

Незнакомые обозначения TCPdump

Рассмотрим подробнее отчет об обмене информацией между хостами `dns.my.com` и `h.root-servers.net`.

```
dns.my.com.53 > h.root-servers.net.53: 12420 (30) (DF)
h.root-servers.net.53 > dns.my.com.53: 12420- 0/3/3 (153) (DF)
```

Первая строка — это запрос от `dns.my.com` к корневому серверу имен. Первое поле, которое не встречалось в стандартных отчетах TCPdump, представляет собой номер `12420`, указанный после номера порта 53 и двоеточия. Это уникальный идентификационный номер DNS, по которому клиент и сервер DNS определяют соответствующие запрос и ответ. Хост `dns.my.com`, скорее всего, одновременно генерирует множество различных запросов, поэтому необходим механизм для определения соответствия ответа и запроса. После этого в строке указывается, что длина полезных данных UDP-пакета (не считая заголовков IP или UDP) составляет 30 байт, а флаг DF установлен для запрещения фрагментации дейтаграммы.

Затем поступает ответ на запрос под номером 12420. Тире после числа `12420` означает, что рекурсия не требуется. Другими словами, сервер `dns.my.com` просит только передать ему ссылку на следующий в процессе поиска DNS-сервер, он не требует, чтобы корневой сервер имен проводил поиск самостоятельно.

Корневые серверы имен обрабатывают огромное количество DNS-запросов и поэтому не могут обрабатывать рекурсивные запросы. Ответы корневых серверов имен предназначены только для предоставления надежных ссылок для обработки запроса. Если заблудившийся в большом городе человек подойдет к полицейскому, который регулирует движение на оживленном перекрестке, и спросит его, как найти конкретный дом в городе, то он вряд ли получит точный ответ. Все, на что можно надеяться, — это получить сведения о месте, где могут дать более четкие ориентиры.

В ответе корневого сервера имен содержится непонятная запись `0/3/3`. Она означает, что записей ответа нет (т.е. нужного IP-адреса не найдено), были обнаружены три записи о полномочиях и три дополнительные записи. Уполномочен-

ный сервер хранит записи соответствий имен хостов и IP-адресов для хостов своего домена. Хотя в отчете TCPdump этого не показано, но в этих записях (0/3/3) предоставляются имена уполномоченных DNS-серверов, а также соответствующие IP-адреса.

Записи о полномочиях

sans.org	nameserver = server1.sans.org
sans.org	nameserver = ns.BSDI.COM
sans.org	nameserver = ns.DELOS.COM

Дополнительные записи

server1.sans.org	Internet address = 167.216.198.40
ns.BSDI.COM	Internet address = 206.196.44.241
ns.DELOS.COM	Internet address = 65.102.83.117

В разделе "Использование DNS в разведывательных целях" продемонстрировано, как получить эту информацию с помощью команды **nslookup**. Благодаря отправке в дополнительных записях IP-адресов уполномоченных серверов имен не требуется дополнительных запросов на преобразование имен этих DNS-серверов в их IP-адреса. Все эти серверы обладают полномочиями на домен sans.org и предоставляют ответы на запросы определения IP-адресов хостов этого домена. В нашем случае сервер dns.my.com выбрал из списка первый официальный DNS-сервер, server1.sans.org.

В заключение рассмотрим последнюю часть отчета TCPdump о процессе определения IP-адреса.

```
dns.my.com.53 > server1.sans.org.53: 12421+ (30) (DF)
server1.sans.org.53 > dns.my.com.53: 12421* 1/3/3 (172)
dns.my.com.53 > host.my.com.1716: 1* 1/3/3 (197) (DF)
```

Таким образом, сервер dns.my.com был проинформирован о существовании нескольких уполномоченных DNS-серверов и для определения IP-адреса был выбран первый из них. Локальный сервер отправляет новый запрос под номером 12421. На этот раз запрос является рекурсивным (на что указывает знак плюс). По существу, dns.my.com ставит перед server1.sans.org задачу найти нужный ему IP-адрес. В данном случае server1.sans.org является официальным сервером для www.sans.org, поэтому он может самостоятельно ответить на запрос. В противном случае ему бы пришлось отправлять рекурсивные запросы другим DNS-серверам до тех пор, пока не был бы найден искомый IP-адрес. Не все DNS-серверы настроены на отправку рекурсивных запросов, поэтому даже при необходимости рекурсии она возможна не всегда.

DNS-сервер server1.sans.org отвечает на запрос. Символ звездочки указывает на ответ, содержащий сведения из достоверного источника. В данном случае возвращается IP-адрес (12.33.247.6) хоста www.sans.org. IP-адрес не показан в отчете TCPdump, но он входит в состав полезной нагрузки UDP-пакета. Также возвращаются рассмотренные выше записи о полномочиях и дополнительные записи. Наконец, после получения сервером dns.my.com искомого IP-адреса, он возвращает его первоначальному источнику запроса — хосту host.my.com.

Кэширование

В этом разделе будет вкратце рассказано, что происходит с полученными ответами на DNS-запросы. DNS-серверы сохраняют или кэшируют полученные DNS-ответы. Это повышает эффективность работы службы DNS и позволяет снизить количество обращений других DNS-серверов, ведь одни и те же DNS-запросы не должны повторяться снова и снова. Вероятность повторения DNS-запроса для определения IP-адреса одного и того же хоста является довольно высокой. Тем не менее, как будет показано в разделе "Подмена содержимого кэша", такой метод имеет определенные недостатки со стороны безопасности, если не подтверждается аутентичность кэшируемых ответов, и они не соответствуют действительности.

При повторном запросе Web-страницы `www.sans.org` через небольшой период времени процесс определения нужного IP-адреса будет несколько иным. Клиент по-прежнему будет отправлять запрос `gethostbyname` DNS-серверу `dns.my.com`. Однако этот сервер имен перед запросом к другим DNS-серверам проверяет свой кэш. Если прошло не слишком много времени, то `dns.my.com` находит нужную запись в кэше и возвращает IP-адрес хосту `host.my.com`.

Как долго кэшированные записи хранятся на DNS-сервере? Точного ответа нет. Каждая отдельная запись может иметь собственный срок хранения на сервере имен. Каждому ответу (записи DNS) присваивается значение времени жизни (TTL). Не путайте это значение со значением поля TTL в заголовках IP-дейтаграмм. Они выполняют абсолютно разные функции. Значение TTL устанавливается DNS-сервером, который отвечает на запрос и кэшируется сервером-получателем. На DNS-серверах, которые часто обновляют свои записи, устанавливаемые значения TTL, как правило, меньше, чем на DNS-серверах, информация которых относительно статична.

BIND

BIND (Berkeley Internet Name Domain) представляет собой де-факто стандартную реализацию службы DNS в современной сети Internet. Старыми версиями BIND являются 4.x.x, а более новые — 8.x.x и 9.x.x. Использование взаимодействующими DNS-серверами в качестве портов получателя и отправителя порта 53 указывает на стандартный режим работы версии BIND 4.x.x. По умолчанию в версиях BIND, начиная с 8.x.x, при отправке DNS-запросов используется один из временных портов (номер больше 1023), как это делается в других приложениях типа клиент-сервер, например `telnet`.

Тем не менее более поздние версии BIND могут быть настроены на стандартный метод работы версии 4.x.x с использованием порта отправителя 53. Для этого в конфигурационный файл добавляется строка `query-source address * port 53`. На некоторых узлах такая конфигурация считается более удобной для соответствия правилам доступа, установленным для брандмауэра или фильтрующего маршрутизатора.

Обратные запросы

Иногда необходимо обратное преобразование IP-адреса в доменное имя соответствующего хоста. Для этого распознаватель клиента выдает запрос `gethostbyaddr`.

Напомним, что DNS представляет собой распределенную иерархическую систему, вершиной которой является корень. Ниже корня располагаются домены верхнего уровня, например `.org`, `.mil`, `.edu` и др. С целью преобразования IP-адресов в имена хостов был зарезервирован специальный домен верхнего

уровня, для указания на который используется суффикс arpa. Уровень второго уровня этого же предназначения называется in-addr. Затем следуют байты IP-адреса. Эту иерархическую структуру можно представить в виде дерева (рис. 6.5). Пусть, например, значением первого байта нашего IP-адреса будет число 12. Затем выбирается следующее поддерево согласно значению второго байта IP-адреса хоста www.sans.org (33). По этой же схеме выполняется поиск следующих поддеревьев (узлы 247 и 6). В данном примере по IP-адресу мы находим имя только одного хоста, но сама древовидная структура охватывает все возможные IP-адреса подобно тому, как иерархия доменных имен позволяет указать доменное имя любого хоста.

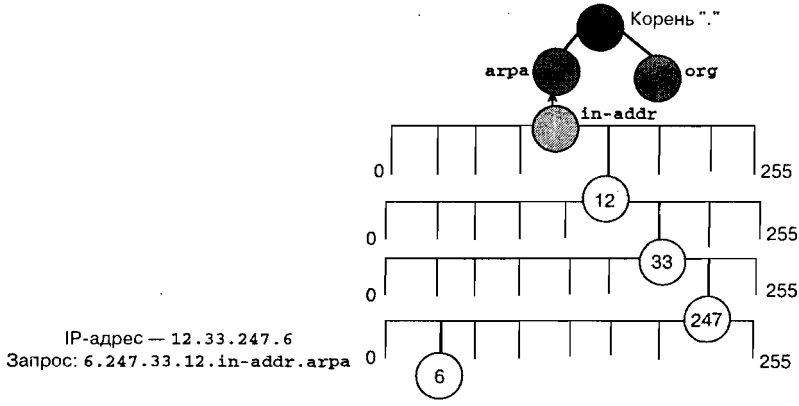


Рис. 6.5. Обратный запрос. Определение имени хоста по IP-адресу

Определение имени хоста по его IP-адресу называют *обратным запросом*. При обратном запросе по IP-адресу 12.33.247.6 приложение преобразует этот запрос в форму 6.247.33.12.in-addr.arpa. Порядок байтов изменяется, чтобы соблюдалось соответствие с принятым порядком записи доменных имен (когда домен высшего уровня указывается последним).

Первичные и вторичные серверы имен

В каждом домене должен быть главный (первичный) сервер, на котором хранятся записи базы данных доменных имен и соответствующих IP-адресов. Для обеспечения избыточности в случае выхода первичного сервера из строя создаются один или несколько вторичных серверов сервер. Если не дублировать первичный DNS-сервер, то при выходе его из строя не будут обслуживаться запросы всех хостов домена, а также внешние запросы на определение имен и IP-адресов хостов этого домена. Вторичные серверы могут взять на себя часть работы по ответам на запросы.

Информация службы DNS хранится на первичном сервере в виде обычных текстовых файлов. Для получения обновлений вторичные серверы имен периодически связываются с первичным DNS-сервером. Если в базу данных DNS были внесены какие-либо изменения, то вторичные серверы, на которых установлены старые версии BIND, полностью копируют всю базу данных. В последних версиях

BIND предусмотрена возможность частичных обновлений, когда копируются только измененные записи.

Перенос зоны

Рассмотрим механизм обновления информации зоны DNS на DNS вторичных серверах. Если при перезапуске или очередном запросе на обновление базы данных вторичного сервера будут обнаружены какие-либо изменения, то выполняется перенос зоны между первичным и вторичным DNS-сервером.

При этом на вторичный сервер просто копируются записи базы данных DNS. В отличие от большинства других операций DNS в данном случае используется протокол TCP, так как требуется надежная передача большого объема данных. Перенос зоны кажется вполне безопасным процессом. Но что будет, если хакер сможет осуществить перенос зоны и получит информацию базы данных чужого DNS-сервера? В распоряжении нарушителя окажутся все имена и IP-адреса хостов целого домена. Это очень важные данные, которые должны быть скрыты от посторонних глаз.

Очевидно, что возможность проведения подобных действий необходимо исключить. В программе BIND начиная с версии 4.9.3 администратор службы DNS имеет возможность указать IP-адреса хостов или подсетей, которым разрешается выполнять перенос зоны. Для контроля операций переноса зоны в BIND 4.9.x предусмотрена директива `xfernets`, а в версиях BIND 8 и 9 — оператор `allow-transfer`.

Если ваша версия BIND не поддерживает этой возможности, то в качестве альтернативы можно использовать блокирование входящего трафика на TCP-порт 53. Это блокирование не только позволяет предотвратить несанкционированные переносы зоны, но и блокирует все полезные операции. Если альтернативы нет, все же лучше блокировать переносы зоны, даже в ущерб другим возможностям.

UDP или TCP

Как уже отмечалось, при работе службы DNS в основном используется протокол UDP, поскольку ответы на запросы DNS достаточно кратки, а потеря пакетов приведет только к их повторной отправке. При переносе зоны требуется надежная передача большого количества данных, поэтому в виде исключения используется протокол TCP.

Максимально допустимый размер UDP-пакета для службы DNS составляет 512 байт. Что произойдет если передаваемый пакет больше этого размера? Во-первых, будет отправлен ответ с установленным битом усечения данных. Этот бит находится в поле флагов сообщения DNS (2 и 3 байт от начала сообщения).

```
dns.my.com.53 > dns.verbose.com.53: 18033 (43) (DF)
dns.verbose.com.53 > dns.my.com.53: 18033| 7/0/0 (494)
dns.my.com.37404 > dns.verbose.com.53: S 518696698:518696698(0) win 8760
↳ <mss 1460> (DF)
dns.verbose.com.53 > dns.my.com.37404: S 199578733:199578733(0) ack 518696699
↳ win 8760 <mss 1460> (DF)
```

В приведенном выше отчете TCPdump обратите особое внимание на вторую строку (ответ `dns.verbose.com.53` хосту `dns.my.com`). После идентификатора сообщения DNS (`18033`) следует вертикальная черта — символ конвейеризации UNIX. Это предупреждение TCPdump о том, что запись DNS была урезана. Дан-

ный ответ, содержащий семь DNS-записей, превысил максимально допустимый размер пакета (512 байт). Как мы видим, в ответе содержалось 494 байт полезных данных, что превышает максимально допустимый размер данных в 484 байт (20 байт требуются для заголовка IP-дейтаграммы, и еще 8 — для заголовка UDP-пакета). По этой причине `dns.my.com` отправляет повторный DNS-запрос, используя протокол TCP. Сначала серверу `dns.verbose.com` отправляется SYN-запрос на установление соединения. В ответ поступает пакет с установленными флагами SYN и ACK, что свидетельствует о готовности принять запрос на TCP-порт 53. Затем необходимая информация передается с помощью TCP-пакетов.

Если с целью запрещения несанкционированных переносов зоны блокировать все входящие пакеты, в которых указан TCP-порт 53 отправителя или получателя, то автоматически удаленные DNS-серверы не смогут получать ответов большой длины. Именно это и происходит в нашем примере. Ответный пакет с установленными флагами SYN и ACK будет заблокирован (четвертая строка отчета). Наше устройство фильтрации пакетов, установленное перед хостом `dns.verbose.com`, блокирует TCP-соединение, для которого указан порт отправителя `domain` (53). Таким образом, эта процедура установки соединения не будет завершена, и DNS-ответ доставлен не будет. Поэтому лучше блокировать прохождение только тех пакетов, в которых указан TCP-порт получателя 53, и разрешать их поступление с порта отправителя 53 при установленном соединении.

Использование DNS в разведывательных целях

DNS представляет собой глобальную базу данных, хранящуюся на множестве серверов. Это прекрасный источник информации. Служба DNS задумывалась для совместного свободного использования всеми пользователями глобальной сети в духе сотрудничества и взаимопомощи. В начале развития Internet с этим можно было согласиться. Но теперь при постоянных атаках хакеров служба DNS превращается в инструмент получения ценных сведений. Ниже рассказывается о некоторых способах применения DNS в разведывательных целях.

Команда nslookup

Работа утилиты `nslookup` аналогична действиям клиента службы DNS, с той разницей, что вся информация выводится на экран. С помощью команды `nslookup` и были получены имена и IP-адреса уполномоченных DNS-серверов домена `sans.org`. Утилита `nslookup` является очень полезным интерактивным средством, которое может использоваться на хостах под управлением UNIX и Windows NT (и следующих сетевых версиях Windows). В некоторых версиях операционной системы UNIX вместо команды `nslookup` применяется команда `dig`.

Запросы к службе DNS позволяют узнать не только доменные имена хостов. С помощью `nslookup` можно формировать различные запросы к DNS-серверам, а специальный параметр настройки позволяет получать из ответов некоторую дополнительную информацию.

Рассмотрим возможности nslookup на следующем примере. Команда nslookup была вызвана на хосте host.my.com. Запускается механизм взаимодействия и принимается сообщение о DNS-сервере по умолчанию (dns.my.com) и его IP-адрес (192.168.4.4), которые будут использоваться при обработке запросов.

```
host.my.com% nslookup
Default Server: dns.my.com
Address: 192.168.4.4

> www.sans.org
Server: dns.my.com
Address: 192.168.4.4

Name: www.sans.org
Address: 12.33.247.6
```

После символа приглашения (>) вводится имя хоста, IP-адрес которого требуется найти. Еще раз повторяются имя и IP-адрес DNS-сервера, использующегося для обработки запроса. Ответом является IP-адрес 12.33.247.6.

Доменное имя DNS-сервера

Как можно определить DNS-сервер какого-либо домена? Используя nslookup, сделать это довольно просто.

```
> set type=ns
> sans.org
Server: dns.my.com
Address: 192.168.4.4

Non-authoritative answer
sans.org        nameserver = NS.DELOS.COM
sans.org        nameserver = server1.sans.org
sans.org        nameserver = NS.BSDI.COM

Authoritative answers can be found from
NS.DELOS.COM            Internet address = 65.102.83.117
server1.sans.org        Internet address = 167.216.198.40
NS.BSDI.COM             Internet address = 206.196.44.241
```

После введения команды nslookup можно набрать следующую команду set type=ns, благодаря которой устанавливается режим возврата имен и IP-адресов всех DNS-серверов, которые будут использованы при дальнейших запросах. Теперь, введя имя sans.org, мы получим полную информацию о DNS-серверах указанного домена. Таким же образом, хакер может определить наиболее интересные цели атаки и провести их сканирование, чтобы обнаружить пробелы в системе защиты или узнать, какие службы или демоны запущены на этих DNS-серверах.

HINFO

Записи HINFO — это еще один тип записей службы DNS. В этих записях хранится потенциально опасная информация, которая может заинтересовать хакера. При создании новой DNS-записи администратор может указать сведения о соответствующем компьютере, в частности тип его центрального процессора и установленную операционную систему. При использовании DNS-сервера для обслуживания хостов локальной сети такой способ позволяет легко определять тот или иной хост, а потенциальный риск совсем невелик.

Из-за того, что записи HINFO предоставляют слишком много полезной информации неизвестным пользователям Internet, многие администраторы в целях безопасности вообще не заносят в них никакой информации.

```
> set type=hinfo
> host49
Server: dns.my.com
Address: 192.168.4.4

host.49.my.com CPU = SunSparc   OS = Solaris
my.com nameserver = dns.my.com
dns.my.com     Internet address = 192.68.4.4
```

Для запроса информации записей HINFO после ввода команды `nslookup` укажите тип запросов `set type=hinfo`. В данном случае запрашивается информация относительно хоста `host49` (здесь указан псевдоним реального хоста). Как мы видим, на хосте `host49.my.com` используется процессор SunSparc и работает он под управлением операционной системы Solaris. Возможно, что все усилия хакера будут тщетны по той причине, что информация записей HINFO просто устарела. Это, пожалуй, один из немногих случаев, когда несвоевременное обслуживание имеет положительный эффект.

Доступ к информации о домене

Одним из самых простых способов для получения информации о хостах домена является запрос на вывод списка всех хостов домена. Предположим, что существует домен `fakeplace.com`. После ввода команды `nslookup` с помощью следующей подкоманды `ls` можно попытаться получить все записи DNS об этом домене.

```
> ls -d fakeplace.com
```

Если на узле сети Internet не запрещено распространение информации, то в ответ на эту команду DNS-сервер предоставит все записи для домена `fakeplace.com`. В данном случае заинтересованное лицо дополнительно получит информацию записей HINFO.

```
Whish       1D IN HINFO     "SGI" "Irix"
1D IN A                 192.168.1.239
susie       1D IN HINFO     "IBM-RS/560F" "unix"
1D IN A                 172.16.16.13
pixie       1D IN HINFO     "IBM-RS/560F" "unix"
1D IN A                 172.12.16.14
bandit      1D IN HINFO     "PC" "Win98"
1D IN A                 192.168.3.107
adder       1D IN HINFO     "IBM-RS/530" "unix"
1D IN A                 172.16.133.4
hub21       1D IN HINFO     "Cabletron-MMAC3" "SNMP"
1D IN A                 192.168.26.80
switch3     1D IN HINFO     "Switch" "3COM"
1D IN A                 192.168.7.130
```

Получение этой информации возможно только в том случае, если разрешается беспрепятственное прохождение пакетов к TCP-порту 53.

Утилита dig

Еще одним способом сбора информации является запрос к DNS-серверу об используемой им версии BIND.

```
dns.my.com% dig @MYDNS.COM version.bind txt chaos

; <<>> dig 8.1 <<>> @MYDNS.COM version.bind txt chaos
; (1 server found)
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0,
   ADDITIONAL: 0
;; QUERY SECTION:
;;      version.bind, type = TXT, class = CHAOS

;; ANSWER SECTION:
VERSION.BIND          OS CHAOS TXT     "4.9.7-REL"
```

Утилита `dig` входит в состав многих версий BIND. Ее возможности практически аналогичны возможностям `nslookup`. С помощью специального параметра командной строки можно узнать версию BIND, запущенную на удаленном DNS-сервере. Формат этой команды выгладит следующим образом.

```
dig@<имя_DNS-сервера> version.bind txt chaos
```

Здесь слово `txt` обозначает запрос из базы данных DNS-сервера информации записей TXT. Завершает запрос слово `chaos` — практически устаревший тип запросов.

В нашем случае с помощью команды `dig` была получена версия BIND хоста `mydns.com`. На этом хосте запущена устаревшая версия BIND 4.9.7, т.е. полученная информация окажется для хакера весьма полезной, и он сможет подобрать подходящую атаку для уязвимой версии BIND. Начиная с версии BIND 8.2 в конфигурационный файл `/etc/named.conf` включен оператор `options`, с помощью которого можно определить сообщение, выдаваемое в ответ на запрос номера версии. Можно ввести сообщение, подобное "unknown version of BIND" ("неизвестная версия BIND"). Также можно обмануть нарушителей, указав неправильную версию BIND.

Опасные ответы DNS

В своей обычной работе DNS-сервер должен взаимодействовать со многими неизвестными хостами. Приходится только надеяться на то, что получаемый ответ будет безопасным. К сожалению, это не всегда правда. В данном разделе изложены различные проблемы, связанные с аутентичностью получаемых записей DNS.

Слабое звено

Одним из недостатков использования имен хостов при разрешении или запрещении доступа к той или иной службе является то, что если хакеру удастся подменить имя чужого хоста, то вся система аутентификации окажется бесполезной. Это касается аутентификации как по именам хостов, так и по полным доменным именам. Разрешается ли в вашей локальной сети доступ к Web-серверу всем хостам ва-

шего домена? Или в вашей сети работают UNIX-хосты, доступ к которым осуществляется без проверки имени и пароля пользователя, только на основе имени доверенного хоста? Такой метод аутентификации может быть очень рискованным, если хакеру удастся проводить свои действия от имени доверенного хоста. Имя хоста можно подменить на самом хосте, на скомпрометированном DNS-сервере или на DNS-сервере, на котором временно подменена запись в кэше.

В BIND начиная с версии 8.3 встроена поддержка протокола DNSSEC (DNS Security Extensions – расширения безопасности DNS), который обеспечивает основанный на криптографических сигнатурах механизм аутентификации DNS-хостов и позволяет гарантировать целостность и подлинность DNS-данных. Для гарантии подлинности набора ответов DNS-сервера этот сервер зашифровывает хэшированный набор DNS-ответов с помощью секретного ключа зоны DNS. Эта сигнатура возвращается распознавателю в качестве новой записи, имя которой SIG. Распознаватель должен получить открытый ключ DNS-сервера для соответствующей зоны, используя другую запись о ресурсах KEY. Затем сервер-получатель дешифрует сигнатуру, чтобы получить исходные хэшированные данные, и вычисляет собственный хэш полученного набора ответов, используя тот же алгоритм, что и DNS-сервер отправителя. Если результат вычисления совпадает с полученным значением от сервера, то данные считаются достоверными.

Подмена содержимого кэша

Бюллетень группы CERT (Computer Emergency Response Team – группа компьютерной "скорой помощи") CA-97.22 за август 1997 года предупреждает о существовании уязвимых мест в различных версиях BIND. В версиях до 8.1.1 существует возможность подмены данных кэша с другого DNS-сервера. Иначе говоря, злоумышленник может использовать один DNS-сервер для добавления некорректных записей в кэш другого DNS-сервера.

Для этой цели нарушителю сначала необходимо заставить атакуемый DNS-сервер обратиться с запросом ко взломанному им серверу имен. Сам запрос не представляет никакой угрозы, но в ответе содержатся подготовленные хакером записи DNS, которые заносятся в кэш атакуемого сервера имен.

После успешной операции подмены на все запросы к скомпрометированному DNS-серверу будет выдаваться информация подложных записей, т.е. нарушится соответствие между действительными именами хостов и их IP-адресами.

Другой метод подмены содержимого кэша основан на использовании отправки ответов совместно с записями запросов. Для передачи запросов и ответов DNS используется один и тот же формат сообщений. В некоторых уязвимых версиях BIND кэшируются все данные из раздела ответа сообщения DNS. При этом не проверяется, запрос это или ответ. Злоумышленник может отправить на уязвимый DNS-сервер запрос, содержащий подложные записи ответа, которые и будут занесены в кэш этого DNS-сервера.

На рис. 6.6 показан пример подмены содержимого кэша DNS-сервера. Предположим, что нарушитель формирует сообщение DNS, содержащее в запросе запись ответа. Затем нарушитель отправляет этот запрос с хоста `evil.dns.net` на атакуемый DNS-сервер `ns04.baweb.com` – уполномоченный сервер имен для домена `www.hillary2000.org`.

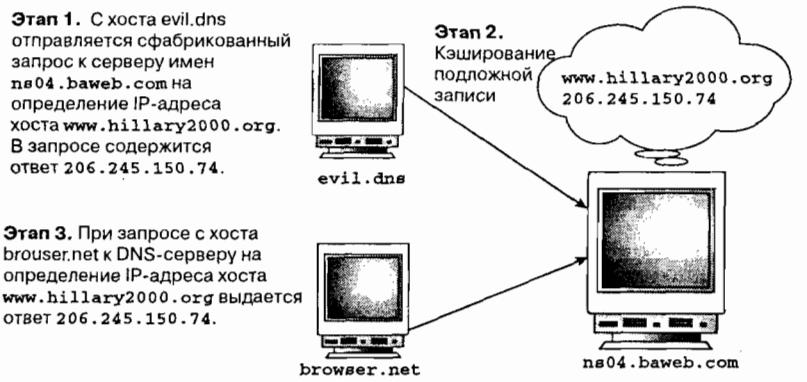


Рис. 6.6. Подмена содержимого кэша DNS

Созданный нарушителем пакет кроме запроса на определение IP-адреса хоста `www.hillary2000.org` содержит в части ответа DNS-сообщения IP-адрес `206.245.150.74`. Как скоро станет ясно, это не настоящий IP-адрес хоста `www.hillary2000.org`.

DNS-сервер `ns04.baweb.com` не в состоянии отличить запрос от ответа и поэтому просто кэширует ответ, переданный в запросе. В кэш заносится подложное соответствие имени хоста и IP-адреса. На последнем этапе этой подмены какой-либо пользователь или процесс должен запросить IP-адрес для хоста `www.hillary2000.org`. Ответ будет выдан из кэша — `206.245.150.74`.

Это реальный пример из политических войн с использованием компьютерных технологий. В июле 1999 года был создан Web-сайт Хиллари Клинтон, информация которого касалась ее предвыборной кампании на место в сенате США от штата Нью-Йорк.

Однако, когда пользователи хотели зайти на этот сайт, открывалась Web-страничка `www.hilllaryno.com` (IP-адрес `206.245.150.74`) сторонников конкурента Хиллари Клинтон, мэра Нью-Йорка Рудольфа Джулиани (в то время Рудольф Джулиани еще не принял решения по поводу борьбы за место в сенате; позже он снял свою кандидатуру).

Как оказалось, это происходило из-за успешной атаки подмены содержимого кэша DNS-сервера. Другими словами, для доменного имени `www.hillary2000.org` был поставлен в соответствие IP-адрес хоста `www.hilllaryno.com`. Естественно, что администраторы сайта `www.hilllaryno.com` отрицали свою причастность к происходящему. Отследить и доказать подмену кэша очень тяжело.

Резюме

Служба DNS представляет собой иерархическую систему серверов имен, в базах данных которых хранится информация, необходимая для преобразования доменного имени хоста в IP-адрес и наоборот. В отличие от большинства других приложений типа клиент-сервер для обработки запроса к службе DNS может потребоваться совместная работа нескольких DNS-серверов, при этом для установления различных соединений возможно применение протоколов TCP или UDP.

DNS-серверы часто используются хакерами для сбора полезной информации. Из-за появления в сети Internet многочисленных нарушителей первоначальную открытость службы DNS следует ограничивать. В целях безопасности DNS-серверы должны предоставлять только минимально необходимую информацию.

История программы BIND пестрит уведомлениями об обнаружении уязвимых мест. За последние годы появилось несколько программ атаки (на основе переполнения буфера), которые позволяли хакерам получить права суперпользователя на хосте с запущенной программой BIND. Но работа в современной сети Internet практически невозможна без использования в той или иной степени возможностей DNS. Желательно организовать надежную защиту своего DNS-сервера и не доверять всем поступающим ответам DNS. Постоянно обновляйте программное обеспечение DNS-сервера. Это позволит воспользоваться наиболее эффективными методами обеспечения безопасности и ограничить объем потенциально опасной информации, предоставляемой вашим DNS-сервером.

II

Анализ трафика

В этой части...

7

Содержимое пакетов

В следующих четырех главах для анализа сетевого трафика используется программа TCPdump. Это средство предоставляет ряд великолепных возможностей при совместном использовании с сетевыми системами обнаружения вторжений (NIDS), основанными на создании сигнатур. В большинстве случаев выявление враждебных действий такими системами обнаружения вторжений осуществляется по предопределенной сигнатуре вредоносного пакета. При обнаружении потенциально опасного трафика системы NIDS выдают сигнал тревоги и перехватывают подозрительные пакеты.

Такой метод имеет несколько недостатков. Во-первых, любой, кто когда-либо использовал системы обнаружения вторжений, знает, что они очень часто сигнализируют о ложных тревогах. Это происходит из-за недостаточной точности заданных сигнатур и отсутствия анализа содержимого предшествующих или последующих пакетов.

Поэтому очень полезно иметь средство наподобие TCPdump, которое позволяет фиксировать входящий и исходящий трафик. Информация, сохраненная таким средством, чем-то напоминает журнал регистрации проходящего трафика. Хотя по умолчанию TCPdump не сохраняет все содержимое проходящих пакетов, но минимально необходимые сведения можно узнать и по заголовкам (откуда пришел пакет, адрес назначения, цель передачи). Системы обнаружения вторжений могут самостоятельно проверять наличие отклонений в полезных данных пакетов.

С помощью отчетов TCPdump можно отличить реальную угрозу от ложной тревоги. Если ваша система обнаружения вторжений выдает вам сведения о сработавшей сигнатуре и пакет, который вызвал тревогу, то с помощью TCPdump можно исследовать данные этого пакета. Кроме того, правильное решение поможет принять информация, сохраненная TCPdump до и после сигнала о тревоге. В нашей практике очень часто записи TCPdump позволяли уточнить причину проблемы и наиболее точно оценить серьезность тревоги, поднятой системой обнаружения вторжений.

Основная цель материала следующих четырех глав — научить наших читателей выполнять квалифицированный анализ сетевого трафика без помощи систем обнаружения вторжений.

Мы начнем с самого начала: рассмотрим пакет на уровне битов. В главе 8, "Исследование полей IP-заголовка", будет рассказано, как исследовать пакет в тех редких случаях, когда анализатор пакетов не в состоянии справиться с этой задачей. В следующих главах 9, "Анализ заголовков вложенных пакетов", и 10, "Практический анализ", описываются методы исследования пакетов на уровне полей. Как мы уже убедились на примере стека протоколов TCP/IP, невозможно сказать, что неправильно, не зная, как должно быть правильно. То же самое касается и полей пакетов. Затем в главе 11, "Необычный трафик", мы перейдем к следующему уровню анализа пакетов, рассматривая их как единое целое. Другими словами, мы будем определять предназначение конкретных пакетов. Будет рассмотрено несколько реальных примеров трафика, перехваченного с помощью TCPdump, а также наборы пакетов, которые применялись при некоторых атаках. В завершающей эту часть главе 12, "Создание фильтров TCPdump", будет подробно рассказано о некоторых реальных случаях анализа трафика. С помощью методов пассивного исследования трафика мы постараемся разобраться, использовалась подмена IP-адресов или действия на самом деле осуществлялись из соответствующих источников.

Операции в фоновом режиме

Как уже не раз указывалось в этой книге, системы обнаружения вторжений позволяют выявить определенные отклонения в передаваемом трафике согласно предопределенным для них сигнатурам. Хотя и существуют NIDS, которые могут быть настроены на динамический перехват всех пакетов, следующих за подозрительным, но все остальные пакеты, которые не касаются текущей проблемы, при этом не сохраняются. Вот почему мы пропагандируем использование TCPdump или другого подобного средства для фонового контроля за проходящим трафиком. Ведь могут возникнуть опасные ситуации, которые не описаны в сигнатурах систем обнаружения вторжений.

Много лет назад, когда я работал в группе CERT военного ведомства, мы получили по электронной почте письмо от администратора другой сети, в котором он утверждал, что один из компьютеров нашей сети был использован для взлома компьютера его сети. Он предоставил сведения о дате, времени, компьютере — отправителе вредоносных пакетов и его IP-адрес. Этот администратор также сообщил крайне полезную информацию о том, что, по его мнению, в файл паролей нашего хоста была добавлена несанкционированная учетная запись для пользователя по имени Darren. Мы немедленно расследовали жалобу и обнаружили, что указанный в сообщении IP-адрес был статическим IP-адресом из диапазона IP-адресов, выделенных для нашей сети. Напомню, что это было много лет назад, когда протокол DHCP еще не приобрел широкой популярности. Владельцем компьютера с этим IP-адресом был менеджер с безукоризненной репутацией, который казался противоположностью принятому стереотипу хакера. После небольшого расследования мы узнали, что у этого человека есть сын — подросток, которого зовут Даррен (Darren). Без какого-либо чувства вины или стыда менеджер охотно рассказал офицеру контрразведки, что он дал своему сыну номер учетной записи, имя пользователя и пароль. Конечно, он горячо возражал против утверждения, что его сын является хакером.

Нам были нужны доказательства. Мы решили просмотреть записи TCPdump за несколько дней до и после случившегося инцидента. Мы обнаружили, что с интересующего IP-адреса было установлено соединение с атакованным компьютером чужой сети, а также пользователь заходил без преувеличения на сотни порнографических сайтов. Тем не менее все эти действия пользователя не вызвали тревоги системы обнаружения вторжений. И только информация, сохраненная с помощью TCPdump, позволила нам определить обоснованность жалобы.

Рассмотренную ситуацию можно сравнить с системой безопасности в супермаркетах. Большинство из них оснащены охранной сигнализацией, которая срабатывает при взломе,

и устройствами, поднимающими тревогу при попытке вынести товар с неоткрепленной биркой. Таким образом, тревога поднимается при определенных обстоятельствах. Но, кроме этого, в различных местах супермаркетов установлены видеокамеры. Эти камеры фиксируют все события. Я помню репортаж о поимке известного похитителя детей. Он использовал свою кредитную карточку при покупке товаров в супермаркете. Камеры сохранили его изображение вместе с похищенным ребенком. Поэтому, хотя его действия и не вызвали немедленной тревоги, но пленка постоянно работающей видеокамеры позволила раскрыть преступление.

Зачем нужен анализ содержимого пакетов

Существует огромное количество средств и коммерческих, и бесплатных, которые самостоятельно анализируют пакеты и выдают информацию о характере их содержимого. Так зачем снова изобретать колесо и выполнять этот анализ своими силами? Если программы наподобие Ethereal позволяют проанализировать пакет на любом уровне, начиная от заголовка кадра и заканчивая полезной нагрузкой, соответственно использованному протоколу, то зачем знать, как интерпретировать записи об этих пакетах в двоичном или шестнадцатеричном формате? Что ж, действительно созданы прекрасные средства анализа пакетов, которые точно определяют предназначение обычных пакетов со стандартными значениями полей. Если нарушитель формирует собственный пакет с нестандартными значениями, то эти средства могут не предоставить полной картины происходящего.

В качестве примера можно привести программу `sidestep`, которая была создана Робертом Грехэмом (Robert Graham) – руководителем технического отдела компании NIDS. С помощью этой программы автор хотел продемонстрировать, что, так как системы обнаружения вторжений должны уметь различать протоколы, используемые при передаче пакетов, то они не выявят скрытого сканирования. Появился даже специальный термин *network grep* (*packet grep*), описывающий системы обнаружения вторжений, которые просто ищут в трафике строку символов. Эта строка служит в качестве сигнатуры обозначения опасного пакета (с помощью UNIX-команды `grep` осуществляется поиск заданной строки символов в тексте или файле). Если система обнаружения вторжений не способна различать пакеты разных протоколов, то для ее обмана хакеру достаточно провести манипуляции с полезной нагрузкой пакетов.

Программа `sidestep` может быть запущена в скрытом режиме для различных протоколов, например DNS, RPC и др. При работе с DNS `sidestep` запрашивает DNS-сервер о запущенной версии BIND. Если не установлен запрет на отправку подобных ответов, DNS-сервер отвечает на этот запрос. Большинство систем обнаружения вторжений выявляют DNS-запрос о версии BIND, отправленный в стандартном формате. При этом осуществляется поиск строки "07version04bind". Числовые приставки, называемые *ярлыками*, всего лишь указывают на количество символов в следующем слове.

В RFC 1035 описано использование указателей в полезной нагрузке сообщений DNS. Согласно этому документу указатели применяются в ответах службы DNS, которые содержат несколько записей с повторяющейся информацией. Например отправляется запрос на определение IP-адресов нескольких хостов сети `veryveryverylongname`. Если в первой записи ответа содержится имя хоста `host1.veryveryverylongname.com`, то во второй записи вместо полного отве-

та host1.veryveryverylongname.com может быть использовано имя хоста host2 плюс указатель на расположение строки veryveryverylongname.com в первой записи. Очевидно, что это позволяет значительно сократить размер ответа. Более подробное описание использования указателей в запросах DNS содержится в разделе "DNS-запросы программы sidestep".

Программа sidestep позволяет продемонстрировать использование указателей не в ответах, а в запросах DNS с целью скрыть истинное назначение запроса. Указатели позволяют отказаться от необходимости последовательного порядка слов "version" и "bind" в DNS-запросе. Теперь порядок их расположения в запросе может быть любым. Значение будет иметь только то, как запрос "расшифровывается" на DNS-сервере. DNS-сервер отвечает на подобный запрос, т.е. сигнатура системы обнаружения вторжений "07version04bind" оказывается абсолютно бесполезной.

До того как был раскрыт механизм работы программы sidestep, я пытался сделать это самостоятельно. Я запускал программу в скрытом режиме и использовал Ethereal для отслеживания проходящего трафика. Ethereal прекрасно справляется с идентификацией пакетов в обычном формате со стандартными значениями, но при определении смысла операций в скрытом режиме это средство оказалось недееспособным. Поэтому я начал разбирать передаваемые пакеты по битам и вчитываться в документы RFC. В большинстве случаев нашим читателям не потребуется проводить такое исследование, современные анализаторы пакетов позволят точно определить предназначение пакетов. Тем не менее в редких случаях, когда программные средства ничем не смогут помочь, вы будете полностью зависеть от собственного умения разбираться в содержимом пакетов. Если вы все еще сомневаетесь в необходимости изучения материала этой главы, то подумайте, насколько понятней станет вам работа различных протоколов.

DNS-запросы программы sidestep

Чтобы глубже осознать необходимость для систем обнаружения вторжений различать пакеты разных протоколов, рассмотрим запросы DNS, сформированные программой sidestep. Прежде всего большинство систем обнаружения вторжений должны определять стандартные запросы. Ниже для сравнения будет рассмотрен скрытый запрос. Такой запрос позволяет продемонстрировать, что системы обнаружения вторжений, для которых в качестве искомых сигнатур заданы строки символов, чаще всего не могут выявлять специально сформированные запросы.

Стандартный запрос

Давайте рассмотрим отчет TCPdump о действиях программы sidestep при использовании стандартного запроса (здесь приведен стандартный отчет TCPdump, а также отчет в шестнадцатеричном формате).

```
12:39:30.027400 10.100.100.201.1128 > DNS.SERVER.domain:  10+ TXT CHAOS)?
↳ version.bind. (30)

4500 003a 052c 0000 8011 c056 0a64 64c9    E..:.,.....V.dd.
0a64 6402 0468 0035 0026 6325 <000a 0100   .....h.5.&c%....
```

```
0001 0000 0000 0000 0776 6572 7369 6f6e     .........version
0462 696e 6400 0010 0003>                   .bind.....
```

В представленном первым стандартном отчете TCPdump хост `10.100.100.201` отправляет запрос на UDP-порт 53 (`domain`) сервера `DNS.SERVER`. Идентификационный номер запроса — `10`, кроме того, задана рекурсия. При этом запрашиваются данные записей TXT и CHAOS, а также версия BIND (`version.bind`).

Теперь обратимся к отчету об этом же запросе в шестнадцатеричном формате. DNS-часть пакета выделена символами < и >. Для всех DNS-запросов предопределен одинаковый формат: последовательность частей запроса, после которой установлено значение `00`. В IP-адресах и именах хостов имена узлов разделяются точками. Например, если нужно найти IP-адрес для хоста `www.yahoo.com`, то генерируемый DNS-запрос будет рассматривать имя хоста как последовательность частей — `www`, `yahoo` и `com`. Указанию каждой части предшествует значение байта (byte count) — число байтов, которые занимает следующая часть адреса.

Запрос `version.bind`, создаваемый в обычном режиме работы программы `sidestep`, в шестнадцатеричном формате выглядит следующим образом.

```
0776 6572 7369 6f6e 0462 696e 6400
```

Жирным шрифтом выделены значения байтов, соответствующие значениям ярлыков. Первый ярлык **07** означает, что первая часть запроса занимает 7 байт. В данном случае следующие после **07** шестнадцатеричные символы являются ASCII-значениями букв слова "version". Затем следует ярлык **04**, указывающий, что следующая часть запроса занимает 4 байт — ASCII-символы слова "bind". Завершает запрос ярлык **00**.

Для каждого запроса службы DNS должен быть указан тип и класс запроса. Для этого используются 2-байтовые поля. Значения различных типов и классов запросов DNS можно найти в RFC 1035. Для запроса о версии BIND типом запроса является TXT (десятичный код 16, представленный шестнадцатеричным значением `0010`), а классом — CHAOS (десятичный код 3, представленный шестнадцатеричным значением `0003`). DNS-сервер, на котором не установлено ограничений на выдачу сведений о версии BIND, ответит на данный запрос.

Скрытый запрос

Теперь рассмотрим отчет TCPdump о действиях программы `sidestep` при отправке скрытого запроса.

```
12:39:56.674320 10.100.100.201.1129 > DNS.SERVER.domain:  42 (32)

4500 003c 0577 0000 8011 c009 0a64 64c9   E..<.w.......dd.
0a64 6402 0469 0035 0028 e445 <002a 0000  .....i.5.(.E.*..
0001 0000 0000 0000 0756 6572 7369 6f6e   .........version
c01a 0010 0003 0442 494e 4400>            .......BIND
     |                          |
     |                          |
     |                          |
     |                          |
     V                          V
Pointer 26          26 Bytes
bytes into DNS
Payload
```

Рассмотрим отчет в шестнадцатеричном формате. Жирным шрифтом выделен текст запроса в скрытом режиме. Как и раньше, запрос начинается с ярлыка 07, после которого указано первое слово запроса, но вместо слова "version" из предыдущего примера теперь используется слово "Version". Так просто можно обмануть любое программное обеспечение, которое при поиске соответствий не выполняет преобразований прописных (строчных) букв.

Это еще не вся хитрость. Посмотрим на значение следующего байта: c0. Максимальное значение ярлыка равно 63, а шестнадцатеричному числу c0 соответствует двоичное число 192. Если значение старших полубайтов ярлыка равно 1 (шестнадцатеричное c), то оно интерпретируется как указатель. Указатель определяет смещение байтов в сообщении DNS, т.е. место, где находится истинное значение ярлыка (или другого указателя). В данном случае значением указателя является шестнадцатеричное число 1a, которое соответствует десятичному значению 26. Таким образом, для того, чтобы найти значение следующей части запроса, нам следует отсчитать 26 байт от начала сообщения DNS (в отчете TCPdump сообщение DNS ограничено символами < и >).

Отсчитав 26 байт от начала сообщения DNS, мы попадаем на строку 0442 494e 4400. Ярлык 04, как и ожидалось, указывает на последующие 4 байт для передачи строки "BIND". Запрос завершается ярлыком 00. Итак, указатель дает ссылку на продолжение строки запроса. Мы снова получили строку "0010 0003", которая обозначает тип запроса TXT и класс CHAOS. DNS-сервер отправляет ответ о запущенной версии BIND.

Программу sidestep можно получить по адресу www.robertgraham.com/tmp/sidestep.html.

Введение в исследование пакетов с помощью TCPdump

При запуске программы TCPdump в обычном режиме сохраняются наиболее важные поля пакетов. Большинство полей содержатся в регистрируемых по умолчанию первых 68 байт любого пакета (14 байт заголовка кадра Ethernet и оставшаяся часть IP-дейтаграммы). Содержимое всех полей по умолчанию не отображается, для этого нужно запустить tcpdump с параметром командной строки -x. До начала любого вида анализа пакетов следует изучить стандартные форматы заголовков пакетов для таких протоколов стека TCP/IP, как IP, TCP, UDP и ICMP.

Рассмотрим следующий отчет как пример выдачи результатов работы TCPdump в шестнадцатеричном формате (используется параметр командной строки -x).

```
11:55:52.069484 192.168.143.5 > 192.168.143.101: icmp: echo request
4500 0054 064b 0000 4001 bc12 c0a8 8f05
c0a8 8f65 0800 620a 850a 0000 889f 4b39
510f 0100 0809 0a0b 0c0d 0e0f 1011 1213
1415 1617 1819
```

На первый взгляд, выдается бессмысленный набор цифр и символов. Давайте приступим к методичному изучению выданного результата. Во-первых, как можно

было догадаться из названия режима, каждый представленный символ является шестнадцатеричным. Значение каждого шестнадцатеричного символа может быть от 0 до 0xf, что соответствует двоичным значениям от 0 до 15 (напомним, что приставка 0x обозначает шестнадцатеричную систему счисления). Для хранения каждого шестнадцатеричного символа используется 4 бит (полубайт). Следовательно, два шестнадцатеричных символа составляют байт данных. В каждой строке отчета TCPdump содержится по 16 байт данных или по 32 шестнадцатеричных символа.

Выдача отчета в шестнадцатеричном формате значительно информативнее стандартного отчета TCPdump. В данном случае мы видим заголовок IP-дейтаграммы, а далее следует вложенный пакет какого-то протокола. На рис. 7.1 представлен стандартный формат IP-заголовка, который мы уже рассматривали в этой книге. Перед тем как двинуться дальше, давайте проверим, можем ли мы найти то или иное поле. Например, первое поле IP-заголовка длиной в 4 бит указывает на использованную версию протокола IP. Если мы посмотрим на предыдущий отчет в шестнадцатеричном формате, то увидим, что значение первого шестнадцатеричного символа равно 4. Таким образом, используется версия 4 протокола IP.

Это было довольно просто. Попробуем что-то посложнее. Еще одним важным полем IP-заголовка является поле протокола, указывающее на протокол, пакет которого следует после IP-заголовка. Это 8-битовое поле можно найти в третьей строке нашей схемы IP-заголовка (см. рис. 7.1). Эта схема отличается от отчета TCPdump в шестнадцатеричном формате, так как каждая строка схемы соответствует 32 бит данных (4 байт). Тем не менее мы можем найти значение поля "Protocol", отсчитав нужное смещение от начала заголовка (не забывайте, что счет начинается с 0). Итак, каждая строка содержит по 4 байт, значит, поле "Protocol" занимает 9-й байт IP-заголовка. Шестнадцатеричное значение этого поля в отчете TCPdump равно `01`. Это означает, что в данную дейтаграмму вложено сообщение протокола ICMP. Другими стандартными значениями этого поля являются 06 для протокола TCP и шестнадцатеричное значение 11 (десятичное 17) для протокола UDP.

Где заканчивается IP-заголовок

Только что мы повторили, как определять протокол, использованный для формирования вложенного в IP-дейтаграмму пакета. Это очень важный момент при анализе пакетов. Следующей важной проблемой является определение окончания IP-заголовка и начала других частей дейтаграммы. Длина стандартного IP-заголовка без параметров (например, маршрутизации от источника) составляет 20 байт. Длина IP-заголовка указывается в младшем полубайте нулевого байта IP-заголовка. В отчете TCPdump это шестнадцатеричный символ, следующий после номера версии IP. В нашем случае он равен 5. Как это согласуется с нормальной длиной 20-байтового заголовка? На самом деле длина IP-заголовка указывается с помощью 32-битовых слов, т.е. значение поля следует умножить на 4.

Хотя было бы здорово и намного проще, если бы все многочисленные длины полей указывались бы в байтах, но это не соответствует действительности. Можно задуматься (или даже посетовать), почему мудрые создатели стека TCP/IP не

соблаговолили стандартизировать всех длины и указывать их в байтах? Наиболее вероятным объяснением является то, что при создании TCP/IP много лет тому, программные и аппаратные средства работали на значительно меньших скоростях и требовалось значительно больше времени на отправку даже нескольких битов. Идея заключалась в том, что если сжимать количество битов, то они будут обрабатываться или отправляться значительно быстрее. Вот почему данное указание длины в битах имеет определенный смысл, хотя и может показаться нелепой случайностью.

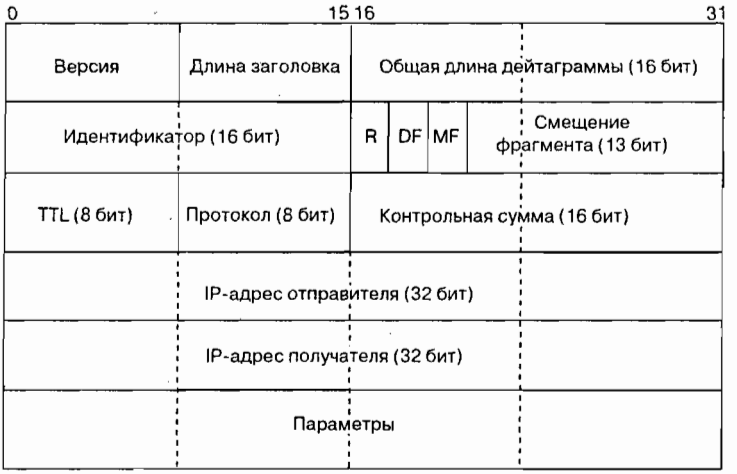


Рис. 7.1. Формат IP-заголовка

Итак, теперь мы знаем длину IP-заголовка — 20 байт, поэтому можно отсчитать 20 байт в отчете TCPdump (шестнадцатеричный формат). При этом можно не задумываться о смещении и не нужно начинать счет с нуля. Мы просто отсчитываем 20 байт от начала отчета. В первой строке содержится 16 байт, следовательно, нужно добавить к ним еще 4 байт из второй строки, и мы найдем нужное место, где заканчивается IP-заголовок и начинается заголовок ICMP-пакета. В нашем случае ICMP-заголовок начинается двумя байтами `0800`.

Другие поля с указанием длины

Рассмотрим другие поля IP-дейтаграммы, в которых указывается длина данных. Очевидно, нужно знать, как интерпретировать их значения для правильного анализа пакета.

Длина IP-дейтаграммы

Еще одним важным полем IP-заголовка является поле, в котором содержится общая длина IP-дейтаграммы. Значение длины указано в байтах, поэтому никаких преобразований не требуется. Это поле занимает второй и третий байты IP-заголовка. Единственная сложность заключается в преобразовании шестнадцатеричного значения в десятичное.

Преобразование шестнадцатеричных значений в десятичные

Преобразование значения, указанного в шестнадцатеричном формате, в десятичный формат выполняется по следующей схеме.

1. Определите количество шестнадцатеричных символов поля.
2. Начинайте с крайнего справа символа.
3. Установите соответствие для каждого символа числу 16, степень которого увеличивается на единицу для каждого следующего символа, начиная со степени 0.
4. Перемножьте числа и установленные им в соответствие числа 16, возведенные в разные степени. Сложите все произведения.

Рассмотрим наш конкретный случай и преобразуем шестнадцатеричное значение `0054` общей длины IP-дейтаграммы в десятичный формат.

1. *Длина поля для хранения общей длины IP-дейтаграммы составляет* 16 бит.
2. Всего есть 4 шестнадцатеричных символа.
3. Начинаем с крайнего справа символа (`4`).
4. Устанавливаем соответствия числу 16 с возрастающими степенями, начиная со степени 0 (от 16^0 до 16^3).
5. Возводим числа в соответствующие степени, выполняем умножение и складываем отдельные произведения.

16^3 16^2 16^1 16^0

0 0 5 4

5*161 + 4*160 = 84

В этом примере мы рассматривали поле, хранящее значение длины. У нас было *4 шестнадцатеричных символа, так как длина поля равна* 16 бит. *Установка соответствий* для числа 16 с различными степенями потребовалась только для двух крайних справа символов, так как только они отличались от 0. Сумма двух произведений дала нам конечный результат 84.

Длина заголовка TCP-сегмента

Как и IP-заголовок, заголовок TCP-сегмента тоже состоит из нескольких полей. Шестнадцатеричное значение длины TCP-заголовка указано как значение полубайта в виде 32-битового слова. Это значение, как и значение длины IP-заголовка, необходимо умножить на 4, чтобы получить значение в байтах. TCP-заголовок без дополнительных параметров имеет длину 20 байт. Значение длины TCP-заголовка содержится в старшем полубайте 12-го байта TCP-заголовка. Это важное значение, так как оно позволяет определить, где заканчивается TCP-заголовок и начинаются сами TCP-данные (полезная нагрузка).

Ниже представлен пример отчета о прохождении стандартного TCP-пакета без дополнительных параметров в обычном и шестнадцатеричном формате.

```
15:43:40.705372 1.2.3.4.63220 > 4.3.2.1.139: S 776342897:776342897(0)
↳ win 3072

4500 0028 e34f 0000 3a06 e534 0102 0304
0403 0201 <f6f4 008b 2e46 0d71 0000 0000
5002 0c00 b85f 0000>
```

TCP-сегмент выделен символами < и >. Жирным шрифтом выделено значение длины заголовка TCP-сегмента, которое, как обычно, равно 5. После умножения этого значения на 4 мы получаем стандартную длину TCP-заголовка — 20 байт.

Теперь рассмотрим обычный и шестнадцатеричный отчет для TCP-заголовка с установленными параметрами.

```
15:48:24.620314 1.2.3.4.3088 > 4.3.2.1.139: S 1212214992:1212214992(0)
↳ win 32120 <mss 1460,sackOK,timestamp 7748460 0,nop,wscale 0> (DF)

4500 003c 11a8 4000 4006 70c8 0102 0304
0102 0304 <0c10 008b 4840 eed0 0000 0000
a002 7d78 92b4 0000 0204 05b4 0402 080a
0076 3b6c 0000 0000 0103 0300>
```

В этом примере шестнадцатеричным значением длины TCP-заголовка является 0xa, что соответствует десятичному значению 10. Умножив его на 4, мы получаем длину TCP-заголовка в 40 байт. Если посмотреть на строку стандартного отчета TCPdump, то мы увидим, что данный TCP-заголовок включает такие параметры, как максимальный размер сегмента (1460), выборочное подтверждение (sackOK), временная метка, заполнитель (nop) для дополнения до 4 байт и размер окна (wscale).

Увеличение фиксируемой длины

Зададим вопрос: почему мы видим только 54 байт в следующем отчете в шестнадцатеричном формате, хотя по умолчанию значение фиксируемой длины пакета составляет 68 байт?

```
4500 0054 064b 0000 4001 bc12 c0a8 8f05
c0a8 8f65 0800 620a 850a 0000 889f 4b39
510f 0100 0809 0a0b 0c0d 0e0f 1011 1213
1415 1617 1819
```

На самом деле TCPdump дополнительно фиксирует 14 байт заголовка кадра Ethernet, хотя и не выводит их без явного указания. Для отображения сохраненной информации заголовка кадра предназначена команда `tcpdump -e`.

```
20:55:48.520619 0:10:b5:39:c6:93 0:10:b5:39:c6:9a ip 102
↳192.168.143.5 > 192.168.143.101: icmp: echo request
```

Бывают ситуации, в которых необходимо проанализировать заголовок кадра Ethernet. Это делается при необходимости узнать MAC-адрес отправителя, чтобы определить, откуда пришел пакет — от хоста или от маршрутизатора.

В предыдущем отчете о транзакции, в которой использовался механизм инкапсуляции Ethernet, определенный в RFC 894, жирным шрифтом выделен текст, отображенный в результате применения параметра командной строки `-e`. Как видите, это MAC-адреса отправителя и получателя. (MAC-адрес отправителя — `0:10:b5:39:c6:93`, а MAC-адрес получателя — `0:10:b5:39:c6:9a`). Может показаться, что эти MAC-адреса подложные (уж слишком они похожи), но на самом деле они настоящие. Просто два компьютера с процессорами Compaq были куплены одновременно. После MAC-адресов указан тип пакета, передаваемого в данном кадре. Типами передаваемых пакетов могут быть IP, APP и RARP. Значения этих полей хранятся в заголовке кадра. Последнее поле содержит значение длины в байтах всего кадра, включая заголовок кадра и инкапсулированные данные. В данном случае заголовок кадра составляет 14 байт, а размер вложенной IP-дейтаграммы равен 88 байт, что в сумме составляет 102 байт. Если в поле "Type" ("Тип") содержится значение 0x0800, это указывает на то, что после заголовка кадра инкапсулирована IP-дейтаграмма. Минимальная длина IP-дейтаграммы равняется

46 байт, а значение длины заголовка кадра согласно RFC 894 вообще *не указывается* в заголовке кадра Ethernet. По умолчанию TCPdump фиксирует 68 байт передаваемого кадра, чего обычно достаточно для сохранения IP-заголовка, заголовка вложенного пакета и небольшого количества данных. Но, если установлено большое количество параметров, будь то параметры протокола IP или параметры TCP, может оказаться, что заголовки будут сохранены не полностью.

Для увеличения установленной по умолчанию фиксируемой длины в TCPdump используется параметр командной строки `-s`. В качестве примера давайте определим сохранение полной дейтаграммы, передаваемой при выполнении любого действия в сети Ethernet. В этом случае нам необходимо увеличить размер фиксируемой длины до максимального размера дейтаграммы плюс заголовок кадра. Максимальная единица передачи данных в сети Ethernet равна 1500 байт. Добавив 14 байт для заголовка кадра, мы получаем значение 1514 байт. После введения команды `tcpdump -s 1514` запустим программу TCPdump и проверим, сохраняется ли теперь вся информация передаваемой дейтаграммы.

```
4500 0054 064b 0000 4001 bc12 c0a8 8f05
c0a8 8f65 0800 620a 850a 0000 889f 4b39
510f 0100 0809 0a0b 0c0d 0e0f 1011 1213
1415 1617 1819 1a1b 1c1d 1e1f 2021 2223
2425 2627 2829 2a2b 2c2d 2e2f 3031 3233
3435 3637
```

Итак, мы сохраняем передаваемые данные в шестнадцатеричном формате, и фиксируемый объем информации превышает принятый по умолчанию размер в 54 байт. Длина дейтаграммы указана во втором и третьем байтах IP-заголовка. Шестнадцатеричное значение `0054` этих байтов соответствует десятичному значению в 84 байт. Как мы видим, сохранены все 84 байт дейтаграммы.

Исследование всего пакета

Мы рассмотрели фундаментальные принципы анализа пакетов. Давайте подведем промежуточные итоги и посмотрим, достаточно ли изложенных сведений для проведения анализа пакетов. Ниже представлен краткий перечень вопросов, необходимых для выполнения этого анализа.

- Определение протокола, использованного при создании вложенного пакета (значение 9-го байта IP-заголовка).
- Поиск места окончания заголовка и проверка длины IP-заголовка.
- Установление длины вложенного пакета.

На первом этапе исследуется содержание дейтаграммы, чтобы определить протокол, который был использован при создании вложенного пакета. Эту информацию можно получить из 9-го байта IP-заголовка. Не забывайте о значениях, соответствующих пакетам наиболее популярных протоколов: 01 – для вложенного ICMP-сообщения, 06 – для TCP-сегмента, шестнадцатеричное 11 (двоичное 17) – для UDP-пакета.

Затем нужно определить длину IP-заголовка. Как правило, она равна 20 байт, но может быть и больше при наличии параметров. Значение длины IP-заголовка определяет младший полубайт нулевого байта IP-заголовка. Не забывайте, что это

значение выражается 32-битовым словом, поэтому, чтобы перевести его в байты, нужно умножить найденное значение на 4. Отсчитав полученное количество байтов от начала IP-дейтаграммы, мы определим место окончания IP-заголовка и начала вложенного пакета.

Затем нужно провести исследование вложенного пакета. Необходимо правильно прочесть информацию его заголовка и преобразовать шестнадцатеричные значения в десятичные. Для UDP-пакета длина заголовка всегда постоянна и равняется 8 байт. Длина заголовка TCP-сегмента бывает различной. Она указывается в старшем полубайте 12-го байта от начала TCP-сегмента. Как и длина IP-заголовка, она выражается 32-битовым словом и для перевода в байты данное значение нужно умножить на 4. Полученный результат позволяет узнать место окончания TCP-заголовка и начала полезной нагрузки.

Ниже дан пример отчета в шестнадцатеричном формате.

```
4500 0054 f23b 4000 ff01 d121 0102 0304
0403 0201 0000 9f00 d646 0000 b4cb 863a
56af 0e00 0809 0a0b 0c0d 0e0f 1011 1213
1415 1617 1819 1a1b 1c1d 1e1f 2021 2223
2425 2627 2829 2a2b 2c2d 2e2f 3031 3233
3435 3637 0000 4e00
```

Давайте разделим исследование этого отчета на два этапа. На первом этапе мы попытаемся определить протокол, использованный для создания вложенного пакета, и длину IP-заголовка. Выделенное жирным шрифтом значение 0x01 поля 9-го байта от начала IP-заголовка указывает на вложенное ICMP-сообщение. В данном случае после IP-заголовка (его последние 2 байт `0201`) следует эхо-ответ ICMP. Поскольку длина IP-заголовка составляет 20 байт, можно найти место начала ICMP-заголовка (два байт отчета `0000`).

На втором этапе нашего исследования рассмотрим заголовок ICMP-сообщения (рис. 7.2). Не забывайте, что в отчете каждый шестнадцатеричный символ соответствует одному полубайту (4 бит).

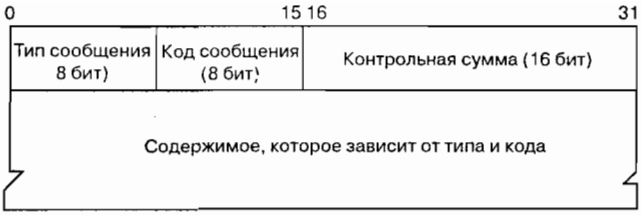
Формат заголовка ICMP-сообщения зависит от типа и кода этого сообщения. Тип и код ICMP-сообщения указываются в первых двух байтах ICMP-заголовка. Существует множество вариантов значений этих полей (полный перечень можно получить по адресу `www.iana.org/assignments/icmp-protocols`). В данном случае мы имеем одну из самых распространенных комбинаций значений: `00` и `00`, обозначающую эхо-ответ. Стандартный отчет TCPdump о прохождении исследованного пакета выглядит следующим образом.

```
1.2.3.4 > 4.3.2.1: icmp: echo reply (DF)
```

Давайте проанализируем еще один пакет.

```
4500 0030 df3c 4000 8006 633f 0102 0304
0403 0201 ob64 0015 48f3 05b1 0000 0000
7002 2000 50b6 0000 0204 05b4 0101 0402
```

В данном случае вложен пакет другого протокола, не ICMP. Наиболее важным является значение порта получателя. Оно позволяет узнать о предназначении этого конкретного пакета. Хотя формат заголовков TCP и UDP пакетов отличается, но общим для заголовков обоих протоколов является хранение в 0-м и 1-м байтах значения порта отправителя, а во 2-м и 3-м байтах — порта получателя.



Тип	Код	Сообщение
0	0	Эхо-ответ
8	0	Эхо-запрос

Рис. 7.2. Формат ICMP-заголовка

Снова размер IP-заголовка равен 20 байт. Но в этом случае выделенное значение поля IP-заголовка указывает, что вложенный пакет является TCP-сегментом. Жирным шрифтом также выделено шестнадцатеричное значение (0015) номера порта получателя, указанное во втором и третьем байтах TCP-заголовка.

Мы определили, что десятичным значением номера порта является 21 (ftp). При этом шестнадцатеричное значение было 0015. Мы сложили произведения $5*16^{0}$ и $1*16^{1}$ и получили искомое значение номера порта получателя.

Таким образом, осуществляется какое-то взаимодействие с использованием службы FTP. В данном случае это только первый этап процедуры согласования параметров соединения, поэтому никаких данных не передается. Тем не менее полезно взглянуть на размер TCP-заголовка, указанный в старшем бите 12-го байта TCP-заголовка. Умножив содержащееся там значение 7 на 4, получим размер TCP-заголовка – 28 байт. Это означает, что установлены какие-то параметры TCP, а изучение стандартного отчета TCPdump дает нам перечень этих параметров: максимальный размер сегмента (mss), два дополнения до 4-байтового значения (nop) и выборочное подтверждение (sackOK).

```
18:26:48.888088 1.2.3.4.2916 > 4.3.2.1.21: S 1223886257:1223886257(0)
↳win 8192 <mss 1460,nop,nop,sackOK> (DF)
```

Бесплатные программы для анализа пакетов

После того как мы научились самостоятельно анализировать содержимое пакетов, перейдем к изучению программных средств, которые призваны упростить эту задачу. Напомним, что умение самостоятельно анализировать пакеты пригодится в случае появления нестандартных, созданных хакером, пакетов, с анализом которых не справятся средства, ориентированные на исследование нормальных пакетов.

Ethereal

Анализатор пакетов Ethereal является бесплатным средством, доступным для использования на хостах под управлением UNIX и Windows. Ethereal удобен в использовании и имеет дружественный интерфейс, упрощающий процесс сохранения и анализа пакетов. Ethereal способен обрабатывать отчеты TCPdump в двоичном формате (создаются при использовании параметра командной строки -w),

а также использовать фильтры TCPdump для выборочного сохранения или отображения записей. Особенно полезным свойством Ethereal является возможность анализа сохраненной информации на различных уровнях абстракции.

На рис. 7.3 представлен пример результата использования Ethereal. В верхнем окне выделена одна запись. При этом в среднем окне отображается информация о заголовке кадра, IP- и TCP-заголовке, включая сведения из различных полей этих заголовков. Ethereal способен различать пакеты разных протоколов и интерпретировать передаваемые данные согласно спецификациям и документам RFC этих протоколов.

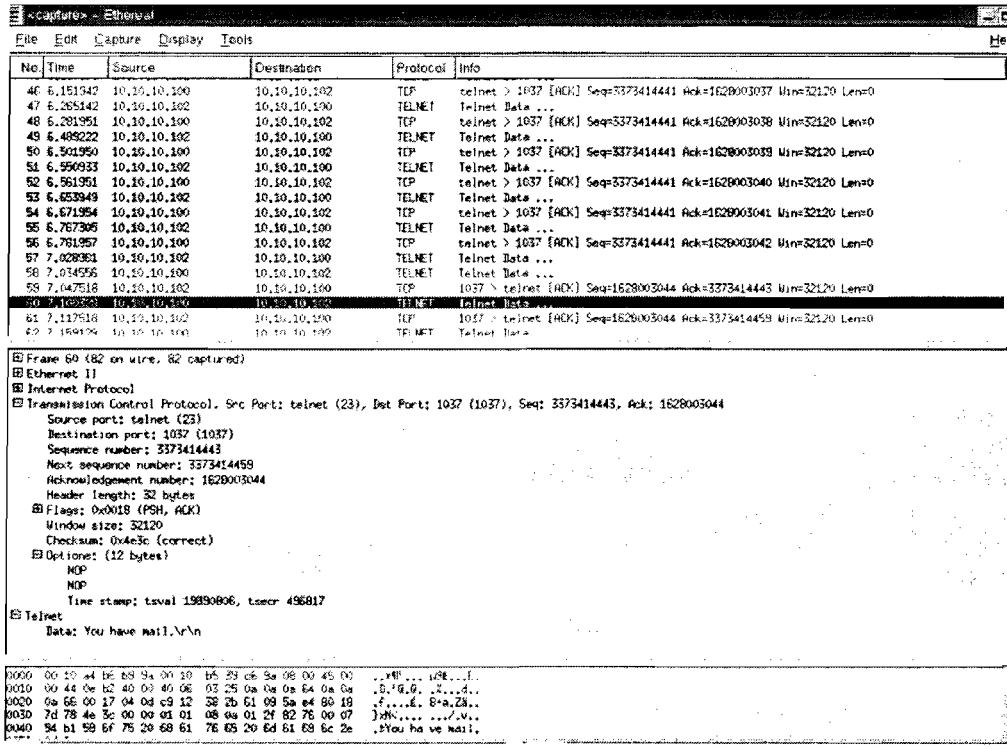


Рис. 7.3. Отчет Ethereal

Анализатор пакетов tcpshow

Анализатор пакетов tcpshow упрощает задачу интерпретации значений полей заголовков, позволяет узнать место хранения конкретного поля, вычисляет длины заголовков и преобразует шестнадцатеричные значения в десятичные эквиваленты. Он также старается исследовать передаваемые данные, и, если полезная нагрузка представляет собой символы ASCII, они выводятся на экран. Однако существующие службы наподобие NetBIOS имеют дополнительные уровни шифрования передаваемых данных, и отчеты tcpshow в этом случае будут бессмысленными. Фиксируемая длина по умолчанию равна 68 байт, что в большинстве случаев не позволяет сохранить полную дейтаграмму.

Для запуска tcpshow служит следующая команда.

```
tcpdump  -enx | tcpshow -nolink
```

Эта команда позволяет организовать чтение фиксируемых записей TCPdump и передачу их анализатору tcpshow. Набор параметров командной строки TCPdump -enx позволяет фиксировать информацию заголовка кадра (параметр -e), не преобразовывать имена хостов (параметр -n) и выводить отчет в шестнадцатеричном формате (параметр -x). Параметр командной строки -nolink команды tcpshow позволяет не выводить всю информацию заголовка кадров (например MAC-адреса). Ниже представлен пример отчета о передаче ICMP-сообщения.

```
Packet 1
IP Header
        Version:                4
        Header Length:          20 bytes
        Service Type:           0x00
        Datagram Length:        40 bytes
        Identification:         0xB5CB
        Flags:                  MF=off, DF=on
        Fragment Offset:        0
        TTL:                    254
        Encapsulated Protocol:  ICMP
        Header Checksum:        0xB229
        Source IP Address:      1.2.3.4
        Destination IP Address: 4.3.2.1

ICMP Header
        Type:                   echo-reply
        Checksum:               0xBC9C

ICMP Data
        .<Q            c.
```

Как видите, tcpshow оказывает значительную помощь при анализе пакета. Это средство позволяет "расшифровать" IP-заголовок, преобразовать сохраненные значения в байты, а шестнадцатеричные значения — в десятичные, и это далеко не все операции, выполняемые tcpshow. Также предпринимается попытка узнать содержимое заголовка вложенного пакета и вывести его данные. В данном случае ICMP-данные не являются ASCII-символами, и их интерпретация tcpshow не приносит корректного результата.

Параметр -X командной строки TCPdump

В версиях TCPdump после 3.4 доступно использование нового параметра командной строки -X. Он служит для попытки интерпретировать полезную нагрузку вложенного пакета из шестнадцатеричного формата в ASCII-символы. Это действие выполняется для всего пакета, включая и информацию полей, хранящих численные значения. Но в данном случае основная цель — интерпретировать ASCII-данные без применения дополнительных средств, например Ethereal или tcpshow. В приведенном ниже примере представлен отчет TCPdump, созданный при указании в командной строке параметра -X.

```
17:21:53.457019 1.2.3.4.ftp > 4.3.2.1.1607: P 1:81(80) ack 1 win 32120 (DF)
↳ [tos 0x10]
```

```
0x0000    4510 0078 1691 4000 4006 6b93 0102 0304  E..x..@.@.k.....
0x0010    0403 0201 0015 0647 a940 1471 309a 93ee  ...e...G.@.q0...
0x0020    5018 7d78 14fa 0000 3232 3020 7665 7262  P.}x....220.verb
0x0030    6f20 4654 5020 7365 7276 6572 2028 5665  0.FTP.server.(Ve
0x0040    7273 696f 6e20 7775 2d32 2e35 2e30 2831  rsion.wu-2.5.0(1
0x0050    2920 5475 6520 5365 7020 3231 2031 363a  ).Tue.Sep.21.16:
0x0060    3438 3a31 3220 4544 5420 3139 3939 2920  48:12.EDT.1999).
0x0070    7265 6164 792e 0d0a                      ready...
```

Обратите внимание на крайнюю справа колонку, там содержится интерпретация данных FTP-пакета. При этом две первые строки этой колонки бессмысленны, так как они соответствуют информации числовых полей заголовка.

Резюме

В большинстве случаев с задачей анализа пакетов полностью справляются программные средства наподобие Ethereal. Такие анализаторы пакетов позволяют раскрыть информацию полей и правильно интерпретировать передаваемые значения. Но бывают крайне редкие ситуации, когда эти стандартные средства или недоступны, или не в состоянии точно интерпретировать содержимое пакета. В этом случае не стоит пугаться непонятного отчета в шестнадцатеричном формате.

Только не забывайте о последовательности анализа любого пакета. Помните о формате заголовка или пакета любого протокола. Попробуйте определить протокол, использованный для создания вложенного пакета. Вычислите длину IP-заголовка, не забывая, что значение соответствующего поля нужно умножить на 4. Затем проанализируйте заголовок вложенного пакета и определите значения важных полей. Используя такой метод, вы сможете разобраться в любом отчете в шестнадцатеричном формате.

8

Исследование полей IP-заголовка

Это первая из двух глав, посвященных изучению полей IP-дейтаграммы. Все внимание в ней сконцентрировано на исследовании IP-заголовка, а в следующей главе рассмотрены заголовки вложенных пакетов (TCP, UDP и ICMP). Мы продолжаем изучение трафика с различных позиций и теперь обращаемся к изучению полей заголовков, а также нормальных и необычных значений этих полей. Теоретические сведения о полях и их стандартных значениях позволят быстро выявить нестандартный или вредоносный трафик. Эти знания точно пригодятся при регулярной проверке отчетов систем обнаружения вторжений или даже отчетов TCPdump.

Атаки со вставкой и скрытые атаки

Прежде чем рассматривать отдельные поля IP-заголовка, сделаем небольшое отступление и изучим типы атак, которые могут помешать системе обнаружения вторжений выявить вредоносный трафик. При изучении полей IP-заголовка мы будем ссылаться на атаки, соответствующие манипуляциям со значениями этих полей.

Томас Птасек (Thomas Ptacek) и Тимоти Ньюшем (Timothy Newsham) в 1998 году написали статью под названием "Вставка, сокрытие и отказ в обслуживании: обман сетевых систем обнаружения вторжений" ("*Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection*"), которая стала стандартной ссылкой при изучении данной проблемы. В этом труде рассмотрены атаки, которые не обнаруживаются сетевыми системами обнаружения вторжений по причине различной интерпретации получаемых пакетов этими системами и конечными хостами-получателями. Описаны условия, при которых системы обнаружения вторжений пропускают потенциально опасный трафик. Теорию подтверждают несколько примеров. Указанный материал можно прочитать по адресу `www.robertgraham.com/mirror/Ptacek-Newsham-Evasion-98.html`.

Кроме атак отказа в обслуживании в этой статье описаны основы других атак, основанных на обмане сетевых систем обнаружения вторжений. Первый тип этих атак

получил название *атак со вставкой* (insertion attack). Хакер отправляет пакеты атакуемому хосту. Один или несколько пакетов, которые пропускает NIDS, так и не достигнут хоста-получателя или будут отброшены как ошибочные. Авторы сделали важный вывод, что системы обнаружения вторжений и хосты-получатели по-разному воспринимают один и тот же трафик или даже "видят" различный трафик.

Второй тип атак называют *скрытыми атаками*. В этом случае хотя и сетевая система обнаружения вторжений, и хост-получатель получают один и тот же трафик, но NIDS отклоняет один или несколько пакетов, а хост-получатель принимает все эти пакеты. Хотя термин "отклоняет" не совсем точен, особенно при сравнении с отбрасыванием пакетов устройствами фильтрации, но именно он применен в цитируемом труде. Успех скрытой атаки объясняется тем, что система обнаружения вторжений ошибается при анализе пакета или его данных, которые выполняет хост-получатель, т.е. хост-получатель получает пакет или данные, которые не замечает сетевая система обнаружения вторжений.

Атака со вставкой

Представим, что наша система обнаружения вторжений установлена в отдельной сети, например в демилитаризованной зоне (DMZ), отдельной от сети, в которой находится группа защищаемых хостов. Также предположим, что эта система выявляет потенциально опасный трафик на основе заданных сигнатур. Одной из таких сигнатур для службы `telnet` (TCP-порт 23) является присутствие в передаваемом трафике строки REWT, что служит признаком попытки использовать потайную учетную запись для службы `telnet`.

Теперь предоставим себе хакера, которому удалось незаметно установить троянскую версию `telnet` на интересующем его хосте, и сейчас ему нужно организовать сеанс связи с помощью учетной записи REWT. Нарушитель смог провести небольшую разведку нашей сети, и теперь он обладает бо́льшим объемом информации о топологии и характеристиках сети, чем нам бы того хотелось. У хакера есть возможность остаться незамеченным системой обнаружения вторжений, если он сможет сформировать пакет, который будет пропущен этой системой, но будет отброшен хостом-получателем.

Таким образом, нарушитель отправляет на TCP-порт 23 атакуемого хоста три отдельных пакета, в полезной нагрузке каждого из которых передается один или несколько символов (рис. 8.1). Первый пакет содержит букву R. Этот пакет получают, исследуют и принимают и система обнаружения вторжений (NIDS), и хост-получатель. В передающемся вторым пакете, содержащем букву O, указана неправильная контрольная сумма. С помощью контрольных сумм проверяется целостность пакета, поэтому такой пакет должен быть отброшен. Давайте представим, что NIDS "видит" этот пакет, но она не запрограммирована на проверку контрольных сумм TCP-пакетов и просто пропускает этот пакет как действительную часть потока символов, передаваемых на хост-получатель. Хост-получатель принимает пакет, обнаруживает ошибку в контрольной сумме и отбрасывает его. Нарушителю удалось вставить символ, который заставляет систему обнаружения вторжений ошибиться при выявлении действительной атаки. Наконец, в третьем пакете, получаемом и принимаемом как NIDS, так и атакуемым хостом в полезной нагрузке передается набор символов EWT.

Система обнаружения вторжений собрала поток TCP-данных и расценила его как безвредный, так как для TCP-порта 23 у нее нет сигнатуры на строку ROEWT. Тем не менее хост-получатель собирает этот поток в виде REWT и организует `telnet`-сеанс с соответствующим пользователем, который остается необнаруженным NIDS. (Примечание: здесь рассмотрена очень упрощенная схема этой атаки, для ее успеха дополнительно необходима синхронизация порядковых номеров TCP-сегментов.)

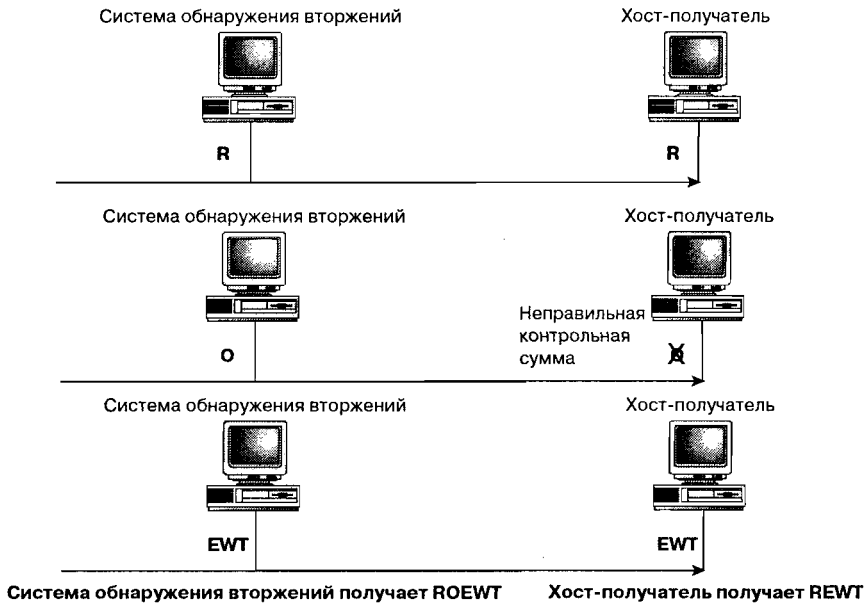


Рис. 8.1. Пример атаки со вставкой

Скрытая атака

При скрытой атаке хост-получатель принимает пакет, который не замечает сетевая система обнаружения вторжений (рис. 8.2). Здесь снова устанавливается `telnet`-сеанс с помощью передачи строки REWT на атакуемый хост-получатель. Если хакеру удастся отправить трафик таким образом, что NIDS отклоняет пакет, а хост-получатель принимает его, то это позволит злоумышленнику избежать разоблачения.

Один из возможных сценариев проведения этой атаки заключается в отправке данных в SYN-пакете. Хотя это и не соответствует стандартному соединению, отправка данных в SYN-пакете допускается документом RFC 793. После завершения полной процедуры согласования параметров соединения эти данные будут учтены как начало потока получаемой информации. Таким образом, на TCP-порт 23 атакуемого хоста первым отправляется SYN-пакет, данные которого представляют собой букву R. Система обнаружения вторжений просматривает полезную нагрузку только после завершения процедуры установки соединения, поэтому данные SYN-пакета полностью игнорируются. Хост-получатель принимает этот пакет

и сохраняет букву R как начало следующего потока данных. После этого завершается процедура установки соединения без дополнительной передачи каких-либо данных. Затем передается обычный TCP-пакет, данные которого представляют собой строку EWT.

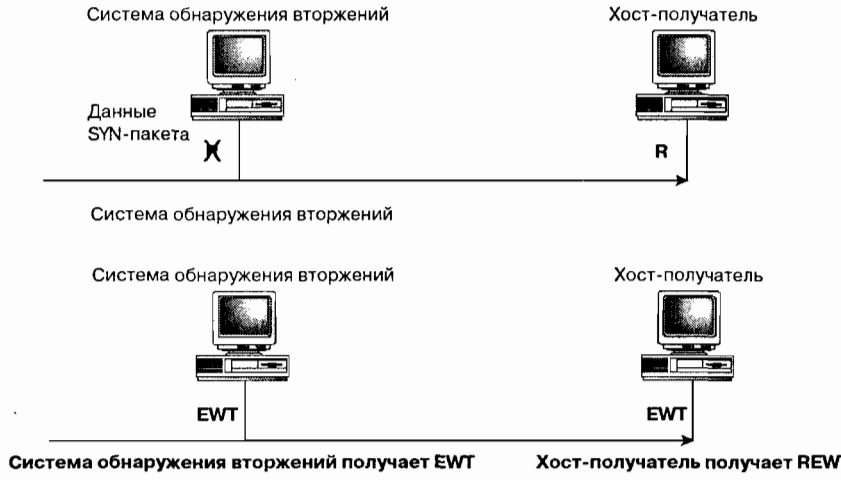


Рис. 8.2. Пример скрытой атаки

Результатом сборки TCP-потока данных на порт 23 хоста-получателя сетевой системой обнаружения вторжений является строка EWT. Эта строка не соответствует ни одной заданной сигнатуре. Хост-получатель собирает поток данных в виде строки REWT и успешно запускает сеанс троянской версии `telnet`.

Чтобы подвести итог, можно сказать, что существует много методов, которые могут быть использованы для атак со вставкой и скрытых атак против сетевых систем обнаружения вторжений. Хотя здесь не раскрыты атаки на уровне приложений, например манипуляции с протоколом HTTP, но в настоящее время прослеживается тенденция увеличения количества именно этих скрытых атак на уровне приложений. Многие из них имеют успех из-за того, что NIDS не в состоянии спрогнозировать реакции стека протоколов TCP/IP хоста-получателя на передаваемый пакет. Существует множество аспектов работы стека протоколов TCP/IP, которые отличаются в различных операционных системах.

Отслеживание всей этой информации сетевыми системами обнаружения вторжений вполне реально, однако нужно принять во внимание, что в этом случае NIDS придется выполнять намного больше операций, следовательно, процесс обработки принимаемого трафика может сильно замедлиться, вплоть до отбрасывания пакетов. При выборе между функциональными возможностями и скоростью в настоящее время побеждает скорость. Единственным способом защиты от атак со вставкой и скрытых атак является установка систем обнаружения вторжений, настроенных на определенный хост для защиты наиболее важных компьютеров. Настройка системы обнаружения вторжений на конкретный хост позволяет обезопасить его от атак со вставкой, но для защиты от этих же атак на уровне приложений придется использовать дополнительные средства.

Поля IP-заголовка

Начнем изучение полей IP-заголовка. При рассмотрении каждого поля будет описано его предназначение, представлены сведения о его нормальных и необычных значениях; о чем можно узнать по информации этого поля, и каким образом осуществляются атаки при манипуляциях со значениями этого поля.

Номер версии протокола IP

Единственными действительными версиями протокола IP являются версии 4 и 6. Наиболее распространенная версия в настоящее время — версия IPv4. IPv6 постепенно внедряется на магистральных линиях Internet.

Значение поля версии протокола IP проверяется хостом-получателем и, если номер недействителен, дейтаграмма отбрасывается без какого-либо уведомления отправителя (согласно RFC 1121). Поэтому создание дейтаграмм с недействительным номером версии IP не имеет никакого смысла, разве что позволяет проверить соблюдение получателем требований RFC.

Пакет с указанием неверной версии IP также будет отброшен на маршрутизаторе без уведомления отправителя. Таким образом, провести атаку со вставкой будет очень сложно, если только злоумышленник не находится в одной сети с системой обнаружения вторжений. В последнем случае злоумышленник может отправить интересующему хосту несколько пакетов с неверными номерами версии IP, и если система обнаружения вторжений их не отбросит, то это можно назвать атакой со вставкой — какие-то пакеты, пропускаемые NIDS, затем отклоняются конечным хостом или маршрутизатором.

Протокол

В поле "Протокол" указывается номер, обозначающий протокол, который был использован при создании вложенного пакета. Список всех возможных номеров протоколов этого поля можно получить по адресу `www.iana.org/assignments/protocol-numbers`. В последних версиях программы `nmap` подерживается возможность сканирования удаленных хостов на предмет поддерживаемых протоколов (все 256 различных вариантов). Для этого применяется параметр командной строки `-sO`. Протокол считается поддерживаемым, если не возвращается ICMP-сообщение "`protocol unreachable`". Ниже представлен пример команды сканирования на предмет поддерживаемых протоколов и результат выполнения этой команды.

```
nmap -sO target

Starting nmap V. 2.54BETA1 by fyodor@insecure.org ( www.insecure.org/nmap/ )
Interesting protocols on myhost.net (192.168.5.5):
(The 250 protocols scanned but not shown below are in state: closed)

Protocol    State       Name
1           open        icmp
2           open        icmp
6           open        tcp
17          open        udp
```

Теперь рассмотрим отчет о зарегистрированном трафике при данном сканировании.

```
07:30:31.405513 scanner.net > target.com: ip-proto-124 0 (DF)
07:30:31.405581 scanner.net > target.com: ip-proto-100 0 (DF)
07:30:31.405647 scanner.net > target.com: ip-proto-166 0 (DF)
07:30:31.405899 target.com > scanner.net: icmp: target.com protocol 124
↳ unreachable (DF)
07:30:31.788701 scanner.net > target.com: ip-proto-132 0 (DF)
07:30:32.119538 target.com > scanner.net: icmp: target.com protocol 166
↳ unreachable (DF)
07:30:34.098715 scanner.net > target.com: ip-proto-236 0 (DF)
07:30:34.098782 scanner.net > target.com: ip-proto-129 0 (DF)
07:30:34.098849 scanner.net > target.com: ip-proto-229 0 (DF)
07:30:32.779583 target.com > scanner.net: icmp: target.com protocol 236
↳ unreachable (DF)
07:30:34.099557 target.com > scanner.net: icmp: target.com protocol 109
↳ unreachable (DF)
```

При сканировании nmap проверяются 256 возможных типов протоколов. Сканируемый хост должен отвечать ICMP-сообщением "`protocol unreachable`" для любого неподдерживаемого им протокола.

Этот тип сканирования также позволяет выявить активные хосты. Это скрытое сканирование, которое не всегда выявляется системами обнаружения вторжений. Однако, если на маршрутизаторе — шлюзе локальной сети с сетью Internet — предусмотрена выдача сообщений во внешние интерфейсы "`no ip unreachables`" ("недоступных протоколов нет") или просто блокируется весь исходящий ICMP-трафик, то данное сканирование не даст никакого полезного результата.

Есть определенный недостаток в используемом программой nmap методе для определения поддерживаемого протокола. Заключение делается на основе отсутствия сообщения "`protocol unreachable`". Но в случае блокирования исходящего ICMP-трафика эти сообщения не поступят. Кроме того, возможна потеря пакетов и ошибочная реакция nmap. Поэтому создатель nmap постарался учесть подобные проблемы. Для устранения возможности случайной потери пакетов и проверки каждого протокола отправляется по несколько пакетов. Кроме того, если не вернулось ни одного ICMP-сообщения о недоступности протокола, nmap предполагает фильтрацию исходящего трафика для сканируемого хоста и выдает соответствующее уведомление.

Простая аналогия

Программа nmap использует отсутствие уведомления как подтверждение использования протокола. Как мы убедились, такой подход имеет определенные недостатки.

Эта ситуация напоминает мне пример из реальной жизни, когда в лечебном учреждении нужно сдать анализ крови. Поскольку и у врачей, и у медсестер очень много работы, то после анализа они обычно говорят, что сообщат вам, если что-то будет не в порядке. Таким образом, отсутствие уведомления является подтверждением того, что вы абсолютно здоровы.

Пусть так, но если быть хоть немного пессимистом, то можно представить неоднозначность этой ситуации. Все что угодно может произойти с вашим анализом крови, он может просто потеряться, вам могут забыть сообщить о его результатах и т.д. В общем, не стоит думать, что все прекрасно, только потому, что вам не перезвонили.

Подобные проблемы могут касаться и доставки пакета. Он может быть потерян по дороге или заблокирован на одном из многочисленных промежуточных пунктов маршрута. В программе nmap учтены некоторые из возможных проблем, но в целом отсутствие уведомления не всегда гарантирует соблюдение условия.

Тип обслуживания

С момента своего первоначального появления в составе IP-заголовка байт "Тип обслуживания" (Type of Service – ToS) претерпел несколько изменений. Одним из них стало предусмотренное в документе RFC 2481 и более новом RFC 3168 использование двух младших битов этого байта для хранения явного уведомления о перегрузке (Explicit Congestion Notification – ECN). Это связано с использованием некоторыми маршрутизаторами метода случайного раннего обнаружения (Random Early Detection – RED) или активного управления очередями с вероятностью потери пакетов.

При высокой нагрузке маршрутизатор может отбрасывать некоторые пакеты. Метод RED предназначен для уменьшения негативного эффекта потери пакетов с помощью вычисления вероятности перегрузки в очереди к интерфейсу маршрутизатора и маркирования пакетов, которые могут быть отброшены при возникновении этой перегрузки.

Существует два возможных значения битов ECN, которые сообщают о поддержке уведомления о перегрузке хостом-отправителем: `01` и `10` (рис. 8.3). Если отправитель поддерживает явное уведомление о перегрузке (ECN), маршрутизатор, использующий метод RED, старается не отбрасывать пакет, а отправить его с уведомлением о возникновении перегрузки (Congestion Experienced – CE). Для этого значения обоих младших битов байта "Тип обслуживания" устанавливаются равными единице (`11`). Хост-отправитель должен отреагировать на получение пакета с этим уведомлением снижением скорости передачи информации. Мы рассмотрим эту тему более подробно при изучении полей TCP-заголовка в следующей главе.

Рис. 8.3. Байт поля "Тип обслуживания", содержащий биты ECN

Флаг DF

Флаг DF (Don't Fragment – не фрагментировать) устанавливается в заголовке IP-дейтаграммы для запрещения ее фрагментации. Если маршрутизатор определяет необходимость фрагментации дейтаграммы, но установлен флаг DF, то такая дейтаграмма отбрасывается, и отправителю возвращается ICMP-уведомление "`unreachable - need to frag <размер MTU>`". Большинство современных маршутизаторов возвращают в этом сообщении размер максимальной единицы передачи данных (MTU) участка сети, на котором требуется фрагментация.

Фрагментация приводит к непроизводительным издержкам, поэтому ее следует избегать. При потере одного из фрагментов должна быть заново отправлена вся последовательность фрагментов. Поэтому при отправке данных некоторыми версиями стека TCP/IP сначала отправляется пробный пакет с установленным флагом DF. Если этот пакет достигнет адреса назначения без возвращения ICMP-уведомлений об ошибках, то для последующих пакетов устанавливается этот же размер дейтаграммы. Если же возвращается ICMP-сообщение "`unreachable - need to frag`", содержащее значение MTU, размер пакета уменьшается настолько, чтобы обеспечить доставку пакета без фрагментации (предполагается, что входящий ICMP-трафик не блокируется).

В стеках TCP/IP некоторых операционных систем флаг DF устанавливается по умолчанию для определенных типов пакетов, и программа `nmap` использует это свойство для определения операционной системы удаленного хоста. Кроме того, нарушитель может установить флаг DF для проведения атаки со вставкой. В этом случае система обнаружения вторжений должна быть установлена в сети с большей MTU, чем сеть, в которой установлен хост-получатель. Атакуемому хосту отправляется набор пакетов, среди которых в одном или нескольких установлен флаг DF. Система обнаружения вторжений получает этот пакет, учитывает его содержимое и отбрасывает его. Таким образом, атакуемый хост не получит этот "лишний" для атаки пакет (или пакеты).

Флаг MF

Установленный флаг MF (More Fragments — следующий фрагмент) указывает, что за этим фрагментом следует еще один или несколько фрагментов. Этот флаг устанавливается во всех фрагментах, кроме последнего. Хост-получатель распознает фрагментированный трафик по наличию этого флага или по значению поля смещения фрагмента IP-заголовка (отличного от нуля).

Сканирование сети с помощью отдельных фрагментов

Один из методов составления схемы сети основан на получении ICMP-сообщения "`reassembly time exceeded`" ("превышено время на повторную сборку пакета") от хостов сканируемой сети. Для этого сканируемым хостам может отправляться незавершенная последовательность фрагментов. Хост-получатель должен ожидать запросов на сканируемый порт, если трафик представляет собой TCP- или UDP-пакеты. Когда сканируемый хост получает первый фрагмент, он запускает таймер. Если время ожидания превышает установленный предел, а хост так и не получил все фрагменты, он возвращает отправителю ICMP-сообщение об ошибке "`IP reassembly time exceeded`".

Важно, что (согласно RFC 792) для отправки этого ICMP-сообщения обязательно должен быть получен первый фрагмент последовательности. В противном случае хост-получатель не запускает таймер. По рекомендациям RFC 1122 лимит времени таймера должен составлять от 60 с до 2 мин, но, как мы увидим, это не всегда соблюдается.

```
hping2 -S -p 139 -x win98

06:49:36.986218 verbo.2509 > win98.netbios-ssn: S 1980004944:1980004944(0)
↳win 512 (frag 38912:20@0+)
06:50:41.636506 win98 > verbo: icmp: ip reassembly time exceeded
```

```
hping2 -S -p 21 -x linux
11:56:04.064978 verbo.2450 > linux.ftp: S 1198423806:1198423806(0) win 512
↳(frag 39067:20@0+)
11:56:34.056813 linux > verbo: icmp: ip reassembly time exceeded [tos 0xc0]
```

Утилита `hping2` является бесплатным программным средством, которое используется для генерации различных типов трафика. В первом случае запуск команды `hping2` позволяет отправить SYN-пакет (параметр `-S`) на порт получателя 139 (`-p 139`) с установленным флагом MF (`-x`). Первый пакет отправляется хосту `win98`, который работает под управлением Windows 98 и ожидает запросов на TCP-порт 139.

Отправленный пакет представляет собой полный SYN-пакет: 20 байт IP-заголовка и 20 байт TCP-заголовка. Никаких данных не передается, но хост получатель ничего об этом не знает, так как установлен флаг MF (об этом говорит знак + в отчете TCPdump). Хост `win98` ожидает приблизительно около одной минуты, пока истечет лимит времени на сборку последовательности фрагментов. Затем он возвращает ICMP-сообщение "`IP reassembly time exceeded`".

Второй пакет, отправленный утилитой `hping2`, используется для проверки хоста под управлением Linux (версия ядра 2.2), на котором запущена служба FTP. Этот хост ожидает около 30 с, чтобы получить оставшиеся фрагменты на свой порт 21.

IP-адреса

В двух 32-битовых полях IP-заголовка содержатся IP-адреса отправителя и получателя. Значение IP-адреса отправителя содержится в 12–15-ом байтах IP-заголовка, а значение IP-адреса получателя – в 16–19-ом байтах.

Какие подложные значения IP-адреса отправителя могут быть указаны во входящих пакетах? Например, если во входящем в локальную сеть пакете указан IP-адрес одного из хостов этой же локальной сети, то, очевидно, проводится атака. Скорее всего, *кто-то сформировал пакет с подложным IP-адресом. Устройство фильтрации* пакетов должно блокировать такой трафик. Кроме того, во входящих пакетах в качестве IP-адреса отправителя никогда не должен содержаться ни адрес петли обратной связи 127.0.0.1, ни один из адресов диапазонов, зарезервированных IANA для частных сетей (определены в RFC 1918). Полный перечень этих адресов доступен по адресу `www.iana.org/assignments/ipv4-address-space`.

В пакетах исходящего трафика должны быть указаны IP-адреса отправителя, одного из хостов адресного пространства, выделенного для данной локальной сети. Если в пакете, исходящем из локальной сети, содержится неизвестный IP-адрес отправителя, то либо IP-адрес был подменен, либо неверно выполнена настройка хостов этой сети. В любом случае такой исходящий трафик должен быть заблокирован. Это предотвратит использование хостов вашей сети в распределенных атаках отказа в обслуживании, так как подчиненные хосты (зомби-хосты) обычно используют подмененные IP-адреса. Также будут заблокированы исходящие попытки скрытого сканирования чужих сетей с использованием подмены IP-адреса. Кроме того, из вашей сети не должны отправляться пакеты с IP-адресами отправителя 127.0.0.1 (адрес петли обратной связи локального хоста) или IP-адресами, зарезервированными для использования в локальных сетях.

И в завершение следует блокировать и входящий, и исходящий трафик с широковещательными адресами получателя. Такие адреса обычно используются для быстрого сканирования сетей или при проведении атаки Smurf.

Идентификатор

Значение идентификатора хранится в 4-м и 5-м байтах IP-заголовка. Для каждой новой отправляемой хостом дейтаграммы должен генерироваться уникальный идентификатор. Это значение обычно увеличивается на 1, хотя в некоторых случаях на 256 для каждой новой дейтаграммы.

Это уникальное значение требуется при фрагментации дейтаграммы. Всем фрагментам одной дейтаграммы присваивается одинаковый идентификатор. Поэтому часто такое поле называют *идентификатором фрагмента*. С помощью значения идентификатора хост-получатель сможет повторно собрать фрагменты одной дейтаграммы.

Возможные значения идентификатора лежат в диапазоне от 1 до 65535, так как это 16-битовое поле (значение 0, как правило, не используется). Когда достигается максимальное значение 65535, значение обнуляется и отсчет опять начинается с 1. Очевидно, что в пакетах из различных источников последовательности идентификаторов должны различаться. Поэтому, если регистрируется подозрительный трафик и, несмотря на то, что указаны различные IP-адреса отправителя, все равно сохраняется последовательное увеличение значений идентификаторов, то, скорее всего, осуществляется подмена IP-адресов отправителя.

Как и любое другое значение поля IP-заголовка, значение идентификатора может быть подменено хакером. Например, если злоумышленник использует программу, которая отправляет все пакеты с одинаковым идентификатором, то вы не сможете получить по значению этого поля какую-либо информацию о хосте нарушителя. Для отображения значений поля идентификатора и поля TTL в программе TCPdump предназначен параметр командной строки `-vv`.

Поле TTL

В поле TTL (Time to Live – время жизни) содержится 8-битовое значение, которое устанавливается отправителем. Начальное значение TTL зависит от варианта стека TCP/IP, используемого на конкретном хосте (табл. 8.1). Узнать варианты начальных значений различных операционных систем можно по адресу `project.honeynet.org/papers/finger/traces`. Как уже указывалось, каждый маршрутизатор на пути следования пакета уменьшает значение этого поля на 1. Если значение становится равным 0, то пакет отбрасывается, и отправителю возвращается ICMP-сообщение "`time exceeded in-transit`". Это позволяет удалить пакеты, которые попали в замкнутый цикл передачи. Значение поля TTL может использоваться для проведения атаки со вставкой, если система обнаружения вторжений учитывает этот пакет, хотя значение поля TTL не позволит этому пакету достигнуть хоста-получателя.

Как можно проверить, что пакет пришел именно от указанного отправителя? Можно узнать значение поля TTL принятого пакета, посмотреть на начальное значение этого поля (см. табл. 8.1) и, отняв от второго значения первое, получить ко-

личество переходов на пути пакета к сети получателя. Затем можно воспользоваться программой Traceroute и проверить количество переходов на обратном маршруте к хосту с подозрительным IP-адресом. Полученное значение может отличаться от количества переходов (hop count) при доставке подозрительного пакета по причине динамического характера маршрутизации. Но обычно эти значения не слишком различаются, если не было каких-либо серьезных проблем с маршрутизаторами или перегрузки в сети во время передачи одного из двух пакетов.

Таблица 8.1. Начальные значения поля TTL для различных операционных систем

Операционная система	Версия	Платформа	TTL
Windows	9x/NT	Intel	32
AIX	4.3.x	IBM/RS6000	60
AIX	4.2.x	IBM/RS6000	60
Cisco	11.2	7507	60
IRIX	6.x	SGI	60
Linux	2.2.x	Intel	64
OpenBSD	2.x	Intel	64
Solaris	8	Intel/Sparc	64
Windows	9x/NT	Intel	128
Windows	2000	Intel	128
Cisco	12.0	2514	255
Solaris	2.x	Intel/Sparc	255

Если пакеты поступают от различных отправителей, существует большая вероятность того, что значения поля TTL в поступивших пакетах будут различными. Если же одновременно от разных отправителей поступают пакеты с одинаковым значением поля TTL, то это похоже на подмену IP-адреса отправителя.

Будьте готовы к тому, что некоторые программы сканирования специально устанавливают случайные начальные значения поля TTL, чтобы устранить возможность отслеживания по этому значению источника дейтаграммы.

Обнаружение подмены IP-адреса по значениям идентификатора и поля TTL

Давайте рассмотрим следующий отчет.

```
07:31:57.250000 somewhere.de > 192.168.104.255: icmp: echo request
↳ (ttl 246, id 5134)
07:34:18.090000 somewhere.jp > 192.168.104.255: icmp: echo request
↳ (ttl 246, id 5137)
07:35:19.450000 somewhere.ca > 192.168.104.255: icmp: echo request
↳ (ttl 246, id 5141)
```

В этом отчете показаны пакеты, поступившие от трех различных отправителей к одному хосту-получателю. Временные метки свидетельствуют о доставке пакетов через несколько минут один после другого. Обратите внимание на значения идентификаторов дейтаграмм. Чем же необычны эти значения полей идентификатора и для чего предназначен этот трафик?

Странно то, что значения идентификаторов постепенно увеличиваются, хотя пакеты как будто поступают от различных источников на один IP-адрес получателя — `192.168.104.255`. В подсети `192.168.104` активных хостов нет, поэтому данный трафик становится еще более подозрительным. Хотя это может быть редким стечением обстоятельств, но, скорее всего, все эти эхо-запросы ICMP отправляются с одного хоста.

Напомним, что возможное значение поля идентификатора IP-заголовка находится в диапазоне от 1 до 65535. Поэтому вероятность случайного поступления трех пакетов от различных отправителей с идентификаторами в диапазоне от 5134 до 5141 за несколько минут крайне мала. Если предположить, что номера идентификаторов не были подменены, то похоже, что пакеты отправляет один, не очень производительный, хост (возможно, простой персональный компьютер).

Как и при изучении любого неизвестного трафика, ответить на вопрос *что* значительно проще, чем на вопрос *зачем*. Может, это попытка сканирования, и указан один действительный IP-адрес, а два IP-адреса подменены? Это позволяет хакеру создать "дымовую завесу": даже если сканирование будет замечено, возможно, не удастся выявить его истинный источник.

Теперь давайте проанализируем тот же трафик, но с точки зрения значений поля TTL. Странно, что все значения поля TTL в разных пакетах совпадают. Это еще больше подтверждает наше предположение об отправке всех трех пакетов с одного хоста. Какова вероятность того, что у трех различных отправителей начальным значением этого поля было 255, и каждый пакет, предназначенный одному и тому же получателю, прошел по девять переходов, и все они поступили приблизительно одновременно?

С помощью параметра командной строки `-vv` программы TCPdump в отчетах дополнительно отображается информация двух полей, которая может послужить для проверки подозрительного трафика.

После регистрации этого трафика была запущена программа Traceroute с целью выявить, были ли использованные IP-адрес настоящими, или же их подменили. Ниже показаны результаты работы этой программы.

```
traceroute somewhere.de
   arriving TTL:              246
   probable initial TTL:      255
   expected hop count back:   9
   actual hop count back:     13

traceroute somewhere.jp
   arriving TTL:              246
   probable initial TTL:      255
   expected hop count back:   9
   actual hop count back:     13

traceroute somewhere.ca
   arriving TTL:              246
   probable initial TTL:      255
   expected hop count back:   9
   actual hop count back:     12
```

В данном случае использование Traceroute не позволяет сделать окончательный вывод. Для обратной доставки пакета каждому из отправителей потребовалось или 12, или 13 переходов. Но это не позволяет однозначно определить подмену или истинность указанных IP-адресов отправителей.

Число переходов на обратном маршруте (12 или 13) довольно близко к ожидаемому числу переходов (9). Тем не менее информация полей идентификатора и TTL указывает на вероятную подмену IP-адресов. Лучшим доказательством этой подмены послужило бы значительное расхождение между ожидаемым и действительным количеством переходов, получаемым с помощью Traceroute.

Нужно дать несколько предупреждений относительно использования Traceroute как средства экспертизы. Во-первых, работа этой программы может быть остановлена из-за блокирования маршрутизатором или устройством фильтрации пакетов ICMP-трафика, в частности сообщений "`time exceeded in-transit`" и "`port unreachable`". Во-вторых, помните, что выполнение команды `traceroute` по отношению к реальному IP-адресу хоста хакера нежелательно, так как явно указывает на интерес к этому хосту.

Контрольная сумма

С помощью контрольных сумм обеспечивается проверка целостности информации, передаваемой по сети. Данные IP-заголовка разбиваются на 16-битовые поля. Для каждого 16-битового поля вычисляется поразрядное дополнение до единицы. Подсчитывается сумма этих поразрядных дополнений, которая и является контрольной суммой.

Контрольная сумма сохраняется в 10-м и 11-м байтах IP-заголовка. Контрольная сумма гарантирует целостность полей только IP-заголовка. Она отличается от контрольных сумм, подсчитанных в заголовках вложенных пакетов, и проверяется на каждом маршрутизаторе на пути следования IP-дейтаграммы, а также на хосте-получателе. Контрольные суммы в заголовках вложенных пакетов (TCP, UDP или ICMP) проверяются только на хосте-получателе.

Если вычисленная контрольная сумма не совпадает со значением, указанным в дейтаграмме, то дейтаграмма отбрасывается без уведомления отправителя. Протоколы верхнего уровня должны самостоятельно обнаружить потерю пакетов и принять меры по ликвидации ошибки.

Формула подсчета контрольной суммы IP-заголовка используется для подсчета всех других контрольных сумм заголовков вложенных пакетов. Итак, сначала IP-заголовок разделяется на 16-битовые поля. Так как длина IP-заголовка всегда кратна 4 байт, то она всегда будет делиться без остатка.

Затем для каждого поля вычисляется поразрядное дополнение до единицы. Эта операция заключается в зеркальном отображении значений битов. Затем значения отдельных дополнений суммируются. Приведем пример.

4	5	0	0	Шестнадцатеричный формат
0100	0101	0000	0000	Двоичный формат
1011	1010	1111	1111	Поразрядное дополнение до единицы

Выше приведены первые 16 бит стандартного IP-заголовка. Каждое шестнадцатеричное значение представляется четырьмя двоичными битами, и каждый из этих битов отображается зеркально, что и является поразрядным дополнением до 1. Это коммутативная операция, поэтому можно сначала сложить шестнадцатеричные значения 16-битовых полей, а затем подсчитать поразрядное дополнение до единицы этой суммы. Результат контрольной суммы от этого не изменится.

Контрольная сумма IP-заголовка проверяется на каждом промежуточном маршрутизаторе на пути следования IP-дейтаграммы. Если значение остается правильным, то маршрутизатор уменьшает значение поля TTL на единицу и передает дейтаграмму далее. Такая проверка действительно имеет смысл. В самом худшем случае в результате доставки пакета хост-получатель будет выведен из строя. Поэтому нет смысла передавать искаженный пакет, так как искажение может затронуть и содержимое пакета.

Хотя с помощью контрольных сумм IP-заголовка и заголовков вложенных пакетов проверяется целостность большинства пакетов, но все же возможны исключения. Например, если поменять местами 16-битовые поля исходной дейтаграммы, контрольная сумма искаженного пакета останется неизменной.

```
4500  003c

4500 = 0100 0101 0000 0000        1011 1010 1111 1111
003c = 0000 0000 0011 1100        1111 1111 1100 0011
                                  1011 1010 1100 0010

003c  4500

003c = 0000 0000 0011 1100        1111 1111 1100 0011
4500 = 0100 0101 0000 0000        1011 1010 1111 1111
                                  1011 1010 1100 0010
```

Рассмотрим этот пример. Мы поменяли местами два первых 16-битовых поля (4500 003c) IP-заголовка. Контрольная сумма для правильной последовательности этих 16-битовых полей равна 1011 1010 1100 0010. Но если переставить эти поля местами и снова подсчитать контрольную сумму, то результат получится тем же. Очевидно, что смысл дейтаграммы при обмене местами полей будет совершенно иным. Это недостаток алгоритма вычисления контрольной суммы.

Почему же не использовать более сложный и надежный алгоритм подсчета контрольных сумм? Ответ прост. Не забывайте, что вычисление контрольной суммы выполняется для каждого пакета на каждом маршрутизаторе. Чем проще вычисление, тем оно быстрее выполняется. Данный алгоритм позволяет очень быстро и достаточно надежно провести вычисления, так как простая перестановка 16-битовых полей встречается крайне редко. Более подробную информацию о контрольных суммах можно прочесть в RFC 1071.

Резюме

Хотя системы обнаружения вторжений позволяют значительно увеличить безопасность работы в сети, они не являются панацеей и не способны выявить любой вредоносный трафик. Так, атаки со вставкой и скрытые атаки приводят к неправильной интерпретации передаваемого потока информации. Существует множество вариантов этих атак, и сетевые системы обнаружения вторжений не в состоянии определить, как будет реагировать на пакет конкретный хост-получатель, так как не известны все нюансы используемого на нем стека протоколов TCP/IP. Кроме того, системы обнаружения вторжений не располагают сведениями о топологии защищаемой сети, поэтому возможны атаки, при которых пакеты с небольшими значениями поля TTL не достигают хоста получателя. По-

этому вместо сетевых систем обнаружения вторжений используют системы обнаружения вторжений для защиты конкретного хоста.

Квалифицированный аналитик сетевого трафика обязан знать типы полей и возможные значения этих полей IP-заголовка. Это позволит ему быстро обнаруживать необычный трафик. Выявление необычных значений полей часто не позволяет ничего сказать о предназначении вредоносного пакета, но должно служить сигналом опасности.

9

Анализ заголовков вложенных пакетов

В этой главе будут рассмотрены поля заголовков TCP-, UDP- и ICMP-пакетов, вложенных в IP-дейтаграмму. Как уже было сказано, для человека, выполняющего анализ трафика, просто необходимо знать предназначение этих полей и хранящиеся в них стандартные значения. Это единственный способ выявить ту или иную форму вредоносных действий.

Так как излагаемая тема довольно обширна, то поля заголовков каждого протокола будут рассматриваться по отдельности. Надеемся, что представленный материал поможет лучше узнать самые популярные сетевые протоколы.

TCP

В главе 2, "TCPdump и TCP", уже было сказано, что TCP считается надежным протоколом. Он гарантирует доставку данных и способен контролировать отсутствие проблем с помощью информации полей порядкового номера и номера подтверждения. В заголовке TCP-пакета полей значительно больше, чем в UDP и ICMP-заголовках, так как протокол TCP требует хранения информации о состоянии сообщения и обеспечивает оптимальную производительность передачи данных между отправителем и получателем. Мы рассмотрим поля заголовка TCP-пакета, а также их нормальные и некорректные значения.

Номера портов

Номера портов отправителя и получателя хранятся в 16-битовых полях TCP-заголовка в 0-м и 1-м байтах (порт отправителя) и во 2-м и 3-м байтах (порт получателя) TCP-заголовка. Возможные значения находятся в диапазоне от 1 до 65535. Использование номера порта 0 не допускается и считается "сигнатурой" неверного значения поля.

Когда отправитель хочет установить соединение с удаленным хостом, то для этой цели, как правило, выбирается один из временных портов, номер которого больше 1023. Для каждого нового соединения (за исключением повторной отправки данных) должен быть выбран другой временный порт. Правила повторной отправки данных при TCP-соединениях будут рассмотрены в разделе "Повторная передача данных". Зачастую при сканировании значение порта отправителя увеличивается на единицу для каждого нового соединения.

Одним из явных признаков проводящегося сканирования nmap с помощью SYN-пакетов является постоянный номер порта отправителя для большого количества новых TCP-соединений.

```
Nmap -sS sparky

09:40:43.964215 verbo.47247 > sparky.1548: S 2401927088:2401927088(0) win 2048
09:40:43.964412 verbo.47247 > sparky.24: S 2401927088:2401927088(0) win 2048
09:40:43.964465 verbo.47247 > sparky.1547: S 2401927088:2401927088(0) win 2048
09:40:43.964553 verbo.47247 > sparky.2564: S 2401927088:2401927088(0) win 2048
09:40:43.964604 verbo.47247 > sparky.1484: S 2401927088:2401927088(0) win 2048
09:40:43.964695 verbo.47247 > sparky.628: S 2401927088:2401927088(0) win 2048
09:40:43.964748 verbo.47247 > sparky.1112: S 2401927088:2401927088(0) win 2048
```

Хотя, казалось бы, номер порта отправителя хоста verbo должен был бы изменяться для каждой новой попытки установить соединение с различными портами хоста sparky, он остается постоянным (47247).

Для сравнения рассмотрим метод, применяемый по умолчанию в другом средстве сканирования hping2. При использовании параметра командной строки -S утилитой hping2 выполняется сканирование с помощью SYN-пакетов. Номер порта отправителя увеличивается, но при этом осуществляется попытка установить соединение с портом 0 хоста-получателя. Очевидно, что в этом случае невозможно узнать открытые порты получателя. Основной целью является получить в ответ пакет с установленным флагом RESET, свидетельствующий только об активности хоста, так как не существует хостов, которые ожидают запросов на порт 0.

```
Hping2 -S spanky

09:44:13.882207 verbo.1788 > sparky.0: S 1553132317:1553132317(0) win 512
09:44:14.876837 verbo.1789 > sparky.0: S 1894028093:1894028093(0) win 512
09:44:15.876836 verbo.1790 > sparky.0: S 2032501562:2032501562(0) win 512
09:44:16.876832 verbo.1791 > sparky.0: S 851202745:851202745(0) win 512
```

Контрольная сумма TCP-заголовка

Контрольные суммы, указанные в заголовках вложенных пакетов, позволяют гарантировать целостность вложенного заголовка и данных TCP-, UDP- или ICMP-пакета. В отличие от контрольной суммы IP-заголовка контрольные суммы вложенных пакетов проверяются только на хосте-получателе. В UDP-заголовках проверка контрольной суммы не является обязательной, но настоятельно рекомендуется.

Контрольные суммы для TCP- и UDP-заголовков вычисляются с помощью псевдозаголовка[1], дополнительного по отношению к заголовку и данным вложенного пакета. Длина псевдозаголовка равна 12 байт. Он состоит из полей IP-адресов отправителя и получателя, поля протокола (по информации IP-заголовка) и копии длины вложенного пакета (длина заголовка плюс количество байтов данных). В 8-м байте от начала заголовка содержится дополнение 8-битового поля протокола до 16 бит, так как для вычисления контрольной суммы требуется разбить заголовок на 16-битовые блоки данных.

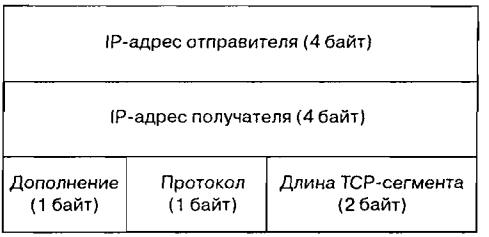


Рис. 9.1. Псевдозаголовок для вычисления контрольной суммы TCP

Зачем нужен псевдозаголовок? Он позволяет хосту-получателю повторно проверить, что на уровне протокола IP не была случайно принята дейтаграмма, предназначенная для другого хоста, или неверно указан вложенный протокол. При изменении IP-дейтаграммы на пути ее доставки контрольная сумма IP-заголовка может не выявить этих искажений. Копирование значений нескольких полей IP-заголовка в псевдозаголовок позволяет снизить вероятность случайных ошибок.

Давайте рассмотрим особый случай, когда псевдозаголовок позволяет предотвратить доставку пакета по ошибочному адресу (рис. 9.2). Предположим, что есть хост, который отправляет пакет по IP-адресу получателя `1.2.3.4`. Для создания вложенного пакета используется протокол TCP, но в данном примере нет принципиальной разницы в использовании протокола TCP или UDP, так как в обоих протоколах применяется псевдозаголовок. При вычислении контрольной суммы заголовка транспортного протокола используются значения полей псевдозаголовка. Поэтому для вычисления контрольной суммы TCP-заголовка использовано значение IP-адреса получателя `1.2.3.4`.

На пути доставки пакета к получателю он проходит через маршрутизаторы, каждый из которых должен проверить контрольную сумму IP-заголовка. Предположим, что один из этих маршрутизаторов проверил контрольную сумму IP-заголовка, уменьшил на единицу значение поля TTL и должен вычислить новую контрольную сумму. По какой-то неизвестной причине на IP-уровне маршрутизатора IP-адрес получателя был изменен на `1.2.3.5`, и новая контрольная сумма IP-заголовка вычисляется на основе именно этого искаженного значения, а пакет передается далее по неправильному адресу получателя.

---

[1] *Несуществующий реально заголовок, состоящий из полей IP-заголовка и TCP- или UDP-заголовка, которые используются для подсчета контрольной суммы вложенного пакета.*

Рис. 9.2. Использование псевдозаголовка

Предположим, что адрес `1.2.3.5` принадлежит активному хосту. Искаженный пакет доставляется по неверному адресу. Проверка хостом-получателем контрольной суммы IP-заголовка дает положительный результат, так как при ее вычислении маршрутизатором уже было использовано неверное значение. Пакет передается на транспортный уровень, где для проверки контрольной суммы используются поля псевдозаголовка. Эта проверка позволит выявить искаженный пакет, так как при подсчете исходной контрольной суммы TCP-заголовка было использовано правильное значение адреса получателя `1.2.3.4`. Полученный пакет отбрасывается хостом `1.2.3.5`.

Призыв о помощи

Из различных источников информации у меня сложилось ясное представление, что предназначением псевдозаголовка является дополнительная проверка правильности доставки пакета указанного протокола на конкретный хост. Тем не менее я не знал, как именно это делается. Я спрашивал у нескольких коллег, но так не получил точного ответа, подкрепленного наглядным примером. Закончилось тем, что я написал разработчику и специалисту в области стека протоколов TCP/IP Дугу Камеру (Doug Camer), который и пояснил мне все это на примере искажения пакета на промежуточном маршрутизаторе. Выражаю ему свою искреннюю признательность.

Порядковые номера

В TCP-заголовке порядковые номера используются для указания начального байта данных каждого TCP-сегмента. Это позволяет контролировать поток передающихся TCP-данных. В большинстве случаев все необходимые для передачи данные не умещаются в одном TCP-сегменте. Или же некоторые службы, например `rlogin`, могут отправлять отдельные символы над потоком TCP-данных, требуя при этом создания нескольких потоков передачи данных в одном сеансе. Так как TCP является надежным протоколом, то необходим механизм, позволяющий сравнить количество отправленных и полученных данных. Именно для этого и применяются порядковые номера.

Порядковые номера не должны повторяться, за исключением случая повторной передачи данных в том же сеансе, если первая попытка оказалась неудачной и отправитель не получил никакого уведомления об ошибке (от самого получателя или же от устройства фильтрации пакетов). Начальный порядковый номер

TCP-сегмента (ISN) указывается в SYN-пакетах (и отправителем, и получателем), отправляемых при установке TCP-соединения.

В различных вариантах стека TCP/IP начальные порядковые номера выбираются по разным формулам. Программа nmap использует эту особенность для определения операционных систем удаленных хостов. Совместно с программой nmap поставляется файл с сигнатурами операционных систем, которые позволяют выявить тип и версию операционной системы удаленного хоста по ответам, приходящим от этого хоста. Для получения характерных ответов программой nmap используется набор специальных тестов, и первый тест из этого набора заключается в том, что nmap проверяет порядковые номера, установленные в ответном пакете после запроса на прослушиваемый порт. В некоторых устаревших версиях операционных систем для каждого нового соединения начальные порядковые номера TCP-сегмента изменяются в определенной последовательности. Если хакер способен прослушивать передающийся трафик, он сможет спрогнозировать и перехватить соединение на основе этой информации, как это было сделано при известной атаке Митника. В других операционных системах значения начальных порядковых номеров выбираются по формуле, зависящей от времени отправки пакета. Это, разумеется, тоже не очень безопасно. Самым надежным способом является выбор случайного значения ISN, которое невозможно спрогнозировать. Хуже всего, что начальный SYN-пакет TCP-соединения используется для синхронизации значений порядковых номеров. В следующем примере представлен результат запуска программы nmap для определения операционной системы удаленного хоста (с помощью параметра командной строки -O), при этом указаны: обнаруженные открытые порты, сложность предсказания порядкового номера и предположительные тип и версия операционной системы.

```
Nmap -O sparky

(The 1495 ports scanned but not shown below are in state: closed)
Port            State       Service
23/tcp          open        telnet
25/tcp          open        smtp
111/tcp         open        sunrpc
513/tcp         open        login
32771/tcp       open        sometimes-rpc5
32772/tcp       open        sometimes-rpc7

TCP Sequence Prediction: Class=random positive increments
                         Difficulty=46112 (Worthy challenge)
Remote OS guesses: Solaris 2.6 - 2.7, Solaris 7
```

Данное сканирование хоста sparky обнаружило, что установка значений начальных порядковых номеров базируется на основе "случайных положительных приращений" ("random positive increments"). Также сообщается, что прогнозирование следующего порядкового номера будет "достаточно сложным" ("worthy challenge"). Хост sparky работает под управлением Solaris 2.7, и, судя по всему, предугадать следующий начальный порядковый номер TCP-сегмента для этого хоста, основываясь на прошлом номере или времени, будет практически невозможно.

Номера подтверждения

Для проверки доставки данных в протоколе TCP используются подтверждения. Для этого хост-получатель устанавливает в ответном пакете флаг ACK и номер подтверждения. Номер подтверждения, отправленный хостом-получателем, указывает на следующий порядковый номер TCP-сегмента, который он ожидает получить от отправителя.

Поскольку номер подтверждения всегда больше порядкового номера, то этот номер всегда больше 0. Здесь нужна маленькая оговорка. Возможно, при TCP-соединении будут использованы все 2 миллиарда положительных порядковых номеров, доступных для 32-битового поля их хранения. Если случайно последний отправленный порядковый номер будет представлять собой максимально допустимое 32-битовое число, то хост-получатель обнулит номера подтверждения и укажет, что следующим порядковым номером должен быть 0. Такое случается очень редко.

Программа `nmap` позволяет определить активные хосты с помощью отправки TCP-сегмента с установленным флагом ACK. Этот метод значительно более эффективен, чем тестовая `ping`-проверка, так как на многих современных узлах блокируются входящие эхо-запросы ICMP. Тем не менее маршрутизатор, который не отслеживает содержимое пакетов, может позволить прохождение пакета "установленного" соединения, содержащего флаг ACK. Целью отправки незатребованного пакета с флагом ACK является получение в ответ от удаленного хоста пакета с флагом RESET, что явно свидетельствует об активности этого хоста. Но такое сканирование выявить довольно просто, так как в современных версиях `nmap` в отправляемом тестовом ACK-пакете значение номера подтверждения равно нулю, как показано в следующем примере.

```
verbo.52776 > win98.netbios-ssn: . ack 0 win 4096 <wscale 10,nop,mss 265,
↳timestamp 1061109567[|tcp]>
```

TCP-флаги

TCP-флаги указывают на предназначение конкретного пакета TCP-сеанса. SYN-флаг используется для организации сеанса, а флаг FIN – для его "нормального" завершения. Флаг RESET указывает на немедленный разрыв соединения, а флаг ACK обозначает подтверждение принятых данных. Флаг ACK устанавливается во всех пакетах после получения первого SYN-пакета сеанса. Флаг PUSH позволяет сообщить отправителю о необходимости немедленной отправки данных из буфера, а хосту-получателю – о передаче всех полученных данных на уровень протокола TCP. Сейчас возможна отправка данных из незаполненного до конца выходного буфера даже без получения пакета с установленным флагом PUSH. И, наконец, с помощью флага URGENT обозначается пакет с наивысшим приоритетом.

Существует множество допустимых комбинаций TCP-флагов. Но также возможны и многие некорректные комбинации TCP-флагов, используемые в различных целях. Раньше, как только сетевые системы обнаружения вторжений появились на рынке, многие из них проверяли трафик лишь на предмет начальных SYN-пакетов. Хакеры учли это и при сканировании стали отправлять пакеты с установленными флагами SYN и FIN, на которые можно получить ответ. В стеках TCP/IP разных операционных систем в ответ на полученный необычный пакет

возвращаются различные ответы, что используется для составления сигнатур. Реальные примеры использования нормальных и недействительных комбинаций TCP-флагов описаны в следующих разделах этой главы.

Искажение TCP-пакетов

По наличию в TCP-сегменте недопустимой комбинации установленных флагов еще нельзя сделать вывод о том, что обязательно действует злоумышленник. При доставке пакетов иногда возникают искажения, и установка TCP-флагов может являться результатом одного из таких искажений.

Рассмотрим следующий отчет о получении пакета. Это зарегистрированная NIDS попытка установить соединение со службой Napster еще в те времена, когда Napster была бесплатным и легальным средством для обмена файлами mp3.

```
host.home.com.1310 > napster.com.6699: SRP [bad hdr length] (DF)
```

В этой записи бросаются в глаза две аномалии. Первая заключается в наличии установленных флагов SYN, RESET и PUSH (строка `SRP`). Второй признак искажений — это уведомление TCPdump "`bad hdr length`".

Сообщение об ошибке "`bad hdr length`" выдается TCPdump, когда указанная длина TCP-заголовка больше действительной длины всего TCP-сегмента (заголовка и данных). Поскольку в IP-дейтаграмме нет поля, в котором бы хранилось значение длины TCP-сегмента, TCPdump вычисляет это значение на основании информации доступных полей. Для этого из общей длины IP-дейтаграммы вычитается значение длины IP-заголовка. В правильно сформированных пакетах в результате получается действительная длина TCP-сегмента. Одна из проверок TCPdump полученного пакета заключается в том, чтобы выяснить, не превышает ли указанное в пакете в байтах значение длины TCP-заголовка вычисленного значения всего TCP-сегмента. При наличии ошибки выдается уведомление.

Рассмотрим следующий отчет в шестнадцатеричном формате, чтобы понять, в чем же тут дело. Здесь IP-заголовок указан в квадратных скобках, а для выделения TCP-заголовка использованы символы "меньше" и "больше".

```
[4500 0028 8974 4000 7406 a9c5 1804 ee22
80f4 4c7b] <051e 1a2b 0000 029d 9efe a721
a7ae 5010 2058 ac31 0047 0050>
```

Сосредоточим наше внимание на значениях полей для хранения длины. Прежде всего, рассмотрим длину всей IP-дейтаграммы, указанную во 2-м и 3-м байтах (выделены жирным шрифтом) IP-заголовка. Это значение равно 0x28, что соответствует длине IP-дейтаграммы в 40 байт. Длина IP-заголовка содержится в полубайте нулевого байта IP-заголовка. Как вы помните, значение 5 соответствует длине в 20 байт.

Жирным шрифтом также выделено значение 9-го байта, которое указывает на протокол, использованный при создании вложенного пакета. Значение `06` обозначает протокол TCP. Затем вычисляется длина вложенного TCP-сегмента: 40 – 20 = 20 байт. Этого вполне хватит для хранения TCP-заголовка без параметров и без данных в пакете, что нормально для SYN-пакета.

Тем не менее судя по выделенному жирным значению старшего полубайта 12-го байта TCP-заголовка (0xa) длина TCP-заголовка равна 40 байт (для этого нужно умножить на 4 десятичное значение 32-битового слова).

Теперь понятно, почему TCPdump генерирует сообщение об ошибке `bad hdr length`? Эта дейтаграмма имеет общую длину 40 байт, включая 20 байт IP-заголовка, и размер TCP-заголовка тоже 40 байт. Следовательно, или IP-дейтаграмма должна быть размером 60 байт или произошло искажение пакета.

Может так случиться, что пакет был искажен и контрольная сумма неверна? Ведь если происходит искажение TCP-заголовка или данных, то обнаружить это можно только на хосте-получателе. Система обнаружения вторжений обычно не проверяет контрольную сумму TCP-сегмента.

Итак, что же можно сказать по поводу данного пакета? Скорее всего, информация IP-заголовка соответствует действительности, так как предыдущий маршрутизатор его не отбросил. Перед тем как попасть на хост-получатель, на котором должна быть проверена контрольная сумма TCP-сегмента, его анализирует TCPdump, который выявляет проблему. Возможно, на маршрутизаторе была искажена контрольная сумма IP-заголовка, но остальная информация этого заголовка должна быть правильной.

Таким образом, мы не можем точно сказать, был пакет искажен случайно или специально и по какой причине. Единственным способом установить искажение пакета является подсчет вручную контрольной суммы его TCP-заголовка на маршрутизаторе локальной сети или изучение реакции хоста-получателя (`napster.com`). Проблема в том, что если контрольная сумма TCP-заголовка окажется неправильной, никакой реакции от `napster.com` вообще не последует. И даже если контрольная сумма будет правильной, некорректная комбинация TCP-флагов может привести к отсутствию ответа. Если же мы действительно увидим неожиданный ответ от `napster.com` (наиболее вероятно возвращение TCP-сегмента с установленным флагом RESET), то контрольная сумма была правильной, и пакет не был искажен на пути его доставки. Это значит, что пакет, скорее всего, был специально сформирован с неверными значениями полей на хосте-отправителе. Кроме того, всегда существует возможность, что 16-битовые поля пакета просто поменяются местами. Это исказит его информацию, но никак не отразится на контрольной сумме.

Верн Паксон (Vern Paxson), создатель системы обнаружения вторжений под названием Bro, в своей статье "Bro. Система обнаружения вторжений в реальном времени" использует специальный термин *crud* для обозначения трафика, содержащего "безвредные ошибки реализации", который можно представить как шаблон вероятных отклонений, подобных настоящим атакам. Автор представил примеры отклонений в работе реальных стеков TCP/IP, регулярно устанавливающих флаг URG в SYN-пакетах, или флаг DF во фрагментах дейтаграмм. Хотя это и не касается искажения пакетов, но очень важно запомнить, что не всегда необычный трафик является вредоносным.

Биты флага ECN

До недавнего времени два старших бита байта TCP-флагов считались зарезервированными. Они не несли никакого значения и должны были хранить значения 0. Однако с появлением средств типа `nmap` оказалось, что эти биты могут оказать помощь при определении операционных систем удаленных хостов. В различных операционных системах стеки TCP/IP по-разному реагируют на получение пакетов с установленными значениями этих битов.

В некоторых операционных системах их значения просто сбрасываются до 0, в других они остаются без изменений. Следовательно, полученные ответы можно использовать при определении операционных систем. Хотя этой информации может оказаться недостаточно для полной уверенности, но при совместном применении с другими тестами определение типа и версии операционной системы удаленного хоста осуществляется с большой долей вероятности.

В главе 8, "Исследование полей IP-заголовка", мы рассказывали о новом предназначении двух *младших* битов ECN (Explicit Congestion Notification – явное уведомление о перегрузке) байта "Тип обслуживания". С их помощью маршрутизатор может уведомлять отправителя о существовании перегрузки в сети и о необходимости снижения скорости передачи данных.

Как именно это происходит? В настоящее время, как указано в RFC 3168, единственным протоколом транспортного уровня, способным поддерживать уведомления о перегрузке, является TCP. В этом RFC предлагается использовать два *старших* бита байта TCP-флагов в качестве полей для хранения битов ECN (рис. 9.3). Правый из этих битов называют *эхо-битом* ECN. Этот бит устанавливается, когда в принятом пакете установлен бит уведомления о перегрузке в байте "Тип обслуживания" IP-заголовка. Это означает, что обе стороны, взаимодействующие по протоколу TCP, поддерживают уведомления о перегрузке, что определяется в процедуре согласования параметров TCP-соединения.

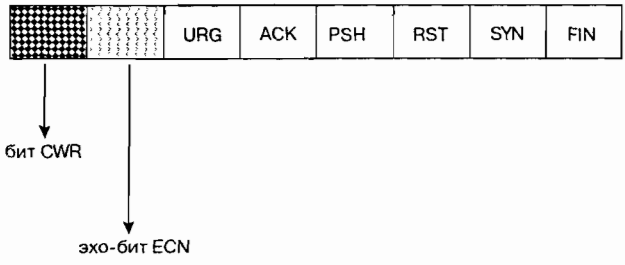


Рис. 9.3. Биты ECN байта TCP-флагов

Эхо-бит ECN установленный в TCP-заголовке, уведомляет отправителя о необходимости снизить скорость передачи данных из-за перегрузки в сети между отправителем и получателем. После получения TCP-сегмента с установленным эхо-битом ECN отправитель в два раза уменьшает *окно перегрузки* – размер буфера отправляемых данных. После этого он устанавливает бит CWR (Congestion Window Reduced – уменьшенное окно перегрузки) для уведомления второй стороны о предпринятых действиях для снижения перегрузки. Этот бит является старшим битом байта TCP-флагов.

Этот механизм позволяет уменьшить количество отброшенных пакетов, но установка дополнительных битов может привести к сообщениям о тревоге от систем обнаружения вторжений. В настоящее время большинство пользователей применяют эти биты только с целью сканирования. Кроме того, некоторые устройства фильтрации пакетов могут не пропустить входящие пакеты с этими установленными флагами. Поэтому придется предпринять еще много действий для постепенного внедрения битов ECN и для возможности отличать их от попыток сканирования.

Определение операционной системы

Когда программа nmap запускается в режиме определения операционной системы удаленного хоста (с помощью параметра командной строки -O), она отправляет несколько пакетов с недопустимой комбинацией установленных флагов на открытый порт. Рассмотрим следующий отчет о сканировании nmap операционной системы удаленного хоста.

```
nmap -O win98

20:33:16.409759 verbo.47322 > win98.netbios-ssn: SFP
861966446:861966446(0)
↳win 3072 urg 0 <wscale 10,nop,mss 265,timestamp 1061109567[|tcp]>
20:33:16.410387 win98.netbios-ssn > verbo.47322: S 49904150:49904150(0)
↳ack 861966447 win 815 <mss 1460> (DF)

nmap -O sparky

20:37:00.738412 verbo.50107 > sparky.echo: SFP 2326441544:2326441544(0)
↳win 2048 urg 0 <wscale 10,nop,mss 265,timestamp 1061109567[|tcp]>

nmap -O linux

20:44:50.370158 verbo.42318 > linux.ftp: SFP 1749165064:1749165064(0)
↳win 1024 urg 0 <wscale 10,nop,mss 265,timestamp 1061109567 0,eol>
```

При сканировании первого хоста win98 на порт 139 отправляется пакет с установленными флагами SYN, FIN, PUSH и URG. Это порт службы NetBIOS, на который ожидается поступление запросов. Все отлично — хост отвечает подтверждением. Это необычная ответная реакция на получение нестандартного пакета.

Во второй попытке сканирования пакет с тем же набором флагов отправляется на открытый порт (echo) хоста под управлением Solaris. Никакого ответа не выдается. В третьем примере сканирования все тот же пакет отправляется на порт службы FTP хоста под управлением Linux, и опять ответ не выдается. Это стандартная реакция на получение подобного пакета, описанная в спецификациях RFC. Таким образом, этот тест продемонстрировал, как можно отличить Windows-хост от других хостов.

Начинающим аналитикам часто бывает трудно отличить вредоносные действия от нестандартной работы стека TCP/IP. Непонятно, почему ответ отличается от ожидаемого. Во многих случаях даже опытные аналитики не в состоянии сказать, означает нестандартный набор TCP-флагов в пакете непредсказуемый сбой в работе TCP/IP или преднамеренные действия хакера.

Повторная передача данных

Допустим, что при попытке установить TCP-соединение инициатору этого соединения не возвращается никакого ответа. Возможно, хост-получатель не активен или вообще отсутствует. Маршрутизатор отправляет ICMP-уведомление о недостижимости хоста-получателя только в том случае, если такие ответы не запрещены. Возможно, перед хостом-получателем установлено какое-то устройство фильтрации пакетов, которое блокирует входящие пакеты и отбрасывает их без какого-либо уведомления отправителя.

Кроме того, существует вероятность, что хост-получатель ответил на отправленный запрос на соединение положительно (установлены флаги SYN и ACK) или

отрицательно (установлены флаги RESET и ACK), но хост-отправитель по какой-то причине не получил этого ответа.

В подобных ситуациях предпринимаются дополнительные попытки или повторные передачи данных. Количество повторных попыток и промежутки времени между отправкой повторных запросов зависят от операционной системы, установленной на хосте-отправителе. В конечном итоге отправитель прекращает безуспешные попытки установить соединение.

Как можно отличить повторные запросы от новых попыток установки TCP-соединения? Для повторной передачи данных неизменными остаются номер порта отправителя и порядковые номера TCP-сегмента. Этот метод не является абсолютно надежным. Возможно, что отправитель специально сформировал пакеты с одинаковыми номерами порта отправителя и порядковыми номерами.

Рассмотрим следующий отчет и обратим особое внимание на изменение времени получения пакетов, а также идентификаторов IP-заголовка. Значения идентификаторов IP-заголовка должны изменяться при повторных запросах, как и другие уникальные параметры попытки соединения, так как хост-отправитель генерирует абсолютно новый пакет для повторной передачи данных.

```
17:14:18.726864 1.1.1.1.62555 > 192.168.44.63.3128: S 20583734:20583734(0)
↳win 8192 <mss 1380> (DF) (ttl 17, id 15697)
17:14:21.781140 1.1.1.1.62555 > 192.168.44.63.3128: S 20583734:20583734(0)
↳win 8192 <mss 1380> (DF) (ttl 17, id 33873)
17:14:27.776662 1.1.1.1.62555 > 192.168.44.63.3128: S 20583734:20583734(0)
↳win 8192 <mss 1380> (DF) (ttl 17, id 46113)
17:14:39.775929 1.1.1.1.62555 > 192.168.44.63.3128: S 20583734:20583734(0)
↳win 8192 <mss 1380> (DF) (ttl 17, id 54353)
```

Рассмотрим временные метки отправленных пакетов. Между первой и второй попытками проходит около 3 с, между второй и третьей в два раза больше – 6 с, а между третьей и четвертой – 12 с. Это удвоение промежутка времени между повторными запросами наблюдается не всегда – в различных стеках TCP/IP применяются различные алгоритмы для определения времени ожидания для следующей попытки отправить запрос.

Часто аналитики сетевого трафика, которые не знакомы с правилами повторной отправки запросов, могут неправильно истолковать происходящее. Они ошибочно принимают подобный трафик за множественные попытки нарушителя установить соединение с атакуемым хостом. На самом деле это автоматически отправляемые повторные запросы TCP.

Использование повторных ответов против хоста нарушителя

Тома Листона (Tom Liston) можно назвать очень опытным специалистом по проблеме защиты от сканирований Web-серверов, выполняемых червем Code Red. Он написал программу, которая блокирует работу сканеров, выполняющих поиск свободных IP-адресов. Как правило, если регистрируются действия по поиску свободных IP-адресов, это означает, что кто-то сканирует хосты вашей сети. Том Линстон назвал свою программу LaBrea по имени знаменитого "кладбища" животных La Brea.

Программа LaBrea устанавливается на локальный хост и ожидает появления ARP-запросов для свободных IP-адресов. Обычно такой ARP-запрос генерирует

маршрутизатор при поиске компьютера с неизвестным IP-адресом. Если в течение трех секунд ни от одного реального хоста не поступает ARP-ответа, то LaBrea подменяет этот ответ своим ARP-ответом.

Если затем со сканирующего хоста (в данном случае с хоста, зараженного Code Red) следует SYN-пакет по тому же адресу "найденного" хоста, то LaBrea генерирует подложный ответ с установленными флагами SYN и ACK (LaBrea не проверяет значение порта получателя, поэтому может использоваться против любого сканирования по протоколу TCP или попытки установить TCP-соединение по свободному IP-адресу). Сканирующий хост завершает процедуру установки соединения и пытается отправить какие-то данные. Программа LaBrea умышленно не выдает никаких ответов и не отправляет никаких подтверждений о получении данных. Таким образом, сканирующий хост попадается в ловушку отправки бесполезных повторных передач данных до тех пор, пока не истечет заданный лимит времени. Это требует расхода ресурсов сканирующего хоста и уменьшает его возможности сканирования, особенно, если он ожидает получения ответа перед тем, как продолжить процесс сканирования.

Давайте рассмотрим все происходящие действия шаг за шагом.

```
Поступает ARP-запрос для свободного IP-адреса 192.168.143.236

18:34:32.757821 arp who-has 192.168.143.236 tell 192.168.143.1
18:34:35.743528 arp who-has 192.168.143.236 tell 192.168.143.1
После 3 с без ARP-ответа хост, на котором установлена LaBrea,
↳ отправляет подложный ответ
18:34:35.743591 arp reply 192.168.143.236 (0:0:f:ff:ff:ff) is-at
↳0:0:f:ff:ff:ff
```

Вначале LaBrea ожидает появления ARP-запросов в локальной сети. Как правило, они поступают от маршрутизатора локальной сети. Если по истечению 3 с (это значение используется по умолчанию, но его можно изменить) никаких ARP-ответов не поступает, то LaBrea подменяет настоящий ARP-ответ. В этом случае мы видим ARP-запрос к хосту `192.168.143.236` (свободный IP-адрес) от локального маршрутизатора `192.168.143.1`. ARP-ответа нет, и через три секунды генерируется повторный ARP-запрос. Практически в тот же момент, через 3 с после первого запроса, программа LaBrea выдает подложный ARP-ответ, в котором указывает, что MAC-аресом хоста `192.168.143.236` является `0:0:f:ff:ff:ff`. Интересно, что ни IP-адрес `192.168.143.236`, ни MAC-адрес `0:0:f:ff:ff:ff` в реальности не существуют. Теперь LaBrea будет просматривать проходящий трафик, предназначенный подложному MAC-адресу.

После получения подложного MAC-адреса сканирующий хост отправляет SYN-пакет запроса на соединение, на который LaBrea генерирует ответ, фальсифицирующий работу запущенной службы на активном хосте, как это показано ниже.

```
Хост, зараженный червем Code Red, отправляет SYN-пакет

18:34:35.743817 codered.victim.com.1113 > 192.168.143.236.www: S
↳301190748: 301190748(0) win 8192 <mss 1460,nop,nop,sackOK> (DF)

Хост с программой LaBrea генерирует подложное подтверждение

18:34:35.743940 192.168.143.236.www  > codered.victim.com.1113: S
↳2516582400:2516582400(0) ack 301190749 win 10
```

Хост, зараженный червем Code Red, завершает процедуру установки соединения

```
18:34:35.744190 codered.victim.com.1113 > 743940 192.168.143.236.www: . ack 1
win 8576 (DF)
```

В приведенном выше отчете зарегистрирована попытка хоста `codered.victim.com` отправить SYN-пакет на порт 80 (www) свободного IP-адреса `192.168.143.236`. Программа LaBrea в ответ генерирует SYN/ACK-пакет от имени несуществующего хоста `192.168.143.236`. Как и ожидалось, сканирующий хост завершает процедуру установки соединения.

Затем хост `codered.victim.com` пытается отправить 10 байт данных по адресу несуществующего Web-сервера.

```
Хост, зараженный червем Code Red, отправляет 10 байт данных
18:34:35.745555 codered.victim.com.1113 > 192.168.143.236.www: . 1:11(10)
↳ack 1 win 8576 (DF)
Повторная передача через 6 с
18:34:41.746643 codered.victim.com.1113 > 192.168.143.236.www: . 1:11(10)
↳ack 1 win 8576 (DF)
Повторная передача через 12 с
18:34:53.743027 codered.victim.com.1113 > 192.168.143.236.www: . 1:11(10)
↳ack 1 win 8576 (DF)
Повторная передача через 24 с
18:35:17.735734 codered.victim.com.1113 > 192.168.143.236.www: . 1:11(10)
↳ack 1 win 8576 (DF)
Повторная передача через 48 с
18:36:05.741181 codered.victim.com.1113 > 192.168.143.236.www: . 1:11(10)
↳ack 1 win 8576 (DF)
Повторная передача через 96 с
18:37:41.911995 codered.victim.com.1113 > 192.168.143.236.www: . 1:11(10)
↳ack 1 win 8576 (DF)
Через 3 минуты 6 секунд попытки повторной передачи прекращаются
```

В отправляемых пакетах установлен только флаг ACK, подтверждающий получение подложного пакета с флагами SYN и ACK от несуществующего хоста `192.168.143.236`.

Вот ловушка и сработала. Подтверждение получения данных, отправленных хостом `codered.victim.com`, не выдается. Поэтому данный хост вынужден проводить повторную передачу данных. При этом используется алгоритм удвоения времени задержки между повторными попытками. В различных случаях запуска программы LaBrea промежуток времени между начальным и первым повторным запросом сканирующего хоста находится в диапазоне от 3 до 12 с.

После отправки начального пакета данных проходит 3 мин и 6 с, за которые хост `codered.victim.com` осуществляет 5 попыток передачи данных, а затем прекращает эти попытки. Но за это время были потрачены ресурсы этого компьютера и был замедлен процесс сканирования (особенно, если сканирующий хост ожидает ответа от несуществующего хоста до продолжения сканирования). Еще больший эффект достигается, когда сканирующий хост снова и снова попадается в ловушку повторных запросов при сканировании всех свободных IP-адресов локальной сети.

Использование программы LaBrea может показаться весьма удобным, однако стоит задуматься и о ее недостатках. Во-первых, в ловушку попадает любая попытка TCP-соединения по свободному IP-адресу локальной сети, независимо от

указанного порта получателя. Если реальный хост локальной сети временно вышел из строя и не способен ответить на ARP-запрос, то по ошибке могут быть пойманы в ловушку легитимные соединения. Во-вторых, работа брандмауэров, поддерживающих таблицы состояний текущих соединений, может быть затруднена бесконечными повторными попытками. Программный код LaBrea можно получить по адресу `www.hackbusters.net`.

Дегтярная ловушка

Знаменитая ловушка для доисторических животных La Brea находится в парке Хенкок города Лос-Анджелес. В этом месте нефть выходила на поверхность и превращалась в деготь. Более 2,5 миллионов лет назад животные забредали в эту ловушку и погибали в ней.

Размер окна

С помощью значения размера окна хост-получатель уведомляет отправителя о текущем размере входного буфера для конкретного соединения. Значение размера окна изменяется динамически в ходе соединения. Оно становится меньше на размер данных, принятых, но еще не обработанных хостом-получателем. Если входной буфер получателя заполняется полностью, то размер окна уменьшается до 0, что сообщает хосту-отправителю о необходимости временно прекратить передачу данных. Обработав часть данных из входного буфера, хост-получатель посылает отправителю обновленную информацию о размере окна, указывающую на возможность возобновления передачи данных.

Очевидно, что хост-получатель управляет потоком TCP-данных в основном с помощью информации о размерах окна. У многих людей сложилось представление, что потоком данных в сети управляет только отправитель. На самом деле управляет процессом в основном получатель.

В различных операционных системах используются различные начальные значения размера окна, поэтому они используются при определении операционной системы удаленного хоста с помощью программы `nmap`.

Программа LaBrea, версия 2

Как вы помните, оригинальная версия программы LaBrea позволяла замедлить работу сканирующего или атакующего хоста на период времени ожидания ответа на получение данных, отправленных после завершения полной процедуры установки соединения.

Том Линстон, автор LaBrea, усовершенствовал свою программу, использовав другой метод, который назвал *таймером удержания* TCP-соединения. Как было сказано, если входной буфер хоста-получателя будет заполнен, то отправитель будет уведомлен о необходимости приостановить передачу данных (в ответном пакете размер окна равен 0 байт). Обычно при освобождении части буфера отправителю немедленно посылается TCP-сегмент с указанием доступного размера окна. А что если это уведомление будет потеряно? И отправитель, и получатель будут заблокированы в ожидании ответа от второй стороны соединения.

Для преодоления такой проблемы предназначен механизм под названием *зондирование окна*. После истечения лимита времени на удержание TCP-соединения, когда отправитель так и не получает уведомления о новом значении размера ок-

на, хост-отправитель отсылает получателю пробный пакет. В этом пакете передается 1 байт полезных данных с единственной целью: получить ответ с обновленным значением размера окна. Хост-отправитель продолжает зондировать размер окна получателя до тех пор, пока это значение не увеличится, или пока этот процесс не прекратит приложение, организовавшее данное соединение.

Как вы увидите в следующем отчете TCPdump, новая версия программы LaBrea позволяет поймать в ловушку хост нарушителя на неопределенный промежуток времени. Как и в прошлой версии, сначала осуществляется полная процедура установки соединения, но в ответ на передачу данных LaBrea отправляет уведомление о получении с указанием размера окна, равного 0. Эта игра "вопрос-ответ" может продолжаться бесконечно. Таким образом обеспечивается длительное соединение атакующего хоста с хостом, на котором запущена программа LaBrea, без передачи каких-либо данных. Рассмотрим отчет.

```
19:28:07.577541 codered.victim.com.2045 > 10.10.10.155.www: S
↳882335286:882335286(0) win 8192 <mss 1460,nop,nop,sackOK> (DF)
19:28:07.577618 10.10.10.155.www > codered.victim.com.2045: S
↳998514038:998514038(0) ack 882335287 win 5
19:28:07.577879 codered.victim.com.2045 > 10.10.10.155.www: . ack 1 win 8576
↳ (DF)

19:28:07.581366 codered.victim.com.2045 > 10.10.10.155.www: . 1:6(5) ack 1
↳ win 8576 (DF)
19:28:07.581437 10.10.10.155.www > codered.victim.com.2045: . ack 6 win 0
19:28:09.820965 codered.victim.com.2045 > 10.10.10.155.www: . :6:7(1) ack 1
↳ win 8576 (DF)
19:28:09.821041 10.10.10.155.www > codered.victim.com.2045: . ack 6 win 0
19:28:14.424567 codered.victim.com.2045 > 10.10.10.155.www: . 6:7(1) ack 1
↳ win 8576 (DF)
19:28:14.424646 10.10.10.155.www > codered.victim.com.2045: . ack 6 win 0
19:28:23.621770 codered.victim.com.2045 > 10.10.10.155.www: . 6:7(1) ack 1
↳ win 8576 (DF)
19:28:23.621845 10.10.10.155.www > codered.victim.com.2045: . ack 6 win 0
19:28:42.016162 codered.victim.com.2045 > 10.10.10.155.www: . 6:7(1) ack 1
↳ win 8576 (DF)
19:28:42.016237 10.10.10.155.www > codered.victim.com.2045: . ack 6 win 0
19:29:18.804962 codered.victim.com.2045 > 10.10.10.155.www: . 6:7(1) ack 1
↳ win 8576 (DF)
19:29:18.805038 10.10.10.155.www > codered.victim.com.2045: . ack 6 win 0
```

Мы начали просмотр отчета с момента после отправки подложного ARP-ответа хостом, на котором установлена программа LaBrea (от имени хоста с IP-адресом `10.10.10.155`). Хост, инфицированный Code Red, здесь указан как `codered.victim.com`. Этот хост отправляет 5 байт данных (выделено жирным шрифтом), так как это позволяет указать размер окна несуществующего хоста `10.10.10.155`. Хост с программой LaBrea отвечает уведомлением о получении данных, но указывает, что размер окна стал равен 0. Хост `codered.victim.com` ожидает несколько секунд и, так как не получает никакого уведомления об увеличении размера окна, то отправляет пробный пакет, содержащий 1 байт данных. В ответ на это он получает "успокаивающий" пакет, которым LaBrea уведомляет, что интересующий нарушителя хост по-прежнему работает, но еще не готов к получению данных. Как вы видите, этот цикл запрос-ответ продолжается бесконечно, при этом промежуток времени между повторными запросами постоянно увеличивается.

UDP

Изучение протокола UDP значительно проще, чем TCP, так как в UDP не существует никаких полей, предназначенных для обеспечения надежной доставки данных. Контроль за доставкой данных при использовании UDP осуществляется самими приложениями. В этом разделе будут рассмотрены поля заголовка UDP-пакета и методы сканирования UDP-портов.

Порты

Как и в TCP-пакете, в пакете UDP для хранения номеров портов отправителя и получателя используются два 16-битовых поля. Действительный диапазон значений — от 1 до 65535, использование номера 0 не предусмотрено в спецификациях протокола.

При установке UDP-соединения на хосте-отправителе, как правило, выбирается один из временных портов с номером выше 1023. Для каждого нового соединения должен быть выбран новый порт.

Сканирование UDP-портов

В отличие от TCP-портов, которые должны выдавать или положительный ответ (пакет с флагами SYN/ACK) для открытого порта или негативный ответ (пакет с установленным флагом RESET) для закрытого порта, в протоколе UDP не предусмотрено ответа на успешную доставку начального пакета. Но при попытке подключения к закрытому UDP-порту активный хост отправляет ICMP-уведомление о недостижимости порта. По наличию этого ответа программы сканирования и определяют, открыт или закрыт конкретный UDP-порт.

Таким образом, вывод об открытости порта делается на основании отсутствия ICMP-сообщения об ошибке "`port unreachable`". Но ведь пакет сканирующей программы может быть утерян на пути к получателю, или ответное ICMP-сообщение может быть заблокировано. Или на узле получателя блокируются входящие UDP-пакеты без какого-либо уведомления отправителя об этом. Все перечисленные варианты могут привести к ошибке при определении открытых портов. Для устранения возможности случайной потери пакетов программа nmap отправляет несколько тестовых пакетов на один и тот же UDP-порт. Кроме того, при полном отсутствии ответов этой программой делается вывод о выполнении фильтрации пакетов, а не о том, что все UDP-порты сканируемого хоста открыты.

Рассмотрим пример сканирования UDP-портов с помощью nmap для диапазона номеров от 32771 до 34000 при поиске открытых портов службы RPC (Remote Procedure Call — удаленный вызов процедур) на хосте под управлением Solaris. На основе отсутствия ICMP-сообщения о недостижимости порта программа nmap делает вывод о наличии многих открытых портов. Как мы знаем, это не всегда соответствует действительности.

```
Nmap -sU sparky -p 32771-34000

WARNING: -sU is now UDP scan - for TCP FIN scan use -sF
Starting nmap V.2.12 by Fyodor (fyodor@dhp.com, www.insecure.org/nmap/)
Interesting ports on sparky (1.1.1.100):
```

```
Port    State       Protocol    Service
32771   open        udp         unknown
32772   open        udp         unknown
32773   open        udp         unknown
32774   open        udp         unknown
32782   open        udp         unknown
32783   open        udp         unknown
32784   open        udp         unknown
32785   open        udp         unknown
32786   open        udp         unknown
32797   open        udp         unknown
```

В следующем отчете TCPdump о данном сканировании UDP-портов показано, что считаются открытыми все порты указанного диапазона, для которых хост sparky не сгенерировал ICMP-сообщения "`port unreachable`".

```
07:09:08.286810 verbo.62865 > sparky.32787: udp
07:09:08.286847 verbo.62865 > sparky.32775: udp
07:09:08.286878 verbo.62865 > sparky.32788: udp
07:09:08.286924 verbo.62865 > sparky.32789: udp
07:09:08.286969 verbo.62865 > sparky.32791: udp
07:09:08.287046 verbo.62865 > sparky.32774: udp
07:09:08.287094 verbo.62865 > sparky.32781: udp
07:09:08.287162 verbo.62865 > sparky.32772: udp
07:09:08.287229 verbo.62865 > sparky.32789: udp

07:09:08.287793 sparky > verbo: icmp: sparky upd port 32788 unreachable (DF)
07:09:08.977544 sparky > verbo: icmp: sparky upd port 32791 unreachable (DF)
07:09:09.657361 sparky > verbo: icmp: sparky upd port 32781 unreachable (DF)
07:09:10.157301 sparky > verbo: icmp: sparky upd port 32787 unreachable (DF)
07:09:10.817315 sparky > verbo: icmp: sparky upd port 32789 unreachable (DF)
```

Поле длины UDP-пакета

Длина UDP-пакета складывается из длин UDP-заголовка и полезных данных. Поскольку минимальный размер UPD-заголовка равен 8 байт, то и минимальный размер UDP-пакета тоже 8 байт. Максимальный теоретический размер IP-дейтаграммы составляет 65535 байт. Таким образом, отняв 20 байт IP-заголовка, получим максимальный теоретический размер UDP-пакета – 65515 байт.

Во многих UDP-приложениях максимальный размер UDP-пакета ограничен 8192 байт, хотя служба DNS ограничивает этот размер 512 байтами. Кроме того, в ядре операционной системы могут быть установлены собственные ограничения размера UDP-пакетов.

ICMP

ICMP представляет собой тоже относительно простой протокол, который не гарантирует доставки сообщений. Рассмотрим стандартные и необычные значения полей заголовка ICMP-пакета.

Тип и код

В ICMP-сообщениях не указываются порты. Поэтому для указания типа ICMP-сообщения используются его первые два байта, хранящие значения типа и кода

соответственно. Код ICMP-сообщения используется для дальнейшего подразделения ICMP-сообщений определенного типа.

Например, существуют два вероятных кода ICMP-сообщения типа 11, который указывает на категорию превышения лимита времени. Если код сообщения равен 0, то это сообщение "`time exceeded in-transit`" ("превышение лимита времени на доставку сообщения"), а если код равен 1, то это сообщение "`reassembly time exceeded`" ("превышение лимита времени повторной сборки пакета"). Доступные значения типов и кодов ICMP-сообщений можно узнать по адресу `www.iana.org/assignments/icmp-parameters`.

Идентификаторы и порядковые номера

Если рассмотреть некоторые ICMP-запросы, например эхо-запрос, то в ICMP-заголовке можно обнаружить дополнительные поля. Это идентификатор, хранящийся в 4-м и 5-м байтах ICMP-заголовка, и порядковый номер, хранящийся в 6-м и 7-м байтах ICMP-заголовка.

Значения этих полей используются для установки соответствий между парами эхо-запрос/эхо-ответ. Для хостов под управлением UNIX значение идентификатора ICMP-заголовка обычно соответствует идентификатору процесса, который послужил причиной отправки ping-пакета. Одновременно могут быть выполнены несколько команд `ping`, а идентификатор эхо-ответа позволяет узнать, какому эхо-запросу соответствует этот ответ. Каждая команда `ping` может являться причиной отправки нескольких эхо-запросов, а порядковые номера позволяют выявить утерянные пакеты. Рассмотрим результат выполнения `ping`-запроса, в котором продемонстрировано изменение порядковых номеров ICMP-заголовка.

```
PING sparky (1.1.1.100) from 1.1.1.5 : 56(84) bytes of data.

64 bytes from 1.1.1.100: icmp_seq=0 ttl=255 time=0.8 ms
64 bytes from 1.1.1.100: icmp_seq=1 ttl=255 time=0.9 ms
64 bytes from 1.1.1.100: icmp_seq=2 ttl=255 time=7.3 ms

16:33:07.400700 verbo > sparky: icmp: echo request

4500 0054 038d 0000 4001 bed1 0101 0105
0101 0164 0800 9e12 c402 0000 0391 8439
1010 0600 0809 0a0b 0c0d 0e0f 1011 1213
1415 1617 1819
```

Давайте рассмотрим на этом примере значения полей идентификатора и порядкового номера ICMP-сообщения. Здесь хостом `verbo` отправляются `ping`-запросы на хост `sparky`. Значения порядковых номеров запросов начинаются с 0 и увеличиваются для каждого нового запроса. В данном случае процесс отправки `ping`-запросов был прерван после третьего эхо-запроса.

Если посмотреть на отчет в шестнадцатеричном формате, можно увидеть, что шестнадцатеричным значением идентификатора является `c402`, что соответствует десятичному значению 50178. Так как запрашивающий хост является Linux-хостом, то можно предположить, что данный идентификатор является идентификатором процесса, вызванного командой `ping`. Это значение остается постоянным для всех эхо-запросов и эхо-ответов, связанных с этой командой. С другой стороны, порядковый номер будет увеличиваться на 1 для каждого нового эхо-запроса и указываться в соответствующем эхо-ответе. Если бы были отображены

все эхо-запросы и эхо-ответы, связанные с данным ping-процессом, мы бы увидели еще 4 записи — 2 эхо-запроса и 2 эхо-ответа. Для них идентификатор оставался бы прежним, а порядковые номера были бы равны 1 для второй пары запрос-ответ и 2 — для третьей пары.

Некорректное использование полей идентификатора и порядкового номера

Раньше значения полей идентификатора и порядкового номера ICMP-сообщения тщательно не проверялись, поэтому они были выбраны хакерами в качестве сигнального трафика для передачи команд на хост-получатель. Например, в распределенной атаке отказа в обслуживании под названием Stacheldraht значение поля идентификатора 667 эхо-ответа ICMP используется для организации соединения между главным и подчиненными хостами. Значение 666 другого эхо-ответа использовалось для ответа подчиненного хоста (агента) в этой атаке. В атаке TFN значение идентификатора 456 ICMP-сообщения позволяет организовать соединение между клиентом и демоном, а значение 123 используется для ответа клиента (тоже в эхо-ответах). И, наконец, в программе атаки Loki много лет назад было задано статическое шестнадцатеричное значение порядкового номера ICMP-сообщения 0xf001 или 0x01f0.

Все перечисленные значения являются допустимыми значениями для этих полей, поэтому необходима точная настройка системы обнаружения вторжений на блокирование пакетов именно с этими значениями. В противном случае будут возникать многочисленные предупреждения о ложных тревогах. Такие пакеты лучше всего анализировать в контексте полученного трафика.

Резюме

В этой главе были рассмотрены поля заголовков вложенных пакетов IP-дейтаграммы. Безусловно, самым сложным является заголовок TCP-сегмента, так как поля этого заголовка должны обеспечивать надежность, порядок данных, а также управление потоком. Нетрудно догадаться, что начальные значения, задаваемые для некоторых из этих полей, представляют собой ценную информацию при определении типа и версии операционной системы удаленного хоста с помощью программы nmap. Кроме того, часть полей может использоваться для атак со вставкой или скрытых атак, как было показано в примере с контрольной суммой TCP в предыдущей главе.

Понять предназначение полей UPD- и ICMP-заголовков значительно проще. Для сканирования UDP-портов с помощью программы nmap используется ответное ICMP-сообщение о недостижимости порта. Исходящие ICMP-сообщения могут использоваться в разведывательных целях, а по значениям полей ICMP-заголовка можно определить операционные системы удаленных хостов. Наконец, значения полей идентификатора и порядкового номера ICMP-заголовка используются для скрытого проведения распределенных атак отказа в обслуживании или обмена данными.

10

Практический анализ

Несомненно, каждый читатель этой книги имеет собственную теорию о правилах проведения анализа пакетов и полей их заголовков. А как насчет применения своих знаний на практике? Именно этой теме посвящена данная глава.

Чтобы обновить ваши знания, рассмотрим примеры трафика с различных точек зрения, начиная от значения отдельных битов и полей пакетов и заканчивая предназначением всех пакетов или их последовательности.

Переход от теоретических знаний к практическим навыкам анализа трафика не всегда проходит легко и просто. Потребуется много времени для того, чтобы набраться достаточно опыта и делать уверенные выводы при исследовании проходящего трафика. Примеры этой главы помогут вам начать этот сложный путь.

Компьютер взломан

Насколько прост наш первый пример реального трафика, настолько же и серьезна представленная ситуация. Когда-то я возглавлял группу по обнаружению взломов (CERT) в одном из департаментов Министерства обороны США. Моя смена начиналась очень рано, около 5.30 утра. Это было связано с предотвращением негативного воздействия мощного потока трафика, поступающего с предместий Вашингтона — одного из крупнейших коммуникационных узлов США. Однажды утром, когда я пришел в офис, там уже звонил телефон. Вряд ли кто-то собирался поздравить меня с выигрышем в лотерею. И в самом деле звонок был от одной из вышестоящих организаций. Мне сообщили, что ночью наша система была взломана.

Необходимо пояснить, что в вышестоящей организации CERT использовался набор средств, предназначенных для дополнительного контроля за нашей сетью. При появлении подозрительного трафика или других важных событиях (как в этом случае) они звонили нам. При этом сообщалось о дате, приблизительном времени, адресах отправителя и получателя пакетов, связанных с попыткой взлома. Больше никаких сведений не предоставлялось.

Хотя, судя по графикам на рис. 11.4 и 11.5, это неочевидно, но полученные пакеты можно было четко разделить по группам, соответствующим стандартным начальным значениям поля TTL. Например, при первом сканировании в полученных пакетах отсутствовали значения поля TTL в промежутках между 22 и 42, а также между 56 и 103. Подобное явление наблюдалось и при повторном сканировании.

Размер окна

Размер окна определяется при попытке установить соединение. Это значение может изменяться и указывает на доступный объем входного буфера для поступающих данных. Буфер позволяет организовать прием нескольких пакетов и установку их в очередь до передачи на уровень TCP и уровень приложений. В каждой операционной системе размер окна устанавливается по умолчанию. Эти начальные значения могут быть использованы для определения операционных систем удаленных хостов. Безусловно, администратор может установить другие значения размера окна, но чаще всего они остаются без изменений.

В большинстве полученных пакетов размер окна был 8192 (рис. 11.6). Согласно табл. 11.1 это указывает на операционную систему Windows9x/NT. Еще одно исследование позволило определить соответствие между стандартным размером окна 16384 и операционной системой Windows 2000. Значение размера окна 65535 соответствует хостам Cisco (см. табл. 11.1). Однако слишком большой процент пакетов с этим значением говорит о том, что это значение используется еще какими-то операционными системами.

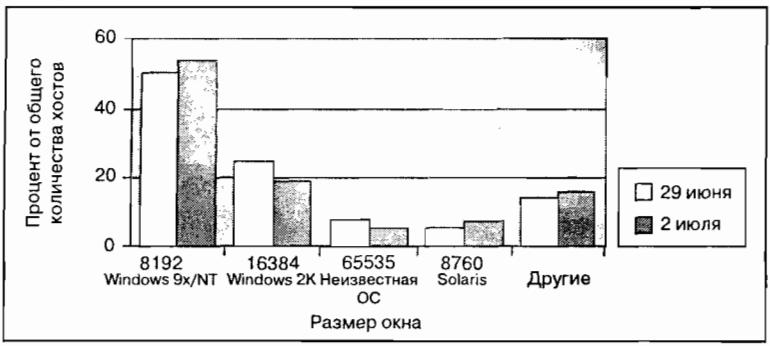


Рис. 11.6. Использованные при сканировании значения размера окна

Поисковые машины не смогли помочь в определении операционных систем, размер окна которых по умолчанию был бы равен 65535. Тогда мы решили проверить все отчеты TCPdump за неделю, чтобы найти пакеты с таким размером окна. Из 5500 хостов, пакеты от которых были зарегистрированы за неделю, было только около десятка, использовавших это значение. Их сканирование с помощью программы nmap не позволило определить операционную систему. На некоторых из этих хостов оказались открытыми порты 135 и 139, что указывает на операционную систему Windows до версии Windows 2000. На других хостах этой группы был открыт порт 445, который в Windows 2000 закреплен за протоколом SMB (Server Message Block – блок сообщений сервера), позволяющим двум ком-

пьютерам обмениваться данными поверх TCP/IP без необходимости применения промежуточного уровня NetBIOS over TCP/IP (NBT). Кроме того, остальные хосты, в пакетах которых был указан размер окна 65535, ожидали запросов на порты 111 (`portmapper`), 515 (служба `lpd` – демон построчной печати) и 6000 (служба X11), что указывает на UNIX-хосты. Таким образом, не удалось сделать никаких определенных выводов об операционных системах, использующих по умолчанию значение размера окна 65535.

Еще одним интересным значением размера окна было 32120, указывающее на хост под управлением Linux, которое было зарегистрировано только при первом сканировании и составило 0,19% от общего числа сканирующих хостов. Размер окна 8760, соответствующий хостам Solaris, был выявлен в обоих случаях сканирования (5,21% в первом сканировании и 6,6% – во втором).

Выводы, сделанные по результатам исследования размеров окна полученных TCP-пакетов, оказались аналогичны анализу значений поля TTL: большинство сканирующих хостов являются Windows-хостами, но в сканировании участвуют и хосты под управлением других операционных систем.

Параметры TCP

Еще одним важным значением является значение поля MSS (Maximum Segment Size – максимальный размер TCP-сегмента), которое входит в набор параметров TCP. Это значение указывает максимальный размер полезных данных TCP-сегмента. При этом не учитываются размеры IP- и TCP-заголовков. Вообще, можно сказать, что значение MSS на 40 байт меньше значения MTU (Maximum Transmission Unit – максимальная единица передачи данных в сети) при условии отсутствия параметров IP и параметров TCP. Значение MTU может использоваться для определения типа сети, в которой установлен хост-отправитель.

В некоторых случаях, хотя и не в данном примере, значение MTU, а значит, и MSS может указывать максимальный размер передаваемого пакета на пути к получателю. С помощью пакета с установленным флагом DF отправитель может послать "тестовый" пакет для определения минимального значения MTU на пути доставки пакета. Если никаких ICMP-сообщений об ошибках не будет получено, то считается, что размер MTU для локальной сети подойдет и для передачи по Internet без фрагментации. Если же возвращается ICMP-сообщение "`unreachable - need to frag (mtu ###)`", то указанный размер MTU (###) определяет минимальное значение размера пакета для участка сети на маршруте передачи пакета. Отправитель может уменьшить размер пакетов, чтобы избежать фрагментации. Обратите внимание на то, что в этом случае размер MSS не может быть использован для определения MTU сети отправителя. Однако, поскольку не было никаких признаков предварительной отправки "пробных" пакетов, мы предположили, что MSS полученных пакетов позволяют узнать MTU сетей, в которых работают хосты-отправители.

В результате анализа мы узнали, что большинство сканирующих хостов установлены в сетях со значением MTU, равным 1500 байт, что указывает на локальные сети Ethernet или DSL-сети (рис. 11.7). Значение MTU 576 связывают с протоколом PPP или ISDN-сетями. И, наконец, значение MTU 1454 указывает на использование PPP в сетях Ethernet или на DSL-соединения.

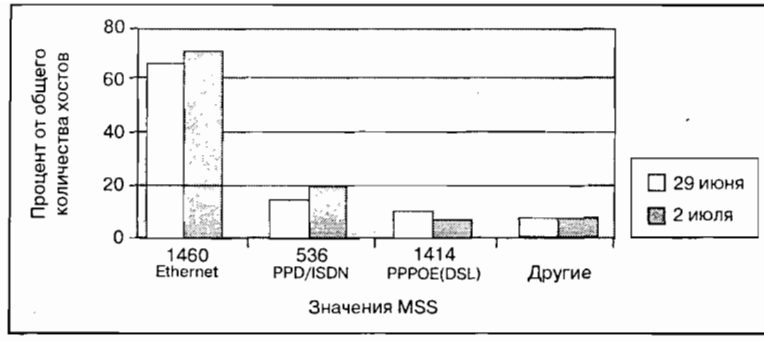


Рис. 11.7. Значения MSS

Хотя значение 536 для MSS указывает на протокол PPP и модемы коммутируемых линий связи, но все же вероятнее, что большинство этих хостов работают в сетях ISDN, для которых используется это же значение. Напомним, что все зомби-хосты должны были начинать сканирование одновременно. То есть каждый из них или должен был получить сигнал к атаке или использовался некий механизм отсрочки по времени, чтобы задать точный момент начала атаки.

Рассмотрим предположение об использовании для сканирования хостов, подключающихся к Internet по коммутируемым линиям. Во-первых, если для подключения к Internet зомби-хоста используется коммутируемая линия связи, то в момент атаки соединение может просто отсутствовать (если нет выделенного канала). Кроме того, для многих коммутируемых соединений выполняется распределение временных IP-адресов с помощью протокола DHCP. Как "командир" передаст сигнал о начале атаки, не зная IP-адреса зомби-хоста? Единственным вариантом является периодическое уведомление зомби-хостов организатора атаки о своих новых IP-адресах. Поэтому участие в атаке смогут принять только активные хосты, подключенные к сети в момент атаки.

Здесь возникает еще один вопрос. Практически точно установлено, что каждому зомби-хосту был выделен свой диапазон адресов сканирования. Использовалась ли какая-то формула, чтобы обеспечить сканирование максимального диапазона IP-адресов?

Мы предположили, что большинство участвующих в сканировании хостов обладали постоянными, выделенными соединениями с Internet. Тем не менее это не могло объяснить пропуски в сканировании отдельных хостов и подсетей.

Повторные запросы

Как уже не раз говорилось, если на попытку установить TCP-соединение отправитель не получает никакого ответа, осуществляются повторные попытки. Хост-отправитель не получает в ответ никакого уведомления, если запрос на соединение не достигает получателя, или если получатель не реагирует на доставку этого запроса. В нашем случае был заблокирован весь трафик, поступающий на порт 27474. Брандмауэр, блокирующий эти пакеты, просто отбрасывал их без какого-либо уведомления отправителя об ошибке. Ответы не возвращались, чтобы лишить злоумышленников возможности провести исследование и разведать средства защиты, установленные на периметре нашей сети.

В дальнейшем будем считать, что повторный запрос на установку TCP-соединения представляет собой TCP-сегмент с теми же IP-адресами и портами отправителя и получателя, а также порядковыми номерами, что и в начальном запросе. Количество повторных попыток и промежутки времени между их отправкой зависят от версии стека TCP/IP.

Повторные запросы связаны с исходным кодом, используемым при соединениях посредством сокетов. Другими словами, исходный код написан таким образом, что вызов функции сокета проходит через необходимые уровни стека TCP/IP. В этом случае сокет использует протоколы TCP и IP для создания соответствующих заголовков и установки значений полей этих заголовков.

Альтернативный механизм называется *сокетом для доступа к протоколам нижнего уровня* (raw socket), при котором для создания пакетов не используется стек TCP/IP. Для создания заголовков и их полей применятся специальная программа. Сформированный пакет передается непосредственно сетевому адаптеру. Многие программы сканирования, например `nmap` и `hping2`, используют сокеты доступа к протоколам нижнего уровня.

При этом сканировании отправляются многочисленные повторные запросы на установку соединения, несмотря на отсутствие какого-либо ответа от хоста-получателя. Что это значит? Были использованы обычные сокеты? Во-первых, сканирующий хост хочет максимально увеличить шанс получения ответа — явный признак сканирования, а не отправки потока данных. При отправке потока данных пакеты чаще посылаются с помощью сокетов доступа к протоколам нижнего уровня с целью ускорить передачу. Во-вторых, использование сокетов доступа к протоколам нижнего уровня усложняет проведение атаки, так как требуется установить интерфейс прикладного программирования, предназначенный для захвата пакетов на уровне пользователя на сканирующем хосте. Для этой цели используются интерфейсы `winpcap` (для Windows) или `libpcap` (для UNIX). Использование стандартных сокетов упрощает установку программного обеспечения, необходимого для сканирования.

Резюме

Основной вывод, сделанный по результатам нашего исследования, таков. Было проведено очень эффективное сканирование с целью выявления хостов, которые ожидают запросов на TCP-порт 27374. Сканирование осуществлялось зомби-хостами, большинство из которых работало под управлением Windows. Хосты, на которых были установлены другие операционные системы, тоже участвовали в сканировании, но играли второстепенную роль. Важно то, что "зараженными" оказались не только Windows-хосты. Неизвестно, насколько количество хостов под управлением Windows превышает число хостов под управлением других операционных систем, и не является ли процентное соотношение сканирующих хостов простым отображением популярности различных операционных систем. Преимущественное использование в качестве зомби-хостов хостов под управлением конкретной операционной системы говорит о простоте ее компрометации.

Является ли единственной целью этого сканирования найти хосты с открытым портом 27374? Мы предположили, что не все хосты были заражены с помо-

щью программы атаки SubSeven. Эта программа ориентирована на только на Windows-хосты. Возможно, здесь применялась улучшенная версия SubSeven, позволяющая также атаковать хосты под управлением других операционных систем. Какая бы программа атаки не управляла работой зомби-хостов, "командир" знает о существовании подчиненных хостов и не нуждается в их поиске с помощью сканирования. Является ли целью данного сканирования поиск других кандидатов в зомби-хосты, управлять которыми будет другой "командир"? В данном случае эти зомби-хосты будут хостами Windows, так как только они могут ожидать запросов на порт SubSeven. Новые зомби-хосты могут использоваться для любых целей, не только для сканирования.

Какой бы ни была цель сканирования, способ его выполнения очень эффективен. За несколько минут было просканировано более 30000 хостов. Это еще раз продемонстрировало увеличивающуюся сложность проводимых сканирований и программ атаки, примерами чему может послужить появление червей Code Red и nimda. Кроме того, очевиден рост количества используемых хостов, которые могут выполнять задания злоумышленников из-за невнимательности или беспечности домашних пользователей, компьютеры которых постоянно подключены к Internet, а также из-за появления операционных систем и прикладных программ с многочисленными уязвимыми местами.

III

Фильтры для контроля сетевого трафика

В этой части...

12

Создание фильтров TCPdump

Это первая из трех глав, в которых обсуждаются фильтры и сигнатуры, предназначенные для обнаружения нестандартного трафика. Для подробного обсуждения конкретных фильтров и сигнатур есть несколько причин. Первая из них – эти сигнатуры предоставляются бесплатно и доступны для широкого круга пользователей. Вторая причина заключается в том, что создано такое большое количество программных пакетов систем обнаружения вторжений, что рассмотреть их всех просто невозможно, из-за чего последуют обвинения в предвзятости. В качестве разумного компромисса мы решили посвятить эту главу сигнатурам TCPdump, а две следующие – сигнатурам Snort.

В этой главе обсуждается, каким образом с помощью фильтров можно выбрать определенные записи из отчета TCPdump. В следующей главе наши читатели познакомятся с бесплатной сетевой системой обнаружения вторжений Snort и ее сигнатурами. В последней из трех глав этой части изложена дополнительная информация о составлении сигнатур Snort.

Знаменитая программа TCPdump поставляется совместно с расширенным языком написания фильтров, который позволяет исследовать значения каждого поля, комбинации полей или даже отдельных битов IP-дейтаграмм. Если вам нравится решать загадки и вы не боитесь работы, то можете использовать фильтры TCPdump для обнаружения любого необычного трафика. Однако это средство не для тех, кто не хочет задумываться. Если вы предпочитаете получить готовое решение, то лучше будет использовать коммерческие программы с удобными графическими интерфейсами и предопределенными фильтрами.

В данной главе мы расскажем об использовании программы TCPdump и ее фильтров для обнаружения необычного сетевого трафика. TCPdump является ядром системы обнаружения вторжений Shadow, поэтому мы рекомендуем загрузить последнюю версию Shadow с сайта `www.nswc.navy.mil/ISSEC/CID`, чтобы изучить и улучшить исходные фильтры TCPdump. Shadow позаботится об автоматизации сборки и обработки трафика, а администратору останется только

сконцентрироваться на создании собственных фильтров TCPdump для улучшения критериев отбора интересующего трафика.

В этой главе также рассмотрен алгоритм создания фильтров TCPdump. Изложены различные методы анализа битов и байтов IP-дейтаграмм с помощью этих фильтров. Примеры создания фильтров TCPdump покажут, как можно получать отчеты об интересующих событиях. Мы начнем с изучения самых простых, а завершим главу описанием более сложных и усовершенствованных фильтров.

Правила создания фильтров TCPdump

По умолчанию TCPdump исследует или всю информацию о событиях в сети или собирает все записи из какого-то файла. Но что делать, если нужно обнаружить записи с определенными значениями полей в IP-дейтаграммах, которые бы указывали на вредоносную деятельность, направленную против хостов локальной сети? С помощью фильтров TCPdump можно найти IP-дейтаграммы с определенным значением какого-либо поля, т.е. провести выборку из общего объема записей. Фильтрация может осуществляться по значениям полей в IP-заголовке (например, по длине IP-заголовка), в TCP-заголовке (например, по наличию флагов TCP), в UDP-заголовке (например, по значению порта получателя) или ICMP-заголовке (например, по типу ICMP-сообщения).

В программе TCPdump предопределены несколько макросов для создания фильтров по значениям наиболее важных полей, например, макрос "port" для фильтрации по номеру порта отправителя или получателя, или макрос "host", позволяющий выбирать записи о пакетах с определенным IP-адресом или именем отправителя или получателя. В наших примерах мы не будем использовать эти макросы не потому, что мы пренебрегаем ими, а потому, что для интересующих нас полей макросы не предопределены. Поэтому при создании фильтра, проверяющего пакеты по значению определенного поля, нам придется указывать, в пакете какого протокола это поле используется, и где оно располагается (смещение в байтах).

В TCPdump заголовкам разных пакетов соответствуют различные имена. Например, имя "ip" используется для указания поля в заголовке IP-дейтаграммы или фрагмента данных в самой IP-дейтаграмме, имя "tcp" – для поля в TCP-заголовке или фрагмента данных в TCP-сегменте, имя "udp" – для поля в UDP-заголовке или фрагмента данных в UDP-пакете и имя "icmp" для поля в ICMP-заголовке или фрагмента данных в ICMP-сообщении.

Затем необходимо создать ссылку на расположение поля в пакете данного протокола, указав смещение в байтах от начала пакета. Например, запись `ip[0]` указывает на нулевое смещение в IP-дейтаграмме, конкретно – на смещение в IP-заголовке (не забывайте, что счет начинается с нуля). Запись `tcp[13]` задает смещение 13 байт от начала TCP-сегмента, что тоже указывает на часть TCP-заголовка. Запись `icmp[0]` обозначает первый байт ICMP-сообщения, т.е. тип ICMP-сообщения.

Таким образом, для создания фильтра TCPdump используется следующий формат.

```
<заголовок_протокола>[смещение:длина] <отношение> <значение>
```

Для всех фильтров, рассматриваемых в этой главе, эталоном является стандартный формат IP-заголовка (рис. 12.1). Обратите внимание на то, что каждая строка содержит 32 бит (с 0 до 31), что равно 4 байт. И не забывайте начинать счет с нуля!

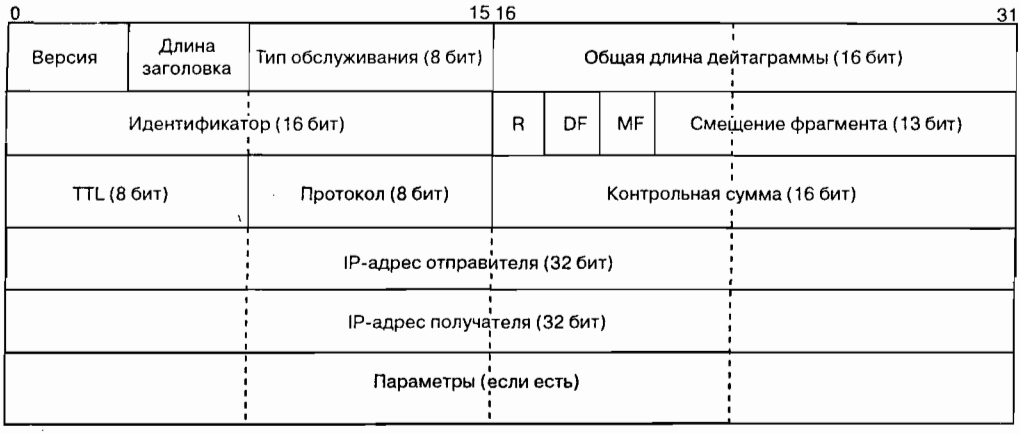


Рис. 12.1. Заголовок IP-дейтаграммы

Предположим, что с помощью TCPdump необходимо выбрать все дейтаграммы с вложенными ICMP-сообщениями. Поле "Протокол" имеет смещение 9 байт (последний раз напоминаем, что счет начинается с 0) в IP-заголовке (см. рис. 12.1). Следовательно, его можно обозначить как `ip[9]`. Обратите внимание на то, что в формате создания фильтра указана форма записи смещение : длина. Однако по умолчанию длина считается равной 1 байт, и указывать ее нужно, только если значение отличается. Определив место расположения 1-байтового поля, необходимо указать на хранящееся в нем значение 1, которое соответствует протоколу ICMP. Полный фильтр для выявления записей об ICMP-сообщениях выглядит как `ip[9] = 1`. Для сбора соответствующих записей в сети следует использовать следующую команду.

```
tcpdump 'ip[9] = 1'
```

Эта команда позволяет организовать чтение информации с заданного по умолчанию сетевого интерфейса и выбирать только записи о передаче ICMP-сообщений. Фильтр записан в одиночных кавычках, чтобы командный интерпретатор UNIX интерпретировал его однозначно. С помощью параметра командной строки `-F` можно задать расположение файла, в котором хранится нужный фильтр. Если сохранить фильтр `ip[9] = 1` в файле `/tmp/filter`, то для его использования потребуется следующая команда.

```
tcpdump -F /tmp/filter
```

Результат будет аналогичен запуску команды `tcpdump` с указанием фильтра в командной строке. Параметр `-F` обычно используется для доступа к файлам с длинными фильтрами или для автоматизированных процессов TCPdump с целью устранить необходимость ввода фильтра из командной строки.

Побитовое маскирование

Теперь необходимо пояснить еще несколько аспектов языка фильтров TCPdump. Этот язык не является сложным по сравнению с конструкциями и операциями, доступными в других языках, например C или Perl. Однако часто для

описания полей, которые выходят за рамки целых байтов, приходится возвращаться к основам языка ассемблера.

Использование TCPdump не вызывает никаких затруднений, когда для хранения значения поля используется целое число байтов, и необходимо исследовать все 8 бит этих байтов. Для объединения нескольких байтов после значения смещения можно указать нужную длину в байтах, но что делать, если нужно исследовать только некоторые биты или диапазон битов? Другими словами, что делать, если не нужно проверять значение всего байта? Эта задача несколько сложнее и требует элементарных навыков работы с двоичными и шестнадцатеричными данными.

Выбор и отбрасывание отдельных битов

Рассмотрим еще раз структуру IP-заголовка, и в частности его первый байт, который состоит из двух 4-битовых полей. Каждое из этих полей является полубайтом (nibble). Как быть, если нужно исследовать только значение поля длины IP-заголовка, не проверяя значения поля "Версия"? Нужно выбрать только значение младшего полубайта, отбросив значение старшего. По существу, четыре бита старшего полубайта должны считаться равными 0. Такой метод позволит при исследовании первого байта IP-заголовка проверять значения только младшего полубайта. Сейчас мы рассмотрим все это подробнее.

Вспомним логические (булевы) арифметические операции. Наверняка, у некоторых читателей вырвался невольный стон. Лично я еще не встречал человека, которому бы нравились таблицы истинности, но, к сожалению, ничего не поделаешь и придется воскресить в памяти действия с логическим оператором И (табл. 12.1).

Таблица 12.1. Таблица истинности для оператора И

Бит А	И	Бит В	Результат
0		0	0
1		0	0
0		1	0
1		1	1

В этой таблице представлены все возможные значения двух битов и результаты их логического умножения (оператор И). Результат будет равен 1, только когда оба бита равны 1. Какое это имеет отношение к фильтрам TCPdump? Вспомните: нам ведь нужно, чтобы все биты старшего полубайта первого байта IP-заголовка хранили значение 0. Необходимо выполнить логическое умножение битов старшего полубайта на 0 (и этот полубайт будет отброшен), а битов младшего полубайта на 1 (и значения битов этого полубайта останутся без изменений).

Посмотрим, как это сделать. На рис. 12.2 в двух прямоугольниках (по 4 бит) указаны значения битов реального IP-заголовка. Исследуем значение дейтаграммы. Значение старшего полубайта равно 0100, где 1 соответствует 2^2, что в результате дает 4. Это стандартное значение версии протокола IP. Значение младшего полубайта определяется как $2^2 + 2^0 = 5$. Это длина IP-заголовка. Но это значение представлено в виде 32-битового слова, и для его преобразования в байты его следует умножить на 4. Таким образом, длина этого заголовка равна 20 байт, что является стандартным значением для заголовка без параметров.

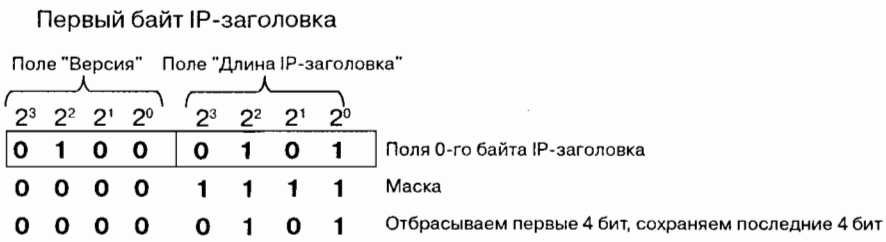


Рис. 12.2. Побитовое маскирование

Создание маски

Вернемся к задаче отбрасывания четырех битов старшего полубайта. На рис. 12.2 строка, указанная ниже значений битов действительной IP-дейтаграмммы, представляет собой *маску*. Маска – это байт, биты которого будут использованы при операции логического умножения с соответствующими битами нулевого байта IP-заголовка. При этом мы добиваемся "отбрасывания" значений старшего полубайта и сохранения значений младшего полубайта. Начнем умножение слева направо. Самый старший бит нулевого байта IP-заголовка хранит значение 0, соответствующий бит маски тоже равен 0. Естественно, что результат логического умножения двух битов тоже будет равен 0. Если при логическом умножении хотя бы одно из значений равно 0, то результат всегда будет равен 0. Согласно этому принципу все биты старшего полубайта при умножении на биты маски получат значение 0 – `0000`.

Поскольку нельзя создать маску только для части байта, нужно маскировать и младший полубайт, сохранив при этом его значение. Снова начнем со старшего бита (крайнего слева), значение которого 0, а значение соответствующего бита маски – 1. Операция И для значений двух битов дает результат 0, то есть оригинальное значение сохраняется. Аналогично умножаем следующую 1 на 1. Таким образом, маска, состоящая из одних 1, позволяет сохранить значения битов младшего полубайта. Таким образом наша цель достигнута – сохранено только значение поля длины IP-дейтаграммы. Нам пришлось применить маску, так как часть байта анализировать нельзя. Прежде чем приступать к изучению методов создания самих фильтров, рассмотрим еще одну тему. Как заставить TCPdump выполнить операцию И, а также как задать маску?

Прежде всего нужно записать значение маски в виде двух шестнадцатеричных символов. Например, `0000 1111` соответствует `0x0f`. Приставка `0x` указывает TCPdump, что это значение шестнадцатеричное (по умолчанию TCPdump работает в десятичной системе счисления). Теперь запишем созданную часть фильтра.

```
ip[0] & 0x0f
```

На данный момент этот фильтр указывает на необходимость выполнить логическую операцию И для значений нулевого байта IP-заголовка и шестнадцатеричного числа 0f.

Выводы

Рассмотрев нулевой байт IP-заголовка, мы выполнили логическое умножение значения этого байта со значением 0x0f, что позволило сохранить только значе-

ние поля длины IP-заголовка. Зачем выделять значение этого поля? Например, для проверки наличия параметров IP-дейтагараммы. Стандартный размер IP-дейтаграммы равен 20 байт или пяти 32-битовым словам. Это означает, что IP-заголовок с такими потенциально опасными параметрами, как маршрутизация от источника, будет иметь длину больше значения 5, указанного в этом поле. Параметры IP-дейтаграмм в основном используются только хакерами с вредоносными целями, поэтому их установку желательно контролировать. Согласно синтаксису написания фильтров TCPdump необходимо указывать оператор отношения и значение. Итог создания фильтра, который задает сигнатуру IP-дейтаграммы без установленных параметров, будет выглядеть следующим образом.

```
ip[0] & 0x0f > 5
```

Вот и все. Этот материал может показаться довольно сложным, требующим фундаментальных теоретических знаний, но на самом деле все станет намного проще после недолгих практических занятий. Конечно, придется внимательно изучить все аспекты создания фильтров, но в результате вы сможете исследовать любое поле и даже отдельные биты IP-дейтаграмм. Не все системы обнаружения вторжений обладают такими возможностями. TCPdump позволяет анализировать и сохранять нужные данные с максимальной точностью. Кроме того, не многие владельцы систем обнаружения вторжений могут этим похвастаться. Именно поэтому следует ближе познакомится с программой TCPdump и ее фильтрами.

IP-фильтры программы TCPdump

Стандартным признаком сканирования сетей является появление широковещательного или фрагментированного трафика, а также наличие установленных параметров IP-дейтаграмм. Входящий широковещательный трафик никогда не является легитимным и его следует блокировать, чтобы предотвратить вышеописанные попытки составления схемы сети и атаки Smurf. Как известно, фрагментация является естественным способом доставки пакетов, MTU которых больше, чем MTU одного из участков маршрута передачи сообщения. Но фрагментация также может применяться для атак отказа в обслуживании или для обмана систем обнаружения вторжений или маршрутизаторов, не поддерживающих хранение содержимого пакетов.

Выявление широковещательного трафика

Будем считать широковещательным адрес, последний байт которого содержит значение 0 или 255. Адрес получателя занимает диапазон с 16-го по 19-й байты (32 бит) IP-заголовка (см. рис. 12.1). Нас интересует только последний, 19-й, байт. Описать широковещательные адреса можно следующим образом:

```
ip[19] = 0xff
ip[19] = 0x00
```

Или как один скомбинированный фильтр:

```
ip[19] = 0xff or ip[19] = 0x00
```

Авторы этой книги привыкли записывать свои фильтры шестнадцатеричными символами, но, если кому-то больше нравится десятичная форма, то этот фильтр можно легко изменить.

```
ip[19] = 255 or ip[19] = 0
```

В зависимости от месторасположения хоста, на котором запущена программа TCPdump с этим фильтром, можно организовать перехват всего широковещательного трафика в своей сети. Предположим, например, что адрес нашей сети `192.168.x.x`. Настроим фильтр на регистрацию широковещательного трафика, отправляемого с удаленного хоста только для этой конкретной сети.

```
not src net 192.168. and (ip[19] = 0xff or ip[19] = 0x00)
```

Оператор `not` обозначает отрицание, а макросы `src` и `net` используются для указания на трафик, исходящий от указанного далее отправителя, и для обозначения сети соответственно. Таким образом, данный фильтр позволяет зарегистрировать весь широковещательный трафик, исходящий из любого внешнего источника, кроме нашей подсети. Если запустить TCPdump, задав данный фильтр (или проверить собранные ранее данные), то будет выдан отчет о всех попытках сканирования локальной сети.

Выявление фрагментированного трафика

В этом разделе рассказано о создании фильтров TCPdump для поиска фрагментированного трафика. Во всех фрагментах стандартной последовательности пакетов, кроме последнего, установлен флаг MF. По этому признаку можно выявить большинство проходящих фрагментов. Еще раз обратимся к рис. 12.1. Бит MF находится во второй строке IP-заголовка. Какой это байт?

Результат подсчета позволяет сказать, что это шестой байт IP-заголовка, а нужный нам бит является третьим справа от самого старшего бита этого байта. Посмотрим на рис. 12.3, чтобы понять, какая маска позволит скрыть все биты этого байта, кроме бита MF. Потребуется маска `0010 0000`, которая в шестнадцатеричном формате выглядит как `0x02`. Фильтр принимает вид `ip[6] & 0x20 != 0`. Используется оператор "не равно" и значение 0. Это означает, что проверяется установка флага MF. А почему просто не задать фильтр `ip[6] & 0x20 = 1`. Разве не так проверяется установка этого бита? Не совсем. Дело в том, что проверяется не само значение конкретного бита, а результат маскирования всего байта (логического умножения исходного байта и байта маски). Установленный бит MF занимает в байте позицию, соответствующую значению 2^5=32. Поэтому проверить результат с помощью выражения `!= 0` несколько проще. Альтернативным вариантом может послужить фильтр `ip[6] & 0x20 = 32`. Не забывайте, что фрагментация является нормальным методом доставки пакетов, и такой фильтр может послужить причиной появления многочисленных сообщений о ложных атаках.

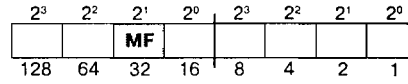


Рис. 12.3. Обнаружение установленного бита MF

Были представлены примеры создания трех фильтров TCPdump для обнаружения потенциально опасного трафика. Теперь рассмотрим пакеты других протоколов и способы создания фильтров TCPdump для этих пакетов.

UDP-фильтры программы TCPdump

Многие программы атаки и троянские программы используют UDP-порты, например, программа Back Orifice использует UDP-порт 31337. Поэтому нужно определить UDP-порты, попытки соединения с которыми будут отслеживаться. Принять решение поможет информация сайта `www.snort.org/ports.html`. Создайте фильтры, отслеживающие запросы к перечисленным на этом сайте портам. Например, следующий фильтр позволяет выявлять пакеты программы Back Orifice.

```
udp and dst port 31337
```

Как видите, создать подобный фильтр очень просто. Сложнее принять решение, какие порты следует контролировать, добавить их в фильтр и постоянно обновлять его при появлении новых программ атаки, использующих UDP-порты.

Рассмотрим популярное UDP-приложение – программу Traceroute. При работе этой программы UDP-пакеты отправляются на привилегированные порты хоста-получателя. Если вы не хотите, чтобы этим хостом оказался хост вашей сети, следует создать фильтр, который будет отслеживать попытки установить UDP-соединение с портами в диапазоне 33000–33999. Этот фильтр позволит выявить большинство действий Traceroute. Однако Windows-программа Traceroute использует эхо-запросы и эхо-ответы ICMP, которые нельзя обнаружить с помощью этого фильтра. Кроме того, для некоторых UNIX-версий Traceroute с помощью параметра командной строки можно задавать номер порта получателя.

Напомним формат заголовка UDP-пакета (рис. 12.4). Обратите внимание на то, что номер порта получателя сохраняется в байтах 2 и 3 UDP-заголовка.

Рис. 12.4. Заголовок UDP-пакета

Резонно возникает вопрос: "А зачем считать смещение поля в байте, если можно просто воспользоваться макросом "port"?". Например, почему бы не применить следующий фильтр.

```
dst port >= 33000 and dst port < 34000
```

Дело в том, что в TCPdump при использовании подобных диапазонов, а не точных значений, необходимо определять значение и смещение этого поля по отношению к пакету базового протокола и отказаться от использования макросов. Правильный фильтр для выявления действий Traceroute должен выглядеть так.

```
udp[2:2] >= 33000 and udp[2:2] < 34000
```

Обратите внимание на первый пример использования параметра длины `[2:2]` для проверки сразу нескольких байтов. Задан анализ двух последовательных байтов начиная со второго (по номеру) байта UDP-пакета. Объем фиксируемого TCPdump трафика можно еще больше ограничить, добавив к этому фильтру

проверку значения поля TTL. Программа Traceroute управляет значением поля TTL IP-заголовка отправляемых пакетов. Последовательно увеличивая значения этого поля на 1 в отправляемых пакетах, она составляет список адресов промежуточных маршрутизаторов на пути к получателю. Поэтому чаще всего в полученном от Traceroute пакете значение поля TTL будет равно 1. Таким образом, фильтр для выявления действий Traceroute можно улучшить, организовав дополнительный поиск по значению поля TTL. Поле TTL находится в 8-м байте (см. рис. 12.1) IP-заголовка, и макроса для него не существует. Новый фильтр принимает следующую форму.

```
udp[2:2] >= 33000 and udp[2:2] < 34000 and ip[8] = 1
```

Вы получили представление о создании UDP-фильтров. Фильтры TCPdump можно использовать и для ICMP-сообщений. В частности, полезно отслеживать поступление ICMP-запросов на получение масок подсети, попытки узнать MTU вашей сети (с помощью отправления пакетов с установленным флагом DF и ожидания ответа от вашего маршрутизатора), а также пакетов программы Loki. Все эти фильтры создать несложно. Попробуйте написать их самостоятельно. Для этого воспользуйтесь перечисленными ниже сигнатурами.

- В первом байте (смещение 0) ICMP-запроса на адрес маски подсети содержится значение 17.
- В ICMP-сообщении с установленным флагом DF в первом байте (смещение 0) содержится значение 3, а во втором (смещение 1) — значение 4.
- Сигнатурой для Loki можно считать эхо-запрос (значение 8 в первом байте ICMP-сообщения) или эхо-ответ (значение 0 в первом байте ICMP-сообщения). Для двух байтов со смещением 6 и 7 можно контролировать шестнадцатеричные значения 0xf001 и 0x01f0.

Проверьте, как вы справились с заданием, с помощью приведенных ниже фильтров.

```
icmp[0] = 17

((icmp[0] = 3) and (icmp[1] = 4)

(((icmp[0] = 0) or (icmp[0] = 8)) and
↳ ((icmp[6:2] = 0xf001) or (icmp[6:2] = 0x01f0)))
```

TCP-фильтры программы TCPdump

С помощью фильтров TCPdump для TCP-трафика в первую очередь отслеживаются попытки установить TCP-соединения с помощью SYN-пакетов и поступление пакетов с нестандартными наборами флагов, которые могут использоваться для разведывательных целей или составления схемы сети. Контролировать поступающие SYN-пакеты необходимо потому, что они указывают на попытку установить соединение с TCP-портом. Пусть даже эта попытка была неудачной. Например, если регистрация трафика выполняется до поступления пакетов на устройство фильтрации, которое блокирует доступ к TCP-портам, то такие пакеты никогда не достигнут адресата. Кроме того, на самом хосте-получателе может не поддерживаться работа запрошенной службы. Но, фиксируя необычный трафик,

вы получаете бесценный опыт обнаружения вредоносной активности, особенно той, что предназначена для открытых портов хостов вашей сети.

Фильтры для исследования TCP-флагов

Информация рис. 12.5 поможет при изучении большинства фильтров, описанных в конце этой главы.

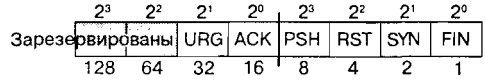


Рис. 12.5. Байт TCP-флагов

Байт TCP-флагов является 13-м байтом TCP-заголовка. Для обнаружения конкретных флагов, установленных в этом байте, придется выполнить маскировки. Начнем с создания фильтра для обнаружения пакетов с единственным установленным флагом SYN.

```
tcp[13] & 0xff = 2
```

Почему фильтр выглядит именно так? Очевидно, что маска состоит из одних единиц. А почему не применить маску, которая содержит нули во всех полях, кроме поля для флага SYN (`tcp[13] & 0x02 = 2`)? Если маскировать значение бита с помощью 0, то результатом всегда будет 0. Значением исходного бита может быть 1, и маска отбросит это значение. Поясним это на примере.

Предположим, нужно отобрать из общего трафика TCP-сегменты, в которых установлен только флаг SYN. Теперь представим, что в байте TCP-флагов установлены флаги ACK и SYN. Двоичным значением такого байта будет 0001 0010. Если это значение маскировать с помощью маски 0000 0010, то конечным результатом будет 0000 0010 или 2. Таким образом, маскирование с помощью нулей всех полей байта TCP-флагов, кроме поля флага SYN, приводит к выбору TCPdump "лишних" TCP-сегментов, а не только искомых пакетов с установленным флагом SYN. Поэтому и необходимо использовать приведенный выше фильтр и сохранять все исходные установленные флаги. Если полученный результат будет отличаться от 2, то это означает, что установлены дополнительные (кроме SYN) флаги. Например, если будет установлен бит ACK, то результатом маскирования будет значение 18.

Поскольку нам необходимо отобрать только те сегменты, в которых установлен один флаг SYN, то наиболее простым фильтром для этой цели может послужить следующий.

```
tcp[13] = 2
```

Этот фильтр позволяет гарантировать, что будут выбираться только TCP-сегменты с установленным флагом SYN, потому что, если будет установлен любой другой флаг, то результат сложения всех битов байта TCP-флагов будет отличаться от 2. Например, представим, что в байте TCP-флагов установлен нестандартный набор флагов: SYN и URG. Если установлен флаг URG, то его позиция соответствует значению 32, флаг SYN дает значение 2. Таким образом, результатом сложения двух битов будет 34, что не соответствует фильтру.

Рассмотрим ряд других фильтров для других возможных комбинаций TCP-флагов.

- `tcp[13] & 0xff = 0` или `tcp[13] = 0`. Позволяет выявить попытки сканирования с помощью null-пакетов (не установлено никаких флагов). Такие пакеты принимать не следует.
- `tcp[13] & 0xff = 3` или `tcp[13] = 3`. Позволяет выявить TCP-сегменты, в которых одновременно установлены флаги SYN и FIN (это явно нестандартная ситуация). Можно заменить указанный фильтр фильтром `tcp[13] & 0x03 = 3`, который позволит обнаружить любые пакеты с двумя установленными флагами SYN и FIN, в том числе совместно с другими флагами.
- `tcp[13] & 0xff = 0x10 and tcp[8:4] = 0`. Позволяет выявить сегменты с установленным флагом ACK, и со значением номера подтверждения, равным 0. Это подозрительная ситуация, так как после проведения полной процедуры установки соединения номер подтверждения должен быть больше начального порядкового номера. Этот фильтр часто позволяет обнаружить попытки сканирования программы nmap для выявления операционных систем удаленных хостов. Программа nmap отправляет TCP-сегменты на различные порты исследуемого хоста с одним установленным флагом ACK и значением 0 в поле номера подтверждения.

  Очень редко значение 0 для номера подтверждения является нормальным. Это происходит, если отправитель передает пакет, в котором указан порядковый номер, состоящий из одних единиц (т.е. значение будет равно $2^{32} - 1$). Тогда для следующего ожидаемого порядкового номера должно быть выполнено обнуление значения.
- `tcp[13] >= 64`. Два старших бита байта TCP-флагов считаются зарезервированными (см. рис. 12.5). Значения этих двух битов должны равняться 0. Если это не так, то что-то неправильно. Первому биту соответствует значение 2^6 (64), а второму – 2^7 (128). Если один или оба бита установлены, то значение байта TCP-флагов будет равняться или превышать 64. Наша старая знакомая, программа nmap, часто устанавливает бит в позиции значения 64 при сканировании операционных систем. Большинство хостов сбрасывают эти установленные значения в 0, но некоторые операционные системы оставляют этот бит без изменений.

  Не так давно эти старшие биты байта TCP-флагов стали использоваться для уведомления о перегрузке (ECN) с целью улучшить работу в сети. Как можно отличить нормальный трафик с ECN-битами от попыток сканирования nmap? При использовании в TCP-сегменте ECN-битов для уведомления о перегрузке поле TOS ("Тип обслуживания") должно хранить ненулевое значение, а программа nmap оставляет значение этого поля равным 0. Более подробную информацию о битах ECN можно получить в документе RFC 3168.

Мы рассмотрели только некоторые из возможных комбинаций TCP-флагов, наличие которых можно проверять в TCP-сегментах.

Выявление данных в SYN-пакетах

Перед тем как вы начнете самостоятельно создавать фильтры TCPdump, рассмотрим пример еще одного расширенного фильтра, использующего уже описанные в этой главе значения битов и полей, а также некоторые другие. В главе 2, "TCPdump и TCP", мы рассказали о том, что данные не должны отправляться до завершения полной процедуры согласования параметров TCP-соединения. Такие действия выполняет программа 3DNS, которая причиняет неудобства, но не является вредоносной. Кроме того, мы описали сканирование, которое выполнялось с помощью отправки данных в начальном SYN-пакете. Это может помешать системе обнаружить взлом, поскольку она начинает собирать поток данных только после завершения процедуры установки соединения.

Поэтому стоит создать фильтр TCPdump, который бы позволял обнаруживать передачу данных в SYN-пакете (конечно, при этом вероятны ложные тревоги, вызванные работой программы 3DNS). Проблема в том, что ни в одном поле TCP-заголовка не хранится значение длины полезных данных TCP-сегмента. Но есть много других полей, хранящих значения других длин. В частности, в IP-заголовке существуют поля длины всей IP-дейтаграммы и поле длины одного IP-заголовка. В TCP-заголовке присутствует поле, в котором указана длина этого заголовка. Таким образом, если от длины всей IP-дейтагарраммы отнять значения длин IP- и TCP-заголовков, мы получим искомое значение полезных данных TCP-сегмента (рис. 12.6).

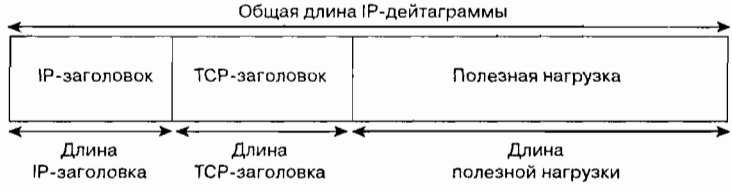


Рис. 12.6. Вычисление длины полезной нагрузки TCP-сегмента

Вы скажете "совсем просто"? Но следует принять во внимание ряд обстоятельств. Не забывайте, что значения разных полей указаны в различных единицах измерения: длина IP-дейтаграммы указана в байтах, а значения длин IP-заголовка и TCP-заголовка являются 32-битовыми словами. Для преобразования двух последних значений в байты их необходимо умножить на 4. Как видите, задача вполне решаемая.

Еще одной небольшой неприятностью является необходимость обращаться к значению поля длины TCP-заголовка (рис. 12.7). Это поле занимает старший полубайт 12-го байта TCP-заголовка. Для исследования значения только старшего полубайта младший полубайт придется маскировать нулями, но и это еще не все. Так как рассматривается старший полубайт, то его значение, умноженное на 16, должно быть нормализовано.

Предположим, что длина TCP-заголовка составляет 24 байт, включая 20 байт самого заголовка и несколько TCP-параметров. Не забывайте, что значение указано в виде 32-битовых слов, поэтому в поле длины TCP-заголовка в этом случае будет содержаться значение 6 (нужно выполнить деление на 4). Предположим, младший полубайт уже замаскирован, и в результате шестнадцатеричное значе-

ние этого байта равно 60. В двоичной форме он будет выглядеть как 0110 0000. Единицы установлены в позициях 2^6 (64) и 2^5 (32), т.е. десятичное значение составляет 96. Поскольку данное поле представляет собой старший полубайт, то его значение в 16 раз больше значения младшего полубайта, и его необходимо нормализовать, поделив на 16, а затем для получения значения в байтах умножить на 4. Только теперь можем создать наш фильтр.

Рис. 12.7. Заголовок TCP-сегмента

Перед написанием самого фильтра давайте вспомним необходимые условия и нужную формулу.

Если установлен только один флаг SYN, вычтем из общей длины IP-дейтаграммы переведенную в байты длину IP-заголовка и нормализованную и переведенную в байты длину TCP-заголовка и убедимся, что полученное значение отличается от нуля.

Единственный установленный флаг SYN.

```
tcp[13] & 0xff = 2  или  tcp[13] = 2
```

Общая длина IP-дейтаграммы.

```
ip[2:2]
```

Значение длины IP-заголовка, преобразованное в байты.

```
((ip[0] & 0x0f)*4)
```

Нормализованное и переведенное в байты значение длины TCP-заголовка:

```
((tcp[12] & 0xf0)/16*4)
```

Это аналогично:

```
((tcp[12] & 0xf0)/ 4)
```

Теперь объединим все части в один фильтр.

```
tcp[13] & 0xff = 2
and
```

```
(  ip[2:2] -
   ((ip[0] & 0x0f)*4) -
   ((tcp[12] & 0xf0)/4)
) != 0
```

Этот фильтр позволяет выявить любые попытки передачи данных в SYN-пакетах.

Резюме

В этой главе было показано, что хотя фильтры TCPdump и не в состоянии сравниться с фильтрами систем обнаружения вторжений, но тоже пригодны для решения многих важных задач. Конечно, при их создании придется немного потрудиться, и не всегда удается легко устранить проблему, если фильтр не работает, но такие фильтры предоставляют полный доступ к данным. Если возникают какие-либо подозрения при получении пакетов, то необходимо иметь средство, выполняющее анализ этих пакетов на уровне битов. Фильтры TCPdump вполне подойдут для этой цели. При проведении подробного исследования они позволят проверить буквально каждый бит.

В большинстве создаваемых фильтров TCPdump используются макросы, и не требуется выполнять операцию побитового маскирования. Но, если нужно проверить отдельные биты байта, их значения нужно выделить с помощью масок. Мы также рассказали о еще одном важном моменте при создании фильтров: не забывайте переводить значения различных полей в одни единицы измерения, а точнее, в байты. Кроме того, важно учитывать положение битов в байте (для значений старшего полубайта, возможно, потребуется нормализация). Овладев техникой создания фильтров и получив возможность всесторонне исследовать поступающие пакеты, вы испытаете чувство глубокого удовлетворения.

13

Система Snort

Snort – это бесплатная система обнаружения вторжений с открытым исходным кодом, разработанная Мартином Рошем (Martin Roesch). Первоначально Мартин создал Snort для перехвата трафика при своей повседневной работе, а затем она была улучшена до многофункциональной сетевой системы обнаружения вторжений. Мартину удалось завоевать симпатии многих программистов, коллективные усилия которых позволили усовершенствовать программный код программы и выпустить ее новые версии. В начале 2002 года за одну неделю программа Snort была загружена более 10000 раз для защиты сетей правительственных организаций и учебных учреждений, корпоративных сетей и компьютеров домашних пользователей.

Принцип работы системы Snort основывается на сигнатурах и использовании комбинаций правил и препроцессоров для анализа трафика. Правила предоставляют простые и гибкие средства создания сигнатур для исследования отдельных пакетов[2]. Программа препроцессора позволяет выполнить более глубокое исследование и обработку данных, которые нельзя осуществить с помощью одних только правил. Препроцессоры позволяют выполнять большое число задач, например дефрагментацию IP-дейтаграмм, обнаружение сканирования портов, сборку потока TCP-данных в пункте получения и др. Препроцессоры предоставляют Snort возможность просмотра и обработки потоков данных, в отличие от правил, которые предназначены для анализа одиночных пакетов.

В марте 2002 года (на момент выхода английского издания этой книги) последней версией Snort была 1.8.3, которая в сжатом виде занимала 1,8 Мбайт. Она полностью переносима и может работать приблизительно на 23 различных платформах, включая Linux, Solaris, BSD, IRIX, HP-UX, Mac OS X и Win 32. Система Snort легко настраивается и позволяет пользователям создавать собственные сигнатуры и изменять исходные функциональные возможности с помощью

---

[2] Правило – это инструкция, определяющая как следует поступить с конкретным пакетом (разрешить или запретить его прохождение через сетевой интерфейс) в зависимости от содержимого и полезных данных этого пакета.

подключаемых модулей. Подключаемый модуль – это программный код, который по желанию может быть скомпилирован совместно с программой Snort во время установки и позволяет расширить возможности базовой программы, например организовать ответные действия против отправителя вредоносного трафика.

В этой части книги основное внимание будет уделено созданию фильтров и сигнатур, поэтому многие другие аспекты работы с системой Snort останутся не рассмотренными, например установка, настройка и получение результатов работы системы. Более подробную информацию по этим темам можно получить по адресу `www.snort.org`. В этой главе мы познакомимся со Snort, изучим ее правила, а также проведем исследование полей и их возможных значений, содержащихся в первой части правила Snort, которую называют *заголовком правила*. В следующей главе мы продолжим анализ правил, рассмотрим вторую часть правила Snort, называемую *параметрами правила*, а также сформулируем несколько более сложных правил.

Режимы запуска Snort

Система Snort может быть запущена в различных режимах: начиная от простого выведения отчета о перехваченном трафике на экран монитора и заканчивая режимом сетевой системы обнаружения вторжений, в котором Snort способна сравнивать пакеты сетевого трафика с заданными сигнатурами (правилами), хранящимися в одном или нескольких файлах. Последний режим используется для Snort чаще всего.

Запуск Snort обычно выполняется из командной строки, независимо от того, на каком хосте (Windows или UNIX) она запущена. Специальная программа ID-Scenter предоставляет графический пользовательский интерфейс для пользователей Windows-хостов, а Demarc/Puresecure позволяет работать в графическом пользовательском интерфейсе как на Windows-, так и на UNIX-системах. Существует множество параметров командной строки, которые можно использовать при запуске Snort, но самым востребованным является параметр (`-c snort.conf`), применяемый для запуска Snort в режиме сетевой системы обнаружения вторжений (NIDS-режим). При этом происходит обращение к файлу конфигурации. Информация этого файла позволяет настроить работу Snort, в том числе присвоить значения для переменных в правилах, указать Snort на необходимость использования тех или иных препроцессоров и правил для анализа трафика. Исходный конфигурационный файл `snort.conf` сохраняется в каталоге установки Snort. Пользователь должен отредактировать этот файл в соответствии с задачами своего узла.

Если система Snort запущена в NIDS-режиме, по умолчанию отчеты о событиях, которые соответствуют заданным сигнатурам, сохраняются в нескольких файлах. Snort позволяет для каждого правила задать *действие*, выполняемое системой при обнаружении пакета, который соответствует этому правилу. *Предупреждение* (alert) определяет запись пакета в файл `alert`, который по умолчанию создается в каталоге `/var/log/snort` многих UNIX-систем. На Windows-хостах файл `alert` создается в подкаталоге `log` того каталога, из которого запускается программа Snort. Рассмотрим примеры записей в файле `alert` системы Snort.

```
[**] NMAP TCP ping [**]
03/21 - 13:33.51:880120 1.2.3.4:1029 -> 192.168.5.5:80

TCP TTL:46 TOS:0x0 ID:19678
******A* Seq: 0xE4F00003 Ack: 0x0  Win: 0xC00
```

Можно сформулировать сообщение, которое будет выдаваться на экран при появлении предупреждения. Это уведомление не является обязательным, но позволяет проинформировать аналитика о возникшей проблеме. Для приведенного выше предупреждения использовалось сообщение "NMAP TCP ping". В следующей строке указаны дата и время события, IP-адрес (`1.2.3.4`) и порт (`1029`) отправителя, направление трафика (отправитель слева от "стрелки", а получатель — справа), IP-адрес (`192.168.5.5`) и порт (`80`) получателя, указанные в опасном пакете. Третья строка содержит тип вложенного пакета (TCP-сегмент), значения полей TTL (`46`), TOS (`0`), а также идентификатор (`19678`). В последней строке перечислены установленные TCP-флаги (символ A обозначает установленный флаг ACK), затем следует значение порядкового номера в шестнадцатеричном формате и размер окна. Вся эта информация позволяет больше узнать о пакете, вызвавшем предупреждение.

Предупреждение регистрируется в результате "срабатывания" правила при получении TCP-сегмента с установленным флагом ACK и значением номера подтверждения, равным 0. В большинстве случаев подобный пакет является характерным признаком попытки сканирования `nmap` с целью выявления активного хоста. Если пакет с установленным флагом ACK достигнет хоста-получателя, то он должен ответить на незапрошенное подтверждение пакетом с установленным флагом RESET, независимо от того, открыт запрошенный порт или нет. Поэтому предупреждению сопутствует сообщение "NMAP TCP ping".

Пакет, вызвавший предупреждение, обязательно регистрируется. Это отдельное действие — запись о случившемся событии. Запись выполняется в формате, удобном для чтения человеком, и предоставляет более подробную информацию об опасном пакете, например, о его полезной нагрузке. Записи сохраняются в файлах и каталогах, имена которым даются в соответствии с указанным в пакете IP-адресом. Записи дополнительно разделяются по различным типам использованных протоколов транспортного уровня, портам отправителя и получателя. Рассмотрим запись о зарегистрированном пакете службы FTP.

```
[**] Attempted anonymous ftp access [**]
04/24-12:11:08.724441 192.168.143.15:3484 -> 192.168.143.16:21
TCP TTL:64 TOS:0x10 ID:30124  DF
*****PA* Seq: 0x93EE0AB7   Ack 0xB8352E61   Win: 0x7D78
TCP Options => NOP NOP TS: 112024246 27551686
55 53 45 52 20 61 6E 6F 6E 79 6D 6F 75 73 0D 0A USER anonymous..
```

Сохраненный отчет содержит ту же информацию, что выдается при предупреждении, а также предоставляет сведения о полезной нагрузке при использовании в командной строке параметра `-d`. Это сообщение указывает на присутствие правила, которое определяет фильтрацию анонимного FTP-трафика к порту 21. Более подробный анализ подобных сообщений содержится в главе 14, "Параметры правил Snort", но по полезной нагрузке зарегистрированного пакета сразу можно сказать, что была предпринята попытка доступа к анонимному FTP-

серверу. В запись о зарегистрированном пакете также добавляются шестнадцатеричные соответствия обнаруженных в полезных данных ASCII-символов.

Создание файлов регистрации и сообщений о зарегистрированных предупреждениях может показаться довольно сложным методом анализа отчетов Snort, но существуют и другие возможности проведения этого анализа, реализуемые с помощью внесения изменений в конфигурационный файл. Например, можно организовать сохранение отчетов о зарегистрированных событиях в буферных файлах с помощью буфера базы данных, который называют Barnyard, или непосредственно в базе данных.

Правила Snort

Система Snort поддерживает методы исследования и заголовка, и полезных данных пакетов, что позволяет полностью определить все контролируемые параметры пакетов в едином правиле. Такая гибкость помогает создавать специализированные правила для фильтрации трафика на конкретном узле и максимально снижать количество ложных тревог. При этом формат правил делает их очень простыми для чтения. Помните все трудности и проблемы создания фильтров TCPdump, в частности поиск пакетов только с определенным установленным флагом? Создание такого правила для системы Snort не составляет никакой сложности.

Сделаем небольшое отступление от темы. Как вы думаете, какими качествами должна обладать хорошая система обнаружения вторжений? Безусловно, их достаточно много, но одним из основных является возможность проверять и изменять заданные сигнатуры. Хотите верьте, хотите нет, существуют сетевые системы обнаружения вторжений, которые не позволяют пользователям просматривать активные сигнатуры или вносить в них какие-либо изменения. Это лишает аналитика сетевого трафика возможности отличать ложные тревоги от действительных атак. Сообщение о тревоге выдается как неопровержимо доказанное, и нет никакой возможности проверить степень опасности с помощью одной только системы обнаружения вторжений. Чтобы принять более точное решение, аналитик сетевого трафика должен иметь возможность исследовать заданные сигнатуры и пакет, который спровоцировал запись-предупреждение.

Кроме того, сигнатуры, позволяющие аналитику сетевого трафика исследовать значения любого поля заголовка или полезной нагрузки с различных точек зрения, потенциально улучшают качество систем обнаружения вторжений. Другими словами, если NIDS помогает формулировать правила для исследования пакетов *только* по конкретному значению IP-адреса, или номеру порта, или протоколу, то, значит, она не способна исследовать полезную нагрузку или поля заголовков на более глубоком уровне, например на уровне установки TCP-флагов. Возможно, аналитику потребуется найти ACK-пакеты с определенным содержимым в полезных данных. Кроме ACK в пакете могут быть установлены и другие флаги, поэтому должны существовать сигнатуры для обнаружения пакетов с конкретным набором флагов.

Возможность анализировать любое поле пакета является отличительной чертой системы Snort. Существует множество различных параметров, позволяющих задать в правиле Snort любое поле и проверить его значение. Кроме того, если

несколько полей не могут быть проверены с помощью доступных параметров создания правил, то Snort помогает проанализировать их с помощью фильтра, указываемого в конце командной строки, или параметра -F, который позволяет использовать сохраненные в файле фильтры BPF (Berkeley Packet Filters). Фильтры BPF мы ранее называли фильтрами TCPdump. С их помощью в пакете конкретного протокола можно определить значение интересующего поля. Например, в Snort нет специального параметра для проверки значения поля "Версия IP", которое занимает старший полубайт нулевого байта IP-заголовка. С помощью BPF-фильтра система Snort может исследовать пакеты прямо из сети или из двоичного файла данных TCPdump, чтобы найти все пакеты, в которых указанное значение версии IP не равно 4. Для анализа пакетов, перехваченных из сети, можно использовать следующую команду.

```
snort -v 'ip[0] & 0xf0 != 0x40'
```

Как уже было подробно рассказано в главе 12, "Создание фильтров TCPdump", использованный здесь фильтр, позволяет замаскировать младший полубайт нулевого байта IP-заголовка и проверить, равно ли значение старшего полубайта 4. Результаты анализа выводятся на экран (параметр -v).

Еще одно преимущество использования Snort заключается в широком наборе предопределенных правил, которые поставляются совместно с системой. Не рекомендуется инсталлировать все доступные правила, так как, чем больше активных правил, тем медленнее выполняется проверка трафика. Аналитик сам должен решить, какие правила лучше подходят для его сети. Удивительно, но новые программные коды правил для Snort предоставляются всего через несколько часов после обнаружения очередной программы атаки. В этом еще раз проявляется преимущество открытого исходного кода, когда многие пользователи и разработчики Snort могут практически мгновенно реагировать на возникающую угрозу разработкой и тестированием новых правил.

Тем не менее необходимо высказать предупреждение относительно некоторых правил Snort. Не следует думать, что правило удачно только потому, что оно стало доступным вскоре после новой атаки. Правило может блокировать работу стандартной скомпилированной версии программы атаки, но может оказаться бесполезным при внесении хакером минимальных изменений в исходный программный код для этой атаки. Очень важно, чтобы разработчик правила разбирался не только в программе атаки и ее стандартных операциях, но и в протоколе, который используется программой атаки для выполнения вредоносных действий.

Хорошие правила можно узнать по наличию проверок для значений полей, которые должны оставаться постоянными для успеха той или иной программы атаки. Например, если в какой-либо программе атаки, направленной против службы DNS, используется стандартное значение идентификатора 0xBEEF, то проверку значения этого поля нельзя назвать составляющей хорошего правила. Это значение обычно меняют в исходном коде программы атаки, ведь ее успех совершенно не зависит от значения идентификатора DNS.

Скрытые сигнатуры

Однажды при выполнении заказа мне пришлось посетить поставщика сетевой системы обнаружения вторжений. Заказчик просил интегрировать отчеты его системы в какую-то кор-

релирующую программу. Честно говоря, я не верил в возможность такого усовершенствования из-за того, что нет никакого способа определить, реальна поднятая NIDS тревога или ложна. Причиной этого является недоступность заданных сигнатур и пакетов, которые спровоцировали предупреждения. Зачем заниматься пустой тратой времени? Но клиент потребовал моего присутствия, поэтому мне пришлось явиться на встречу.

Первым делом я спросил, есть ли способ узнать активные сигнатуры? Представитель клиента возмутился и спросил, зачем мне это нужно. "Мне необходимо знать, является атака настоящей или имеет место ложная тревога", — вежливо ответил я. В ответ я услышал, что, если я считаю, что столкнулся с редким случаем ложной тревоги, то могу позвонить в службу поддержки и попросить о помощи. При том количестве ложных тревог, которое генерировала система обнаружения вторжений этого поставщика, я мог только вообразить, что бы произошло, если бы эта глупость стала реальностью. Я с негодованием продолжал настаивать и спросил, почему же все-таки мне нельзя увидеть сигнатуры? Ответ был следующим: "Если вы узнаете наши сигнатуры — их смогут узнать хакеры!". Честное слово, это было самое нелепое объяснение, какое я только слышал. Я стал догадываться, что на самом деле он боится, чтобы сигнатуры его программы не узнали конкуренты, но не решается этого сказать. Как вы считаете, следует серьезно воспринимать таких людей и их "собственные" сигнатуры? К сожалению, мы не всегда можем оказывать влияние на решение о покупке той или иной системы обнаружения вторжений. Что если вы работаете в сети, в которой установлена NIDS, ограничивающая просмотр активных сигнатур и перехваченного трафика или не допускающая его? Вы безропотно примиритесь с такой ситуацией? Нет, следует проявить изобретательность и попытаться улучшить работу NIDS. Всегда можно запустить TCPdump в фоновом режиме, одну или совместно с системой Shadow. Кроме того, можно попытаться внести правки в выдаваемые отчеты с помощью других источников информации, например брандмауэров, маршрутизаторов или журналов хостов. Конечно, такое решение не идеально, но очевидно лучше, чем полное неведение.

Мы запустили Shadow совместно с системой обнаружения вторжений этого клиента. Через некоторое время мне сообщили, что система обнаружения вторжений выявила проводимую атаку Loki, и попросили проверить отчеты TCPdump, чтобы определить, была ли эта атака действительно, или тревога была ложной. Я помнил, что много лет назад стандартной сигнатурой атаки Loki было значение 0xf001 или 0x01f для порядкового номера ICMP-сообщения. Мне сообщили IP-адреса отправителя и получателя подозрительных пакетов. Просмотрев записи TCPdump, я обнаружил нужный отчет. Оказалось, что в данном случае имело место случайное использование этих значений для порядкового номера обычной пары эхо-запроса и эхо-ответа ICMP. Этот метод определения ложной тревоги был слишком трудным и потребовал много времени, но оказался лучше, чем полное доверие данной NIDS.

Создание правила Snort

Каждое отдельное правило Snort состоит из двух основных частей. Первая часть — заголовок — определяет, пакеты для каких хостов и портов попадают под действие этого правила. Во второй части — параметрах — определяется, что должно исследоваться, т.е. информация заголовка пакета (например, установленные TCP-флаги) или содержимое полезной нагрузки.

Вообще, не обязательно должны существовать обе части правила. Для некоторых правил можно указать только заголовок, и тогда данное действие будет применяться для всех пакетов, в которых в качестве получателей или отправителей указаны определенные хосты и/или порты. Обычно такие правила применяются для игнорирования трафика между определенными хостами и портами, например, для разрешения трафика, поступающего с порта 53 DNS-серверов локальной сети.

Для выдачи предупреждения или какого-то другого действия должны быть выполнены все условия, указанные и в заголовке правила, и в параметрах правила. Кроме того, важно понимать, что правила Snort анализируют каждый пакет отдельно и не рассматривают их в контексте других (предшествующих или последующих) пакетов. В Snort добавлена возможность анализа последовательности пакетов, которая реализуется с помощью препроцессора, например, дефрагмен-

тация IP-трафика или сборка потока TCP-данных, но все равно остаются определенные ограничения такого метода исследования отдельных пакетов.

Кроме того, если система Snort определяет, что пакет удовлетворяет первому правилу, то остальные правила уже не проверяются. Порядок правил, в котором они указаны в файлах, имеет определенное значение, но все же Snort зачастую самостоятельно решает, какое правило должно иметь приоритет. По умолчанию Snort располагает правила в порядке заданных ответных действий: предупреждение, пропуск и регистрация. Этот порядок можно изменить с помощью параметра командной строки. Однако Snort этим не ограничивается и систематизирует правила, объединяя их по одинаковым заголовкам (более подробную информацию по этой теме можно получить по адресу `www.snort.org/docs/faq.html#3.13`).

Рассмотрим пример правила Snort (рис. 13.1). Заголовок этого правила содержит сведения о действиях, которые должны быть предприняты при выявлении пакета, соответствующего правилу, и информацию об отправителе и получателе этого пакета. В данном случае должно выдаваться предупреждение (alert), если поступает TCP-трафик из любого порта хостов сети, отличной от 10.1.1.x, на любой порт локальной сети (мы считаем локальной сеть `10.1.1.x`). Таким образом, правило сработает при попытке установить TCP-соединение любым удаленным хостом.

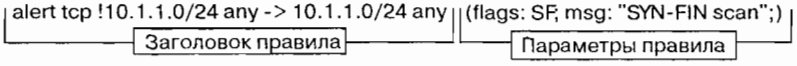


Рис. 13.1. Строение правила Snort

Теперь рассмотрим параметры правила, чтобы узнать, какими особенностями должны обладать подозрительные пакеты. В данном случае должна присутствовать нестандартная комбинация TCP-флагов: SYN и FIN, и, если они будут обнаружены, в качестве предупреждения должно выводиться сообщение "`SYN-FIN scan`" ("сканирование с помощью пакетов с установленными флагами SYN и FIN"). В следующих разделах будут более подробно рассмотрены ключевые слова, используемые при создании правил Snort.

Ходят слухи, что в версии Snort 2.0 синтаксис написания правил будет полностью изменен. Поэтому, если вы читаете эту главу после выхода версии 2.0, следует сначала просмотреть документацию программы Snort, ибо изложенные ниже сведения могут оказаться устаревшими.

Поля заголовка правила

Как мы уже рассказали, в заголовке правила указывается, какое действие должно выполняться при его "срабатывании", протокол и заданные IP-адреса, а также порты отправителя и получателя. При соблюдении всех этих условий будут проверены параметры правил (параметры правил Snort подробно рассмотрены в главе 14, "Параметры правил Snort").

Поле "Действие"

Поле "Действие" (Action) является первым полем в заголовке правила. В этом поле Snort уведомляется о том, какое действие должно выполняться при "срабатывании" правила. Ниже перечислены доступные значения для этого поля.

- **Alert (предупреждение)**. Если в поле "Действие" содержится это значение, то при получении пакета, соответствующего данному правилу, Snort создает запись в специальном файле (файл регистрации предупреждений) и регистрирует этот пакет. По умолчанию информация, сохраняемая в файле регистрации предупреждений, представляет собой копию заголовка подозрительного пакета. Регистрация пакета заключается в сохранении этой же информации в отдельном файле в создаваемом каталоге, имя которого обычно совпадает с IP-адресом отправителя подозрительного пакета (при использовании в командной строке параметра `-d` может также сохраняться полезная нагрузка пакета).
- **Log (регистрация)**. Это значение поля определяет только регистрацию пакета, удовлетворяющего правилу. Записи в файл регистрации предупреждений (alert file) не добавляются. В файлы регистрации (log file) могут записываться данные вложенного пакета, если в командной строке будет указан параметр `-d`.
- **Pass (пропуск)**. Если полученному пакету соответствует правило, в поле "Действие" которого содержится значение `pass`, система Snort прекращает исследование пакета. Это может пригодиться, например, при отслеживании попыток установить анонимное соединение с обычным (неанонимным) FTP-сервером. В этом случае можно сформулировать правило, которое бы позволило организовать прохождение пакетов с попытками анонимного подключения к анонимному FTP-серверу локальной сети. Второе правило предупреждения позволит выявить все остальные попытки анонимных FTP-соединений.
- **Activate (запуск)**. Это значение позволяет не только выдать предупреждение, но и запустить другие (динамические) правила.
- **Dynamic (динамическое)**. Правила с этим значением поля "Действие" остаются неактивными до тех пор, пока не будут выполнены правила со значением `activate`. После активирования этих правил их действия аналогичны обычным правилам регистрации.

Обратите внимание на то, что действия `activate` и `dynamic` замещаются параметром `tag`, который может быть задан во второй части правила — в его параметрах. Параметр `tag` позволяет организовать динамическую регистрацию пакетов на определенный период времени или регистрацию указанного количества пакетов после "срабатывания" правила.

Кроме того, можно определить собственные действия, что позволяет указать, куда следует вывести информацию о "срабатывании" правила. Здесь мы не станем рассматривать такие возможности, а все желающие могут получить интересующую информацию на официальном сайте Snort (`www.snort.org`). Как уже указывалось, по умолчанию правила выполняются в следующем порядке: предупреждение, пропуск, регистрация. Изменить этот порядок позволяет параметр `-o` в командной строке запуска Snort. Параметр `-o` позволяет организовать обработку первыми правил пропуска, затем — правил предупреждения и последними — регистрации. Принятая по умолчанию последовательность выполнения правил предназначена для неопытных пользователей и не позволяет случайно заданному правилу пропуска отключить все предупреждения и регистрацию опасных пакетов.

Поле "Протокол"

Информация этого поля указывает Snort на то, пакеты какого протокола следует проверять. Snort поддерживает работу со следующими протоколами: TCP, UDP, ICMP и IP. В будущем планируется добавить поддержку ARP, RARP, GRE, OSPF, RIP и IPX. Система Snort работает с дейтаграммами протокола IP только версии 4, поэтому она не определит использование для пакета протокола IPv6. Snort не поддерживает спецификацию IPSec, поэтому не способна расшифровать информацию полей пакетов этого набора протоколов.

Поля "IP-адрес отправителя" и "IP-адрес получателя"

По информации полей IP-адресов отправителя и получателя можно определить, откуда исходит и куда направляется подозрительный трафик. В качестве IP-адресов может быть указан IP-адрес хоста, подсети или нескольких хостов или подсетей. IP-адреса записываются согласно простому и понятному стандарту записи CIDR (Classless Inter-Domain Routing – бесклассовая междоменная маршрутизация). Этот формат записи содержит необходимый IP-адрес и количество битов, используемое для маски подсети. Давайте рассмотрим этот формат на примере нескольких IP-адресов.

Формат:

```
адрес/маска_подсети или any или [адрес/маска_подсети, адрес/маска_подсети]

адрес = x.x.x.x
маска подсети = количество битов сетевой части адреса

24.0.0.0/8 =              Сеть класса A
135.1.0.0/16 =            Сеть класса B
192.168.5.0/24 =          Сеть класса C
192.168.5.5/32 =          Адрес хоста
```

Ключевые слова:

any — все возможные адреса;
! — символ отрицания;
$HOME_NET — переменная, значение которой определено в файле правила.

Таким образом формат CIDR определяет базовый адрес и количество битов, которое отведено для хранения сетевой части адреса. Например, запись `24.0.0.0/8` означает, что это адрес сети класса A, первый байт которого (24) является сетевой частью адреса, а остальные байты используются для обозначения хостов этой сети. Хотя в наших примерах использованы стандартный формат CIDR для сетей классов A, B и C, основное преимущество этого формата заключается в том, что разделительная линия между сетевой и узловой частью адреса может отделять любое число битов 32-разрядного IP-адреса, и не обязательно разделять его по байтам.

Можно задать перечень контролируемых IP-адресов, указав их в квадратных скобках и разделив запятыми (никакие пробелы не допускаются). Например, если необходимо отслеживать трафик, предназначенный хосту `1.2.3.4` и подсети `2.3.4.x`, можно использовать следующую запись.

```
[1.2.3.4,2.3.4.0/24]
```

Ключевое слово any означает любой IP-адрес. Символ отрицания (!) позволяет сделать в правиле исключение для какого-то IP-адреса. И, наконец, можно за-

давать IP-адреса с помощью значения переменной, что обеспечивает дополнительную гибкость и переносимость правил. Для указания адреса локальной сети в правилах Snort часто применяется переменная `$HOME_NET`. Можно использовать любое другое имя переменной, но так как во многих правилах уже существуют ссылки на переменную `$HOME_NET`, то лучше воспользоваться именно ею. Эта переменная должна быть определена в файле правила, в конфигурационном файле или в командной строке (с помощью параметра `-S`) до ее использования. Переменные могут использоваться и в других полях правил Snort.

Поля "Порт отправителя" и "Порт получателя"

Эти поля используются для указания контролируемых портов отправителя и получателя. Может быть задан отдельный номер порта, диапазон номеров или использовано ключевое слово `any`, которое обозначает все возможные порты. Ниже показано несколько примеров определения портов.

статический порт	`111`
все порты	`any`
диапазон	`33000:34000`
исключение	`!80`
меньше или равно	`:1023`
больше или равно	`1024:`

Первый пример соответствует указанию номера статического порта, например, порта 111, который обычно используется RPC-службой `portmapper`. Как и для IP-адресов ключевое слово `any` позволяет обозначить все возможные порты. Также может быть указан диапазон контролируемых портов, например 33000–34000 (`33000:34000`) для выявления действий UNIX-программы Traceroute. С помощью символа отрицания можно организовать исключение конкретного порта из контролируемого диапазона (`!80`). Можно задать проверку пакетов для портов, номера которых меньше (`:1023`) или больше (`1024:`) определенного значения, например для проверки доступа к временным портам. Кроме того, можно задать номер порта как переменную, значение которой должно быть присвоено до ссылки на эту переменную.

А зачем указывать порты для ICMP-пакетов? Ведь в этом протоколе порты вообще не используются. Тем не менее синтаксис написания правил требует задать номер порта, поэтому в данном случае нужно использовать какой-то заполнитель, чаще всего это ключевое слово `any`.

Указатель направления

Поле направления трафика позволяет указать направление, в котором должен передаваться пакет. Для использования доступны два параметра. Первый из них выглядит как стрелка (`->`) и указывает, что пакет должен передаваться от отправителя получателю. Отправитель находится слева от стрелки, а получатель — справа. Если пакет передается в обратном направлении, то он не будет удовлетворять правилу.

Использование второго параметра, напоминающего двустороннюю стрелку (`<>`), позволяет указать, что пакет может передаваться в любом направлении. При такой форме записи не имеет значения, какой хост является отправителем, а какой получателем.

Резюме

Snort является очень хорошей и бесплатной системой обнаружения вторжений. Максимально ее возможности проявляются в режиме перехвата трафика, когда проходящие пакеты сравниваются с набором заданных правил. Анализ пакетов может выполняться в реальном времени, или же трафик сохраняется в двоичном формате и впоследствии анализируется системой Snort как входной файл.

Правила Snort представляют собой гибкие и простые средства для исследования значений большинства полей заголовков, а также для анализа любой части полезных данных. Они предоставляют пользователю различные способы проверки значений конкретных полей и даже использования переменных для хранения этих значений. Возможность точного указания в правилах Snort атрибутов различных пакетов позволяет добиться высокой достоверности результатов проверки по вопросу блокирования или пропуска пакета. При этом число ложных тревог уменьшается до минимального значения.

14

Параметры правил Snort

В предыдущей главе наши читатели ознакомились с основными принципами работы системы Snort и методами написания правил для этой системы. Напомним, что правило Snort состоит из заголовка и параметров, которые и будут подробно рассмотрены в этой главе.

Заголовок правила определяет действие, выполняемое при получении соответствующего правилу пакета. В нем также задаются IP-адреса и порты отправителя и получателя, используемый протокол и направление передачи пакета. Заголовок правила может использоваться и как самостоятельное правило, но чаще после него указываются параметры, с помощью которых более точно задаются атрибуты интересующих пакетов. Интересно, что существуют коммерческие системы обнаружения вторжений, в которых сигнатуры анализа трафика предоставляют возможности, аналогичные возможностям заголовков правил системы Snort. Другими словами, в этих системах нельзя задать более подробный анализ пакета, чем проверка IP-адресов, используемого протокола и номеров портов. Безусловно, подобные сигнатуры никак нельзя назвать точными. Для системы Snort параметры правил являются основой функциональных возможностей обнаружения взлома.

Формат параметров правила

В Snort параметры правила отделяются от его заголовка с помощью круглых скобок. Рассмотрим следующее правило.

```
alert tcp !$HOME_NET any -> $HOME _NET any (flags: SF; \
msg: "SYN-FIN scan";)
```

В данном случае параметры заданы в строке `(flags: SF; \ msg: "SYN-FIN scan";)`. Каждый параметр включает в себя ключевое слово и (возможно) его значение. В этом примере для ключевого слова `flags` установлено значение `SF`, а для ключевого слова `msg` — значение `SYN-FIN scan`. Для некоторых параметров в качестве значений ключевого слова должны устанавливаться численные

значения, для других это должны быть специальные коды. Между ключевым словом и его значением устанавливается символ двоеточия (:), а параметры отделяются друг от друга точкой с запятой (;). Кроме того, точка с запятой должна быть установлена после последнего параметра. В противном случае будет выдано сообщение об ошибке. Хотя значения большинства ключевых слов требуется устанавливать, но есть и несколько исключений. Например, не существует значений для ключевого слова nocase, которое определяет, что при исследовании содержимого полезных данных пакета не должен учитываться регистр символов.

Для Snort не имеет значения наличие нескольких или полное отсутствие пробелов между символами разделения (; и :), параметрами и значениями. Например, оба следующих набора параметров являются равнозначными.

```
(flags:SF;msg:"SYN-FIN scan";)
(flags:  SF    ; msg  : "SYN-FIN scan"   ;)
```

Символ обратной косой черты (\) является символом продолжения правила. Он устанавливается в конце строки незавершенного правила и позволяет продолжить его написание в следующей строке. Символ "решетки" (#) используется для добавления комментариев в правила Snort.

Параметры правил

Теперь приступим к рассмотрению наиболее популярных параметров правил Snort, которые позволят нам продемонстрировать всю мощь возможностей этой системы. Описание всех доступных параметров правил Snort можно найти на сайте www.snort.org, открыв в ссылке Documentation раздел Snort Users Manual.

Параметр msg

С помощью параметра msg можно задать специальное сообщение, выдаваемое при "срабатывании" правила. Записи в файле регистрации или в файле зарегистрированных предупреждений представляют собой только отчеты о подозрительных пакетах. В этих записях не указано правило, по которому была поднята тревога. Поэтому необходимо создать какое-то описание, которое бы позволило связать уведомление о тревоге с конкретным правилом. Значение параметра msg будет сохранено перед записью о подозрительном пакете, что позволит быстрее разобраться в причине тревоги.

Ниже приведены формат параметра, пример правила и запись о зарегистрированном предупреждении в результате "срабатывания" этого правила.

Формат:

```
msg: "<текст сообщения>";
```

Пример правила:

```
alert udp any any ->192.168.5.0/24 31337 \
(msg: "Back Orifice";)
```

Пример отчета:

```
 [**] Back Orifice [**]
04/24-08:49:21.318567 192.168.143.15:60256 -> 192.168.5.16:31337
UDP TTL:41 TOS:0x0 ID:49951
Len:  8
```

Это правило Snort вызывает предупреждение (и регистрацию) при получении с любого IP-адреса и порта отправителя пакета, предназначенного хостам подсети 192.168.5 на порт 31337. При этом должно выдаваться сообщение "Back Orifice".

Параметр logto

Параметр `logto` позволяет задать имя файла, в который будут записываться отчеты. Он используется только для правил, для которых определено действие предупреждения или регистрации. Обычно при "срабатывании" таких правил информация сохраняется в каталоге по умолчанию (`var/log/snort` для UNIX-хостов, а для хостов под управлением Windows – в подкаталоге `log` каталога, в который была установлена Snort) или в каталоге, указанном после параметра командной строки `-l` при запуске Snort. При этом предполагается, что пользователь не изменил применяемую по умолчанию регистрацию на сохранение информации в двоичном формате (с помощью параметра командной строки `-b`) или на передачу данных демону системного журнала `syslog` (с помощью параметра командной строки `-s`) и полностью не отменил регистрацию (с помощью параметра командной строки `-N`).

Воспользовавшись параметром `logto`, можно организовать отправку отчетов о "срабатывании" определенного правила или группы правил в отдельный файл. Использование этого параметра является отличным методом для отделения отчетов о действительно опасных и вредоносных атаках от других отчетов. В приведенном ниже примере при подозрении в попытке использовать пакет `trinoo` для проведения распределенной атаки отказа в обслуживании или попытке осуществить эту же атаку с помощью какой-либо другой программы можно просмотреть записи в специальном файле регистрации подобных тревог. Кроме того, все отчеты об этих атаках также будут сохраняться в файле регистрации, используемом по умолчанию, так как для данного правила определено действие предупреждения.

Формат:

```
logto: "<имя_файла>";
```

В Snort не нужно задавать полное имя файла (с указанием пути). Если будет указан путь к файлу, то Snort выдаст сообщение об ошибке. Имя файла необходимо заключать в кавычки, иначе перед этим именем может быть добавлен пробел.

Пример правила:

```
alert udp any any -> 192.168.5.0/24 31335 \
(msg: "trinoo port"; logto: "DDoS";)
```

Пример отчета (при "срабатывании" правила запись об этом событии на UNIX-хосте заносится в файл `/var/log/snort/DDoS`):

```
 [**] trinoo port [**]
04/24-09:07:41.320938 192.168.143.15:56881 -> 192.168.5.16:31335
UDP TTL:42 TOS:0x0 ID:4011
Len: 8
```

Параметр tcl

С помощью параметра `tcl` можно проверить значение поля TTL полученного пакета. Для использования этого параметра существует множество причин. Одной из них является поиск пакетов с небольшими значениями поля TTL, что сви-

детельствует об использовании программы Traceroute удаленным UNIX-хостом или программы Tracert удаленным Windows-компьютером. Если при этом используется один из UDP-портов в диапазоне от 33000 до 34000, то скорее всего применяется программа Traceroute. Windows-программа Tracert работает с помощью эхо-запросов ICMP.

Следующее правило позволяет обнаружить запросы программы Traceroute к сети 192.168.5 с указанным портом отправителя в диапазоне от 33000 до 34000 и значением поля TTL, равным 1.

Формат:

```
ttl: <число>;
```

Пример правила:

```
alert udp any any ->192.168.5.0/24 33000:34000 \
(msg: "Unix traceroute"; ttl: 1;)
```

Пример отчета:

```
[**] Unix traceroute [**]
04/24-09:29:37.971353 192.168.143.15:40920 -> 192.168.5.16:33437
UDP TTL:1 TOS:0x0 ID:40923
Len: 18
```

Параметр id

Для хранения значения идентификатора в IP-заголовке используется 16-битовое поле. Для каждой новой дейтаграммы устанавливается уникальный идентификатор, и его значение обычно увеличивается на 1 для каждого следующего пакета. При фрагментации это число называют *идентификатором фрагмента*, по которому выполняется сборка получаемой последовательности фрагментов. В следующем примере представлено правило, с помощью которого можно выявить пакеты с необычным значением идентификатора, равным 0. К сожалению, ядро операционной системы Linux 2.4 устанавливает значение 0 для идентификатора IP-дейтаграмм, когда в дейтаграмме также установлен флаг DF (Don't Fragment – не фрагментировать). Объясняют это тем, что если пакет не может быть фрагментирован, то зачем думать об идентификаторе фрагмента.

Формат:

```
id: <число>
```

Пример правила:

```
alert icmp any any -> 192.168.5.0/24 any \
(msg: "Suspect IP Identification #"; ID:0;)
```

Пример отчета:

```
[**] Suspect IP Identification # [**]
04/25-09:21:36.371005 192.168.143.15 -> 192.168.5.16
ICMP TTL:64 TOS:0x0 ID:00
```

Параметр dsize

Параметр `dsize` позволяет Snort исследовать размер полезной нагрузки пакета. Можно установить проверку по точному значению или задать поиск пакетов с

большей или меньшей по размеру полезной нагрузкой. Это может пригодиться при создании правила для блокирования атак на переполнение буфера. Эти атаки легко выявить, если организовать поиск пакетов, размер полезных данных которых превышает ожидаемое значение. Следующее правило предназначено для выявления ICMP-пакетов, полезная нагрузка которых превышает 1024 байт.

Формат:

```
dsize: [<|>] число
```

Пример правила:

```
alert icmp any any ->192.168.5.0/24 any \
(msg: "Превышение стандартной длины ICMP-данных"; dsize: >1024;)
```

Пример отчета:

```
[**] Превышение стандартной длины ICMP-данных [**]
04/24-11:24.110169 192.168.143.100 -> 192.168.5.16
ICMP TTL:255 TOS:0x0 ID:5487 DF
ID:7564   Seq:0 ECHO
```

Параметр sequence

Параметр sequence позволяет проверить значение поля порядкового номера TCP-сегмента. Распределенную атаку отказа в обслуживании, осуществляемую с помощью программы Shaft, можно выявить по неизменному порядковому номеру TCP-сегментов. В шестнадцатеричном формате это значение равно 28374839 для всех TCP-сегментов наводнения, направленного на атакуемый хост. Несомненно, это значение установлено где-то в исходном коде атаки Shaft и может быть изменено. Поэтому не стоит думать, что данное правило позволит полностью заблокировать эту атаку. Кроме того, и в абсолютно нормальном пакете может использоваться данное значение порядкового номера.

Формат:

```
seq: <число>
```

Пример правила:

```
alert tcp any any -> any any \
msg: "Возможно, проводится DDoS-атака Shaft"; seq:0x28374839;)
```

Пример отчета:

```
[**] Возможно, проводится DDoS-атака Shaft [**]
04/25-07:19:58.582562 192.168.143.100:35680 -> 192.168.143.15:23
TCP TTL:255  TOS:0x0 ID:7705 DF
******S* Seq: 0x28374839  Ack: 0x0  Win: 0x2238
TCP Options => MSS: 1460
```

Параметр acknowledgement

С помощью параметра acknowledgement можно проверить значение номера подтверждения TCP-сегмента. Основным предназначением такой проверки является выявление попыток сканирования, осуществляемых программой nmap. Для выявления активного хоста nmap отправляет TCP-пакет с установленным флагом ACK, а значение номера подтверждения в этом пакете равно 0. В нормальном TCP-сеансе полу-

чение пакета с таким номером подтверждения является довольно редким, так как это означает, что в предыдущем полученном TCP-сегменте значение порядкового номера достигло $2^{32}-1$ (только тогда номер подтверждения обнуляется).

Формат:

```
ack: <число>;
```

Пример правила:

```
alert tcp any any -> any any \
(msg: "nmap TCP ping"; flags: A; ack: 0;)
```

Пример отчета:

```
[**] nmap TCP ping [**]
04/25-07:27:13.578488 192.168.143.15:63367 -> 192.168.143.16:80
TCP TTL:42 TOS:0x0 ID:26253
***A**** Seq: 0x16680003   Ack: 0x0   Win: 0xC00
```

Параметры itype и icode

Параметр `itype` используется для указания типа ICMP-сообщения. Поле "Тип" находится в нулевом байте ICMP-пакета. Возможные значения этого поля и поля для хранения кода ICMP-сообщения, которое проверяется с помощью параметра `icode`, можно узнать по адресу `www.iana.org/assignments/icmp-parameters`. Чаще всего параметры `itype` и `icode` используются совместно. Код ICMP-сообщения указывается в первом байте ICMP-пакета. Многие ICMP-сообщения отличаются только кодом и имеют один и тот же тип. Например, существует довольно много различных кодов для ICMP-сообщений типа 3. Чтобы выявить ICMP-сообщения о недостижимости порта, нужно создать правило Snort, согласно которому значение параметра `itype` равнялось бы 3, и значение параметра `icode` тоже бы составляло 3.

Формат:

```
itype: <число>;
icode: <число>;
```

Пример правила:

```
alert icmp 1.1.1.0/24 any -> 192.168.5.0/24 any \
(msg: "port unreachable"; itype: 3; icode: 3;)
```

Пример отчета:

```
[**] port unreachable [**]
04/25-07:56:37.129338 1.1.1.16 -> 192.168.5.15
ICMP TTL:255 TOS:0xC0 ID:33569
DESTINATION UNREACHABLE: PORT UNREACHABLE
```

Параметр flags

С помощью параметра `flags` можно организовать различные проверки установленных в TCP-сегменте флагов. Перечислим их, начиная с самого младшего флага (справа налево в байте TCP-флагов):

- F – установлен флаг FIN;
- S – установлен флаг SIN;

- R — установлен флаг RESET;
- P — установлен флаг PUSH;
- A — установлен флаг ACK;
- U — установлен флаг URG;
- R — установлен флаг RESET;
- 2 — установлен эхо-бит ECN (ранее зарезервированный бит);
- 1 — установлен бит CWR (ранее зарезервированный бит);
- 0 — не установлено ни одного флага.

Кроме того, для проверки установленных комбинаций флагов допускается использование трех модификаторов (+,*,!). Например, запись A+ означает, что в пакете должен быть установлен флаг ACK. При этом он может быть единственным установленным флагом или вместе с ним могут быть установлены и другие флаги. Описание подходит и для пакетов со стандартной комбинацией флагов PUSH (т.е. новые данные отправляются одновременно с подтверждением получения прошлой порции). Модификатор * используется для выявления пакетов, в которых установлен один или несколько флагов из заданной комбинации. Например, строка SFP* определяет поиск пакетов с установленным хотя бы одним из трех флагов SYN, FIN и PUSH, а также пакетов с любой их комбинацией. Последний модификатор ! позволяет задать поиск пакетов, в которых не установлен данный флаг. Параметр `flags:!S`, например, позволяет найти все пакеты, в которых не установлен флаг SYN.

Формат:

```
flags: <код_установленных_флагов>
```

На рис. 14.1 показано графическое представление кодов Snort для флагов TCP. Доступные модификаторы флагов:

- + — все пакеты, в которых установлен заданный флаг (или флаги) и любые другие флаги;
- * — все пакеты, в которых установлен один из указанных флагов;
- ! — все пакеты, в которых не установлен указанный флаг (флаги).

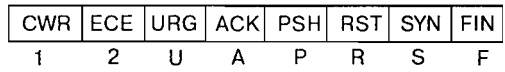


Рис. 14.1. Байт TCP-флагов в кодах системы Snort

Пример правила:

```
alert tcp any any -> any any (msg: "Сканирование с помощью null-пакетов"; /
 flags:0;)
```

Пример отчета:

```
[**]Сканирование с помощью null-пакетов [**]
04/25-05:49:51.914748 192.168.143.15:54746 -> 192.168.143.16:21
TCP TTL:51 TOS:0x0 ID:23446
******** Seq: 0x1CED3E2E  Ack: 0x0  Win:0x1000
TCR Options => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL EOL
```

В приведенном выше примере отчета содержится строка из восьми символов звездочек (********). Snort заменяет звездочки соответствующими кодами для установленных в пакете флагов (12UAPRSF), если получение этого пакета вызвало предупреждение. Так как в данном случае проводилось сканирование с помощью null-пакета (*в пакете не установлено никаких флагов*), *то все звездочки остались на месте.*

Параметр content

Один из самых важных параметров — content — зачастую используют неверно. С его помощью осуществляется проверка содержимого полезных данных пакетов. Доступны различные способы задания значений для параметра content, и можно организовать проверку по нескольким таким значениям. Просмотр содержимого полезных данных требует значительного расхода вычислительных средств, другими словами, неправильное использование параметра content может серьезно замедлить работу Snort. Хотя создатели Snort максимально оптимизировали алгоритм для просмотра полезных данных пакетов, но все равно эта операция выполняется значительно медленнее, чем, например, поиск заданных значений в полях заголовков. Это объясняется тем, что длина поля заголовка составляет максимум 4 байт, а размер полезной нагрузки обычно намного больше.

По возможности параметр content должен использоваться совместно с такими параметрами, как flags, или другими, которые позволят сузить диапазон поиска. Например, можно задать смещение в полезных данных, с которого должен начинаться поиск информации, и размер исследуемого фрагмента полезных данных. Параметр content проверяется в последнюю очередь, даже если он указан первым в перечне параметров правила. Это делается для оптимизации поиска.

Строка поиска может быть задана как текст, как шестнадцатеричное представление двоичных данных или как комбинация текста и шестнадцатеричных символов. Текстовые строки заключаются в кавычки и при поиске учитывается регистр символов (если не использован параметр nocase). Шестнадцатеричный код отделяется с помощью символов конвейеризации (|). С помощью параметра content (или нескольких подобных ему) для поиска в пакете могут быть заданы различные значения, при этом для "срабатывания" правила в пакете должны присутствовать все заданные значения. Значения, заданные с помощью нескольких параметров content, могут находиться в подозрительном пакете в любой последовательности, независимо от порядка параметров content. Существует и другой параметр content-list, который позволяет считать правило "сработавшим" при нахождении соответствия хотя бы с одним из заданных параметров content-list. Описание этого параметра и пример его использования можно найти на сайте www.snort.org в разделе The Snort Users Manual.

Формат:

```
content: <"value">;
content: <"value">; content: <"value">;
```

Пример правила:

```
alert udp $EXTERNAL_NET any -> $HOME_NET 53 \
(msg: "Атака BIND попытка переполнения буфера tsig"; \
content: "|00 FA 00 FF|"; content: "/bin/sh";);
```

Пример отчета:

```
02/22-15:33:19.472301 ATTACKER:1024 -> VICTIM:53
UDP TTL:64 TOS:0x0 ID:6755 IpLen:20 DgmLen:538
Len:518

<пропущенные строки для сокращения отчета>

00 3F 90 E8 72 FF FF FF 2F 62 69 6E 2F 73 68 00 .?..r.../bin/sh.
0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D ................
1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D .. !"#$%&'()*+,-
2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C EB ./0123456789:;<.
07 C0 00 00 00 00 00 3F 00 01 02 03 04 05 06 07 .......?........
08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 ................
18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 ........ !"#$%&'
28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 ()*+,-./01234567
38 39 3A 3B 3C EB 07 C0 00 00 00 00 00 3F 00 01 89:;<........?..
02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 ................
D8 FA FF BF D8 F7 FF BF D0 7C 0D 08 04 F7 10 40 .........|.....@
22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 "#$%&'()*+,-./01
32 33 34 35 36 37 38 39 3A 3B 3C EB 07 C0 00 00 23456789:;<.....
00 00 00 3F 00 01 02 03 04 05 06 07 08 09 0A 0B ...?............
0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B ................
1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B .... !"#$%&'()*+
2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B ,-./0123456789:;
3C EB 07 C0 00 00 00 00 00 00 00 FA 00 FF <.............
```

В этом отчете обнаруженные в полезных данных шестнадцатеричные символы отображены слева, а справа представлены соответствующие им ASCII-символы. Созданное правило предназначено для обнаружения входящего UDP-трафика, отправленного на порт 53 одному из компьютеров защищаемой сети. В частности, отслеживается передача двух строк: первая из них представлена в шестнадцатеричном формате `00 FA 00 FF`, а вторая — в текстовом (строка `/bin/sh`). Проверяется присутствие обоих этих строк в полезной нагрузке в любом порядке. Это правило будет использовано только после проверки всех остальных параметров для поступившего пакета.

Некоторые параметры правил используются только в качестве модификаторов для параметра `content`, другими словами, при их самостоятельном использовании без этого параметра будет выдано сообщение об ошибке. Такими параметрами являются: `offset`, `depth`, `nocase` и `regex`. Они уточняют характеристики поиска только для указанного непосредственно перед ними параметра `content`.

Какой флаг

Если исследовать правила Snort, поставляемые совместно с самой системой, то можно обнаружить, что в очень многих правилах кроме параметра content используется параметр `flags` со значением A+. Таким образом, для "срабатывания" правила в пакете должен быть установлен флаг ACK (и, возможно, другие флаги). Это может показаться неправильным, и многие могут логично возразить: "А почему не использовать параметр flags со значением P+?". В конце концов, разве система Snort не должна исследовать содержимое, когда в пакете передаются байты полезной нагрузки?

Все это так, и обработка пакетов становится более эффективной, если правило с использованием параметра content будет применяться только для тех пакетов, в которых передаются полезные данные. Но, как сказано в книге Ричарда Стивенса *TCP/IP Illustrated, Volume 1*, несмотря на то, что стеки TCP/IP многих BSD-систем устанавливают флаг PUSH для каждого пакета, в котором передаются данные, в других операционных системах этот флаг устанавливается только тогда, когда пользователь инициирует немедленную отправку данных

из исходящего буфера. Это означает, что если получатель устанавливает небольшой размер TCP-окна, и отправитель не освобождает весь свой выходной буфер, то в отправляемых пакетах устанавливается только флаг ACK. Поэтому для параметра `flags` используется значение `A+`. Несмотря на то что многие пакеты с флагом ACK не несут полезной нагрузки, они также будут проверяться.

Альтернативным вариантом проверки наличия в пакете полезных данных является использование параметра `dzise > 0`. Это позволяет обнаружить необычный трафик, например, передачу данных в SYN-пакетах, в которых флаг ACK не устанавливается.

В качестве примера отправки полезных данных в пакете, в котором установлен только флаг ACK, рассмотрим два отчета о действиях программы LaBrea версии 2. Как указывалось в главе 9, "Анализ заголовков вложенных пакетов", эта программа позволяет замедлить атаку злоумышленника с помощью искусственного занижения размера окна. В первой записи хост с программой LaBrea (IP-адрес `10.10.10.155`) выдает себя за Web-сервер и устанавливает необычно малое значение размера окна, равное 5. Хост нарушителя `attacker.net` отправляет 5 байт полезных данных. Как вы видите, в передаваемом пакете установлен только флаг ACK и нет флага PUSH, так как размер отправленных данных не позволил полностью освободить буфер хоста `attacker.net`.

```
10.10.10.155.www > attacker.net.2045: S 998514038:998514038(0) ack
↳ 882335287 win 5
attacker.net.2045 > 10.10.10.155.www: . 1:6(5) ack 1 win 8576 (DF)
```

Параметр offset

Поиск информации в полезных данных требует значительных затрат ресурсов, но его можно сделать более эффективным, ограничив размер исследуемой части данных с помощью значения смещения (`offset`), которое указывает на то, сколько байтов данных можно пропустить от начала полезной нагрузки. По умолчанию поиск в содержимом начинается с первого байта полезных данных, что соответствует смещению 0.

Формат:

```
offset: <число>;
```

Пример правила:

```
alert tcp any any -> 192.168.5.0/24 21   \
(msg: " Попытка анонимного доступа к ftp-серверу ";  \
content: "anonymous"; offset: 5;)
```

Пример отчета:

```
[**] Попытка анонимного доступа к ftp-серверу [**]
04/24-12:11:08.724441 192.168.143.15:3484 -> 192.168.5.16:21
TCP TTL:64 TOS:0x10 ID:30124  DF
***AP*** Seq: 0x93EE0AB7  Ack: 0xB8352E61  Win: 0x7D78
TCP Options => NOP NOP TS: 112024246 27551686

55 53 45 52 20 61 6E 6F 6E 79 6D 6F 75 73 0D 0A  USER anonymous..
```

Строка "anonymous" находится в шестом байте полезной нагрузки, но так как при вычислении смещения счет начинается с нуля, то значение смещения 5 позволяет обнаружить заданную строку.

Параметр depth

Параметр `depth` представляет собой еще один параметр, с помощью которого можно ограничить размер исследуемых Snort полезных данных пакета. Он определяет количество байтов, исследуемых после заданного смещения. Если смещение не задано, оно считается равным 0. Этот параметр позволяет значительно

повысить эффективность работы Snort при обработке пакетов с большими объемами полезных данных, для которых известна структура этих передаваемых данных (в какой части пакета находятся те или иные данные).

Формат:

```
depth: <число>
```

Пример правила:

```
alert udp !$HOME_NET any -> $HOME_NET 5632 \
(msg: "PCAnywhere Startup"; content: "ST"; depth: 2;)
```

Пример отчета:

```
[**] PCAnywhere Startup [**]
04/24-12:11:08.724441 192.168.143.15:3484 -> 192.168.143.16:5632
UDP TTL:64 TOS:0x10 ID:30124  DF
73 74 61 72 74 75 70 STARTUP
```

Пакет будет соответствовать этому правилу, если в его первых двух байтах (по умолчанию смещение равно 0) полезной нагрузки будет обнаружена строка "ST".

Параметр nocase

С помощью параметра `nocase` определяется, что при поиске заданной строки не будет учитываться регистр символов. Это один из немногих параметров, который не требует присвоения значения.

Формат:

```
nocase
```

Пример правила:

```
alert tcp any any -> any 21  \
(msg: "FTP warez snooping"; content: "warez"; nocase;)
```

Пример отчета:

```
[**] FTP warez snooping [**]
04/25-05:28:28.146374 192.168.143.15:3487 -> 192.168.143.16:21
TCP TTL:64 TOS:0x10 ID:30637  DF
***AP*** Seq: 0xE1977C8D   Ack: 0x452F7F9   Win: 0x7D78
TCP Options => NOP NOP TS: 118248207 33775174
43 57 44 20 57 61 52 65 5A 0D 0A          CWD WareZ..
```

Параметр regex

Параметр `regex` позволяет в строке символов, задаваемой для поиска с помощью параметра `content`, использовать универсальные символы. Возможно использование двух универсальных символов: знак ? можно применять для обозначения одного любого символа, а звездочка * позволяет заменить любое количество символов.

С помощью параметра `regex` можно выявить признаки атак на переполнение буфера. При успехе этой атаки на хосте под управлением UNIX злоумышленник, скорее всего, захочет получить доступ к командному интерпретатору, например интерпретатору Bourne (файл `/bin/sh`), хотя существует много других командных интерпретаторов, таких как командный интерпретатор C (`chs`), Korn (`ksh`) и Bourne again (`bash`). Таким образом, с помощью одной строки и универсального символа можно обнаружить попытки доступа ко всем командным интерпрета-

торам. До появления параметра regex единственным способом проверить попытки доступа к командным интерпретаторам являлось создание отдельных правил. Не забывайте, что параметр regex стал полнофункциональным только в версии Snort 2.0.

Формат:

```
regex;
```

Пример правила:

```
log tcp any any -> 192.168.5.0/24 515/
(msg: "Попытка получения доступа к командному интерпретатору /
посредством lpd"; content: "/bin/*sh"; regex;)
```

Пример отчета:

```
[**Попытка получения доступа к командному интерпретатору посредством
lpd**]
03/23-07:41:11.282960 1.1.0.1:1892 -> 192.168.5.55:515
TCP TTL:64 TOS:0x0 ID:63821 IPLen:20 DgmLen:60
***AP*** Seq: 0x32A77D55  Ack: 0x0  Win: 0x200  TcpLen: 20
2F 62 69 6E 2F 63 73 68 0A 00 00 00 00 00 00 00 /bin/csh.......
00 00 00 00
```

Приведенное выше правило позволяет контролировать попытки доступа к командному интерпретатору с помощью пакетов, отправленных на порт 515 (порт lpd). Уточняющее значение параметра regex (/bin/*sh) для параметра content позволяет выявить любые попытки доступа к командному интерпретатору.

Параметр session

Параметр session позволяет организовать проверку данных, вводимых пользователем во время TCP-сеансов. Сохраненная информация о вредоносных действиях пользователя может использоваться при проведении судебного расследования.

Для параметра существует два доступных ключевых слова: printable и all. С помощью ключевого слова printable сохраняются печатаемые символы. Ключевое слово all позволяет зарегистрировать также и все непечатаемые символы с их шестнадцатеричными эквивалентами.

Будьте готовы к тому, что применение параметра session может снизить производительность Snort, поэтому его лучше использовать для уже зарегистрированных данных: можно сохранить данные в двоичном формате (файлы TCPdump) и затем просмотреть их с помощью Snort. Кроме того, при применении этого параметра должны проверяться данные, передаваемые в обоих направлениях (как показано в примере). И последнее: желательно добавлять в командную строку параметр -d для сохранения данных на уровне приложений, в противном случае использование параметра session не имеет особого смысла.

По умолчанию отчеты о сеансах сохраняются в каталоге log. Затем информация разделяется по подкаталогам, названия которых аналогичны IP-адресам хостов, инициировавших соединение. Формат имени файла регистрации сеанса – SESSION:sourceport-destport, где sourceport и destport – номера портов отправителя и получателя для данного сеанса соответственно.

Формат:

```
session: [printable/all]
```

Пример правила:

```
log tcp any any <> 192.168.5.0/24 21 (session: printable;)
```

Если предположить, что IP-адресом хоста-отправителя является `1.2.3.4`, а порт отправителя 1025, то следующий отчет будет сохранен в каталоге `log` в подкаталоге `1.2.3.4` в файле `SESSION`.

Пример отчета:

```
220 linux2 FTP server (Version wu-2.5.0(1) Tue Sep 21 16:48:12 EDT 1999)
ready.
USER jsmith
331 Password requied for jsmith.
PASS snorty-the-pig
230 User jsmith logged in.
SYST
215 UNIX Type: L8
QUIT
221-You have transferred 0 bytes in 0 files.
221-Total traffic for this session was 239 bytes in 0 transfers.
221-Thank you for using the FTP service on linux2.
221 Goodbye
```

Параметр resp

Параметр `resp` позволяет организовать автоматическое ответное действие при обнаружении вредоносной деятельности нарушителя. При обнаружении атаки прежде всего предпринимается попытка оборвать текущее соединение. В правиле можно задать различные комбинации ответных действий, которые назначаются с использованием разных опций параметра `resp`.

Для разрыва TCP-соединений на сокет хоста-отправителя, хоста-получателя или сокеты обоих хостов может быть отправлен пакет с установленным флагом RESET. Если атака осуществляется с помощью UDP-пакетов, то для разрыва соединения посылаются различные ICMP-сообщения (о недостижимости сети, хоста, порта или комбинация этих ICMP-сообщений).

Параметр `resp` не входит в состав автоматически устанавливаемых. Чтобы его использовать, следует задать конфигурацию системы Snort с помощью следующей команды.

```
./configure -enable-flexresp
```

Она позволяет добавить необходимый программный код для последующей компиляции. Возможно, что в состав вашей версии UNIX не входит файл `libnet.h`, который требуется для выполнения компиляции. Этот файл можно загрузить с сайта `www.packetfactory.net`.

Совершенно ясно, что ответное действие выполняется после выдачи нескольких предупреждений. Во-первых, следует хорошенько подумать перед тем, как безрассудно использовать ответное действие. Его следует применять в ситуациях, когда весьма вероятным является причинение серьезного ущерба, например, при атаках на переполнение буфера. При этом не забывайте, что хакеры могут подменять IP-адреса отправителя, и в результате ответное действие может оказаться направленным против ни в чем не повинного пользователя (или пользователей), который никогда не отправлял вам никаких пакетов. Представьте себе последствия ответных действий, если кто-то проведет атаку, использовав IP-адреса хостов вашего партнера. В этом случае

вы организуете атаку сами против себя. Кроме того, может возникнуть ложная тревога, и в обслуживании будет отказано обычным пользователям.

Еще одна проблема заключается в согласовании по времени. Часто обмен запросами и ответами осуществляется практически мгновенно, особенно запросами и ответами службы DNS по протоколу UDP. Попытка отреагировать на полученный вредоносный DNS-запрос может оказаться бесполезной, так как за время реакции Snort ответ уже может быть отправлен.

Формат:

```
resp <resp-параметр[, resp-параметр...]>;
```

Доступными вариантами ответов являются:

- `rst_snd` — отправить TCP-пакет с установленным флагом RESET на сокет хоста-отправителя;
- `rst_rcv` — отправить TCP-пакет с установленным флагом RESET на сокет получателя;
- `rst_all` — отправить TCP-пакеты с установленным флагом RESET на сокеты хоста-отправителя и получателя;
- `icmp_net` — вернуть отправителю ICMP-сообщение о недостижимости сети (`network unreachable`);
- `icmp_host` — вернуть отправителю ICMP-сообщение о недостижимости хоста (`host unreachable`);
- `icmp_port` — вернуть отправителю ICMP-сообщение о недостижимости порта (`port unreachable`);
- `icmp_all` — вернуть отправителю все три перечисленных выше ICMP-сообщения.

Пример правила:

```
alert tcp any any -> $HOME_NET 21
(msg: "Получение файла паролей FTP";
flags: A+; resp: rst_all; content: "passwd";)
```

Пример отчета о сеансе:

```
[rootiaverbo hping2-beta53]# ftp sparky
Connected to sparky.
220 sparky FTP server (SunOS 5.7) ready.
Name (sparky:root): jsmith
331 Password required for jsmith.
Password:
230 User jsmith logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd /etc
250 CWD command successful.
ftp> get passwd
local: passwd remote: passwd
200 PORT command successful.
421 Service not available, remote server has closed connection
```

В предыдущем правиле было задано ответное действие при попытке подключиться к FTP-серверу и получить доступ к файлу паролей `passwd`. Snort пытается

оборвать соединение на обоих концах, так как была выбрана опция `rst_all` для параметра `resp`.

Обратите внимание на последнюю строку FTP-сеанса. Сразу же после того, как нарушитель ввел команду **`get passwd`**, соединение было разорвано. Но, к сожалению, есть вероятность, что файл паролей был отправлен еще до разрыва соединения.

Параметр tag

При использовании параметра `tag` система Snort регистрирует все пакеты, поступающие после "срабатывания" правила. Если не указывать этот параметр, сохраняется только пакет, непосредственно соответствующий правилу. Таким образом, можно увидеть, какие события происходят после получения опасного пакета, что поможет узнать, была тревога ложной или нет.

Формат:

`tag: <`*тип*`>, <`*количество*`>, <`*показатель*`>, [`*направление*`]`

- **Тип.** Указывает, пакеты какого типа следует регистрировать.
  - `session` — сохранять все пакеты, которыми обмениваются участники соединения;
  - `host` — сохранять только те пакеты, которые поступают от хоста — отправителя пакета, вызвавшего "срабатывание" правила (необходимо использовать модификатор *направление*).
- **Количество.** Позволяет определить количество единиц, заданных с помощью модификатора *показатель*.
- **Показатель**. Определяет, в каких единицах число указано в модификаторе *количество*.
  - `packets` — сохранить <*x*> пакетов сеанса или отправленных с атакующего хоста;
  - `seconds` — сохранить пакеты за <*x*> секунд после "срабатывания" правила для данного сеанса или отправленных с атакующего хоста;
- **Направление**. Используется только для типа `host` и указывает, по какому IP-адресу (отправителя или получателя) следует выбирать регистрируемые пакеты.
  - `scr` — сохранить все пакеты с IP-адресом отправителя, совпадающим с IP-адресом отправителя в пакете, получение которого привело к "срабатыванию" правила;
  - `dst` — сохранить все пакеты с IP-адресом получателя, совпадающим с IP-адресом получателя в пакете, получение которого привело к "срабатыванию" правила.

Пример правила:

```
alert tcp any any -> any 21 (msg: "FTP-доступ к файлу passwd"; flags: A+; \
content: "passwd"; tag: session, 10, packets;)
```

Пример отчета.

В файле регистрации предупреждений сохранены сокращенные данные о потенциально опасном соединении через порт 21.

```
[**] FTP-доступ к файлу passwd [**]
03/21-20:31:05.610035 10.10.10.101:1454 -> 10.10.10.100:21
TCP TTL:128 TOS:0x0 ID:50697 IpLen:20 DgmLen:58 DF
***AP*.* Seq: 0x17806739  Ack: 0x121C07E5  Win: 0x1FD3  TcpLen: 20
```

Создается каталог по имени `10.10.10.101` с файлом `TCP:1454-21`, в котором сохраняются отчеты о всех операциях, выполненных во время сеанса с момента попытки получения файла паролей (еще 10 последующих записей). Обратите внимание на то, что, указав при запуске Snort в командной строке параметр `-d`, можно сохранить все полезные данные передаваемых пакетов. Ниже приведен фрагмент такого отчета.

```
03/21-20:31:05.610035 10.10.10.101:1454 -> 10.10.10.100:21
TCP TTL:128 TOS:0x0 10:50697 IpLen:20 DgmLen:58 DF
***AP*** Seq: 0x17806739  Ack: 0x121C07E5  Win: 0x1FD3  TcpLen: 20
52 45 54 52 20 2F 65 74 63 2F 70 61 73 73 77 64  RETR /etc/passwd
0D 0A                                            ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

03/21-20:31:05.610731 10.10.10.100:21 -> 10.10.10.101:1454
TCP TTL:64 TOS:0x10 ID:1752 IpLen:20 DgmLen:109 DF
***AP*** Seq: 0x121C07E5  Ack: 0x1780674B  Win: 0x7D78  TcpLen: 20
31 35 30 20 4F 70 65 6E 69 6E 67 20 41 53 43 49  150 Opening ASCI
49 20 6D 6F 64 65 20 64 61 74 61 20 63 6F 6E 6E  I node data conn
65 63 74 69 6F 6E 20 66 6F 72 20 2F 65 74 63 2F  ection for /etc/
70 61 73 73 77 64 20 28 36 37 39 20 62 79 74 65  passwd (679 byte
73 29 2E 0D 0A                                   s)...
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
<несколько пропущенных записей>
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
03/21-20:31:08.924038 10.10.10.101:1454 -> 10.10.10.100:21
TCP TTL:128 TOS:0x0 ID:52489 IpLen:20 DgmLen:58 DF
***AP*** Seq: 0x17806764  Ack: 0x121C0860  Win: 0x1F58  TcpLen: 20
52 45 54 52 20 2F 65 74 63 2F 73 68 61 64 6F 77  RETR /etc/shadow
00 0A                                            ..
```

Создание нового правила

После того как были рассмотрены многочисленные правила Snort, может возникнуть вопрос, а как написать правило для блокирования новой программы атаки? Весьма вероятно, что многочисленные пользователи/разработчики Snort быстро выпустят новое правило. Но, предположим, что вы столкнулись с программой атаки, для которой еще не существует правила Snort.

Первым делом следует запустить программу атаки в изолированной сети тестирования, например у себя дома или в изолированной лаборатории. Если программа атаки работает согласно ее описанию, сохраните отчеты о передаче пакетов между атакующим и атакуемым хостами. Затем постарайтесь найти уникальные и повторяющиеся значения полей пакетов, которые можно будет использовать для создания правила. Возможно, придется прочесть несколько документов RFC, чтобы ближе познакомиться с протоколом, используемым для этой атаки, и запомнить, какие значения полей могут повторяться, а какие должны изменяться.

Предположим, что у нас есть программа атаки, использующая ошибку переполнения буфера записей `TSIG` (transaction signature) базы данных DNS-сервера. Это реаль-

ная атака, которая была эффективна против пакетов BIND начиная с версии 4.x (за исключением версии 8.2.3), для которых не была установлена специальная заплата. Запись TSIG является одной из записей о ресурсах наподобие адресной записи (A) или записи указателя доменного имени (PRT). Она используется распознавателями и при динамическом обновлении информации базы данных DNS-сервера с целью поддержания целостности передаваемой DNS-записи с помощью одностороннего хэширования и совместно используемого секретного ключа.

Программа атаки предназначена для получения доступа к командному интерпретатору на привилегированном уровне, на котором запущена BIND (демон named). Исходя из этого и следует создавать сигнатуру для обнаружения вредоносных пакетов. Ниже представлен пакет, в котором организовано переполнение буфера BIND и затем следует попытка получить доступ к командному интерпретатору.

```
02/22-15:33:19.472301 ATTACKER:1024 -> VICTIM:53

UDP TTL:64 TOS:0x0 ID:6755 IpLen:20 DgmLen:538
Len: 518

DE AD 01 80 00 07 00 00 00 00 00 01 3F 00 01 02 ............?...
03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 ................
13 14 15 16 17 18 19 1A 1B 1C 10 1E 1F 20 21 22 ............. !"
23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 #$%&'()*+,-./012
33 34 35 36 37 38 39 3A 3B 3C EB 0A 02 00 00 C0 3456789:;<......
00 00 00 00 00 3F 00 01 EB 44 5E 29 C0 89 46 10 .....?...DA)..F.
40 89 C3 89 46 0C 40 89 46 08 8D 4E 08 B0 66 CD @...F.@.F..N.,f.
80 43 C6 46 10 10 66 89 5E 14 88 46 08 29 C0 89 .C.F..f.^..F.)..
C2 89 46 18 B0 90 66 89 46 16 8D 4E 14 89 4E 0C ..F...f.F..N..N.
8D 4E 08 EB 07 C0 00 00 00 00 00 3F EB 02 EB 43 .N.........?...C
B0 66 CD 80 89 5E 0C 43 43 B0 66 CD 80 89 56 0C .f...^.CC.f...V.
89 56 10 B0 66 43 CD 80 86 C3 B0 3F 29 C9 CD 80 .V..fC.....?)...
B0 3F 41 CD 80 B0 3F 41 CD 80 88 56 07 89 76 0C .?A...?A...V..v.
87 F3 8D 4B 0C B0 0B CD 80 EB 07 C0 00 00 00 00 ...K............
00 3F 90 E8 72 FF FF FF 2F 62 69 6E 2F 73 68 00 .?..r.../bin/sh.
0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D ................
1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D .. !"#$%&'()*+,-
2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C EB ./0123456789:;<.
07 C0 00 00 00 00 00 3F 00 01 02 03 04 05 06 07 .......?........
08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 ................
18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 ........ !"#$%&'
28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 ()*+,-./01234567
38 39 3A 3B 3C EB 07 C0 00 00 00 00 00 3F 00 01 89:;<........?..
02 03 04 05 06 07 08 09 0A 0B 0C 00 0E 0F 10 11 ................
D8 FA FF BF D8 F7 FF BF D0 7C 0D 08 04 F7 10 40 .........|.....@
22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 "#$%&'()*+,-./01
32 33 34 35 36 37 38 39 3A 3B 3C EB 07 C0 00 00 23456789:;<.....
00 00 00 3F 00 01 02 03 04 05 06 07 08 09 0A 0B ...?............
0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B ................
1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B .... !"#$%&'()*+
2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B ,-./0123456789:;
3C EB 07 C0 00 00 00 00 00 00 90 FA 00 FF <.............
```

Первой очевидной сигнатурой атаки является строка **/bin/sh**, с помощью которой предпринимается попытка получить доступ к командному интерпретатору после успешного переполнения буфера. Другой сигнатурой для этого отчета является определенная проверка использования записи TSIG для DNS-сервера.

Тип DNS-записи хранится в 2-байтовом поле, и TSIG-записи соответствует значение 250 (0x00FA). Кроме того, каждому типу записи о ресурсах соответствует свой класс записи, который также хранится в 2-байтовом поле. Для TSIG-записи в поле класса используется значение 255 (0x00FF), которое соответствует любому классу. Таким образом, в полезных данных DNS для обозначения TSIG-записи должна присутствовать строка (0x00FA00FF). Обычный TSIG-запрос не содержит строки /bin/sh, поэтому одновременный поиск двух указанных значений позволит выявить вредоносные действия без вызова ложных тревог. Хотя в этом конкретном пакете для создания правила могут быть использованы и другие значения, но злоумышленник может изменить исходный код программы атаки таким образом, что, несмотря на изменение DNS-заголовка и TSIG-записи, программа атаки все равно будет работать. Следующее правило позволяет выявить описанную атаку.

```
alert udp $EXTERNAL_NET any -> $HOME_NET 53  \
(msg: "tsig-атака на BIND посредством переполнения буфера";  \
content: "|00 FA 00 FF|"; offset: 12;          \
content: "/bin/*sh"; regex; offset: 12;)
```

В рассмотренном примере выявляются пакеты, передаваемые посредством протокола UDP, которые поступают от удаленного пользователя на порт получателя 53. Два отдельных параметра content позволяют выявлять в пакетах искомые строки. Параметр regex применяется на случай, если будет предпринята попытка доступа к командному интерпретатору, отличному от Bourne. Параметр regex стал полнофункциональным в текущей версии Snort и имеет ограниченный набор возможностей в версии Snort 1.8.3 (недоступно использование универсального символа *).

Кроме того, при поиске строк используется параметр offset со значением 12 для указания того, что поиск должен начинаться с 12-го байта от начала DNS-сообщения. Это позволяет повысить эффективность поиска и его точность. Первые 12 байт DNS-сообщения занимает DNS-заголовок, а наш поиск должен выполняться в полезных данных.

Программа атаки, использующая TSIG-записи

Более подробную информацию о TSIG можно получить в RFC 2845, озаглавленном "Secret Key Transaction Authentication for DNS (TSIG)" ("Аутентификация для DNS-службы с помощью передачи секретного ключа (TSIG)"). Об описанной атаке можно прочитать на сайте www.cert.org, ссылка CA-2001-02. Еще одно полное исследование этой атаки провел Пол Асадориан (Paul Asadoorian). Его статью можно найти по адресу www.sans.ofg/newlook/resourses/INFAQ/TSIG.htm. Большое спасибо, Пол, за совместное обсуждение правил Snort и отчетов об атаках.

Резюме

Параметры правил Snort предоставляют широкий диапазон атрибутов и способов определения значений, которые могут быть проверены в пакете. Использование параметров не представляет особой сложности. Достаточно небольшой практики их применения и краткого знакомства с документацией Snort. С выходом каждой новой версии Snort добавляются новые параметры, что обеспечивает функциональное богатство правил Snort и их равенство или превосходство по сравнению с сигнатурами других коммерческих систем обнаружения вторжений.

Для создания правила Snort, выявляющего действия определенной программы атаки, следует запустить эту программу в изолированной среде и с помощью Snort или TCPdump целиком сохранить пакеты, передаваемые от организатора атаки к ее цели. Используйте любые доступные поля заголовков правил Snort или параметры этих правил для четкой идентификации уникальных значений в пакетах атаки. Не забывайте, что исходный код программы атаки может быть изменен, поэтому при создании правила постарайтесь найти такие значения или поля, изменить которые будет сложно. Выбор нужного поля или значения представляет собой сложную задачу, может считаться своеобразным искусством и требует знаний о языке написания сигнатур, атаке и протоколе, используемом при атаке.

IV

Технология взлома

В этой части...

15

Атака Митника

В последней части книги мы рассмотрим обычные и автоматические методы ответных действий, а также архитектурные и организационные вопросы. Эта глава, посвященная атаке Митника, может считаться переходным звеном между рассмотренным выше базовым материалом и знаниями более высокого уровня. Атака Митника (Mitnick) является одной из самых знаменитых. Любой специалист в области компьютерной безопасности должен знать о методе, использованном Митником для взлома систем Цутому Шимомуры (Tsutomu Shimomura). В этой главе также рассматриваются такие важные проблемы, как разведывательные действия и сканирование с целью обнаружения доверительных отношений. Мы обсудим защиту от взлома на периметре локальной сети и на отдельном хосте.

Основным источником изложенного материала являются сообщения самого Шимомуры об атаке Митника. Более подробную информацию по этой теме можно получить, написав письмо по адресу `tsutomu@ariel.dssc.edu` (Tsutomu Shimomura).

Использование недостатков TCP

Для проведения атаки Митник использовал недостатки протокола TCP, которые были хорошо известны в научных кругах, но никак не учитывались разработчиками систем. Использовалось два типа атак: наводнение с помощью SYN-пакетов и перехват TCP-сеанса. Хотя и в наше время наводнение SYN-пакетами способно вывести систему из строя, операционные системы, использовавшиеся во время атаки в 1994 году, были значительно более уязвимыми. С помощью потока SYN-пакетов блокировалась передача данных от одной из систем. Благодаря этому нарушитель смог подменить одну из сторон соединения и перехватить TCP-сеанс. Митник обнаружил установленные доверительные отношения между двумя компьютерами и воспользовался ими в своих целях. Как ни странно, но с того времени почти ничего не изменилось, например, между компьютерными системами по-прежнему используются практически не контролируемые довери-

ные отношения. Чаще всего это делается для удобства системных администраторов или пользователей.

Недостатки стека TCP/IP

Большинство программ для проведения разведки, атаки и отказа в обслуживании используют недостатки в архитектуре или реализации стеков протоколов Internet. В главе 4, "Протокол ICMP", мы рассматривали использование широковещательных запросов ICMP для проведения сканирования сети и отказа в обслуживании с помощью атаки Smurf. В главе 3, "Фрагментация пакетов", обсуждались методы проникновения в защищенный периметр с помощью фрагментированных пакетов, а также применение фрагментации с вредоносными целями.

В некоторых атаках используются старые, хорошо известные методы, а новые, постоянно создаваемые программы, основаны на ошибках программирования при реализации стеков IP. Следующий отчет TCPdump содержит запись о пакете, созданном средством тестирования PROTOS, которое использует ошибку в реализации SNMP.

```
18:49:54.519006 10.0.0.1.59108 > 10.0.0.2.161: GetRequest(33)
.1.3.6.1.2.1.1.5.0[len3<asnlen4294967295] (DF)
0x0000 4500 004c 0000 4000 4011 269f 0a00 0001
0x0010 0a00 0002 e6e4 00a1 0038 0efc 302e 0201
0x0020 0004 0670 7562 6c69 63a0 2102 0206 9202
0x0030 0100 0201 0030 1530 1306 082b 0601 0201
0x0040 0105 0044 84ff ffff ff02 0100
```

Когда мы впервые использовали этот тест против компьютера, работающего под управлением Red Hat Linux 7.0, произошел аварийный сбой в работе серверного приложения SNMP и прекратил работу анализатор пакетов `ethereal`. Почему это случилось? Обратите внимание на длину пакета, указанную согласно формату ASN.1 в квадратных скобках в начале отчета. Как видите, передается 4 миллиарда байт какой-то информации. Попытка выделения памяти для приложений SNMP и `ethereal` приводит к аварийному завершению работы этих приложений. Наличие доступной памяти является основной проблемой и для атаки Митника.

Проще всего организовать растрату всех ресурсов памяти ("утечку памяти") с помощью незавершенной процедуры установки соединения. Используемые Митником недостатки присутствовали в первых реализациях стеков TCP/IP, но даже сейчас подобные атаки иногда могут иметь успех.

История TCP/IP

Когда разрабатывался протокол TCP, купить большой объем оперативной памяти для компьютера было невозможно. Оперативная память размером в 4 Мбайт считалась хорошим показателем для сервера. Поэтому создатели стеков протоколов IP были крайне сдержаны по отношению к требуемым ресурсам.

Сеть Internet является развитием проекта, осуществлявшегося в 1970-х годах Министерством обороны США, и в частности ARPA (Advanced Research Projects Agency — агентство перспективных исследований и разработок). Первоначально он носил название ARPANET и представлял собой крайне ненадежную службу обмена информацией между удаленными компьютерами. В 1973–1974 годах в результате усилий различных исследовательских и образовательных организаций был разработан стандартный набор протоколов для обмена информацией в сети. Этот набор протоколов стали называть *стеком протоколов TCP/IP* или просто *стеком IP*. Протоколы из набора TCP/IP по-

зволили компьютерам сети ARPANET взаимодействовать независимо от их операционных систем или аппаратных средств.

Более подробную информацию по данной теме можно получить по адресу www.ie.cuhk.edu.hk/~shlam/cstdi/hystory.html.

Давайте внимательней рассмотрим проблему утечки памяти. Для приложений наподобие FTP или `telnet` сокеты являются самым низким уровнем — программным интерфейсом для доступа к сетевым аппаратным средствам. Протокол IP представляет собой другой уровень и работает поверх сокетов. TCP работает поверх IP. Так как протокол TCP ориентирован на установку соединений, он должен хранить сведения о состоянии сеанса, включая размер окна и порядковый номер пакета.

Представленный ниже код на языке C взят с моей рабочей станции UNIX. Его можно считать записью в базе данных с большим количеством полей. Хранение данных каждого из этих полей требует ресурсов оперативной памяти.

```
struct ip {
#if defined(bsd)
                u_char  ip_hl:4,     /* длина заголовка */
                ip_v:4;              /* версия */
#endif
#if defined(powerpc)
                u_char ip_v:4,       /* версия */
                ip_hl:4;             /* длина заголовка */
#endif
        u_char  ip_tos;              /* тип обслуживания */
        short   ip_len;              /* общая длина */
        u_short ip_id;               /* идентификатор */
        short   ip_off;              /* поле смещения фрагмента */
#define IP_DF 0x3000                /* флаг DF */
#define IP_MF 0x4000                /* флаг MF */
                u_char  ip_ttl;      /* время жизни */
                u_char  p;           /* протокол */
                u_short ip_sum;      /* контрольная сумма */
                struct  in_addr ip_src, ip_dst; /* адреса отправителя и
получателя */
};
```

Приведенный выше фрагмент является фрагментом файла IP-заголовка для системы SunOS 4.1.3. Его структуру (в данном случае `struct ip`) можно считать записью базы данных, а внутренние элементы — полями этой записи. Каждый раз при создании нового соединения необходимо создавать такие структуры для сокета, протокола IP и других протоколов. На все это требуются ресурсы памяти. После того как сервер отвечает на SYN-пакет, область использованной памяти фиксируется и должна оставаться без изменений до истечения срока действия таймера (обычно около 60 секунд). Если за это время соединение так и не будет установлено, то память освобождается. Поскольку объем оперативной памяти не бесконечен, создателям стеков пришлось предусмотреть определенные ограничения. При атаке с помощью наводнения SYN-пакетами используется проблема установленного предела на размер очереди одновременно ожидаемых соединений для определенной службы. Хотя некоторые современные операционные системы не так подвержены подобным атакам, еще для многих платформ проблема остается актуальной. Компьютер, на котором установлена оперативная память размером 1 Гбайт, и который работает под управлением Solaris 2.5 без установленной заплаты, по-прежнему не сможет нормально работать после практически одновременного получения 32 SYN-пакетов.

SYN-наводнение

В современных атаках SYN-наводнение заключается в простой отправке на сервер сотен или тысяч пакетов в секунду. Это приводит к растрате или ресурсов сервера, или даже сетевых ресурсов при достаточной интенсивности трафика.

При организации наводнения SYN-пакетами злоумышленник не старается провести полную процедуру установки TCP-соединения. Его целью является превысить допустимый предел числа одновременно устанавливаемых соединений для определенной службы. В 1994 году стеки TCP/IP были не в состоянии установить никаких новых соединений для атакованной службы до тех пор, пока число запросов на соединение не станет меньше установленного предела. До момента превышения установленного предела сервер на каждый получаемый SYN-пакет отвечает пакетом с установленными флагами SYN и ACK. Запросы остаются в очереди (которая обычно составляет от 5 до 10 одновременно устанавливаемых соединений). В современных стеках очереди стали больше, и максимальное количество одновременно устанавливаемых соединений составляет от 100 до 1000.

SYN-наводнения пять лет спустя

В феврале 2000 года SYN-наводнение было в заголовках всех новостей, когда знаменитая распределенная атака отказа в обслуживании была использована против Yahoo! и других популярных Internet-сайтов. За годы, прошедшие со времени атаки Митника, в стеки сетевых протоколов были внесены значительные усовершенствования и была улучшена защита по периметру. Ответ хакеров был очень простым — количество SYN-пакетов увеличилось на несколько порядков. Описанное в нашей книге SYN-наводнение проводится довольно элегантно, а многие современные наводнения в Internet являются примерами грубой силы.

Для каждого соединения запускается таймер, устанавливающий предел времени, в течение которого система ожидает установки соединения. Песочные часы на рис. 15.1 символизируют таймер, который устанавливается приблизительно на одну минуту. По истечению этого срока освобождается память, выделенная для хранения информации о соединении, и очередь для службы уменьшается на единицу. После достижения предельного количества одновременно устанавливаемых соединений для того, чтобы очередь оставалась заполненной, и система не могла установить новых соединений, на порт должны приходить около 10 новых SYN-пакетов в минуту.

Сокрытие следов

Поскольку единственная цель данной атаки заключается в обеспечении условий отказа в обслуживании, то злоумышленнику нет смысла использовать свой настоящий IP-адрес. TCP-соединение не устанавливается, а только заполняется очередь. Злоумышленнику совсем не нужны ответные пакеты с установленными флагами SYN и ACK, и он уж точно не хочет быть обнаруженным. Поэтому IP-адрес отправителя, как правило, подменяется. Следующий IP-заголовок взят из реального кода атаки для SYN-наводнения. Обратите внимание на адреса dadd и sadd, использующиеся соответственно в качестве адресов получателя и отправителя.

```
/* Введите информацию IP-заголовка */
   packet.ip.version =4;              /* Версия. 4 бит */
   packet.ip.ihl=5;                   /* Длина заголовка. 4 бит */
   packet.ip.tos=0;                   /* Тип обслуживания. 8 бит */
   packet.ip.tot_len =htons(40);      /* Общая длина. 16 бит */
   packet.ip.id=getpid();             /* Идентификатор. 16 бит */
```

```
packet.ip.frag_off=0;              /* Смещение фрагмента. 13 бит */
packet.ip.ttl=255;                 /* Поле TTL. 8 бит */
packet.ip.protocol=IPPROTO_TCP;    /* Протокол. 8 бит */
packet.ip.check=0;                 /* Контрольная сумма заголовка */
packet.ip.saddr=sadd;              /* IP-адрес отправителя */
packet.ip.daddr=dadd;              /* IP-адрес получателя */
```

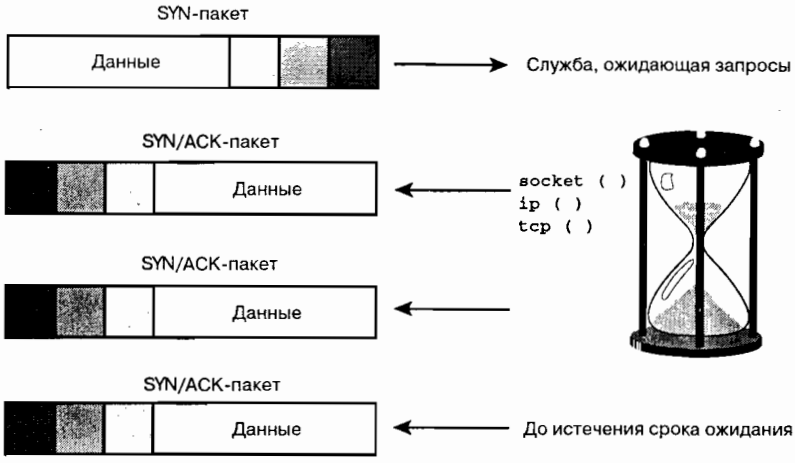


Рис. 15.1. Начало атаки

В этом методе применяется даже подпрограмма исправления ошибок, как мы увидим в следующем фрагменте кода. С ее помощью гарантируется, что выбранный для подмены адрес является маршрутизируемым, но принадлежит неактивному хосту. Когда нарушитель вводит адрес, программа атаки тестирует его (обратите внимание на строку `slickping`). Ведь если адрес принадлежит активному хосту, то в ответ на получение пакета с установленными флагами SYN и ACK будет возвращен пакет с флагом RESET. Атакуемая система, получив этот пакет (с флагом RESET), освобождает память и уменьшает число ожидаемых соединений в очереди, т.е. атака становится неэффективной. Итак, хакерам нужны специальные пакеты с подложными IP-адресами, которые будут использоваться при той или иной атаке. Очень часто разработчики программного обеспечения, позволяющего автоматическое создание пакетов для атаки, допускают небольшие погрешности или упрощают себе задачу, благодаря чему эти пакеты получают уникальный признак, или сигнатуру. Эта сигнатура позволяет выявить подобные пакеты в трафике. При обнаружении специально подготовленного пакета можно не сомневаться в последующей атаке.

```
case 3:                 if(!optflags[1]){
                                fprintf(stderr,"Сначала введите адрес
                                хоста\n");
                                usleep(MENUSLEEP);
                                break;
                        }
     /* ICMP-сокет для доступа к протоколам нижнего уровня */
if((sock2=socket(AF_INET,SOCK_RAW,IPPROTO_ICMP))<0){
                                perror("\nHmmm.... проблема с сокетом\n");
                                exit(1);
                        }
                        printf("[число ICMP_ECHO запросов]-> ");
```

```
        fgets(tmp,MENUBUF,stdin);
        if (! (icmpAmt=atoi(tmp)))break;
        if(slickPing(icmpAmt,sock2,unreach)){
        fprintf(stderr,"Хост недостижим...Выберите другой\n");
      sleep(1);
```

Этот метод может использоваться в атаках отказа в обслуживании. Атакуемая система бомбардируется SYN-пакетами до тех пор, пока не потеряет возможность устанавливать новые соединения. В этом состоянии хакер может держать систему до тех пор, пока он не даст команду на прекращение отправки SYN-пакетов. Цель атаки Митника заключалась в блокировании одной стороны TCP-соединения, чтобы переключить это соединение с доверительными отношениями на свой хост.

Выявление доверительных отношений

Как Митник узнал, какую систему нужно заблокировать? Как он убедился в существовании доверительных отношений между двумя хостами? Многим тщательно подготовленным атакам предшествует предварительный сбор информации, или *разведывательное зондирование* (recon probes). Рассмотрим пример зондирования, приведенного в сообщении Цитому, которое было зарегистрировано с помощью `tcpdump` — анализатора пакетов, созданного в лаборатории Лоренса Ливермора (Lawrence Livermore) Министерства энергетики США.

"Атака с подменой IP-адреса отправителя началась примерно в 14:09:32 по тихоокеанскому времени 25.12.94. Первые пакеты поступили от `toad.com`" (информация из журналов регистрации пакетов).

```
14:09:32 toad.com# finger -l @target
14:10:21 toad.com# finger -l @server
14:10:50 toad.com# finger -l root@server
14:11:07 toad.com# finger -l @x-terminal
14:11:38 toad.com# showmount -e x-terminal
14:11:49 toad.com# rpcinfo -p x-terminal
14:12:05 toad.com# finger -l root@x-terminal
```

Каждая из приведенных команд `finger`, `showmount` и `rpcinfo` предназначена для сбора информации о системе UNIX. Если вы работаете в UNIX и давно не пользовались этими командами, то, возможно, стоит изменить имена хостов вашей сети на `target`, `server` и `x-terminal` и посмотреть, какая информация будет выдана в результате приведенных выше команд.

- Команда `finger` сообщает о пользователях, работающих в системе, а именно о том, когда пользователь вошел в систему, откуда, как долго он не выполняет никаких действий, адрес электронной почты и когда его день рождения (ладно, на счет дня рождения — шутка). Аналогичной командой для систем под управлением Microsoft Windows является команда `nbtstat`.

```
finger Example:
[root@toad /tmp]# finger @some.host.net
[some.host.net]
Login       Name            TTY         Idle     When    Where
chap  Bill Chapman x1568  pts/6     3:11 Tue 17:26  picard
chap  Bill Chapman x1568  console   8:39 Mon 14:44  :0
[root@toad /tnp]#
```

- Команда `showmount -e` предоставляет сведения о файловых системах, смонтированных с помощью NFS (Network File System — сетевая файловая систе-

ма). Особый интерес для хакера представляют файловые системы, монтируемые с неограниченными правами для чтения и записи (т.е. доступные для всех пользователей).

```
showmount Example:
[root@toad /tmp]# showmount -e some.host.net
Export list for some.host.net:
/usr       export-hosts
/usr/local export-hosts
/home      export-hosts
[root@toad /tmp]#
```

С помощью команды `rpcinfo` можно узнать о запущенных службах удаленного доступа. Команда `rpcinfo -p` выводит также список портов, которые используются каждой из поддерживаемых служб.

```
rpcinfo  Example
[root@toad /tmp]# rpcinfo -p some.host.net
   program vers proto   port
    100000    3   udp    111  rpcbind
    100000    2   udp    111  rpcbind
    100003    2   udp   2049  nfs
    100024    1   udp    774  status
    100024    1   tcp    776  status
    100021    1   tcp    782  nlockmgr
    100021    1   udp    784  nlockmgr
    100005    1   tcp   1024  mountd
    100005    1   udp   1025  mountd
    391004    1   tcp   1025
    391004    1   udp   1026
    100001    1   udp   1027  rstatd
    100001    2   udp   1027  rstatd
    100008    1   udp   1028  walld
    100002    1   udp   1029  rusersd
    100011    1   udp   1030  rquotad
    100012    1   udp   1031  sprayd
    100026    1   udp   1032  bootparam
```

В настоящее время на внешних брандмауэрах или фильтрующих маршрутизаторах большинства локальных сетей блокируются запросы к TCP-порту 79 (`finger`), но все равно стоит проверить, например, учетную запись для своего домашнего компьютера. Не забудьте *перед* проверкой получить разрешение у своего провайдера. Также лучше проверить блокирование TCP/UDP-порта 111 (`portmapper`). Не очень давно появились так называемые "защищенные службы преобразования портов" (secure portmapper). Они предоставляются различными поставщиками, а также в виде пакета, созданного Витсом Винима (Wietse Venema). Этот пакет доступен по адресу `ftp://coast.cs.purdue.edu/pub`.

Исследование сетевых трассировок

Во время атаки Митника ни один из перечисленных портов заблокирован не был. Поэтому хакер (`toad.com`) получил информацию, которой он воспользовался на следующем этапе своей атаки. Вот цитата из отчета Цутому:

> "От `apollo.it.luc.edu` поступило двадцать запросов на установление соединения с `x-terminal.shell`. Цель этих попыток заключалась в определении принципа работы генератора порядковых номеров TCP-соединений хоста x-

terminal. Обратите внимание на то, что в полученных SYN-пакетах начальные порядковые номера увеличиваются на 1 для каждого нового соединения. Это означает, что они не были сгенерированы системным стеком TCP/IP. В ответ на каждый пакет с флагами SYN и ACK отправляется RST-пакет, и в результате очередь соединений на x-terminal не заполняется".

При анализе следующей трассировки tcpdump обратите внимание на то, что пакеты сгруппированы по три: SYN-пакет от apollo к x-terminal, SYN/ACK-пакет (второй этап процедуры установки TCP-соединения) и пакет с флагом RESET для предотвращения наводнения SYN-пакетами хоста x-terminal.

Как читать трассировки TCPdump

```
Временная метка  хост-отпр.порт-отпр.  > хост-получ.порт-получ: Флаги TCP
14:18:25.906002 apollo.it.luc.edu.1000 > x-terminal.shell: S
↳ Порядк_номер:номер_подтв Размер окна
↳ 1382726990:1382726990(0)  win 4096
```

В следующих трассировках показано начало наводнения SYN-пакетами хоста x-terminal. Символы +++ добавлены для обозначения группы из трех пакетов.

```
+++
14:18:25.906002 apollo.it.luc.edu.1000 > x-terminal.shell: S
↳ 1382726990:1382726990(0) win 4096

14:18:26.094731 x-terminal.shell > apollo.it.luc.edu.1000: S
↳ 2021824000:2021824000(0) ack 1382726991 win 4096

14:18:26.172394 apollo.it.luc.edu.1000 > x-terminal.shell: R
↳ 1382726991:1382726991(0) win 0
+++

+++
14:18:26.507560 apollo.it.luc.edu.999 > x-terminal.shell: S
↳ 1382726991:1382726991(0) win 4096

14:18:26.694691 x-terminal.shell > apollo.it.luc.edu.999: S
↳ 2021952000:2021952000(0) ack 1382726992 win 4096

14:18:26.775037 apollo.it.luc.edu.999 > x-terminal.shell: R
↳ 1382726992:1382726992(0) win 0
+++
```

Обратите внимание на значения, выделенные жирным шрифтом. Во второй группе пакетов это порядковый номер SYN/ACK-пакета от x-terminal, который равен 2021952000. В предыдущей группе пакетов аналогичный порядковый номер имел значение 2021824000. Если из 2021952000 вычесть 2021824000, получится 128000. Означает ли что-то это число? Посмотрим, будет ли оно повторяться для следующих групп пакетов.

```
14:18:27.014050 apollo.it.luc.edu.998 > x-terminal.shell: S
↳ 1382726992:1382726992(0) win 4096

14:18:27.174846 x-terminal.shell > apollo.it.luc.edu.998: S
↳ 2022080000:2022080000(0) ack 1382726993 win 4096

14:18:27.251840 apollo.it.luc.edu.998 > x-terminal.shell: R
```

```
↳ 1382726993:1382726993(0) win 0

14:18:27.544069 apollo.it.luc.edu.997 > x-terminal.shell: S
1382726993:1382726993(0) win 4096

14:18:27.714932 x-terminal.shell > apollo.it.luc.edu.997: S
2022208000:2022208000(0) ack 1382726994 win 4096

14:18:27.794456 apollo.it.luc.edu.997 > x-terminal.shell: R
↳ 1382726994:1382726994(0) win 0
```

И снова 2022208000 – 2022080000 = 128000. Таким образом, это число повторяется, т.е. значение порядкового номера является *предсказуемым*. Теперь известно, что при отправке SYN-пакета на хост `x-terminal` в ответном SYN/ACK-пакете значение порядкового номера будет больше на 128000 (если это соединение будет следующим). Если есть возможность заблокировать одну сторону TCP-соединения, используются доверительные отношения и можно спрогнозировать порядковый номер пакета, значит, все готово для перехвата этого соединения (рис. 15.2).

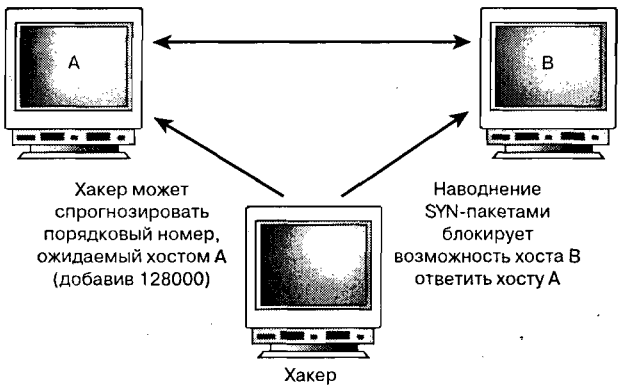


Рис. 15.2. Готовность к атаке

Взлом системы

Почему стала возможной эта атака с использованием доверительных отношений? Безусловно, любой компьютер обнаружил бы некорректный IP-адрес хоста хакера. Однако IP-адрес был подменен, поэтому ничего не обнаруживается. Значение поля TTL может отличаться, но это значение используется на уровне протокола IP, а все действия после установки соединения выполняются на уровне TCP. Изменяется маршрут доставки пакетов, но без специальных программ это определить невозможно, так как в протоколе IP не предусмотрена запись информации о маршруте передачи сообщения. Поэтому все упирается в порядковые номера. Если хакер отправит пакет с неправильным порядковым номером, то получатель ответит RST-пакетом и разорвет соединение. Именно поэтому в атаке Митника такое большое значение имеет возможность предсказать порядковый номер. Теперь хакер может заблокировать одну сторону соединения (сервер), перехватить соединение и убедить другую сторону (`x-terminal`) в том, что его хост и есть сервер. Что дальше? Вернемся к рассказу Цутому:

"Мы получили подложный SYN-пакет (запрос на соединение) якобы от `server.login` к `x-terminal.shell`. Предполагалось, что между этими двумя компьютерами установлены доверительные отношения, поэтому `x-terminal` выполнял все, что требовал `server` (или тот, кто выдавал себя за `server`). Хост ответил на запрос пакетом с флагами ACK и SYN. Так как заблокированный сервер не отвечал на пакеты, отправленные к `server.login`, то хакеру пришлось также подменить следующее подтверждение".

При стандартной установке соединения порядковый номер в SYN/ACK-пакете необходим для генерации соответствующего номера в пакете с флагом ACK. Но хакер знает о принципе назначения порядковых номеров на хосте `x-terminal.shell`, поэтому он может отправить "правильный" ACK-пакет без получения SYN/ACK-пакета.

Все это подтверждает следующая трассировка TCPdump. В первой строке хост `x-terminal.shell` получает от имени `server` запрос на установку соединения. Сервер так и не получит ответного SYN/ACK-пакета, который пропущен в трассировке. Но хакер знает, что в ACK-пакете нужно только добавить к исходному порядковому номеру число 128000 плюс 1. После доставки ACK-пакета соединение считается установленным.

```
14:18:36.245045 server.login > x-terminal.shell: S 1382727010:1382727010(0)
↳ win 4096
14:18:36.755522 server.login > x-terminal.shell: . ack 2024384001 win 4096
```

Итак, Митник использует доверительные отношения между `x-terminal` и `server`. Он отправляет SYN-пакет с подмененным IP-адресом отправителя. Этот пакет отправляется вслепую, хакер не может узнать ответ (если только в локальной сети `x-terminal` или `server` не установлен анализатор пакетов). Хост `x-terminal` отправляет ответ (SYN/ACK-пакет) действительному серверу. Сервер, который не отправлял никакого SYN-пакета, должен был бы ответить RST-пакетом и разорвать соединение. Но этого не происходит. Как показано ниже, за 14 секунд до начала основной фазы атаки очередь соединений порта `login` заполняется подготовленными SYN-пакетами. Сервер не в состоянии ответить на новый пакет.

```
14:18:22.516699 130.92.6.97.600 > server.login: S 1382726960:1382726960(0)
↳ win 4096
14:18:22.566069 130.92.6.97.601 > server.login: S 1382726961:1382726961(0)
↳ win 4096
14:18:22.744477 130.92.6.97.602 > server.login: S 1382726962:1382726962(0)
↳ win 4096
14:18:22.830111 130.92.6.97.603 > server.login: S 1382726963:1382726963(0)
↳ win 4096
14:18:22.886128 130.92.6.97.604 > server.login: S 1382726964:1382726964(0)
↳ win 4096
14:18:22.943514 130.92.6.97.605 > server.login: S 1382726965:1382726965(0)
win 4096
```

r-службы

Некоторые думают, что служба `telnet` и r-службы уже полностью заменены службой SSH (secure shell — защищенная оболочка), но это не соответствует действительности. И `telnet`, и r-службы по-прежнему используются очень широко. Служба `login` еще известна как `rlogin` и `rshell` (для доступа к командному интерпретатору). Эти службы удаленного вызова процедур созданы для "удобства" пользователей и позволяют выполнять операции на удаленном хосте без необходимости ввода пароля. На UNIX-компьютерах для создания

доверительных отношений для всех пользователей, кроме суперпользователя (root), достаточно добавить имя доверенного компьютера или учетной записи в файл /etc/hosts.equiv. Для установки доверительных отношений с привилегиями суперпользователя предназначен файл /.rhosts. Так называемые r-службы уже устарели и использовать их не следует. Служба SSH обеспечивает большую безопасность. Для файлов /etc/hosts.equiv и /.rhosts символ + (плюс) является универсальным. Например, строка /.rhosts ++ означает "доверять всем хостам и всем пользователям этих хостов".

Пока настоящий сервер заблокирован SYN-наводнением, соединение с доверительными отношениями используется для выполнения следующей команды: **rsh x-terminal "echo + + >>/.rhosts"**. Эта команда приказывает хосту x-terminal доверять всем компьютерам и всем пользователям этих компьютеров. Ниже представлена соответствующая трассировка.

```
14:18:37.265404 server.login > x-terminal.shell: P 0:2(2) ack 1 win 4096
14:18:37.775872 server.login > x-terminal.shell: P 2:7(5) ack 1 win 4096
14:18:38.287404 server.login > x-terminal.shell: P 7:32(25) ack 1 win 4096
```

В этот момент соединение разрывается хакером с помощью пакета с установленным флагом FIN. Теперь Митник может подключиться к x-terminal с любого хоста и выполнить любую команду. Атакованная система скомпрометирована.

```
14:18:41.347003 server.login > x-terminal.shell: . ack 2 win 4096
14:18:42.255978 server.login > x-terminal.shell: . ack 3 win 4096
14:18:43.165874 server.login > x-terminal.shell: F 32:32(0) ack 3 win 4096
```

Если бы Митник так и оставил компьютер по имени server в заблокированном состоянии, то какой-то другой пользователь мог бы безуспешно пытаться установить соединение, что только привлекло бы внимание к ситуации. Поэтому очередь ожидаемых соединений на server.login освобождается с помощью отправки группы RST-пакетов.

```
14:18:52.298431 130.92.6.97.600 > server.login: R 1382726960:1382726960(0)
↪ win 4096
14:18:52.363877 130.92.6.97.601 > server.login: R 1382726961:1382726961(0)
↪ win 4096
14:18:52.416916 130.92.6.97.602 > server.login: R 1382726962:1382726962(0)
↪ win 4096
14:18:52.476873 130.92.6.97.603 > server.login: R 1382726963:1382726963(0)
↪ win 4096
14:18:52.536573 130.92.6.97.604 > server.login: R 1382726964:1382726964(0)
↪ win 4096
```

Выявление атаки Митника

Как уже упоминалось, эта глава служит для двух основных целей: рассказать об атаке Митника и подготовить читателей к информации заключительного раздела. Поэтому давайте познакомимся со средствами, которые необходимы для обнаружения и реакции на подобные атаки. Атака Митника могла быть выявлена как с помощью сетевых систем обнаружения взлома, так и с помощью тех же систем для защиты конкретного хоста. Обнаружение возможно на различных этапах проведения атаки, начиная со сбора хакером разведывательной информации и заканчивая изменением файла /.rhosts, когда система уже полностью скомпрометирована. Средство обнаружения взлома представляет собой не отдельную программу, а совокупность, набор методов и инструментов. Многие производи-

тели, включая NAI и ISS, разрабатывают гибридные системы обнаружения взлома, которые могут выполнять анализ как системных журналов хоста, так и поступающих пакетов. В этой книге есть несколько примеров выявления атак с помощью брандмауэров, а также двух типов систем обнаружения взлома.

Перехват TCP-сеансов становится для хакеров все более сложной задачей, так как во многих операционных системах теперь применяется случайное назначение порядковых номеров. Тем не менее системы, созданные Microsoft, являются исключением и по-прежнему уязвимы. Потому для опытного хакера такая атака не потеряла своей привлекательности. SYN-наводнения все еще могут блокировать работу многих реализаций стека TCP/IP, хотя современные операционные системы стали намного устойчивей к этим атакам. И, конечно, даже если SYN-наводнение не позволит заблокировать одну из сторон соединения с использованием доверительных отношений, то это можно проделать с помощью другой атаки отказа в обслуживании. Несмотря на существование более безопасных служб, например, службы SSH, многие системные администраторы продолжают использовать r-утилиты. Если нет возможности обнаружить атаку Митника, то что же мы можем обнаружить? Еще раз напомним, что выявление атаки Митника является прекрасным тестом возможностей любой системы обнаружения взлома. Почему ей уделяется столько внимания? Оказывается, что почти десятилетие после появления атаки Митника перехват TCP-сеансов по-прежнему почти невозможно выявить с помощью одного средства. Хорошо лишь то, что большую часть действий атаки Митника можно выявить без особого труда. Посмотрим, как это делается.

Сетевые системы обнаружения вторжений

Сетевые системы обнаружения взлома обычно легко выявляют разведывательное зондирование, представленное в следующей трассировке. Аналитик сетевого трафика, возможно, не станет обращать внимание на отдельную попытку использования команды `finger`, но набор таких команд абсолютно точно указывает на атаку, что уж никак нельзя игнорировать. Рассмотрим некоторые способы выявления этого зондирования сетевыми системами обнаружения вторжений.

```
14:09:32 toad.com# finger -l @target
14:10:21 toad.com# finger -l @server
14:10:50 toad.com# finger -l root@server
14:11:07 toad.com# finger -l @x-terminal
14:11:38 toad.com# showmount -e x-terminal
14:11:49 toad.com# rpcinfo -p x-terminal
14:12:05 toad.com# finger -l root@x-terminal
```

Доверительные отношения

Целью зондирования Митника является обнаружение и последующее незаконное использование доверительных отношений между двумя компьютерами. Чтобы определить цели атаки, сначала необходимо провести сбор разведывательной информации. Если Митник смог получить нужные сведения со своего компьютера, то в пределах локальной сети можно сделать то же самое и даже лучше. Опытным специалистам, хорошо знающим свои сети, часто достаточно одного взгляда на трассировку, чтобы понять ее предназначение и цель исследо-

вания. Современные системы обнаружения вторжений еще не обладают подобными способностями.

Сканирование портов

Системы обнаружения вторжений обычно настраивают на выявление отдельных нарушителей, которые пытаются подключиться к нескольким портам одного хоста. Средства сканирования портов позволяют хакерам собрать много ценной информации. Как мы видели, Митник (`toad.com`) отправил три запроса на хост `x-terminal`. Однако два из них (`showmount` и `rpcinfo`), скорее всего, поступят на один и тот же TCP/UDP-порт 111. Безусловно, можно определить выдачу предупреждений о попытках соединения с двумя различными портами одного хоста в течение одной минуты. Однако на практике это приведет к большому числу сообщений о ложных тревогах. Не потребуется долго ждать, пока аналитик сетевого трафика установит более высокий предел для выдачи уведомлений об атаке. Таким образом, система обнаружения вторжений, вероятнее всего, не уведомит об этом зондировании, как о попытке сканирования портов.

Зондирование сети

Зондированием сети считается попытка доступа одного удаленного хоста к различным хостам одной сети за короткий промежуток времени. В нашем примере хост `toad.com` подключался к трем различным системам с промежутком в несколько минут. Мощные средства обнаружения сканирования хостов заставляют хакеров осуществлять свои попытки зондирования с различных адресов, чтобы избежать обнаружения. За время работы нам приходилось сталкиваться с различными системными администраторами. Некоторые применяют абсолютно глупые алгоритмы выявления атак (например, требуют уведомлять обо всех подключениях, выполненных с одного адреса к пяти различным хостам сети за один час), другие используют грамотные методы, позволяющие максимально снизить число ложных тревог. Если уменьшить контролируемый промежуток времени с часа до, например, пяти минут и задать выдачу уведомления об атаке при подключении к трем или более хостам, то частота ложных тревог останется низкой для большинства узлов. Если в системе обнаружения взлома можно изменить правило уведомления о зондировании сети и устранить хосты или условия, которые часто вызывают ложные тревоги (например, не учитывать обращения к популярным Web-серверам, радио в реальном времени или другим широковещательным службам), то пороговое значение для предупреждения об опасности можно задать даже меньшим, чем подключение извне к пяти хостам за час или к трем хостам за пять минут. Специальная программа системы обнаружения вторжений, предназначенная для выявления зондирования сети, должна обнаружить приведенное выше зондирование Митника.

Подключение к важным портам

В разведывательном зондировании осуществлялось подключение к портам, используемым хорошо известными и потенциально уязвимыми службами. Поэтому подобное зондирование практически наверняка должно быть обнаружено.

Сетевые системы обнаружения вторжений с высокой надежностью выявляют попытки подключения к службам удаленного вызова процедур. В целом, хакер имеет определенное преимущество в выборе методов, позволяющих ему избежать обнаружения. Он может действовать медленно и постепенно, использовать отвлекающие маневры и только затем переходить к достижению истинной цели. Однако для запуска программы атаки хакеру приходится подключаться к порту популярной службы, и здесь преимущество уже на стороне системы обнаружения вторжений. Службы удаленного доступа являются одними из самых распространенных целей атак, поэтому любая система обнаружения вторжений обязана выявлять попытки доступа к этим службам.

Системы обнаружения вторжений для хостов

Поскольку атака Митника проводилась против UNIX-системы, в этом обзоре мы рассмотрим два типа широко используемых UNIX-программ для защиты от взлома: TCP Wrapper (упаковщик TCP) и Tripwire. Программа TCP Wrappers регистрирует попытки установить соединение с защищаемыми службами и разрешает или запрещает доступ на основе заданного списка контроля. Утилита Tripwire позволяет отслеживать состояние отдельных файлов и уведомляет при их изменении. Для хостовых систем обнаружения взлома эти возможности являются минимально необходимыми. Программы PortSentry и LogSentry (доступны по адресу `www.psionic.com`) обеспечивают даже более высокий уровень обнаружения взлома и защиты с помощью контроля за системными журналами и анализа входящих пакетов.

TCP Wrappers

С помощью программы TCP Wrappers или демона `xinetd` можно защититься от попыток зондирования или атак на уровне хоста. Для использования TCP Wrappers нужно отредактировать файл `/etc/inetd.conf` и перечислить в нем защищаемые службы, например `finger`. Кроме того, в TCP Wrapper следует добавлять списки контроля доступа. Если в системе запущена какая-то служба, например `finger`, то можно определить, каким компьютерам будет разрешен доступ к демону `finger`. При этом будут регистрироваться как успешные, так и неудачные попытки доступа. Приведенная ниже вымышленная запись в системном журнале (`syslog`) показывает, как будут выглядеть три попытки подключения к службе `finger`, зарегистрированные с помощью TCP Wrappers.

```
Dec  24 14:10:29 target in.finger[11244]: refused connect from toad.com
Dec  24 14:10:35 server in.fingerd[21245]: refused connect from toad.com
Dec  24 14:11:08 x-terminal in.fingerd[11066]: refused connect from
toad.com
```

При обнаружении попыток взлома на хостах очень важно правильно определить, какая часть информации должна сохраняться и анализироваться на локальном хосте, а какая — отправляться на центральный сервер. В нашем примере показано, что три различных компьютера (`target`, `server` и `x-terminal`) отправляют уведомления на центральный сервер регистрации. Отдельная попытка установить соединение с портом службы `finger` может быть проигнорирована. Но три такие попытки подключения к трем хостам, зарегистрированные на центральном сервере, скорее всего, не останутся незамеченными.

Для специалиста в области безопасности попытки установить соединение со службой `portmapper` должны быть важнее, чем аналогичные попытки по отношению к `finger`. Во время проведения атаки Митника защищенные службы преобразования портов еще не получили широкого распространения. Сейчас ситуация изменилась, поэтому отсутствие возможностей регистрации и контроля доступа для службы преобразования портов (`portmapper`) явно свидетельствует об устаревшей или неправильно сконфигурированной системе UNIX. Система обнаружения вторжений, предназначенные для защиты отдельного хоста, обязана выявлять попытки доступа к службе `portmapper`.

Программа Tripwire

Программу Tripwire не стоит использовать для выявления попыток разведывательного зондирования. Дело в том, что ее основным предназначением является создание и хранение контрольных сумм для критически важных файлов с целью обнаружения любых изменений в этих файлах. С помощью Tripwire можно обнаружить взлом системы только в тот момент, когда будет перезаписан файл `/.rhosts`. К сожалению, даже если сигнал тревоги поступит немедленно, все равно будет уже слишком поздно. Система уже скомпрометирована, и подготовленная атака может нанести огромный ущерб практически мгновенно. Поэтому, чем раньше обнаруживается вторжение, тем лучше. Шансы на блокирование настоящей атаки значительно увеличиваются, если нарушитель и цель его атаки будут установлены еще на этапе сбора разведывательной информации.

Предотвращение атаки Митника

Конечно, атаку Митника можно предотвратить несколькими способами на разных ее этапах. Правильно настроенный брандмауэр или фильтрующий маршрутизатор не требуют крупных капиталовложений, просты в настройке и эффективны по отношению к блокированию попыток сбора разведывательной информации и атак по Internet. Даже на то время на рассмотренном нами узле Шимомуры было доступно больше служб, чем следовало.

При блокировании возможности сканирования и работы r-утилит провести успешную атаку становится сложнее, а иногда и абсолютно невозможно. В общем случае на сетевом узле должны блокироваться все входящие пакеты, кроме тех, которые поступают на открытые порты. По адресу `www.sans.org/top20.htm` можно найти файл, в котором перечислены наиболее важные порты (не только те, доступ к которым следует блокировать, но и те, попытки доступа к которым следует регистрировать). Как будет рассказано в главе 16, "Вопросы архитектуры", основные возможности обнаружения взлома обеспечиваются на периметре локальной сети.

Читатели этой книги уже узнали о средствах защиты локальных хостов и использовании списков контроля доступа. Конечно, на каждой системе запускаются различные службы, но часто можно указать компьютеры, которые могут обращаться только к той или иной службе (например, с помощью TCP Wrappers). В этом случае хакеру придется сначала взломать доверенный хост и уже с него запускать программу атаки. В атаке Митника просто использовался IP-адрес доверенного хоста, что намного проще, чем действительно скомпрометировать этот хост.

Даже после начала атаки Митника она может быть остановлена при своевременном обнаружении и правильных ответных действиях. В главе 18, "Ответные действия", мы рассмотрим методы замедления и даже полного блокирования начатых атак.

Резюме

При ретроспективном анализе успешной атаки или компрометации системы чаще всего выясняется, что атаке предшествовал сбор разведывательной информации. Основной проблемой является выявление попыток разведывательного зондирования, серьезное отношение к этим попыткам и улучшение защиты системы. Очень часто предварительная разведка выполняется для поиска доверительных отношений между компьютерными системами.

Злоумышленники нередко используют доверительные отношения при организации своих атак. Устанавливают доверительные отношения сами системные администраторы "для удобства", хотя при этом они и осознают, что своими руками пробивают брешь в системе защиты.

В атаке Митника сознательно обрывалась процедура установки TCP-соединения, чтобы организовать SYN-наводнение на одну из сторон доверительных отношений. Этот метод используется во многих атаках и средствах сканирования.

Специально подготовленные пакеты атаки, в которых подменяется IP-адрес отправителя, часто имеют сигнатуры, по которым и можно выявить проводимую атаку.

Однократная проверка является общей проблемой для всех средств обеспечения компьютерной безопасности. При разработке программного обеспечения предусматривайте возможность перепроверки.

Признаком перехвата TCP-сеанса является изменение IP-адресов в течение сеанса при одновременной "правильности" порядковых номеров в поступающих пакетах. Надежное выявление перехвата TCP-сеанса с помощью одного средства защиты по-прежнему остается недостижимой мечтой для реальной работы в компьютерных сетях.

Атака Митника – прекрасный тест для определения качества системы обнаружения вторжений. Системы, которые не способны выявить эту атаку в реальных сетях с реальной нагрузкой (например, в загруженных сетях T-1 или в сетях с большей пропускной способностью), только вводят своих пользователей в заблуждение. Даже самая лучшая система обнаружения вторжений не сможет выявить атаку, обнаружение которой не заложено в ее программу. Многие аналитики сетевого трафика предпочитают использовать системы обнаружения вторжений, позволяющие им создавать собственные фильтры для выявления новых или необычных атак. В следующей главе приведено несколько примеров пользовательских фильтров.

16

Вопросы архитектуры

В этой главе обсуждаются свойства систем обнаружения вторжений, возможные альтернативы и проблемы, с которыми могут столкнуться пользователи и разработчики. Здесь больше теоретической информации, чем в других частях книги, но в нее добавлены примеры из реальной жизни. Будет рассмотрена концепция *значимых событий* (event of interest). Это важное понятие, так как аналитик сетевого трафика добивается лучшего результата при использовании системы обнаружения вторжений, если он понимает, что он ищет. Следовательно, он может правильно настроить систему обнаружения вторжений, а не позволять самой этой системе уведомлять его о тех или иных событиях. Кроме того, мы рассмотрим вопрос точности определения атак. Каждое происшествие уникально и должно рассматриваться отдельно. Это давний предмет ожесточенных споров теоретиков о том, где должен быть установлен датчик системы обнаружения вторжений: за или перед брандмауэром. В данной главе рассмотрены этот и другие вопросы, касающиеся размещения датчиков.

Один из самых известных мифов, которые распространяют производители программного обеспечения для обнаружения взломов, заключается в работе программ *в реальном времени*. Под работой в реальном времени в данном случае понимается то, что аналитики сетевого трафика будут немедленно реагировать на выдаваемые предупреждения и сигналы тревоги. Конечно, реагирование в реальном времени практически невозможно, по крайней мере для человека, так как скорость передачи пакета сравнима со скоростью распространения света. На рис. 16.1 показано обнаружение атаки непосредственно в реальном времени (сразу же после события). Этот рисунок пригодится вам, если руководство будет акцентировать внимание на времени ответа. В действительности компьютерные системы UNIX и Windows NT не поддерживают ни реагирования в реальном времени, ни даже точно подсчитанных задержек по времени. Мы рассмотрим эти вопросы при сравнении систем доставки и получения сообщений (push and pull architectures). Более того, аналитики сетевого трафика могут применять фильтры для повторного или даже третьего просмотра данных в поисках значимых событий.

Ни один производитель систем обнаружения вторжений не предоставляет идеального интерфейса для аналитика сетевого трафика. Мы обсудим, каким же должен быть этот интерфейс.

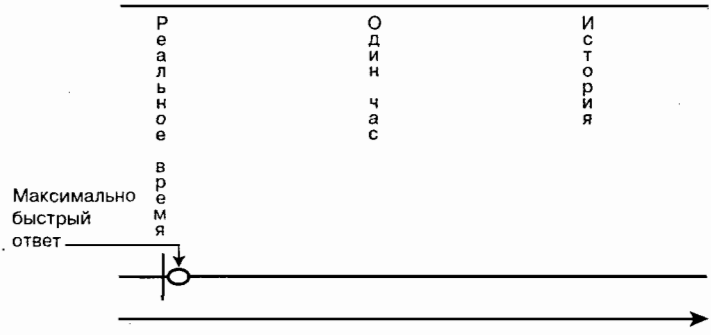


Рис. 16.1. Реакция системы обнаружения вторжений

В следующем разделе рассмотрены некоторые варианты настройки, которые должны учитываться при создании или усовершенствовании средств обнаружения вторжений. К ним относятся проблемы определения истинных тревог и места установки датчиков.

Значимые события

В главах 13, "Система Snort", и 14, "Параметры правил Snort", уже упоминалось, что при написании фильтра необходимо задать условие его "срабатывания". Допустим, для правила Snort используется параметр `content`, определяющий поиск шестнадцатеричных значений `0xdead` или `0xbeef` (стандартный шаблон, который иногда используется нарушителями в их программах). Тогда получение пакета, удовлетворяющего данному шаблону, можно считать потенциально значимым событием. В теории обнаружения взлома значимые события связаны с тремя основными проблемами:

- баланс между сообщениями о настоящих атаках и ложными тревогами;
- точная настройка средства обнаружения (датчика) на гарантированное выявление значимого события;
- влияние ограничений системы на возможности обнаружения атак.

Проблема невыявленных атак и ложных тревог является одной из самых серьезных для систем обнаружения вторжений. Мы потратили огромные усилия при создании собственных фильтров, позволяющих обнаружить значимое событие и при этом не создающих ложных тревог. С другой стороны появление ложных тревог означает, что что-то обязательное для обнаружения не было задано в фильтре. Покажем, что может сделать аналитик, на простом примере. Злоумышленники обычно используют определенные строки, числа и шестнадцатеричные значения в своих программах для проведения разведки, атак отказа в обслуживании и других атак. Некоторыми "классическими" значениями являются:

- десятичные значения 31337 и 666;
- строка ASCII-символов skillz;
- шестнадцатеричные шаблоны 0xdead и 0xbeef.

Предположим, нам нужно создать фильтр для выявления шестнадцатеричного значения 0xdead.

```
alert icmp  any any -> 192.168.5.0/24 any      \
      (msg: "Обнаружена строка шестнадцатеричных символов 0xdead";  \
      content: "|DE AD|")
```

Приведет ли использование этого правила к появлению ложных тревог? Безусловно. Если в ICMP-пакете будут содержаться эти шестнадцатеричные символы в указанном порядке, будет выдано предупреждение об атаке. Стоит ли применять это правило для обработки данных в реальном времени? Нет, лучше не надо. С другой стороны, если во многих входящих пакетах будут содержаться 0xdead или 0xbeef, то это может оказаться важным. Для нас одним из самых полезных уроков работы с системой Shadow явилась необходимость применения вторичного анализа. Сохраните пакеты за несколько дней и затем проведите их повторный анализ с целью выявления значимых событий. Вероятнее всего, не стоит обращать внимание на единичное появление значений 0xdead или 666 в данных за несколько дней, но, если будут обнаружены десятки таких значений, то, безусловно, следует внимательно проанализировать полученные пакеты, содержавшие эти строки.

Почти все приведенные в главах 10, "Практический анализ", и 11, "Необычный трафик", случаи похожи. Аналитик сетевого трафика, изучая входящие данные, замечал что-то необычное. Когда мы с Джуди совместно работали над анализом сетевого трафика, мы выявляли огромное количество атак. Люди спрашивают, как нам это удавалось. Обычно я отвечаю: "Просто повезло". Но наши читатели теперь знают больше. Мы старались разделить полученные данные с целью выявления наиболее значимых событий.

Еще одним подходящим сценарием является сбор данных на протяжении недели или около того, и их последующий анализ с целью обнаружения необычного использования возможностей протоколов, как показано, например, в следующем фильтре.

```
not tcp and not udp and not icmp and not igrp and not igmp
```

Этот фильтр, очевидно, не следует использовать в реальном времени, но он, несомненно, будет полезен при ретроспективном исследовании полученных данных в поисках значимых событий, которые могли остаться незамеченными. После того как аналитик изучил свою сеть и оптимизировал используемые фильтры, с помощью этого фильтра очень редко удается что-нибудь обнаружить. Я не рекомендую запускать его в интерактивном режиме и просматривать результаты. Вы быстро устанете и прекратите это занятие. Однако, если задать регулярный запуск этого фильтра раз в неделю и настроить систему обнаружения вторжений на выдачу предупреждений только в случае получения каких-то результатов, то однажды это средство позволит обнаружить проведенную атаку. Если вы находитесь в процессе поиска новых улучшенных средств или программ обеспечения защиты корпоративных сетей, то обратите внимание на возможность запланированного запуска сценариев для анализа полученных данных.

Итак, мы закончили наше изучение значимых событий, остановившись на рассмотрении общих ограничений систем при обнаружении атак. Начнем с самого важного: важно четко осознавать, что именно требуется найти, и какие события должны иметь значение, потому что нельзя регистрировать или получать отчеты обо всех событиях. На рис. 16.2 изображены как события, которые может "видеть" ваша система обнаружения взлома, так и другие события, которые остаются вне поля зрения.

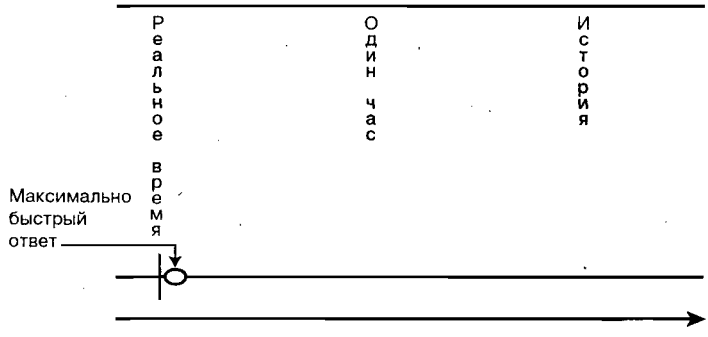


Рис. 16.2. Источники информации

Ограничения контроля

Как показано на рис. 16.2, средство обнаружения не способно контролировать все происходящие события. Это часто удивляет тех, кто потратил крупные средства на приобретение системы обнаружения вторжений, а со временем понял, насколько она ограничена в использовании на практике. Так какие же события мы не можем контролировать?

- **События в других сетях**. В наше время довольно распространены несанкционированные подключения к сети через потайные ходы. Каждый компьютер с модемом потенциально может предоставлять этот потайной ход. Это явление доказывает несостоятельность рекламных заявлений производителей систем обнаружения взлома хоста наподобие "мы здесь, мы там, мы – везде".
- **Не работает средство обнаружения**. События могут быть очевидными, но система обнаружения может выйти из строя и будет не способна их выявить. Выражение "выйти из строя" в данном случае означает промежуточное состояние между полным отказом (с появлением голубого экрана) и временной невозможностью провести эхо-тестирование. Хорошим показателем надежности системы обнаружения вторжений может служить время, требующееся для перезагрузки системы, так как это может быть проблемой как для систем под управлением Windows NT, так и для UNIX-систем. Я многократно проверял эту характеристику, работая с Shadow, NFR, NID, Snort и RealSecure. По закону невезения эти системы требуется перезагружать в дождливые дни, и при этом они обязательно находятся в другом здании. Конечно, некоторые из систем надежнее других. А какое самое эффективное средство для удаленного управления Windows NT? Разумеется, автомо-

биль :-). Если жесткий диск, использующийся для сохранения отчетов, будет заполнен, то сбор данных прекратится.

- **Протоколы SNA и SS7**. Невозможно проверять данные протоколов, с которыми не поддерживает работу система обнаружения взлома. А нужна ли система обнаружения взлома, которая способна декодировать пакеты протоколов SS7 или SNA? Для большинства случаев ответ будет отрицательным, но довольно часто мы обнаруживаем пакеты неизвестных протоколов. Например, я знаю людей, которые обнаруживали пакеты протоколов IP Protocol 54, NHRP (Next Hop Routing Protocol – протокол определения адреса следующего перехода) в своих демилитаризованных зонах, и никогда не видел системы обнаружения вторжений, способной декодировать эти пакеты.
- **Превышение максимальной пропускной способности**. Интенсивность событий может превысить максимальную пропускную способность для обработки данных средством обнаружения. Тогда датчик начинает пропускать пакеты, и наступает режим, который аналитики сетевого трафика называют *статистической выборкой*. На вопрос о максимальной скорости поступающей информации поставщики систем обнаружения взлома дают весьма любопытные ответы (от "80 Мбит/с" и до "когда как"). *Совет*: лучше верить человеку, который отвечает "когда как", чем тому, кто называет точную цифру, особенно если она превышает скорость передачи данных в канале T-3 (45 Мбит/с). Одним из важнейших факторов для определения верхней границы скорости работы средства обнаружения вторжений является количество активных правил для проверки пакетов. Если обработка конкретного пакета не будет закончена на момент поступления следующего, то последний пакет будет отброшен.

Таким образом, системы обнаружения вторжений не могут отслеживать все происходящие события по следующим причинам:

- событие произошло в другой сети;
- вышла из строя система обнаружения вторжений;
- нельзя обработать пакеты неизвестного протокола;
- достигнута максимальная скорость обработки данных системой обнаружения вторжений.

Плохо, что мы не можем отслеживать абсолютно все события. Но хорошо, что есть события, которые можно регистрировать. Из всех регистрируемых пакетов какая-то часть будет соответствовать заданным фильтрам (область внутреннего круга). И, наконец, некоторые из общего числа подозрительных пакетов будут действительно важными. Они и представляют собой значимые события безо всяких ложных тревог. Выявление значимых событий является основной целью всей работы по установке и настройке систем обнаружения вторжений. Обнаружение атаки, особенно сложной атаки, приносит огромное удовлетворение.

Принцип низко висящего плода

На сегодняшний день базовым стандартом в теории обнаружения вторжений является набор правил Snort. Ранее использовались два основных набора правил, но

некоторые проблемы с законом у Макса Вижина (Max Vision) привели к тому, что теперь его набор правил недоступен. Было очень интересно наблюдать совместный труд многих пользователей над созданием правил, уточнением списка контролируемых портов и объяснением уязвимых мест. Мне даже стыдно уделять так мало места описанию этого важного труда, но здесь нужно быть осторожными. Мы уже рассматривали проблему ложных тревог при обсуждении сигнатур и фильтров для выявления сигнатур. Теперь рассмотрим влияние принципа низко висящего плода на ложное отсутствие предупреждений. При чем здесь низко висящий плод?

Я живу на острове в тропиках. Многих вещей мне не хватает, но зато есть достаточно бананов и бесплатных кур. Семь лет назад был ураган и многие курятники развалились, а куры разбежались. Хищников на острове нет, и теперь он переполнен курами. У моего соседа недавно был невиданный урожай бананов. Я никогда не задумывался, сколько бананов может вырасти на одном дереве, но по всей видимости они весят не меньше сотни фунтов. Когда дерево начинает сгибаться под тяжестью, нижние бананы становятся добычей кур. Куры выстраиваются под банановыми деревьями и, подпрыгивая, откусывают куски раскрывшихся бананов. Таким образом, низко висящий плод – это плод, который легко сорвать, и до которого может добраться каждый желающий.

Предположим, что большинство поставщиков систем обнаружения вторжений втайне загрузили набор правил Snort и используют его как базу для своих собственных правил. Что если они затем обращаются на несколько хорошо известных сайтов, загружают программы атаки и запускают их в своих лабораториях с целью определения сигнатур, создания эффективных фильтров для выявления этих атак и сохраняют эти фильтры в новых системах обнаружения вторжений? Если бы так было, мы бы могли найти что-то общее во всех системах обнаружения вторжений. На первый взгляд, это может показаться даже удачным: пользователи всех систем обнаружения вторжений точно могут рассчитывать на минимальный уровень выявления атак, установленный набором правил Snort. Проблема в том, что злоумышленник может проанализировать набор правил Snort и внести минимальные изменения в свою атаку, что не позволит ее выявить системе обнаружения вторжения. Если многие поставщики коммерческих систем скопируют правила Snort, это может стать интересной проблемой. Один и тот же набор правил станет доступным для огромного количества пользователей, как низко висящий плод.

Хотя ситуация, описанная в предыдущем абзаце, отчасти справедлива, но есть множество методов, чтобы избежать этой проблемы. Многие исследователи и поставщики систем обнаружения вторжений поддерживают связи с хакерами и имеют доступ к большему количеству атак, чем предоставляется на сайтах. Несколько исследователей стараются собрать все программы атаки с целью узнать все уязвимые места. К сожалению, они используют различные названия при описании одних и тех же атак. Организация Mitre (`http://cve.mitre.org`) руководит проектом под названием *CVE* (Computer Vulnerabilities and Exposures – уязвимые места и дефекты компьютерных программ), который получил широкую поддержку у производителей. Целью проекта является создание единой системы имен, которую можно было бы использовать как словарь (или справочник) для описания уязвимых мест и создания систем обнаружения вторжений.

Кроме того, можно создать фильтр, позволяющий выявить целую группу программ атаки. Мы уже рассматривали один такой фильтр для выявления атак на Web-серверы. При его изучении мы рассказывали о большом количестве атак CGI-BIN на Web-серверы, в которых предпринималась попытка получить файл паролей для их последующего взлома. Самой известной является атака на сценарий PHF. Кроме того, существует еще около сотни подобных атак, включая атаки на PHP и `aglimpse`. В прошлом в пакете каждой из этих атак содержались строки `cgi-bin` и `/etc/passwd`, поэтому все подобные атаки можно было обнаружить с помощью одного общего фильтра. С появлением теневых файлов паролей количество атак с указанием файла `/etc/passwd` уменьшилось. Вместо этого в пакете обычно содержится следующая строка.

```
id;uname -a; w
```

Команда `id` предоставляет действительный идентификатор пользователя, точка с запятой отделяет команды, команда `uname -a` выводит отчет об операционной системе (текущую версию, установленные заплаты), и, наконец, команда `w` предоставляет сведения обо всех зарегистрированных пользователях. Также можно (и крайне желательно) создать общие фильтры для выявления необычных событий (тех, что не должны происходить) и выдачи отчета об этих событиях. Примером такого события является получение пакетов со всеми установленными TCP-флагами, вообще без установленных флагов и пакетов неизвестных IP-протоколов. Существует много способов для усовершенствования средств обнаружения, однако следует задуматься: если система обнаружения взлома основана на сигнатурах и в ней не предусмотрен фильтр для выявления этой сигнатуры, как она сможет выявить атаку?

Человеческий фактор

На уровень обнаружения значимых событий и уведомления о них также влияет участие человека. В течение обычного рабочего дня оператор системы обнаружения вторжений регистрирует несколько происшествий и, возможно, сообщает о них. Если просмотреть годовой отчет о событиях на крупном узле, можно обнаружить отчеты о приблизительно 12 атаках на IMAP, 5 на `portmap`, 25 эхотестированиях по ICMP, 30 атаках Smurf, 8 сканированиях `msscan`, 5 сканированиях портов, 5 попытках переноса зоны DNS, 4 атаках WinNukes и т.д. Если обратиться к группе CIRT крупной сети, то мы узнаем, что обо всех этих инцидентах сообщали только некоторые администраторы узлов этой сети. Почему так происходит? Не только потому, что система обнаружения вторжений не смогла отреагировать на атаку из-за отсутствия нужной сигнатуры. Зачастую аналитики сетевого трафика сами не сообщают о выявленных инцидентах.

Если потратить день или два на поиск в Internet, можно без труда собрать сотни различных программ атаки. Компиляция некоторых из них потребует определенных усилий, для других не предоставляется полной документации. Но факт остается фактом — можно легко найти большее число атак, чем было обнаружено и описано. А в чем же проблема? Частично это объясняется тем, что не так просто определить нужную сигнатуру. Если система работает на основе сигнатур, а фильтра не существует, выявить событие невозможно. Другие

факторы, снижающие способности выявления значимых событий системами обнаружения вторжений, связаны с аналитиками сетевого трафика и группами CIRT (Computer Incident Response Team – группа реагирования на угрозы безопасности компьютеров).

Ограничения, связанные с аналитиками

Часть ответственности за необнаруженные атаки лежит на аналитиках сетевого трафика. Тому есть несколько причин. Иногда аналитики пренебрегают попытками взлома и считают их недостаточно серьезными для глубокого анализа. Я и сам так поступал много раз. Вот классический пример: вирус Code Red попрежнему действует только потому, что некоторым людям не хватает соображения установить заплаты для своих IIS-серверов. Однажды на моем узле было зарегистрировано большое число попыток подключения к порту 80, но я не стал глубоко их анализировать, сообразив, что это действие Code Red. Однако в феврале 2002 года, когда появились сведения о выявлении уязвимого места в сценарии PHP сервера Apache, мне пришлось изменить свое отношение. В конце концов, ведь на моем хосте также запущен Apache.

Сообщают ли аналитики о тех атаках, которых они сами не поняли? Чтобы понять смысл неизвестных методов атаки, необходимо обладать глубокими знаниями о теории стека TCP/IP и процессах, происходящих в компьютерных системах. А что если аналитик не доверяет собственной системе обнаружения вторжений? Требуется немалая уверенность в своей системе, чтобы принимать решение только по информации предупреждения, выдаваемого на экран. И практически слепое доверие необходимо, когда система обнаружения вторжений сообщает о двух атаках Wiz в день (Wiz – это очень старая атака на электронную почту) и шести SYN-наводнениях в час (очевидно, что имеют место ложные тревоги). Таким образом, аналитики явно являются слабым звеном в системе обнаружения вторжений. Основные причины этого:

- нежелание сообщать о событиях, выявленных системой обнаружения вторжений;
- недостаток знаний, необходимых для выявления новых атак;
- недостаточные знания о протоколах стека TCP/IP и сетевых службах;
- недоверие к самой системе обнаружения вторжений.

Ограничения, связанные с группами CIRT

Могут ли группы CIRT не уведомлять о каких-то выявленных атаках? Если эта группа получит отчет об атаке на IMAP, `portmap`, сканировании ICMP, атаках Smurf, `mscan`, `portscan`, переносе зоны DNS или чем-либо подобном, то все будет нормально. Они внесут это сообщение в свою базу данных, доступную всем пользователям. Но что будет, если группа CIRT получит приблизительно такое уведомление "Неизвестный тип зондирования. Вот его трассировка. Эта атака привела к отказу моего компьютера"? Человек, который сообщил об атаке, вероятно, новичок. Кроме того, возникает проблема отсутствия нужного раздела в базе данных. Профессиональные аналитики обычно перегружены работой, и

опытные сотрудники групп CIRT обычно выходят из себя при получении отчетов о (как часто оказывается впоследствии) ложных тревогах. Отношение будет другим только при получении второго подобного отчета из другого источника. Такая задержка может быть очень опасной при активном использовании новой атаки. С момента, когда я впервые узнал об уязвимом месте в `klogin` в мае 2000 года, прошло менее 8 часов до компрометации нашей первой системы.

Это серьезная проблема, потому что группы CIRT практически всегда недоукомплектованы. Люди звонят по телефонам и просят о помощи, так как их компьютеры скомпрометированы, а в их организации никогда не относились серьезно к безопасности. Этим людям помощь необходима в первую очередь. В конце месяца или квартала CIRT издает свой отчет, в котором уведомляется об обнаружении такого-то количества попыток проведения каждой конкретной атаки. Неопытный аналитик сетевого трафика, сообщивший о неизвестном типе атаке, видит, что в отчете CIRT о его случае нет и слова, и решает больше никогда не сообщать о происшествиях. При этом он не знает, сочли сотрудники CIRT его уведомление ошибочным, или им просто все равно. Вот почему мы сделали сознательный выбор в пользу Incidents.org – добровольного объединения всех групп CIRT и организаций обеспечения безопасности в Internet. Именно этой организации мы прежде всего отправляем наши уведомления обо всех неизвестных шаблонах атаки и добавляем следующий комментарий: "Мы понятия не имеем, что это такое. Может ли нам кто-нибудь помочь?". Не один раз полученный ответ меня смущал, так как я должен был узнать эту атаку. Со временем мы научились работать подобно CIRT и отправлять запросы только в случае крайней необходимости. Мы стали осторожнее в призывах о помощи, и это может привести к успеху атаки прежде, чем мы самостоятельно сможем ее выявить и заблокировать. Мы знаем, что не следует быть слишком замкнутыми, но так трудно бороться со своим характером.

Итак, краткое описание значимых событий закончено. Мы не можем узнать о каждом событии. Из числа наблюдаемых событий некоторыми пренебрегают, хотя они являются настоящими атаками, а другие вызывают ложные тревоги. Цель разработчика системы обнаружения вторжений и аналитика сетевого трафика заключается в максимальном увеличении регистрируемых значимых событий и одновременном сведении к минимуму ложных тревог. Здесь возникает множество архитектурных и программных препятствий, но какую-то роль также играет участие человека в процессе выявления атаки.

Степень угрозы

В некоторых научных школах предлагаются способы сведения степени угрозы атак к метрике – значению, которое можно вычислить. В этом разделе будут рассмотрены некоторые из основных факторов, на основе которых должно выполняться подобное вычисление. Давайте начнем с базового философского принципа: степень угрозы лучше всего могут оценить пользователи атакуемой системы. Это очень важный принцип, так как чем дальше находится лицо, оценивающее атаку (наблюдатель), тем менее серьезной она кажется (по крайней мере с точки зрения наблюдателя).

Это происходит постоянно

Однажды группу обнаружения вторжений, в которой я работал на протяжении нескольких лет, пригласили провести день в обществе очень большой группы CIRT. В состав этой группы CIRT входил отдел аналитиков сетевого трафика, которые только что получили новые возможности обнаружения атак с помощью интерфейса, позволяющего контролировать большое количество датчиков. Нам всем показалось очень интересным понаблюдать за работой группы аналитиков, использующих систему Shadow, и оценить эффективность нового программного интерфейса. Через четыре минуты одна из аналитиков обнаружила сигнатуру, указывающую на взлом с правами суперпользователя на одном из близких нам узлов. Она хотела позвонить на узел и сообщить им об атаке, но сотрудники CIRT рассмеялись, сказав: "Это происходит постоянно". Несомненно, с их точки зрения это было справедливо. За год они, вероятно, получают уведомления о таком числе взломов, с каким я не сталкивался за свою жизнь. Мне было не по себе, так как я знаю, сколько проблем и неприятностей несет взлом системы ее владельцам и тем, кто им помогает. Поэтому степень угрозы лучше всего оценивать владельцам взломанных систем.

Хотя при обсуждении степени опасности атак следует помнить и о человеческом факторе, но для надлежащего блокирования нужна какая-то классификация проводимых атак. В каждом приемном отделении больниц есть человек, который отвечает за порядок оказания помощи и следит, чтобы она предоставлялась тем, кто нуждается в ней в первую очередь. Поэтому пациент с угрозой для жизни не будет ждать, пока медицинский персонал окажет помощь пациенту с пораненным пальцем. При реагировании на широкомасштабную атаку ресурсы компьютера очень быстро истощаются, поэтому необходим способ их упорядочения. Рассмотрим эту концепцию на высшем уровне (рис. 16.3).

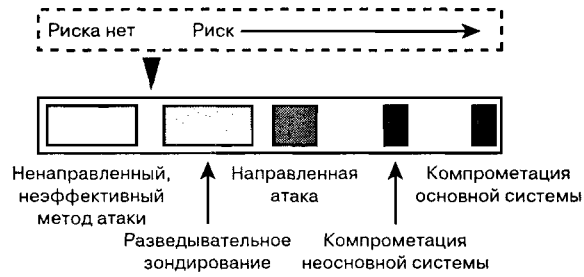


Рис. 16.3. Степень угрозы

Действительно ли ненаправленные атаки на уязвимые места, которые отсутствуют на атакуемом компьютере, не представляют никакой опасности? С научной точки зрения одним из элементов уравнения для оценки риска считается уровень уверенности — насколько вы уверены, что в программном обеспечении ваших систем нет уязвимых мест. Я скорее консерватор. Поэтому считаю, что я ненаправленные, "безвредные" атаки не представляют опасности только тогда, когда они дополнительно блокируются брандмауэром или фильтрующим маршрутизатором. В этом случае можно говорить об отрицательном риске. Злоумышленник сам подвергается большей опасности, когда использует ненаправленную атаку против хорошо защищенного узла, чем этот узел, так как об атаке будет сообщено. Если он добьется успеха и причинит ущерб на других узлах, то, по крайней мере, повышается вероятность его обнаружить.

Как можно оценить степень угрозы? Какие основные факторы? Можно ли написать уравнение? Насколько высока вероятность того, что атака нанесет ущерб, и насколько серьезным он будет? Очевидно, все это должно быть учтено.

Важность цели

В теории оценки риска одним из важнейших факторов является важность атакуемой цели. Когда-то, выступая в Вашингтоне, я хотел привлечь внимание к антивирусам и брандмауэрам на персональных компьютерах. Я спросил у аудитории: "Сколько из вас путешествуют по несколько раз в год?". Большинство слушателей подняли руки вверх, что вполне нормально для офицеров штаба. Тогда я спросил: "Сколько из вас берут с собой "Кипро?". "Кипро" – это антибиотик против сибирской язвы. Поскольку до октября 2001 года никто не слышал о рассылке бактерий по почте, то об этом никто и не думал. Однако я только могу представить себе, что было бы, будь я в незнакомом городе, и вдруг у меня начались боли, сильнее которых не было в моей жизни. Как мне получить высококвалифицированную медицинскую помощь? Дома у меня есть личный врач, который меня знает, медицинская карточка и друзья-доктора. В Хьюстоне, Сиэтле или Нью-Йорке придется обращаться в отделение скорой помощи. Практически невозможно ждать 12 часов, чтобы тебя осмотрели в приемном отделении. Насколько это серьезно? Эта загадка может любого вывести из равновесия. Теперь, чтобы вы не думали, что я не встаю со своего кресла-качалки без "Кипро", уточню, что тоже много путешествую и, хотя стараюсь не пить сырой воды и чищу фрукты, все равно лучше иметь средство наподобие "Кипро" на всякий случай. Какое это имеет отношение к антивирусам или персональным брандмауэрам? Если, не имея этих средств, вы будете "путешествовать" по Internet, то можете оказаться в сложной ситуации, сравнимой с заболеванием сибирской язвой и одновременным отсутствием "Кирпо".

Будет не очень хорошо, если я заболею сибирской язвой, но еще хуже, если ею заболеет президент Соединенных Штатов. Основным критерием для оценки опасности является важность цели атаки. При взломе настольного компьютера могут быть потеряны время и результаты работы. Кроме того, эта система может использоваться как стартовая площадка для организации последующих атак. Однако при компрометации сервера DNS или сервера электронной почты возникающие проблемы гораздо серьезнее. Действительно, если взломщик сможет контролировать сервер DNS, то он сможет манипулировать доверительными отношениями и скомпрометировать большинство или все системы локальной сети. Для создания метрики требуется определить способ оценки важности цели. Для этого можно использовать простую пятибалльную шкалу.

5 баллов	Брандмауэр, DNS-сервер, основной маршрутизатор
4 балла	Сервер электронной почты
2 балла	Настольная пользовательская UNIX-система
1 балл	Система MS DOS 3.11

Уровень риска

Уровень риска атаки определяет то, какой ущерб она может причинить. Программы атаки обычно предназначены для использования уязвимых мест опреде-

ленных приложений или операционных систем. Компьютер под управлением Macintosh не уязвим для программы атаки на переполнение буфера на сервере объектной базы данных ToolTalk системы UNIX или атаки на службу `rpc.statd`. Если же эти атаки будут использованы против хоста Sun Microsystems, на котором запущена версия Solaris без установленных заплат, то атакованный хост может быстро перейти в "собственность" хакера. Как аналитик сетевого трафика, я всегда беспокоюсь, когда нарушитель атакует конкретную цель при помощи подходящей программы атаки. Это означает, что он уже провел предварительную разведку, и что нужно предпринимать дополнительные меры противодействия для защиты атакуемой цели. Для оценки степени риска снова воспользуемся пятибалльной системой.

5 баллов	Злоумышленник получает права суперпользователя на всех хостах локальной сети
4 балла	Полное блокирование с помощью атаки отказа в обслуживании
4 балла	Доступ посредством учетной записи пользователя (например, с помощью взломанного пароля)
1 балл	Малая вероятность успеха атаки (например, для атаки Wiz в 2002 году)

Последний пример в 1 балл для атаки Wiz позволяет продемонстрировать действительно важный момент при оценке степени риска атак — новизну атаки. Для описания этого аспекта используют кривую риска. У хакеров есть специальный термин *точка отсчета* (zero day), который относится к началу использования атаки, которая еще не известна общественности. Программа атаки работает хорошо, и ее держат в секрете несколько человек, применяющие ее для взлома систем. Это высшая точка кривой риска, но при этом число атак еще сравнительно мало. Со временем атака обнаруживается и описывается. Теперь о ней уже знает и сообщество пользователей, и другие злоумышленники. Возникает условие "гонки на выживание" — злоумышленники стремятся скорее использовать атаку и взломать чужие системы, а специалисты по безопасности стараются быстрее создать заплаты, загрузить новые сигнатуры для систем обнаружения взлома или предпринять другие меры защиты. На этом этапе атака еще очень опасна, но кривая риска начинает снижаться, в то время как количество случаев взлома возрастает. Наконец, мы достигаем гребня волны. Все больше системных администраторов устанавливают заплаты или используют другие контрмеры, и со временем атака становится все менее и менее опасной.

Меры противодействия

Что можно сказать о брандмауэрах, программных заплатах или запуске операционных систем с компакт-дисков? Меры противодействия, безусловно, влияют на степень опасности и с логической точки зрения могут быть разделены на две категории: системные и сетевые. По пятибалльной шкале системные меры противодействия можно оценить следующим образом.

5 баллов	Современная операционная система со всеми доступными заплатами, с установленными средствами обеспечения безопасности, такими как TCP Wrappers и служба SSH
3 балла	Устаревшая операционная система, некоторые заплаты отсутствуют

1 балл	Никаких средств защиты, разрешено использование постоянных незашифрованных паролей

Пятибалльная шкала для сетевых мер противодействия выглядит так.

5 баллов	Единственный маршрут соединения с Internet контролируется брандмауэром, максимально ограничивающим прохождение пакетов
4 балла	Брандмауэр, максимально ограничивающий прохождение пакетов. Используется несколько внешних соединений (модемные, ISDN)
2 балла	Брандмауэр, разрешающий прохождение большинства пакетов (основной вопрос: "позволяет ли брандмауэр провести атаку?")

Вычисление степени угрозы

Аналитики сетевого трафика, прошедшие курс обучения в GIAC, для вычисления степени угрозы атаки используют следующую формулу.

```
(Важность цели + Уровень риска) - (Системные + сетевые меры противодействия)
= Степень угрозы
```

Рассмотрим несколько примеров. Они взяты из практического курса получения сертификата сетевого аналитика в GIAC. Чтобы не вырывать эти примеры из контекста, опишем весь процесс анализа, сосредоточив основное внимание на степени угрозы.

Описанный здесь метод подтверждает аксиому о том, что степень угрозы атак различна. При этом учитывается несколько факторов. Можно ссылаться на эти факторы, когда кто-то просит помочь в исследовании атаки. Метод для вычисления степени угрозы пригодится при классификации атаки и выборе средств защиты. Для владельца системы она является самой важной во всем мире. Подобный метод оценки степени опасности может пригодиться при обосновании выбора средств защиты для конкретного пользователя.

Поиск троянских программ

Первый пример взят из трассировки, выбранной Дэвидом Липхартом (David Leaphart) для использования в своих практических занятиях. Для начала скажем, что запись первой строки свидетельствует о том, что 24 марта в 1:54 ночи хост-отправитель `24.3.57.38`, использовав порт `11111`, подключился к TCP-порту получателя `12345` хоста `24.3.21.199`.

```
Mar 24 01:54:58 cc1014244-a kernel: securityalert: tcp if=ef0 from
↳ 24.3.57.38:11111 to 24.3.21.199 on unserved port 12345
Mar 24 03:14:13 cc1014244-a kernel: securityalert: tcp if=ef0 from
↳ 171.214.113.228:2766 to 24.3.21.199 on unserved port 1243
Mar 24 04:45:01 cc1014244-a kernel:  securityalert:  tcp if=ef0 from
↳ 208.61.109.243:3578 to 24.3.21.199 on unserved port 1243
Mar 24 04:45:06 cc1014244-a kernel: securityalert: tcp if=ef0 from
↳ 208.61.109.243:3832 to 24.3.21.199 on unserved port 27347
Mar 24 05:40:42 cc1014244-a kernel: securityalert: udp if=ef0 from
↳ 24.24.100.172:2147 to 24.3.21.199 on unserved port 137
Mar 24 14:56:08 cc1014244-a kernel: securityalert: udp if=ef0 from
63.17.79.40:4294 to 24.3.21.199 on unserved port 137
Mar 24 17:20:44 cc1014244-a kernel: securityalert: tcp if=ef0 from
```

```
↳ 62.6.100.45:1828 to 24.3.21.199 on unserved port 27374
Mar 24 20:50:47 cc1014244-a kernel: securityalert: tcp if=ef0 from
↳ 194.27.62.179:4857 to 24.3.21.199 on unserved port 27374
```

Анализ

Ниже перечислены вопросы, ответы на которые весьма полезны при определении степени угрозы любого взлома. В данном случае они применяются по отношению к предыдущей трассировке.

- **Есть ли доказательства направленности атаки?**

  Да. Трафик от отправителя поступает на интерфейс конкретного хоста.

- **Есть ли записи в архиве?**

  Нет. Все переданные пакеты зафиксированы в отчете.

- **В чем заключается метод атаки?**

  TCP- и UDP-пакеты были направлены на конкретный хост. SYN-пакеты были направлены на TCP-порты 12345, 1243, 27347 и 27374. UDP-пакеты предназначались UDP-порту 137. Отправитель надеется на получение в ответ SYN/ACK-пакета или отсутствие ответа на UDP-пакет. Сканирование портов осуществлялось с разных компьютеров в течение нескольких часов. Все IP-адреса отправителей принадлежат активным хостам, подключенным к Internet, и, по всей видимости, не подменялись.

- **Есть ли доказательства злого умысла хакера?**

  В данном случае выполняется сканирование портов с целью выявления уязвимых мест, а именно:

  - порт 12345 программы Netbus и TrendMicro;
  - порт 1243 троянские программы SubSeven и Backdoor-G;
  - порт 27374 программа SubSeven 2.0;
  - порт 27347 возможно, случайность, опечатка вместо 27374;
  - порт 137 NetBIOS.

  Аналитик сетевого трафика должен сам проверить атакованный хост на наличие троянских программ и убедиться в нормальном функционировании NetBIOS.

- **Отдельный злоумышленник или группа?**

  Судя по информации службы `whois`, IP-адреса отправителей принадлежат хостам различных сетей. Они не связаны ни географически, ни хронологически. На последние адреса, однако, следует обратить особое внимание, так как они принадлежат хостам в Турции. Это вредоносное сканирование, но атакованный хост надежно блокирует атаку.

Степень опасности

Оценить степень опасности этой атаки можно следующим образом.

- **Важность цели** – 2 балла при условии, что сервер не является критически важным.

- **Уровень риска** — 4 балла, так как эти программы атаки способны причинить серьезный вред.
- **Меры противодействия** — 5 баллов при условии, что для операционной системы установлены все программные заплаты.
- **Сетевые меры противодействия** — 0 баллов, так как, по всей вероятности, брандмауэр отсутствует.

Сканирование хостов по FTP

Рассмотрим еще один пример (табл. 16.1), который предоставил нам Эрик Брок (Eric Brock). Для сбора этой информации он использовал брандмауэр WireWall-1.

Анализ

Перед тем как начать анализ атаки, хотим обратить внимание на тот факт, что пакеты поступили в нашу демилитаризованную зону. Очень важно найти записи в архиве. В следующем исследовании мы рассмотрим записи в архиве нашей демилитаризованной зоны. Однако с помощью программы `dshield` (`http:/www.dshield.org/ipinfo.php`) мы можем выполнить поиск по IP-адресам отправителей также и в архивах на других узлах сети. Мы опишем использованный метод сканирования и попытаемся наиболее точно определить предназначение полученных пакетов.

- **Исходные данные**. К нам обращается кто-то, заявляющий, что его IP-адрес `195.243.30.140`.
- **Архив**. В архиве не содержится никаких записей о предыдущих соединениях с этим хостом.
- **Методы**. Отправитель посылает по одному FTP-пакету на каждый хост нашей сети. Пакеты передаются очень быстро.
- **Намерения**. Отправитель стремится найти в нашей сети хосты, которые отвечают на запрос к порту службы FTP.
- **Направление атаки**. Целью сканирования является вся наша сеть без выделения конкретных серверов.
- **Анализ**. Отправитель сканирует всю сеть в поисках FTP-серверов. Возможно, он планирует атаку отказа в обслуживании против FTP-сервера или ищет анонимный FTP-сервер с целью выяснить, можно ли скачать с него какую-то ценную информацию или загрузить в него свои данные.

Степень опасности

При оценке степени опасности атаки используется ряд показателей.

- **Важность цели** — 3 балла, так как не было направленного сканирования отдельных серверов.
- **Уровень риска** — 4 балла, так как существует много уязвимых мест в программном обеспечении службы FTP.

Таблица 16.1. Сводные данные о сканировании хостов по FTP

Идентификатор	Дата	Время	IP-адрес отправителя	Порт отправителя	IP-адрес получателя	Порт получателя	Протокол	Размер
661530	21Feb2000	9:09:24	195.243.30.140	4858	10.10.1.1	FTP	TCP	len 60
661531	21Feb2000	9:09:24	195.243.30.140	4857	10.10.1.0	FTP	TCP	len 60
661532	21Feb2000	9:09:24	195.243.30.140	4860	10.10.1.0	FTP	TCP	len 60
661533	21Feb2000	9:09:24	195.243.30.140	4859	10.10.1.0	FTP	TCP	len 60
...	...	...	...	...	...	...	...	...
661632	21Feb2000	9:09:25	195.243.30.140	1144	10.10.1.252	FTP	TCP	len 60
661633	21Feb2000	9:09:25	195.243.30.140	1145	10.10.1.253	FTP	TCP	len 60
661634	21Feb2000	9:09:25	195.243.30.140	1146	10.10.1.254	FTP	TCP	len 60

- **Системные меры противодействия** — 2 балла, на всех операционных системах установлены последние заплаты, но на некоторых хостах открыты порты службы FTP.
- **Сетевые меры противодействия** — 4 балла, так как брандмауэр блокирует все входящие FTP-пакеты.
- **Степень угрозы** равна 1. Она определяется по формуле:

```
(Важность цели + Уровень риска) - (Системные + сетевые меры
противодействия) = Степень угрозы
```

Размещение датчиков

Сетевая система обнаружения вторжения не будет работать без средств обнаружения (датчиков), а если датчики установлены неправильно, действует неэффективно. Датчики следует устанавливать рядом с брандмауэром.

Перед брандмауэром

Обычно датчики систем обнаружения взлома устанавливают перед брандмауэром в демилитаризованной зоне[3] (рис. 16.4). Таким образом, датчик может регистрировать все атаки, поступающие из Internet. Однако если для проведения атаки используется протокол TCP, и она блокируется брандмауэром или фильтрующим маршрутизатором, то система обнаружения вторжения может ее не выявить. Многие атаки обнаруживаются только при совпадении полученных данных с заданной строкой сигнатуры. Строка символов не отправляется до завершения полной процедуры установки TCP-соединения.

Несмотря на то что некоторые атаки не будут выявлены при установке датчика перед брандмауэром, все равно это лучшее место для размещения датчика. Преимущество заключается в том, что аналитики сетевого трафика могут видеть все атаки, которым подвергается их узел и брандмауэр. Один из рецензентов книги сказал об этом так: "Перед брандмауэром мы выявляем атаку, а за брандмауэром — взлом".

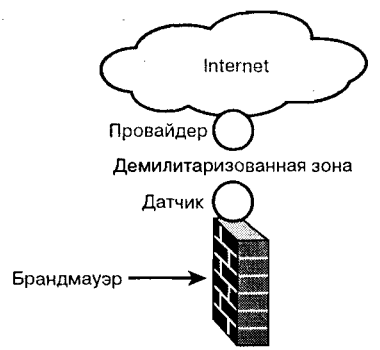


Рис. 16.4. Датчик (детектор событий) установлен в демилитаризованной зоне

[3] Демилитаризованная зона представляет собой зону между провайдером Internet и внешним интерфейсом брандмауэра.

В конце 1997 и начале 1998 годов на многих узлах были зарегистрированы попытки доступа к TCP/UDP-порту 111 (`portmapper`). На компьютерах с открытыми портами `portmapper` скорее всего был запущен демон `rpc.statd`. Я использовал программу сканирования уязвимых мест на двух узлах, чтобы оценить степень риска. Сканирование выявило более 50 систем, которые бы ответили на запрос `rpcinfo -p` (что означает наличие незащищенной службы `portmapper`). Дальнейшее исследование показало, что на этих хостах был запущен демон `statd`. Таким образом, хосты сетей, которые подвергались атакам, были уязвимы. Однако на обоих узлах брандмауэры блокировали атаки на службу и любые попытки непосредственного доступа к демону `statd`. Это лишний раз убедило меня в необходимости приложить все усилия, чтобы защитить службы преобразования портов и контролировать установку всех последних заплат для службы `statd`. Более подробную информацию по этой теме можно получить по адресу `www.cert.org/advisories/CA-97.26.statd.html`.

Датчики за брандмауэром

Некоторые специалисты полагают, что датчики следует устанавливать за брандмауэрами. Для этого утверждения есть несколько причин. Если злоумышленник сможет обнаружить датчик, то, проведя атаку на этот датчик, он сможет снизить риск регистрации своих действий. Системы за брандмауэрами менее уязвимы, чем системы перед ними. Если датчик установлен за брандмауэром, он будет получать меньше ненужной информации, и снизится число ложных тревог. Кроме того, такое размещение датчика позволяет проверить правильность настройки брандмауэра (например, если он пропускает атаки, которые должен был бы блокировать).

Абсолютно справедливо, что правильно настроенные брандмауэры блокируют большинство простых попыток проведения атак. И правда то, что слишком много внимания уделяется обнаружению и исследованию этих простых атак.

И перед, и за брандмауэром

Чем больше, тем лучше. Для меня это не просто поговорка. Я устанавливаю датчики с обеих сторон брандмауэра. Если ваша организация может себе позволить установить датчики и перед, и за брандмауэром, то это даст определенные преимущества, например:

- не нужно угадывать, пропустит ли брандмауэр атаку;
- есть возможность выявлять атаки внутри локальной сети;
- можно обнаружить неверную настройку некоторых систем, которая не позволяет их пакетам проходить через брандмауэр (и тем самым оказать помощь системному администратору).

Если в организации используется дорогостоящая система обнаружения вторжений, то, может, она того и не стоит. Лучше установить датчики перед и за брандмауэром, и внутренний датчик уведомит об опасности.

Неверно сконфигурированные системы

Системы обнаружения вторжений, а точнее обслуживающие их специалисты, должны уметь находить и устранять неисправности в локальных сетях. Когда я устанавливал систему Shadow, мы

обычно тратили неделю или две на устранение неполадок в локальных сети. Ситуация не изменилась и по сей день. Ниже перечислены основные причины неполадок в сетях.

- Широковещательные рассылки `localhost` `127.0.0.1` или `127.0.0.2` во внутренней подсети.
- Неправильная конфигурация файлов DNS. Они читаются справа налево. Например, если адрес сети `172.20.0.0/24`, и вы обнаружили хост (`172.20.30.40`), выполняющий широковещательную рассылку по адресу `255.30.20.172`, то, скорее всего, кто-то не знает, что файлы конфигурации домена читаются справа налево.
- Некорректная маска подсети. Широковещательные рассылки отправляются по адресу `172.20.255.255` вместо `172.20.30.255`.
- Потайные ходы. Когда регистрируется пакет, поступающий из Internet на адрес `172.20.30.255` (*используем адрес сети из предыдущего примера*), то весьма вероятно, что хакером установлен потайной ход. Пакет, предназначенный для передачи в локальной сети, не должен существовать за пределами брандмауэра.

Другие места установки датчиков

Чаще всего датчики устанавливают перед брандмауэром, хотя это не единственное приемлемое место для установки датчика. Многие системы обнаружения вторжений используются организациями в других местах локальной сети, включая следующие:

- сети партнеров, с которыми устанавливаются прямые соединения, заказчиков или поставщиков, нередко работающих на хостах вашей сети, защищенной брандмауэром;
- особо важные места, например научные лаборатории или сети бухгалтерии;
- сети с большим количеством временных пользователей;
- подсети, которые могут стать мишенью для атак извне, или в которых выявлены признаки взлома или другие отклонения от нормальной работы.

Последний вопрос, связанный с размещением датчиков, – к чему их подключать. Современные сети практически полностью представляют собой коммутируемые сети VLAN (Virtual Local Area Network – виртуальная локальная вычислительная сеть). Датчики *могут* работать в таких средах. Однако, если связующие порты коммутатора сконфигурированы неверно, то обнаружение взлома будет невозможным. Не забывайте, что использование несколькими сетями VLAN общего связующего дерева (Spanning Tree) дает большую нагрузку на коммутатор. Если датчик должен работать в коммутируемой сети, то его обязательно нужно протестировать. Протокол TCP является дуплексным, и аналитик сетевого трафика должен убедиться, что датчик получает данные от каждой стороны соединения. Кроме того, датчик должен без ошибок передавать данные из коммутируемой сети. Возможно, что для датчика потребуется установка двух сетевых адаптеров. Первый из них, присоединенный к связующему порту, позволяет контролировать трафик, передающийся в неразборчивом режиме (перехват всех пакетов, независимо от того, предназначались ли они датчику). Второй сетевой адаптер, который будет взаимодействовать с хостом аналитика, можно "разместить" в отдельной сети VLAN. Безусловно, трата денег на решение проблемы – один из лучших методов в технологии обнаружения вторжений. Для того чтобы устранить проблемы с конфигурацией или нагрузкой в сети, можно предпринять следующие действия.

- Установить сетевые ответвители, которые подключаются непосредственно к среде передачи данных и позволяют датчику перехватывать все данные, проходящие через этот ответвитель.
- Использовать коммутатор компании TopLayer (`www.toplayer.com`), позволяющий копировать передаваемые в сети данные в систему обнаружения вторжений.
- Задействовать коммутаторы Cisco Catalyst 6000, поддерживающие дополнительную опцию Policy Feature Card, которая также позволяет управлять процессом копирования трафика в систему обнаружения вторжений.

Системы доставки и системы отправления сообщений

После определения места установки датчика нужно решить, как из него извлекать данные. Обычно (по крайней мере при первой установке датчика или генератора событий) мы рекомендуем использовать режим, в котором отчеты о произошедших событиях доставляются на компьютер аналитика. Когда датчик выявляет событие, он создает пакет с отчетом и отправляет его на хост аналитика сетевого трафика. Очевидно, что при этом должен применяться какой-то протокол наподобие SMTP. В большинстве коммерческих систем обнаружения вторжений используются собственные протоколы для организации соединений между датчиком и компьютером аналитика. Покупатели систем обнаружения вторжений при сравнении разных систем в первую очередь хотят иметь возможность реагирования на атаки "в реальном времени".

Системы обнаружения вторжений на основе доставки сообщений

Одним из важнейших аспектов при выборе системы обнаружения вторжений является то, насколько их работа может отличаться от нормальной. Все кажется очень удачным, когда вы смотрите на систему обнаружения вторжений, уведомляющую об атаках на консоль по электронной почте, пейджеру или по сотовому телефону. Но проходит только несколько недель, и все эти удобные уведомления в реальном времени приходится отключать. Даже самому лучшему аналитику, полностью преданному своей работе, вряд ли захочется получать уведомления о ложных тревогах в три часа ночи.

Реагирование в реальном времени невозможно, пока средства обнаружения взлома не будут интегрированы в сетевые коммутаторы, операционные системы и программы. Несмотря на это потенциальные пользователи будущих систем обнаружения вторжений хотят, чтобы информация об атаках передавалась им как можно быстрее. Поэтому принцип доставки сообщений будет правильным выбором для сетевых систем обнаружения вторжений.

Доставка сообщений имеет один очень серьезный недостаток. Если система обнаружения вторжений на основе этого принципа в ответ на выявление атаки должна генерировать пакет, а за датчиком можно проследить, то со стороны можно легко выяснить его настройки. Со временем хакер сможет определить, какие пакеты датчик игнорирует. Такие усилия и терпеливость не присущи неопытным хакерам, применяющим готовые атаки, но профессиональные взломщики почти наверняка будут вести себя именно так. Очевидным решением этой проблемы является выдача отчетов о событиях по расписанию через определенные промежутки времени в виде потока данных. Это позволяет провести те же ответные действия, что и в реальном времени, только с небольшой задержкой и при этом скрыть, что именно обнаруживает датчик. Если никаких атак не выявлено, поток данных состоит из зашифрованных нулей.

Рассмотрим различия систем доставки и получения сообщений (рис. 16.5). В целом принцип доставки сообщений более подходит для обнаружения взлома.

Одной из лучших реализаций систем получения сообщений является скрытый датчик, который может применяться при расследовании вредоносных действий. Его можно настроить на конкретную компьютерную систему. Кроме того, такой датчик может беспрепятственно пропускать информацию, пока в передаваемом трафике не будет обнаружена ключевая строка, после чего датчик начинает фиксировать поток передаваемых данных. Большинство анализаторов пакетов, созданных хакерами для перехвата идентификаторов и паролей пользователей, являются системами получения сообщений. Они ведут поиск до тех пор, пока интересующая их информация не будет обнаружена.

Рис. 16.5. Доставка или получение?

Консоль аналитика

Итак, выбраны место для установки датчиков и метод получения уведомлений о значимых событиях (доставка, получение или оба метода). Наконец-то можно приступить к работе. Аналитик обнаружения взлома работает за своей консолью. Система обнаружения вторжений часто выбирается исходя из ее стоимости, но не стоит экономить, выбирая консоль, поскольку она — очень важный элемент. При покупке системы обычно обращают внимание на наличие следующих возможностей:

- работа в реальном времени;
- автоматический ответ на атаку;
- обнаружение абсолютно всех атак;
- запуск на платформе Windows XP/UNIX/Commodore 64 (на любой платформе, используемой в организации).

Хорошо, система приобретена, но будет ли она использоваться? Я был на нескольких узлах, на которых коммерческие системы обнаружения вторжений были установлены очень давно, и, хотя они были по-прежнему подключены к сети, клавиатура консоли аналитика уже покрылась слоем пыли. После использования системы обнаружения вторжений на протяжении нескольких месяцев приоритеты ее основных характеристик формируются так:

- быстродействующая консоль аналитика;
- лучшая обработка ложных тревог;

- фильтры для вывода определенной информации;
- возможность отметить ранее рассмотренные события;
- возможность детализации зафиксированных событий;
- возможность сопоставления;
- улучшенная выдача отчетов.

Подавляющее большинство консолей коммерческих систем обнаружения вторжений были настолько неудачны, что Министерство обороны было вынуждено разрабатывать свои собственные проекты. Некоторые из них стали весьма популярными. Большинство организаций не могут позволить себе создавать альтернативные интерфейсы, поэтому приведенный перечень поможет выбрать хорошую систему обнаружения вторжений. В следующих разделах характеристики консоли рассмотрены более подробно.

Быстродействие

Способности человека часто растрачиваются впустую. Именно это происходит, когда опытный аналитик сетевого трафика ожидает реакции компьютера. Обнаружив атаку, аналитик начинает сбор дополнительных сведений. При этом он ждет появления окна и вдруг понимает, что не может вспомнить, что он вообще делает.

Недавно в разговоре с менеджером по продажам компании, производящей системы обнаружения вторжений, я заметил, что интерфейс системы работает слишком медленно. Он, естественно, посоветовал приобрести быстродействующий компьютер (это был двухпроцессорный Pentium IV с частотой 1,2 ГГц с объемом оперативной памяти 1 Гбайт – вполне современный компьютер для января 2002 года). Существует простой способ улучшить производительность консоли аналитика, при котором система находится в постоянной готовности к выдаче информации об атаках с высоким приоритетом. При этом анализ начинается по первому щелчку кнопкой мыши – компьютер ждет аналитика, а не наоборот.

Лучшая обработка ложных тревог

Ложные тревоги случаются постоянно. Иногда их невозможно устранить без одновременного пропуска настоящих атак. Что же с ними делать?

Хорошим примером может послужить Web-атака с помощью Code Red. Если мы создадим фильтр, который будет пропускать запросы к порту 80 (а большинство так и делает), то мы рискуем пропустить огромное количество атак. Если не использовать этот фильтр, можно столкнуться с большим числом ложных тревог (ложными эти тревоги будут при условии, что не запущена уязвимая версия сервера IIS, и не следует беспокоиться о возможности успеха атаки Code Red). Поскольку червь Code Red поражает Windows-системы, можно постараться решить эту проблему, создав более точный фильтр. Если язык написания фильтров для системы обнаружения вторжений это позволяет, то мы можем добавить в него информацию, полученную в результате пассивного анализа атаки для систем Windows. Например, по умолчанию Windows-системы устанавливают в отправляемых пакетах значение поля TTL равное 128, а значение размера окна – в диапазоне от 5000 до 9000 для Windows NT, и от 17000 до 19000 для Windows 2000.

Поэтому в фильтре можно задать игнорирование пакетов со значениями TTL больше 128 и значениями размера окна, которые не попадают в используемые диапазоны. Данные таких пакетов по-прежнему регистрируются, но аналитику не выдается предупреждение об атаке. Когда аналитик относит какое-либо событие к разряду ложных тревог, на консоли должна отображаться обычная информация, но с дополнительными данными, которые позволят аналитику принять правильное решение.

Ответственность за ложные тревоги

Поставщики систем обнаружения вторжений обязаны стремиться улучшать методы обработки ложных тревог. Набор правил Snort становится все лучше, предоставляя дополнительную информацию в файле помощи, которая позволяет аналитику определить возможность ложной тревоги. Но этого недостаточно. Поставщики должны минимизировать число ложных тревог, которые представляют собой основное затруднение для правильной интерпретации значимых событий. Для этого следует усовершенствовать фильтры (т.е. в эти фильтры должна быть добавлена возможность настройки), а ненужные фильтры, вызывающие слишком много ложных тревог, нужно удалять. Кроме того, в документации должны быть точно описаны ситуации, при которых фильтры сообщают о ложных тревогах.

Фильтры для вывода определенной информации

Рассмотренный выше метод обработки ложных тревог используется в некоторых коммерческих системах обнаружения вторжений и должен считаться минимально допустимым уровнем. Для выявления максимального числа значимых событий придется иметь дело с некоторым числом ложных тревог. Единственным способом борьбы с этим является применение фильтров для вывода на монитор только определенной информации. Эта идея не нова, в сетевых средствах обнаружения вторжений, например Sniffer, всегда наряду с фильтрами сбора данных применялись фильтры для вывода информации.

Маркировка ранее рассмотренных событий

Если вы не являетесь аналитиком сетевого трафика высшего уровня (например, инспектором или инструктором), то не стоит тратить драгоценное время на исследование событий, которые уже были проанализированы. После изучения события его нужно пометить. Все очень просто. Даже Web-броузеры отмечают другим цветом посещенные ссылки. В идеале это должно быть похоже на функцию редактирования в современных текстовых процессорах, таких как Microsoft Word, — событию должен присваиваться ярлык с указанием даты и времени его анализа, именем пользователя, который выполнял этот анализ, а также с информацией о том, было оно расценено как ложная тревога или занесено в отчет как настоящая атака.

Детализация событий

Интерфейс системы обнаружения вторжений не должен пугать начинающего пользователя. Когда в организации начинают работать с системой обнаружения вторжений, пользователям системы достаточно иметь удобный графический интерфейс, который бы выдавал пиктограмму атаки, дату, время, а также IP-адреса отправителя и получателя. Радость пользователей обычно заканчивается, когда они понимают, что выдается отчет о ложной тревоге. С этого момента аналитик

хочет видеть все данные, и они должны быть доступны по одному щелчку кнопкой мыши. Возможность детализации — очень эффективное средство. Она позволяет увидеть общую картину событий. А при желании, один раз щелкнув кнопкой мыши, аналитик может узнать подробности любого события. Для этого не требуется менять интерфейс (что значительно замедляет анализ) и применять отдельную программу для исследования данных.

Детализация событий невозможна при отсутствии зарегистрированных данных (по крайней мере, заголовков пакетов). Аналитик не должен сообщать о выявлении события, которое он не может проверить!

Сопоставление

Каждому аналитику, обнаружившему атаку, доводилось размышлять, не видел ли он этого IP-адреса раньше? Аналитики сетевого трафика на постоянно атакуемых узлах обнаруживают от 15 до 60 значимых событий в день. После нескольких недель работы накапливается слишком много IP-адресов нарушителей, чтобы всех их помнить. Консоль аналитика должна хранить список узлов, о которых уведомлялось ранее, и выделять различными цветами IP-адреса отправителей согласно этому списку.

Улучшенная выдача отчетов

В повседневной работе аналитиков сетевого трафика используются отчеты двух видов: отчеты о выявлении событий и итоговые отчеты. Отчеты о событиях содержат подробную информацию о выявленных происшествиях, итоговые отчеты позволяют аналитику следить за тенденциями атак с течением времени, а руководителю — понимать, на что расходуются деньги.

Отчеты о выявленных событиях

Отчеты о выявленных событиях выдаются или непосредственно по факту события или в конце дня. Как правило, они отправляются по электронной почте. Система обнаружения вторжений должна позволять отправку отчетов по разным адресам и их шифрование с помощью PGP. Отчеты могут отправляться группам, специализирующимся на сборе и анализе информации о происшествиях в сетях, например на Incidents.org или Security Focus, или специалистам по безопасности в группах CIRT и FIRST. Если нужно скрыть действия от хакера или планируются ответные действия, то можно сохранять отчет как уведомление, добавленное к записи. Аналитик должен иметь возможность сообщать о каждом обнаруженном и отображенном на консоли событии с помощью одного щелчка кнопкой мыши. В ответ на это система должна составить отчет, который аналитик просматривает и снабжает пояснениями перед отправкой в вышестоящую инстанцию.

При покупке системы обнаружения вторжений сядьте за консоль и проверьте, сколько времени потребуется для сбора информации, необходимой для составления отчета о событии и отправки этого отчета по электронной почте (или в другом формате, например XML) команде CIRT или FIRST. Если доступ к необработанным (raw) или вспомогательным данным получить невозможно, то эта система не стоит вашего внимания. Если этот процесс потребует более пяти минут, а ваша

организация планирует уведомлять других о значимых событиях, то над покупкой еще стоит подумать. Если же сбор информации, включая необработанные и дополнительные данные, будет выполнен в течение двух минут, то, пожалуйста, отправьте мне письмо по электронной почте, я тоже хочу купить такую систему.

Итоговые отчеты

Руководство организации зачастую также хочет получать сведения о выявленных атаках, направленных на узлы их сетей. Однако составление отчетов по факту события или ежедневных отчетов будет слишком утомительным и не позволит рассмотреть ситуацию в целом. Поэтому принято создавать еженедельные или ежемесячные отчеты. В общем случае, чем выше ранг руководителя, тем реже ему нужно отправлять отчеты.

Обнаружение вторжений на хостах и в сети

Чем больше информации будет у аналитика, тем выше вероятность, что он решит проблему, связанную с обнаружением вторжения. Откуда лучше получать сведения: с отдельных хостов или с точки входа в сеть? Если прочитать описания программ для обнаружения вторжений на хостах, то можно сделать вывод, что метод контроля взломов на хостах, безусловно, лучше. Но если прочитать описания сетевых систем обнаружения вторжений, то мнение будет противоположным. Очевидно, что в своей сети хотелось бы иметь обе возможности, при этом, предпочтительно, интегрированные. Пожалуй, лучше всего охарактеризовать преимущества обоих методов можно с помощью описания минимально необходимых возможностей обнаружения взлома для средней локальной сети, подключенной к Internet (рис. 16.6).

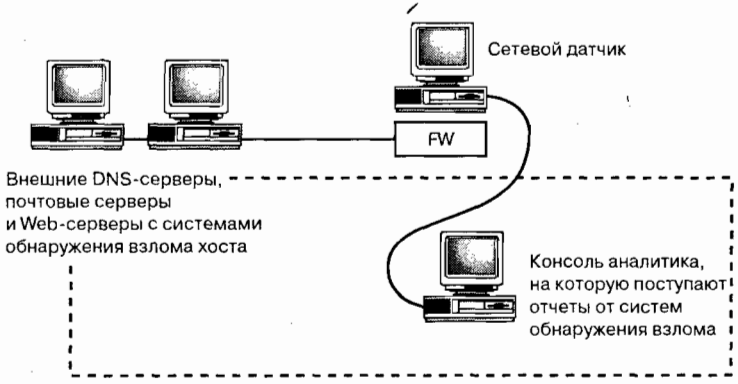


Рис. 16.6. Стандартная архитектура сети для небольших организаций

Датчик размещается перед брандмауэром и позволяет выявлять все атаки, поступающие из Internet. На DNS, почтовые и Web-серверы направлено около трети от общего числа атак. Эти компьютеры должны взаимодействовать с другими удаленными компьютерами по Internet, и поэтому защитить их можно только частично. В связи с высокой степенью риска компрометации этих компьютеров на них должны устанавливаться системы обнаружения вторжений, которые будут

отправлять уведомления на консоль аналитика. Таким образом, даже для небольших организаций необходимы оба вида обнаружения взломов: и на хостах, и в сети. Чем больше организация, тем больше требуется мер противодействия атакам.

Эти минимальные функции защиты не учитывают внутренних угроз. В литературе (особенно по защите хостов) часто указывается опасность атак локальных пользователей. Я сам не раз видел результаты исследований и статистические выкладки, которые ясно дают понять, что *большинство* взломов осуществляются именно локальными пользователями. Но ситуация меняется, и многие эксперты сходятся во мнении, что теперь большинство атак проводятся по Internet. Программы атаки стали огромной проблемой, однако троянские программы и вирусы для сбора сведений после установки в системы можно считать "локальными". Если главной проблемой организации являются атаки своих же пользователей, для создания минимально приемлемого средства обнаружения вторжений необходим ряд дополнительных мер:

- используйте ответвители или связующие порты коммутаторов таким образом, чтобы иметь возможность перехватывать весь проходящий трафик;
- настройте фильтры на сенсоре в демилитаризованной зоне так, чтобы они не игнорировали пакеты, передаваемые между локальными хостами. Исходящий трафик должен контролироваться так же, как и входящий. Это особенно важно в связи с увеличением числа доступных программ атаки;
- настройте фильтры своего пограничного маршрутизатора или брандмауэра на пропуск только тех исходящих пакетов, в которых IP-адреса отправителей соответствуют IP-адресам хостов вашей сети. Это называется *фильтрацией исходящего трафика* (egress filtering), о которой есть очень много информации на Web-сайте Incidents.org (`www.incidents.org/defend/egress.php`);
- устанавливайте сетевые датчики в особо важных участках, например, в научных лабораториях или сетях бухгалтерии;
- устанавливайте хосты-ловушки в привлекательных местах и сохраняйте на них подложную информацию, которая может заинтересовать хакера;
- время от времени добавляйте дополнительные датчики в пользовательские сети для случайной проверки;
- обязательно установите системы обнаружения вторжений на всех серверах, а также на хостах всех высших должностных лиц организации. Многие персональные брандмауэры можно приобрести менее чем за $75 на один компьютер, и они очень просты в использовании (например, Tiny, ZoneAlarm, BlackIce и Internet Security);
- установите систему поощрений для тех, кто уведомляет о сотрудниках, которые неверно обращаются с данными или крадут информацию.

Резюме

Очень часто свойства, которые казались наиболее важными при покупке системы обнаружения вторжений, при использовании на практике становятся менее привлекательными. В первую очередь это относится к возможности отправки уведомлений об атаках на пейджер аналитика.

По различным причинам системы обнаружения вторжений не могут выявлять все происходящие события. Почему? В этой главе рассмотрено несколько причин. Событие происходит в другой сети. Система обнаружения вторжений вышла из строя или не поддерживает работу с конкретным протоколом. Возможно, системой достигнута максимальная скорость обработки пакетов, и пакет отброшен. Более того, сетевые системы обнаружения вторжений ограничены возможностями связующих портов коммутатора, а зашифрованные пакеты не идентифицируются.

Аналитик может добиться наибольшей пользы от системы обнаружения вторжений только тогда, когда понимает, что он ищет, и соответствующим образом настраивает систему.

Если у вас есть только один датчик, установите его перед брандмауэром.

Когда существует уверенность, что узел подвергается целенаправленной атаке, и злоумышленник знает о типе операционной системы атакуемого хоста, немедленно принимайте дополнительные меры противодействия.

По возможности устанавливайте системы обнаружения вторжений как для отдельных хостов, так и для всей сети.

17

Безопасность в организации

Какое отношение имеет управление рисками к обнаружению взлома? В каждой организации так или иначе принимается решение относительно допустимых рисков. Распределенные атаки отказа в обслуживании, которые стали широко известны в феврале 2000 года, и атаки Code Red в 2001 году продемонстрировали, что нужно учитывать и то, какой риск позволяют себе другие пользовали. Безопасность моей сети частично зависит от безопасности вашей. В этой главе рассказывается, как доказать своему начальству, что обнаружение взломов является частью общей системы безопасности. Главная цель сетевых систем обнаружения вторжений – выявление атак, направленных против систем защиты на периметре локальной сети таким образом, чтобы пользователи могли быть уверены в успешном отражении этих атак. Другими словами, обнаружение взлома может служить средством для определения показателей, необходимых для грамотного управления степенью риска. В этой главе раскрывается связь между методами управления рисками и концепциями обнаружения вторжений.

Модель безопасности организации

Для управления риском нужна какая-то модель – способ описать проблему и действия, которые позволяют ее решить. Простым примером является модель списка Top Twenty (`www.sans.org/top20.htm`). В этом списке перечислены двадцать основных уязвимых мест, используемых хакерами, а также средства для их поиска и методы устранения. Если многие люди воспользуются этой информацией и защитят свои системы, то хакерам потребуется гораздо больше времени на компрометацию какого-то компьютера, и общий риск снизится. Эту модель построил мой хороший друг Алан Пэллер (Alan Paller), работающий начальником отдела исследований института SANS. За время нескольких долгих перелетов, консультируясь с лучшими специалистами в области компьютерной безопасности, Алан создал еще одну более сложную модель.

Используя эту модель, я был поражен результатами. Она мне очень помогла при написании программ курса для получения сертификата GIAC и курса повышения квалификации в институте SANS. После двадцати лет напряженной работы на правительство уже трудно вспомнить, с чего все начиналось.

Работая в войсках противоракетной обороны (Ballistic Missile Defense Organization), я использовал модель безопасности для определения основных приоритетов из набора часто несовместимых требований. В правительственных организациях настолько много работы, что нужно записывать все, что ты делаешь. В организациях SANS и GIAC слово не расходится с делом. Если мы чему-то учим, то мы это и используем. Поэтому я стремился применять одну и ту же модель безопасности в различных организациях нашего постоянно меняющегося мира. Я не разрабатывал эту модель. Это сделали Алан Пэллер, Джин Шульц (Gene Schultz), Мэт Бишоп (Matt Bishop) и Хэл Померанц (Hal Pomeranz). Но я успешно использовал ее в прошлом, и, надеюсь, что она поможет нашим читателям в будущем. Здесь она описана с точки зрения обнаружения взлома, но, конечно, ее можно применять и для решения более глобальных проблем. Результаты работы моих коллег представлены ниже. Вместо трех компонентов самой простой модели безопасности (определение двадцати наиболее уязвимых мест, сканирование системы и устранение обнаруженных уязвимых мест) в этой модели заданы семь основных принципов обеспечения безопасности.

1. Установить политику безопасности.
2. Проанализировать возможные риски; исследовать уязвимые места.
3. Организовать инфраструктуру безопасности.
4. Разработать методы контроля и установить стандарты для каждой технологии.
5. Определить, какие ресурсы будут доступны пользователям, задать приоритеты для мер противодействия незаконным операциям и реализовать наиболее строгие меры противодействия, какие только можно себе позволить.
6. Выполнять периодические проверки и (по возможности) тестирование безопасности системы.
7. Установить систему обнаружения вторжений и определить порядок ответных действий в случае атаки.

Политика безопасности

Остановитесь! Не закрывайте книгу только из-за словосочетания *политика безопасности*. По своему опыту обучения аналитиков сетевого трафика и специалистов по обнаружению взлома я знаю, что, как только речь заходит о составлении правил политики безопасности в организации, мои студенты сразу теряют интерес к лекции. Но ведь установка фильтров для выявления взлома — это только одна из мер обеспечения защиты, не так ли?

Вспомните, что набор правил для фильтров системы обнаружения взлома называется *политикой*. Это название используется в большинстве коммерческих систем, и оно абсолютно справедливо, так как именно фильтры определяют политику безопасности. Брандмауэр — это только механизм, который реализует заданную политику работы в сети. Поэтому давайте не будем думать о политике

безопасности как о куче ненужных бумаг, на написание которых требуется несколько недель, и которые затем просто покрываются пылью. Политика безопасности является пропуском, набором полномочий, предоставленных аналитику сетевого трафика с целью создания активной системы защиты с использованием, например, брандмауэров и систем обнаружения взлома. Все верно, политика предоставляет полномочия для тех или иных действий! Ее стержнем является система обнаружения вторжений, которая служит как средство контроля. Но никогда не следует контролировать действия пользователей, не получив на это необходимых полномочий.

Должное отношение к организации работы

И риски, и уязвимые места будут рассмотрены далее, но важное место занимает и *правильный метод организации работы*. Хотя в каждой организации есть свой опыт организации труда, ответа на все вопросы нет ни у кого. Технологии изменяются так быстро, что аналитик не может обладать знаниями во всех областях. Лучшим решением проблемы является обмен опытом. Одни из лучших моих воспоминаний связаны с участием в совместных проектах института SANS. Многие из них назывались "*Step by Step*" ("*Шаг за шагом*"), например "Обеспечение безопасности Windows NT. Шаг за шагом". Это была работа многих профессионалов, которые объединились в работе над проектом, чтобы поделиться своими знаниями.

Инфраструктура безопасности

Роберт Пиви (Robert Peavy), начальник отдела безопасности и контрразведки в BDMO, выступил с докладом на Федеральной конференции по компьютерной безопасности, который назывался "Безопасность как источник дохода. Как продать средства защиты своему руководству".

Как и многие из тех, с кем я был знаком, Роберт Пиви понял, что для обеспечения защиты, особенно надежной защиты, нужны люди. Это так же справедливо для обнаружения взломов, как и в других областях безопасности. Специалисты по обнаружению взломов постоянно находятся на передовой. Они часто несут персональную ответственность за успешные атаки и компрометацию систем. Они работают на износ и часто сталкиваются с различными проблемами, особенно если они должны организовать ответные действия на выявленную атаку. Им необходимо постоянно обновлять свои знания, но мест повышения квалификации не так уж много. Какой из этого вывод? Вывод такой, что мудрые руководители организаций оказывают особое внимание специалистам в области компьютерной безопасности и совершенствуют необходимую инфраструктуру.

Основные меры противодействия

Прошлой ночью я очень испугался. Я запустил программы сканирования уязвимых мест для проверки сетей нескольких организаций. Компьютеры этих сетей работали под управлением UNIX и Windows 2000/XP. Я был поражен количеством систем, программное обеспечение которых имело хорошо известные уязвимые места, а также количеством компьютеров, по-прежнему использующих протокол SNMP. На тот момент прошло две недели со времени появления набора

тестов CERT для проверки SNMP и PROTOS. Этот набор тестов выявил тысячи проблем. Неужели повторится ситуация с демоном `rstatd`?

Начиная с 1997 года продолжает увеличиваться число систем UNIX на платформе Sun Solaris, скомпрометированных с помощью программы атаки на переполнение буфера посредством использования демона `rstatd`. Существует несколько программ атаки на переполнение буфера для взлома DNS-серверов, так что такой взлом вполне возможен. Я выполнил сканирование UNIX-системы, не защищенной брандмауэром, и обнаружил открытые порты для служб `echo`, `chargen`, `portmapper` и еще нескольких RPC-служб. Это напоминает мне младшие классы в школе, когда мы вешали на спины своих товарищей наклейки с надписью "Ударь меня".

Как определить основные приоритеты мер противодействия, когда все кажется одинаково важным? Если злоумышленник может по Internet легко провести атаку на уязвимое место, то нужно решать, когда придется устранять проблемы: до компрометации хоста или после? Я не устаю удивляться количеству организаций, которые не находят времени на установку нормальной защиты, но находят время на устранение последствий взлома.

Периодические проверки

Проснитесь! Специалист по обнаружению взлома обязан знать это! Время от времени нужно проверять набор фильтров системы обнаружения вторжения. Работая над проектом системы Shadow, я заставлял себя каждые несколько месяцев запускать дополнение к нашему набору фильтров и искать в накопившихся отчетах необычные события. Необходимо постоянно обновлять набор фильтров для снижения количества невыявленных атак. Если в месячном отчете о выявленных атаках перечислены те же атаки, что и три месяца назад, то это плохой знак.

Как еще улучшить фильтры, кроме их обновления при выявлении обычно игнорируемых событий? Отличным источником информации о новых атаках, для каждой из которых может потребоваться отдельный фильтр, является список рассылки `bugtraq`. Еще один прекрасный способ быть постоянно в курсе всех последних событий заключается во взаимодействии с другими специалистами по борьбе со взломом компьютеров.

Выполнение периодических проверок — более общий принцип безопасности, чем просто контроль текущего набора фильтров. Аналитик сетевого трафика должен регулярно проверять набор фильтров брандмауэра. Можно найти какие-то ошибки. При установке брандмауэра используется четкий и весьма жесткий набор ограничивающих правил. Однако со временем в коммерческих интересах и для поддержки новых служб могут быть сделаны какие-то исключения из этого набора правил. Кроме того, от случая к случаю администратор брандмауэра может добавлять специальные правила "только для тестирования", а потом забывать о их существовании. Аналитик может считать UDP-порты полностью заблокированными, а на самом деле это не будет соответствовать действительности. Проконтролировать эти изменения очень сложно, они происходят неожиданно в течение длительного периода нормальной работы и чем-то напоминают крайне медленное сканирование. Я даже думаю, что существуют специальные сканирующие службы, которые отслеживают подобные изменения и сохраняют информацию в базах данных. Подтверждение этому можно найти на сайте `www.qualys.com`.

Ответные действия

Подробное обсуждение последовательности ответных действий при обнаружении взлома выходит за рамки этой книги, но я хочу остановиться на вопросах, которые касаются модели безопасности. Вы пытались получить сертификат администратора CPR? Как бы вы себя чувствовали, если бы пришлось заняться администрированием через 3, 6 или 12 месяцев по окончанию обучения? Я думаю, что меня можно назвать более-менее опытным специалистом по обнаружению вторжений, но если бы я не выявлял атак на протяжении нескольких месяцев, то и мне пришлось бы нелегко.

Какое отношение имеет реакция на компрометацию к обнаружению вторжения? Прямое! Аналитик, скорее всего, сам поднимет тревогу. В организации с многоуровневой системой обработки событий аналитик может предоставлять свою информацию другим специалистам более высокого ранга. В других организациях обработку событий выполняют специалист по взлому и один или два системных администратора. Прочтите внимательно в главе 18, "Ответные действия", раздел "Атака: ответные действия аналитика" и обратите внимание на действия, которые необходимо выполнить перед ответом на серьезный взлом. Эта информация наверняка поможет в сложный момент.

Определение риска

Невозможно эффективно управлять риском, если не знать, в чем он заключается. Риск лежит в области неопределенного. Если все ясно, неопределенности нет, то нет и риска. Прыжок без парашюта с высоты двух километров не является рискованным. Это просто самоубийство. Вероятность смерти крайне близка к 100%, а выживания — практически равна 0. Однако все равно есть риск при прыжке с самолета даже с самым лучшим парашютом.

Давайте применим эту концепцию к фильтрам маршрутизатора. Во многих случаях эти фильтры работают при "попытке соединения", т.е. проверяются номера портов устанавливаемого соединения. Попытка установить соединение с TCP-портом 25 расценивается как попытка доступа к службе `sendmail`, и выполняется предопределенное действие. Однако любая служба может работать через любой порт. Здесь есть неопределенность, а значит, и риск принять неверное решение. Особенно часто происходят ошибки при назначении предопределенных действий в ответ на попытки доступа ко временным портам (номера больше 1024).

Аналитик сетевого трафика должен знать степень неопределенности для конкретного фильтра. Например, фильтры для выявления SYN-наводнений обладают высокой степенью неопределенности. Если аналитик будет уведомлять о каждой ложной тревоге, то CIRT будет пренебрегать его сообщениями. Таким образом, вероятность ошибочного действия при использовании фильтра повышает риск для организации аналитика, особенно в сложных ситуациях. И, наоборот, опытный аналитик знает об условиях, при которых фильтр работает правильно, и условиях, которые приводят к ошибке. Такие аналитики способны "читать между строк".

Возможно, вас не впечатляет простая проблема потери репутации. Та же проблема неопределенности фильтров становится намного интереснее, если на узле применяются автоматические ответные действия.

Необходимо вкратце упомянуть еще один потенциально опасный результат неопределенности применения фильтров для обнаружения атак. Некоторые поставщики коммерческих систем обнаружения вторжений предоставляют перечень используемых фильтров. Иногда эти фильтры перечисляются в порядке вероятности появления ложных тревог, а иногда указываются и причины, по которым это может происходить. Это очень полезно для аналитика. Но что если компания заявляет: ложные тревоги для этого фильтра невозможны? Значит, нет неопределенности и нет никакого риска. После небольшого исследования оказывается, что ложные тревоги все таки случаются. Это может подорвать всякое доверие к компании-производителю. Так было и у меня. Я решил протестировать несколько фильтров, которые в соответствии с документацией не генерировали ложных тревог, и обнаружил, что дела обстоят не совсем так. Это меня разозлило. Почему они сказали, что нет никаких ошибок, если они есть? Но потом я успокоился и подумал: зачем ругать компанию, которая предоставила наиболее полную документацию по фильтрам по сравнению с другими коммерческими системами обнаружения вторжений? Поэтому я просто обновил свою копию документации и отправил трассировки моих тестов компании-производителю. Зачем мне все это было нужно? Я узнал о том, какие фильтры не работают со стопроцентной гарантией, и об условиях, в которых может появиться ошибка, и фильтр сгенерирует ложное предупреждение.

А что можно сказать насчет набора правил Snort? Они открыты для всех желающих и их можно проверить – есть ли какая-то неопределенность? Открытый доступ дает огромные преимущества и уверенность (не сомневайтесь, такие фильтры найдут применение в других системах обнаружения вторжений), но в то же время хакер будет знать об опасности и постарается изменить атаку, чтобы не быть обнаруженным.

Три самых неприятных слова для аналитика сетевого трафика произносит сердитый пожилой начальник, который должен принять важное решение. Он пристально смотрит в глаза и спрашивает: "А вы уверены?".

Риск

Риск есть всегда. Смешно говорить, что я не хочу никакого риска. Правильней будет управлять степенью риска. По телевизору я слышал, что для резервных систем космического челнока существуют свои резервные системы. При полете в космос есть множество опасностей: радиация, жара, холод, вакуум и, наконец, возвращение на землю. Если войти в атмосферу под слишком большим углом, то произойдет взрыв, если же угол будет недостаточным, то челнок отбросит обратно в космос. Риск очень велик, и космонавты недаром пользуются такими привилегиями.

Если внимательно рассмотреть ситуацию, весь процесс космического полета очень опасен, и ни один здравомыслящий человек на это не пойдет. НАСА применяет критерии делать-не делать. Например, если что-то не так, запуск откладывается, даже несмотря на то, что есть резервные системы. Это называется *недопустимым риском*. Другие риски считаются допустимыми, например, посадка в космический корабль. Для любого риска нужно принять решение, как с ним поступить. Есть три варианта:

- принять риск таким, какой он есть;
- снизить риск;
- переложить риск на кого-то другого (модель страхования).

Допустимый риск

Подключение к Internet без установленного брандмауэра подобно посадке в космический корабль без всякой тренировки и знаний. Это весьма рискованно, и мы допускаем этот риск. Если на хостах хранится важная информация, и при этом хосты никак не контролируются, а также нет системы обнаружения вторжений, значит, мы рискуем еще больше.

Концепция допустимости риска довольно проста, но есть один важный аспект. Водитель школьного автобуса, который пьет пиво перед поездкой с детьми, конечно, тоже рискует, но такой риск недопустим. Администратор брандмауэра, который проверял работу службы и по ошибке оставил ее запущенной, также заставляет рисковать всю организацию, что никак не планировалось. В конце концов зачем тогда было вообще покупать и устанавливать брандмауэр? Одна из важных проблем информационной безопасности заключается в том, что один человек может допустить риск, не имея на это никаких прав. Проиллюстрируем такую ситуацию на примере из области обнаружения взлома.

На прошлой неделе мы обнаружили системы, которые инициировали передачу файлов из контролируемой нами сети. Это было достаточно необычно, и мы решили разобраться, в чем тут дело. После анализа полезной нагрузки FTP-пакетов стало очевидно, что каждый из компьютеров отправлял информацию о своей системе. Мы не были в этом уверены, пока не нашли строку "Preferred Customer", которая обычно используется для поля регистрации в продуктах Microsoft Office. Наши подозрения быстро подтвердились. Когда-то сотрудник отдела кадров этой организации отправил всем служащим высшего ранга уведомление в качестве приложения к сообщению электронной почты. Интересно, что эти руководители говорили мне о необходимости проверки FTP-сеансов, а сами даже не читали своей электронной почты. Их секретари просматривали почту, распечатывали ее и заносили в папку входящих сообщений только самые важные. Отправленное из отдела кадров сообщение было инфицировано вирусом, который отправлял в Internet информацию о локальных хостах. В данном случае никакого серьезного ущерба не было нанесено. Но такая ситуация меня обеспокоила. Сотрудник, который не использовал современные антивирусные программы, подверг высокому риску всю организацию. Как насчет еще одного примера?

На той же неделе в той же сети мы обнаружили намного больше систем, которые получали файлы, чем обычно. За один день мы выявили пять таких систем. Когда мы исследовали содержимое пакетов, то оказалось, что все они имеют одинаковый IP-адрес получателя, и получение файлов инициировано от имени одного и того же пользователя с неизменным паролем. Файлы загружались на настольные компьютеры. В данном случае причиной тревоги стала условно бесплатная программа PKZip. Нет, это не троянская версия и не атака. На Web-сайте этой программы было сказано, что при установке PKZip также устанавливается и дополнительный компонент, который при подключении к Internet загружает с сайта рек-

ламные объявления. Ни один из пяти пользователей не задумался над тем, что он делает. Все они просто хотели использовать новую программу. Так в чем же проблема? Все в порядке, если программа действительно загружает только рекламу. Однако не следует забывать, что многие брандмауэры бесконтрольно пропускают все пакеты для исходящих соединений, установленных из локальной сети. Это означает, что вместе с рекламой может быть организована успешная атака.

Троянская версия

Мне попадалось несколько троянских версий популярных программ, например ICQ или IRC (Internet Relay Chat). Пользователь не знает о том, что троянская программа передает важные сведения о его системе или загружает из Internet средства, позволяющие атаковать другие хосты его локальной сети.

А что если компания, занимающаяся рекламой, взяла на работу хакера или эксперта в промышленном шпионаже? Представьте себе, что он может сделать с программным кодом дополнения, которое предназначено для загрузки случайных файлов из Internet при каждом включении системы! Если кому-то это кажется фантастикой, подумайте об использовании жучков в сетях (`www.bugnosis.org`) и других шпионских штучках.

Вредоносные соединения

Создано огромное количество атак на службу DNS, одна из которых заключается в подмене содержимого кэша DNS-сервера для того, чтобы пользователь вместо реального сервера подключался к серверу хакера. Это часто происходит, когда клиент отвечает на заданный вопрос.

Проблема довольно сложна. Пользователи настольных Windows-систем обычно не знают, какие соединения устанавливает их компьютер. Честно говоря, я и сам не знал, что программы на моем компьютере под управлением Windows могут самостоятельно устанавливать исходящие соединения. Несколько лет назад я купил программный пакет McAfee Office, в основном, чтобы получить систему шифрования PGP (Pretty Good Privacy), но решил ознакомиться с большей частью пакета. Одна из программ называлась GuardDog и была предназначена для защиты Windows-систем. Я установил пакет, и представьте мое удивление, когда после перезагрузки системы я получил предупреждение о том, что одна из программ пытается установить соединение с Internet. Это была программа Real Audio. У меня не было времени на установку средств слежения у себя дома, поэтому я просто удалил эту программу. Позднее я узнал, что таким образом производители собирали сведения о пользователях своего программного обеспечения. Теперь на своем домашнем компьютере я использую персональные брандмауэры, такие как ZoneAlarm и Norton Internet Security.

Настало время сделать некоторые выводы. В двух предыдущих примерах (вирус и программа PCZip) компьютеры пользователей инициировали соединение с Internet, о чем сами пользователи и не догадывались. В обоих случаях для организации есть потенциальная опасность. Таким образом, из-за бездействия (первый пример) или неверных действий (второй пример) отдельные пользователи приняли решение о допустимости риска, который несет угрозу всей организации.

Наша точка зрения на обнаружение взлома

На конференции SANSFIRE в 2000 году Нейл Джонсон (Neil Johnson), научный сотрудник и преподаватель университета Джорджа Мейсона, представил прекрасный документ по обнаружению взлома и защите с помощью водяных знаков. Если на создание графических образов затрачены время и деньги, то на них можно нанести логотип для защиты авторских прав. Для этого существуют специальные средства. Затем можно использовать технологию червя в WWW для поиска ваших изображений на серверах, которые не имеют лицензии на их использование. Нейл пояснил технологию и продемонстрировал возможности атаки и защиты от таких атак. Теперь вам понятно, какое отношение имеют водяные знаки к обнаружению взлома?

Мы продолжаем изучать теорию риска и ее применение на практике по отношению к обнаружению вторжений, поэтому помните, что опасен не тот злоумышленник, который бесцельно применяет устаревшие атаки трехлетней давности, а тот, кто знает, чего хочет, и для достижения своей цели применяет сложный для выявления метод. Газета "USA Today" рассказала, что Бен Ладен использовал шифрованные сообщения для проведения атаки 11 сентября 2001 года. Но есть и более прагматичные примеры. Для компании, которая занимается созданием графики, созданные ею изображения имеют огромную ценность. Для этой компании самой страшной ситуацией будет та, при которой хакер, способный удалить логотип защиты авторских прав на изображениях, начнет их продавать как свою собственность.

Снижение риска

А что если мы решим все-таки совершить полет на космическом челноке, пусть даже это очень рискованно? Может быть, мы очень хотим стать первыми людьми на Марсе? Такое предприятие весьма опасно, но мы убеждены в его необходимости. В этом случае, если мы не безрассудны, то постараемся сделать все возможное, чтобы снизить риск.

Вы когда-нибудь задумывались об атаках на переносные компьютеры? В наше время ими пользуются многие специалисты, которые часто хранят на них очень важную информацию. Мы уже рассказывали о троянских программах для сбора информации, и эти программы могут использоваться против любой системы. Как именно можно получить данные из переносного компьютера? Что можно сделать для снижения риска?

Атака по сети

Если организация для получения электронной почты и доступа в Internet использует услуги провайдера, а не защищенную выделенную линию, то существует возможность атаки на переносные компьютеры этой организации во время их соединения с Internet. Они не защищены брандмауэром и поэтому доступны для атак посредством NetBIOS и других атак на Windows-системы.

Хватай и беги

Я просто ненавижу ставить мой переносной компьютер на конвейерную ленту рентгена при проверке в аэропортах. Если я не сразу пройду через металлодетектор, то кто-то имеет хорошие шансы украсть мой компьютер в толпе. Из-за постоянного поиска террористов меня могут задержать и даже проверить мои туфли. Если же кто-то заберет мой переносной компьютер, а я попытаюсь его догнать, то рискую быть застреленным охранником. Иногда я сомневаюсь, оставлять компьютер в комнате гостиницы или брать его с собой в ресторан.

Я не знаю, насколько важна информация на переносных компьютерах в вашей организации. В конце концов, кому нужен проект или бизнес-план, сведения о

продажах и ситуации на рынке? Я специально немного утрировал, но если вы спросите владельцев переносных компьютеров, то многие скажут, что их информация совсем не существенна и не нуждается в защите.

Зато я знаю свою ситуацию. Я пишу книги, преподаю и часто работаю с конфиденциальной информацией. Я подписал несколько документов о неразглашении и всегда стараюсь быть аккуратным. Если крупная компания по обеспечению безопасности в сети решит, что я не выполнил требований безопасности, мне придется столкнуться с целой армией адвокатов. Поэтому я стараюсь защитить мой переносной компьютер как можно лучше. Я ищу средства для снижения риска. Так как я знаю, что подключение к Internet имеет определенный риск, то стараюсь узнать, какие программы помогут защитить мою систему.

Я проверил работу нескольких средств. Программа ZoneAlarm бесплатна для использования частными лицами и работает вполне надежно. Некоторые мои друзья в восторге от BlackIce, но, хотя трассировки этой программы очень точны, качество фильтров стало ухудшаться с тех пор, как компания-производитель была продана. Пакет Norton Internet Security может работать на платформе Windows XP, что несомненно является плюсом. Система PGP имеет встроенный персональный брандмауэр, но, когда мой начальник установил ее на свой компьютер под управлением Windows XP, он не смог подключиться к Internet. У меня тоже был такой случай, когда я запустил PGP на платформе Windows Me. В обоих случаях причиной неполадок являлась программа PGPNet. Так как PGP была мне крайне нужна, то я просто удалил операционную систему и установил Windows 2000. В состав пакета PGP также входит программа PGPdisk, с помощью которой можно защитить важные файлы, если переносной компьютер будет украден или взломан по сети. Для защиты файлов на платформах Windows 2000 и XP можно использовать шифрующую файловую систему EFS (Encrypting File System) компании Microsoft. Хотя в пакет PGP входит подпрограмма для перезаписи диска и уничтожения данных, мне больше понравилась программа BC Wipe (`http://www.jetico.sci.fi`). Вот я и проиллюстрировал мой личный опыт использования средств защиты для снижения риска.

Страхование

На прошлой неделе вместо работы с исходящими пакетами FTP я занимался потопом в моем доме. Сломался туалетный бачок, вода полилась на пол и начала свое путешествие к океану. Но это еще не все. В этот день в городе решили промыть пожарные гидранты, поэтому дом наполнила грязная вода. Когда моя жена вернулась домой, вода била фонтаном из люстры в столовой. Штукатурка поотлетала, а деревянный пол покоробился. Вода продолжала накапливаться, пока не провалился потолок в комнате, где жена занималась шитьем. Она позвонила мне на работу и спросила, что делать. "Перекрой воду и уходи из столовой. Я выезжаю", — ответил я.

Я использую один и тот же метод реагирования на происшествия как на работе, так и в быту. Положив трубку, я осознал, что нанесенный мне ущерб составит от 20 до 30 тыс. долл. Очень расстроенный, я приехал домой, и мы с женой стали спасать то, что еще можно было спасти. Мое настроение оставалось плохим, пока уже ночью я не вспомнил, что застрахован в хорошей компании! Я всегда знал, что владение домом сопряжено с определенным риском. Не существует таких

средств, которые бы могли защитить от поломки туалетного бачка в тот день, когда в городе промываются пожарные гидранты. Как и большинство домовладельцев, я переложил свой риск на страховую компанию. Я позвонил, они приехали и пообещали обо всем позаботиться. В результате, потеряв только $2500, я получил лепку на потолке, о которой мечтал довольно давно.

Как применить страхование по отношению к защите информации и обнаружению взлома? Можно использовать тот же метод. Несколько агентств, включая Lloyds и IBM, предлагают страховки на случай атак хакеров. Обычно перед страхованием они требуют установить средства защиты, например, брандмауэры и провести поиск уязвимых мест.

Мы рассмотрели, как связаны понятия неопределенности и риска. Выяснили, что один риск можно допускать (разрешено нам это или нет), а другой – нет. В последнем случае нужно или уменьшить риск или застраховаться. Теперь нужно разобраться с причиной, которая может привести к нанесению ущерба, – мы называем это *угрозой*. Уязвимые места – это ворота, через которые проникают угрозы.

Описание угрозы

– На вашем месте я бы туда не ходил.
– Это почему?
– Всякое может случиться.
– Что может случиться?
– Всякое.

Не очень то внятное предупреждение, правда? Большинство из нас не стали бы к нему прислушиваться. Представьте себе, что такие аргументы используются для установки системы обнаружения вторжений.

Нам нужно четко определить возможные угрозы.

- Что может произойти?
- Насколько это серьезно?
- Как велика вероятность повторения происшествий?
- Откуда вы знаете?
- На основании чего делаются такие выводы?

Ответы на эти вопросы нужно давать отдельно для каждой вероятной угрозы.

Насколько серьезна угроза

В конечном итоге риск оценивается в денежном эквиваленте. Это справедливо даже при несчастных случаях. Для каждой конкретной угрозы мы берем цену имущества, которое подвергается риску, и умножаем ее на коэффициент незащищенности (exposure factor). В результате мы получаем ожидаемые убытки, или *вероятность разового ущерба* (single loss expectancy – SLE):

```
Цена имущества x коэффициент незащищенности = SLE
```

Коэффициент незащищенности может меняться в диапазоне от 0 до 100 % возможного ущерба имуществу. Рассмотрим, как оценить угрозу от взрыва атом-

ной бомбы над небольшим городком, имущество которого оценивается в 90 млн. долл.: 90 млн. долл. x 100 % = 90 млн. долл.

Вернемся к нашей теме. Выше уже говорилось, что, выполняя поиск уязвимых мест на UNIX-хостах, я нашел ряд систем, на которых присутствовала уязвимая версия базы данных ToolTack. Можно ли применить формулу оценки вероятных убытков для этого случая? Прежде всего нужно описать угрозу. Представим себе локальную сеть класса С. Угроза заключается в получении хакером прав суперпользователя, использовании им доверительных отношений и приобретении контроля над всеми хостами. Хакер может зашифровать все файловые системы и потребовать выкуп в размере 250 тыс. долл. Для достижения своей цели злоумышленник выполняет зондирование сети и находит шесть уязвимых компьютеров. С помощью атаки на переполнение буфера он быстро получает права суперпользователя. Установив доверительные отношения, он компрометирует еще четыре системы и зашифровывает диски 10 рабочих станций UNIX. Утром директор этой организации по электронной почте получает письмо с требованием выкупа в 250 тыс. долл. за возврат зашифрованных данных.

Каким будет значение SLE в данном случае? Можно сказать, что оно составит 250 тыс. долл., но не все так просто. Если данные можно восстановить с резервных копий, то будет потеряно только несколько дней работы. Если резервных копий не было (а они должны быть всегда), проблема становится сложнее. В данный момент мы не знаем, получим ли когда-нибудь ключ для расшифровки. Есть угроза, что не получим. Таким образом, цена имущества равна цене данных на дисках взломанных систем плюс время на восстановление этих систем после взлома плюс стоимость простоя. Как можно оценить стоимость данных?

Приблизительно оценить стоимость данных можно по средней заработной плате людей, которые работали на этих системах. Для простоты будем считать, что все взломанные UNIX-системы принадлежали специалистам. Все они работали над одним проектом в течение 2 лет, каждый получал по 60 тыс. долл. в год, т.е. с учетом дополнительных затрат (премии, аренда помещения и т.д.) потрачено по 100 тыс. долл. Десять человек по 100 тыс. в год, за два года получается 2 млн. долл. А каким будет коэффициент незащищенности? Он будет равен 100%, все файлы уже зашифрованы. Таким образом, становится очевидным, что лучше без вмешательства правоохранительных органов тихо выплатить 250 тыс. долл. Мы платим деньги, получаем ключ к шифру, возвращаемся к работе, и все довольны. А что произойдет, если уязвимое место так и не будет устранено?

Повторяемость угроз

О *среднегодовой вероятности ущерба* (annualized loss expectancy — ALE) говорят, когда конкретная пара "угроза — уязвимое место" может стать причиной ущерба чаще одного раз в год. В отчете на конференции по безопасности информационных технологий, которая проходила в мае 2000 года, доктор Джин Шульц (Gene Schultz) указала, что такая оценка может быть неточной. В примере с атомной бомбой сколько бы атомных бомб мы не бросали на город, навряд ли бы это привело к дальнейшим разрушениям. Показатель ALE хорошо работает при оценке таких событий, как кража в магазинах, возврат заказанных товаров и невозвращение ссуд. Сколько событий класса ALE можно выдержать в условиях конкурен-

ции (например, в электронной торговле)? Если Web-магазин будет прекращать работу четыре или пять раз в месяц, то покупатели начнут обращаться к конкурентам. Как их удержать? Какое значение имеет показатель ALE для защиты информации и обнаружения взлома?

Как известно, системы обнаружения вторжений позволяют выявить несанкционированный доступ. Правда, я считаю, что для квалифицированного аналитика это просто потеря времени. Тем не менее есть весьма убедительные аргументы в пользу применения системы именно таким образом. В следующем примере обратите внимание на то, что, специально взяв для начала небольшие суммы, мы получим в результате довольно крупные деньги, которых будет вполне достаточно, чтобы оправдать наем специалистов по компьютерной безопасности. Для вычисления параметра ALE используется следующая формула.

```
SLE x число происшествий в год = ALE
```

Таким образом, показатель ALE — это результат умножения SLE на количество происшествий в год. Вот почему раздел, в котором был дан пример шифрования хакером информации жестких дисков, мы закончили вопросом: "А что случится, если уязвимое место так и не будет устранено?". Хакер забирает деньги, тратит их, его друзья интересуются источником этого богатства, и наша игра начинается сначала.

Давайте рассмотрим распространенную ситуацию, когда сотрудники вместо работы "бродят" по Internet. Для начала нужно вычислить значение SLE. Пусть у нас будет 1000 сотрудников, 25 % из которых тратят 1 час в неделю на развлечения в Internet.

```
$50/час x 250 = $12500
```

Теперь подсчитаем ALE при условии, что сотрудники работают все время, кроме отпусков.

```
$12500 x 50 = $625000
```

Теперь понятно, почему организации вкладывают деньги в средства обнаружения вторжений и привлекают специалистов, чтобы запретить несанкционированные действия своих сотрудников. Не забывайте, что цифры были специально занижены, в реальности они намного выше. Кроме того, в действительности пустая трата времени не одинаково присуща всем сотрудникам. Если выявить и уволить таких злостных нарушителей (в конце концов, если они не работают, то и не нужны), то организация получит серьезную экономическую выгоду.

Осознание неопределенности

Насколько точны результаты подсчета значений SLE и ALE? Это необходимо знать, если мы планируем принимать решения на основе полученных значений. Однажды меня полдня убеждали в необходимости покупки одной системы обнаружения вторжений. Это средство могло делать практически все: объединять показатели датчиков, автоматически определять взаимосвязь уязвимых мест с входящими атаками и даже выдавать в графическом виде отчеты о выявленных вирусах. "Основное преимущество, — сказал мой собеседник, — в том, что здесь используется экспертная система".

Он продолжал говорить, я время от времени кивал, но уже не слушал. Я просто вспомнил несколько своих фраз из курса лекций по искусственному интеллекту. "Причина, по которой экспертные системы не оправдывают возложенных на них надежд, заключается в том, что используемые в них правила не очень точны. Создатель базы знаний опрашивает специалистов в определенной области, но многие мнимые эксперты совсем ими не являются. Одна из основных проблем систем искусственного интеллекта состоит в том, что эти системы не знают, чего именно они не знают. В этом отношении они похожи на людей."

При вычислении показателей SLE и ALE нужно учитывать то, чего мы не знаем, приблизительные цифры и те ситуации, когда использование моделей просто невозможно. Некоторые могут подумать, что не очень то и нужны эти подсчеты. Что ж, может быть, вы делаете то же самое, только безо всяких правил или обоснования.

Например, работая в организации, контролирующей локальные сети, я слышал, как новых сотрудников предупреждали о недопустимости поиска порнографии в Internet и о том, что тот, кто будет в этом замечен, скорее всего, будет уволен. Попробуем представить, что при этом думал молодой сотрудник. Возможно, его заинтересовало, сможет ли он незаметно ввести в поисковую систему неправильно написанное слово сексуальной тематики или использовать косвенные ссылки. А вдруг у руководства нет чувства юмора, да и показатель SLE слишком велик. Конечно, его мысли могли быть немного другими, но вы, наверняка, поняли, что я имел в виду.

Хотите еще один пример неопределенности? В середине февраля 1999 года я был участником группы по созданию Президентского указа 63 (Presidential Decision Directive 63). Были собраны около 50 ведущих специалистов в области обнаружения вторжений и информационной безопасности, призванных описать четыре основных направления исследования и получить денежный грант размером в полмиллиарда долларов. Одно из направлений получило название "вредоносное поведение пользователей", что в Вашингтоне понимали как проблему атак локальных пользователей. Когда мы представили результаты работы, то группа, занимающаяся этой темой, обнаружила, что было выделено в 100 раз больше денег на блокирование атак внешних пользователей, чем локальных. На вопрос: "Откуда такое соотношение и на чем оно базируется?" многоуважаемые ученые ответили: "Мы взяли это значение с потолка, но оно близко к реальному".

Деньги на управление риском

Если прийти к начальству и сказать, что нужно 10000 долларов на покупку системы обнаружения вторжений, то, скорее всего, начальство потребует дополнительной информации. Хороший признак, если начальник спросит, как долго устанавливать эту систему. Умный руководитель знает, что затраты на аппаратные средства и программное обеспечение – это только верхушка айсберга. Он хочет понять, как система обнаружения вторжений впишется в модель работы организации. Управление риском (включая и систему обнаружения вторжений) зависит от денежных вложений.

Когда мы сталкиваемся с каким-либо риском, мы начинаем думать, как его можно уменьшить. Мы ищем меры противодействия и подсчитываем, во сколько

это может обойтись в год. Если вы хотите внести предложение, то смета явно не помешает, а еще лучше предоставить несколько вариантов. Не забывайте про показатели SLE и ALE. Здесь они и пригодятся. Меры противодействия стоят денег, но посмотрите, во что обходится риск!

Есть очень важный момент в общении с руководством: не мелочитесь! Чем более глобальную картину вы нарисуете с описанием всех рисков, уязвимых мест, мер противодействия и тщательно разработанных планов действий, тем лучше будут восприняты ваши предложения.

Оценка риска

Я очень люблю слушать, как продавцы предлагают купить системы обнаружения вторжений для защиты отдельных хостов. Это настоящие сражения, а в наше время, когда каждый может получить персональный брандмауэр всего за 60 долларов, это становится смешным. Продавцы прежде всего делают упор на угрозы со стороны пользователей той же локальной сети. Им приходится так поступать. Они пытаются заставить потенциального покупателя поставить один риск выше другого. Если подумать, то это обычная стратегия любой продажи.

В Вирджинии не выпадает много снега, но с наступлением зимы рекламисты машин просто заставляют купить вас полноприводной автомобиль. Неважно, что он дороже, сложнее в устройстве и потребляет больше горючего, чем автомобиль с передним или задним приводом. Купив такой автомобиль, снега можно не бояться. На основании этого можно сделать следующие выводы: нужно оценивать все риски и просчитывать развитие событий. Нужно выбирать меньший риск. Оценка риска может быть количественной и качественной.

Количественная оценка риска

Этот метод оценки риска позволяет подсчитать риск как числовое значение. Самый распространенный способ такого подсчета заключается в использовании значений SLE и ALE. Не стоит подсчитывать риск для каждого отдельного компьютера вашей организации! Но на уровне всей локальной сети такая оценка может оказаться эффективной, и нужные цифры найти совсем не сложно. Для вычисления стоимости имущества используется следующая формула.

```
Стоимость имущества = аппаратные средства + коммерческие программы +
собственное программное обеспечение + данные
```

Материально ответственное лицо организации должно предоставить сведения о стоимости аппаратных средств и программного обеспечения за несколько минут. Определить стоимость собственных созданных программ сложнее. Нужно учесть размер зарплат для всех, кто участвовал в разработке проекта за несколько лет. Но самое ценное — это данные! В вашей организации компьютерами пользуются все сотрудники? Если да, то стоимость данных определяется по затратам на их создание в период жизненного цикла данных (как правило, его принимают равным трем годам). Эта сумма будет очень большой! Определение ценности данных организации не должно занимать больше часа времени. Этот способ оценки риска пригодится, когда нужно убедить руководство в приобретении чего-либо или заставить прекратить потенциально опасные действия.

Качественная оценка риска

Для определения иерархии рисков можно составить контрольный список. Как правило, составляют список вероятных угроз и для каждой из них указывают степень опасности: высокая, средняя или низкая. Этот метод лучше работает при оценке риска отдельно для каждого компьютера, а не для всей организации. Несколько примеров использования модифицированного количественного метода оценки риска, а также примеры контрольных списков для количественного метода можно посмотреть по адресу `http:/www.nswc.navy.mil/ISSEC/Form/AccredForms/index.html`.

Еще одним ресурсом, который будет полезен при оценке риска, являются контрольные списки `score` по адресу `www.sans.org/score`. И, наконец, на сайте `www.cisecuroty.org` можно найти много средств, которые позволяют провести оценку безопасности вашей сети. Сначала кажется, что эти средства используют количественный метод, так как они оперируют цифрами, но при более глубоком их анализе оказывается, что они основаны на качественном методе оценки риска.

Неточность оценки

Оба метода оценки риска очень хорошо работают в теории. На практике это не так. Это объясняется тем, что люди не очень любят говорить правду, так как, признавшись в существовании уязвимого места, придется что-то делать, чтобы его устранить. Таким образом, те, кто используют количественный метод оценки, получают очень большие цифры. Они понимают, что не могут позволить себе такой большой риск, поэтому стараются рассмотреть проблему по частям, пока не приходят к количественной оценке риска для одного компьютера, а это нелепо. Не нужно гадать, какую систему (с высокой, средней или низкой степенью риска) выберут для оценки риска. Если степень риска мала, то зачем вообще о чем-то беспокоиться?

Резюме

Со времен Карибского кризиса и до падения берлинской стены лучшим способом получения денежного пособия для сотрудника Министерства обороны было заявление в Конгресс с предупреждением "Русские идут". Несмотря на критику по телевидению, законодатели не так уж глупы, а многие из находятся у власти уже много лет. В какой-то момент они начинают понимать, что финансирование было выделено на основании ничем не подкрепленной угрозы.

Теперь все хотят остановить террористов, в частности — хакеров компьютерных систем. Истерические предупреждения помогут получить деньги в первый раз, но в целом это ошибочный метод. В этой главе была описана модель надежной защиты организации. Мы рассказали, как оценивать риск и определять приоритеты опасности. Дана хорошая база для разговора с руководством о том, что и зачем нужно. В следующей главе мы рассмотрим ответные действия после атаки на систему или ее компрометации. Когда у нас есть надежные средства защиты, можно спокойно рассуждать о том, как действуют хакеры и как пережить компьютерную войну.

18

Ответные действия

Описывая методы анализа сетевых трассировок, мы подробно рассматривали вопрос воздействия и реакции. Теперь рассмотрим этот же вопрос на уровне возможных ответных действий целой организации. Ответную реакцию вызывают успешные атаки или попытки атак. При выявлении успешно проведенной атаки следует выполнить восстановление после взлома. Что такое успешная атака? Это "любая атака, которая является причиной дополнительных действий, за исключением фильтрации пакетов". Согласны? Если нет, то не забывайте, что любые неавтоматические ответные действия требуют затрат дополнительных ресурсов. Давайте рассмотрим три примера атак и попытаемся определить, являются ли они успешными.

- **Зондирование сети**. Поступает набор эхо-запросов ICMP для исследования компьютеров локальной сети. Как правило, зондирование сети выполняется с удаленного хоста по Internet, а ICMP-сообщения отправляются по широковещательным адресам подсетей. Эта атака может быть выявлена брандмауэром или системой обнаружения вторжений.
- **Анкетирование с помощью дискет**. Сотрудник получает письмо с приложенной к нему дискетой. Если он откроет эту дискету на своем компьютере, ответит на предлагаемые вопросы и отошлет дискету обратно, то получит в подарок футболку.
- **Соединения через TCP-порт 53**. Internet-компания, занимающаяся созданием рекламных баннеров для Web-страниц, выполняет тестовое подключение к тем системам, пользователи которых заходили на эти Web-страницы. Для этого организовывается попытка соединения с TCP-портом 53 проверяемой системы.

Какая из этих атак успешна? Я бы ответил, что если маршрутизатор на периметре сети или брандмауэр блокирует входящие эхо-запросы ICMP, то попытка зондирования сети будет неудачной. Многие считают, что это просто разведывательное зондирование, а не атака, но вопрос в том, будут ли использованы дополнительные ресурсы. Недавно я видел одну трассировку, в которой хакер отправ-

лял пакеты только по IP-адресам реальных компьютеров, подключенных к Internet. Это очень опасно, когда злоумышленник точно знает, что он ищет.

Анкетирование с помощью дискет можно назвать успешной атакой. Большинство пользователей никогда не узнают, какие файлы были скопированы, а какие добавлены в их системы. И уж наверняка, злоумышленник почерпнет полезную информацию из анкет, за что придется расплачиваться всей организации. Специалист в области информационной безопасности обязан уведомить пользователей, что они должны сразу выбрасывать такие дискеты в мусорный ящик, а если им так уже хочется ответить на вопросы, то пусть это делают дома.

А что насчет DNS-запросов? Эти запросы поступают постоянно и очень трудно определить, какие запросы могут служить для разведывательных целей, а какие применяются для обычной работы пользователя в Internet (запрос `gethostbyaddr`). Тем не менее HTTP-заголовки предоставляют значительный объем информации о клиенте, работающем в WWW. Только по нескольким полям этого заголовка можно получить следующие сведения:

- операционная система хоста;
- версия используемого броузера;
- *последний посещенный* Web-сервер (*поле* `referrer`).

Web-серверы собирают всю эту информацию для маркетинговых исследований. С помощью сохраненных данных создатели Web-страниц могут точнее определить круг своих клиентов, а также узнать ключевые фразы, которые вводят пользователи в строку поиска. Но эта же информация может быть использована и хакерами. Если добавить к ней информацию службы DNS и сведения, добытые с помощью утилиты `netstat`, то получится довольно значительный объем информации о конкретном IP-адресе или диапазоне IP-адресов.

Как можно заметить, я больше не использую примеров элементарных атак. Мне очень нравится философия боевого искусства эскрима (escrima), которая заключается в том, чтобы использовать все ошибки, допускаемые противником в конкретный момент времени. Это можно назвать фундаментальным принципом информационной войны. Хакеры используют различные методы атак на избранную цель, выискивая наиболее уязвимые места. Вот почему тщательно продуманная (эшелонированная) система защиты и автоматические ответные действия имеют такое большое значение.

Автоматические ответные действия

В этом разделе рассматриваются принципы построения схемы автоматического ответа, доступные для этой цели механизмы, наиболее популярные реализации такой защиты (программа PortSentry), а также возможность автоматического ответа персональных брандмауэров. Очевидно, что автоматические ответные действия являются самыми дешевыми и простыми методами реакции на атаки. При грамотном и аккуратном использовании они обеспечивают достаточный уровень безопасности. Здесь придется дать несколько предупреждений. Поскольку системы обнаружения вторжений часто сообщают о ложных тревогах, то по ошибке ответные действия могут быть автоматически направлены против абсолютно невиновных людей. С другой стороны, можно организовать набор методов пассивной

защиты. Такие пассивные ответные действия никому не причиняют вреда. Нужно быть очень глупым человеком, чтобы организовать автоматический запуск ответной атаки против вероятного хакера. Ведь могла произойти ошибка, или хакер подменил свой IP-адрес.

Еще одна проблема заключается в том, что злоумышленник может определить наличие автоматических ответных действий и использовать их в своих целях. Представьте себе создание замкнутого цикла обмена сообщениями (с помощью подмены двух IP-адресов), передаваемыми между двумя автоматизированными системами обнаружения вторжений, наподобие того, как это делалось для атаки отказа в обслуживании с помощью "замыкания" друг на друга служб `echo` и `chargen`. Еще хуже, если злоумышленник осуществляет атаку от имени партнера, заказчика или поставщика, вызывает автоматические ответные действия и срывает сроки договоренностей.

Архитектурные решения

Большинство сетевых систем обнаружения вторжений являются пассивными, они просто перехватывают поток данных и не реагируют на действия хакеров. Однако многие коммерческие реализации этих систем могут быть подключены непосредственно к брандмауэру, и такой тандем обеспечивает возможность автоматических ответных действий. В качестве примера можно назвать программу `hogwash` – реализацию брандмауэра, основанную на системе Snort, которая совмещает функции обоих продуктов. Разрабатываются еще несколько подобных систем. Систему обнаружения вторжений с автоматическими ответными действиями обычно устанавливают в демилитаризованной зоне или на линии подключения к Internet, но возможны и другие удачные решения, например, брандмауэры в локальной сети и сами системы обнаружения вторжений, установленные на хостах.

Автоматический ответ на линии подключения к Internet

Чем ближе система с автоматическими ответными действиями будет расположена к точке подключения к Internet, тем эффективнее она будет работать. Однако при этом возрастает риск подмены и манипуляции IP-адресами. Основная причина состоит в том, что пакеты в точке подключения к Internet обычно не фильтруются, они только затем поступают на брандмауэр или фильтрующий маршрутизатор. Это означает, что устройства, установленные в точке подключения к Internet, могут быть атакованы с использованием IP-дейтаграмм, в которых указаны любые адреса (в том числе и подложные), TCP-пакетов с указанием любых портов отправителя (в диапазоне 0–65535) и комбинаций флагов и параметров, UDP-пакетов тоже с любыми параметрами, ICMP-сообщений, фрагментированных пакетов, а также с помощью всех типов пакетов протокола IP. Это обширное поле деятельности для организации защиты. Политика "запрета всего, что не разрешено" позволяет блокировать бо́льшую часть атак на периметре локальной сети, но обработка всех входящих пакетов с использованием автоматических ответных действий довольно рискованна. И все-таки автоматические ответные действия используются довольно широко благодаря своему основному преимуществу – возможности при внезапном возникновении угрозы отвечать автоматически с задержкой по времени, необходимой только для обработки одного пакета. Создание средств наподобие `hogwash` и UnityOne компании Tippingpoint's (`www.tippingpoint.com`) подтверждает принцип, согласно которому

эффективность автоматического ответа повышается с приближением к точке подключения к Internet.

Брандмауэры в локальной сети

Автоматические ответные действия, осуществляемые брандмауэрами, которые находятся в локальной сети, намного безопаснее, поскольку на эти брандмауэры поступает как минимум частично отфильтрованный трафик. Кроме того, можно значительно точнее определить политику безопасности. Например, если брандмауэр предназначен для защиты пяти локальных хостов, то можно точно сказать, какие службы и порты должны на них использоваться. Конечно, при этом автоматические ответные действия выполняются значительно реже. Здесь необходимо учитывать затраты на аппаратные средства, программное обеспечение и администрирование. Положительным моментом является наличие устройств, которые практически не нуждаются в настройке. Благоприятные условия для этого создала революция на рынке кабельных модемов и DSL-модемов, и в данное время существует множество решений от таких фирм, как Cisco, Linksys, Netgear и Symantec. Например, стоит приобрести для своей организации средство NAT (Network Address Translations), которое обеспечивает высокую степень защиты от атак при сравнительно низкой стоимости ($250). Если люди начнут повсеместно использовать такие устройства, то скоро мне придется подыскивать себе новую работу, не связанную с обнаружением взлома.

Защита на хостах

Автоматические ответные действия, организуемые на хостах, реагируют на самое небольшое количество внешних воздействий, но широко распространены, и риск подмены IP-адресов отправителя намного ниже, чем на периметре локальной сети. Существуют два основных метода: установка брандмауэров в локальных сетях и защита с помощью брандмауэров конкретных хостов. Многие люди целиком полагаются на защиту с помощью персональных брандмауэров, например, на программу PortSentry компании Psionic для UNIX-систем. Эта программа блокирует хост нарушителя, не разрешает устанавливать с ним новые соединения и даже удаляет маршрут, что полностью предотвращает возможность обмена данными. Систем персональных брандмауэров довольно много, но так как эта глава посвящена автоматическим ответным действиям, то следует особо выделить программу BackOfficer Friendly (`www.nfr.com/products/bof/index.html`). Это нечто большее, чем просто персональный брандмауэр! Ее можно считать активным средством защиты. Тем, кто работает в Windows и хочет узнать об автоматических ответных действиях, следует загрузить эту программу и проверить ее в боевых условиях. Единственным ее недостатком является отсутствие обновлений. Тем не менее это очень эффективная программа для защиты хоста, которая проста в установке, настройке и обслуживании. Почему пользователи столь зависимы от подобных программ? Очень часто ими пользуются администраторы локальных сетей, если при подключении к Internet вообще не осуществляется никакой фильтрации трафика. Основными сетями, в которых не осуществляется фильтрация входящих пакетов, являются:

- сети, в которых используются подключения с помощью кабельных модемов или DSL-модемов;
- сети коммерческих организаций, в которых защита просто игнорируется;

- сети университетов (отсутствие защиты во имя свободы);
- сети Ethernet в гостиницах.

Использование кабельных модемов и модемов для цифровых абонентских сетей (DSL-модемов) становится все большей угрозой для специалистов по безопасности (поэтому, скорее всего, мне не придется менять работу). Я лично создал много соединений с Internet с помощью кабельных модемов и наблюдал приблизительно от 5 до 20 попыток сканирования каждый день. Сотни людей подключаются к Internet по кабельным и цифровым сетям, и на большинстве их систем нет никаких средств защиты. Вот почему мы были так озабочены в 2001 году, когда появились вирусы Code Red, nimda и Leaves. Основными средствами защиты при использовании кабельных модемов являются программы наподобие NAT и персональные брандмауэры. Эти средства не обеспечивают автоматических ответных действий, но это неплохая сделка: защита от взлома взамен его обнаружения.

Коммерческие организации, которые не придают значения защите при подключении к Internet, не выдержат перехода к экономике, основанной на информационных технологиях. Просто поразительно, сколько существует узлов, на которых не установлены брандмауэры или не задействована какая-либо адекватная защита на периметре локальной сети. Любое соединение работающего компьютера с Internet будет проверено хакерами. Если системы не готовы к защите, то они будут взломаны. Если повезет, то хакеры просто позабавятся со взломанными хостами, но и в этом случае такие хосты, скорее всего, будут использованы для проведения следующих атак. Гораздо хуже, если скомпрометированные хосты организации используются для кражи коммерческих тайн. Если бы я был серьезным бизнесменом, то в дополнение к основному брандмауэру обязательно установил несколько брандмауэров в своей локальной сети. В противном случае атаки хакера, которому удалось взломать защиту на периметре, не смогут быть ни заблокированы, ни выявлены. Самые важные компьютеры обязательно должны быть защищены дополнительными средствами.

Я несколько лет наблюдал интересные сражения в университетских сетях. На многих узлах университетов брандмауэры или средства фильтрации пакетов отсутствуют напрочь. Но некоторые факультеты не хотят впускать в свои сети всех желающих и покупают собственные брандмауэры. Кроме того, сознательные администраторы пользуются программами защиты своих хостов. Ничем не ограниченное соединение с Internet уже устарело и может считаться пережитком университетских свобод. Вряд ли подобная практика просуществует более четырех лет. Будет любопытно понаблюдать за этим процессом. Ученые, которые заявляют, что все пакеты должны свободно поступать и передаваться в Internet, наверняка скоро изменят свое мнение. Нужно только дождаться, когда 50% бюджета факультета будет потрачено на удовлетворение судебного иска, поданного какой-то коммерческой организацией, которая понесет убытки от атаки, осуществленной со скомпрометированных хостов сети университета.

Подключения к Internet во время путешествий требуют особой осторожности. Я часто вожу с собой небольшой концентратор Linksys с возможностью маршрутизации и таким образом создаю два эшелона защиты: с помощью программы NAT и с помощью персонального брандмауэра. Кроме того, это позволяет одновременно рабо-

тать в Internet и мне, и моей жене. Использование NAT позволяет снизить риск взаимодействия с другими пользователями и доступа к моим секретным документам.

Автоматические ответные действия следует применять с осторожностью. Я предпочитаю использовать их в сетях учебных учреждений, особенно во время рождественских каникул. В это время людей обычно нет, а компьютеры остаются и их можно использовать для экспериментов с автоматическими ответными действиями. Поскольку никакой работы практически не выполняется, риск использования автоматических ответных действий снижается до минимума, и можно проверить, на что они способны.

В следующих разделах мы рассмотрим варианты автоматических ответных действий. При этом не забывайте о том, где эти возможности ответа будут иметь наибольший эффект.

Увеличение задержки ответа

Это хороший метод реагирования на сканирование портов, зондирование сети, SYN-наводнения и составление схем сети. Идея заключается в том, чтобы, обнаружив наводнение SYN-пакетами или сканирование, постепенно увеличивать время задержки ответов. Так можно заблокировать сканирование нескольких типов, например составление схемы сети с помощью `ping`-запросов (широковещательные адреса 0 или 255). Подобные методы сканирования основаны на сценариях, согласно которым определение хостов под управлением UNIX основано на времени их ответа. В маршрутизаторах Enterasys и Cisco поддерживается возможность ограничения интенсивности пропускаемого трафика. Вообще, любое устройство с функциями обеспечения *качества обслуживания* (Quality of Service – QoS) должно предоставлять подобные возможности.

Увеличение задержки может осуществляться и на уровне протокола. Для протокола UDP "глушится" хост-отправитель. Для протокола TCP, если трафик проходит через брандмауэр, в ответных пакетах может занижаться размер окна. Я стараюсь при использовании программы LaBrea не устанавливать размер окна со значением 1. В этом случае злоумышленники могут догадаться о причине задержки, но значение 5, например, великолепно подходит для замедления атаки.

Обрыв соединения

Описание обрыва соединения взято из руководства по определению соответствия атак выявленным в трафике строкам символов. В данном случае термин "соединение" относится к TCP-соединению, но этот же термин применим и к UDP-соединениям при использовании метода блокирования нарушителей (описан в следующем разделе).

Нарушитель устанавливает соединение с открытым портом. Затем он отправляет пакет или пакеты (для таких систем, обладающих возможностями повторной сборки получаемых пакетов, как Cisco Secure IDS или Snort), в которых содержится строка символов для проведения атаки или программа атаки. Система обнаружения вторжений выявляет эту строку и выдает команду брандмауэру оборвать соединение. Возможно, что система уже скомпрометирована, но хакер не может сразу же воспользоваться этим. В случае атаки на переполнение буфера в этот

момент на взломанном компьютере запускается программный код (вероятно, с привилегиями суперпользователя), который содержался в специальной строке, выходившей за пределы размера буфера. Если же это программа типа "абордажный крюк" (небольшой демон `telnet`, запускающийся на заданном порту), то обрыв соединения даст только несколько секунд передышки.

Блокирование нарушителя

Блокирование является одним из важнейших методов защиты как при автоматических ответных действиях, так и при самостоятельных действиях системного администратора.

В ходе атаки на скомпрометированном хосте запускаются новые процессы с привелегиями суперпользователя (root), например, хакер может организовать `telnet`-сеанс, возврат запрошенных данных с помощью системы X Window или создать любой другой потайной ход для доступа к системе. Обрыв соединения ничего не дает, так как немедленно будет организовано новое соединение, а в X Window инициируется исходящее соединение со взломанного хоста. В этом случае блокирование будет намного эффективнее. Блокирование позволяет полностью запретить обмен данными (как получение, так и отправку) с хостом злоумышленника. Можно заблокировать отдельный хост или всю его подсеть. При использовании этого метода просматривается файл, в котором сохранены запрещенные для блокирования IP-адреса хостов (так называемый "белый список" – IP-адреса постоянных клиентов и поставщиков). Это позволяет предотвратить атаку с использованием подложных IP-адресов, целью которой является нарушить взаимодействие организации с ее партнерами.

Блокирование не поможет при использовании хакером нескольких IP-адресов, принадлежащих хостам различных подсетей. Такое случается довольно часто. Мой друг, Педро Васкес (Pedro Vasques), прислал мне из Бразилии трассировку именно одной такой скоординированной атаки на переполнение буфера сервера DNS. Атака осуществлялась с одного хоста, а окно X Window открывалось на другом хосте нарушителя. Тем не менее не стоит отказываться от метода блокирования только потому, что он не помогает абсолютно во всех ситуациях.

Предварительное блокирование

Как ни странно, но многие поставщики услуг Internet и даже целые страны не могут или не хотят управлять работой своих хостов. Со временем, когда вы наберетесь опыта в обнаружении взломов, то увидите, что огромное количество атак осуществляется из одних и тех же сетей. Зачем играть с огнем? В конце концов они найдут способ взломать защиту. Лучше блокировать их до этого. Более того, используйте для блокирования два байта адреса (шестнадцатибитовую маску). Можно заблокировать все хосты целой страны, если в ней не преследуют хакеров. Если страны, которые не контролируют свои "исследовательские сети", окажутся в изоляции и не смогут получать доступ к целым областям Internet, то скоро им придется изменить свое беспечное поведение.

Мы проверили этот вопрос на Web-сервере SANS и нашли одного провайдера услуг Internet, proxy-серверы которого пропускали больше атак, чем поступало от любых других групп IP-адресов. Мы заблокировали этого провайдера на несколько месяцев и получили письмо с извинениями. Но в тот же день, когда мы сняли запрет, мы снова подверглись атаке. Запрет доступа из сетей этого провайдера стал постоянным.

Отключение питания

Метод отключения питания — это крайнее средство автоматических ответных действий. Смысл его в том, что при выявлении большого числа атак за короткий промежуток времени (обычно, когда аналитик сетевого трафика отсутствует на работе) система обнаружения вторжений отправляет сигнал на X10 или другое подобное реле с управляемыми логическими схемами и отключает питание маршрутизатора. В результате узел оказывается изолированным от Internet. Хотя такой метод защиты создает благоприятные условия для проведения хакерами атак отказа в обслуживании, это вполне приемлемая стратегия на время трехдневных выходных на узлах с высоким уровнем безопасности. Для возможности автоматического отключения следует добавить несколько строк кода в любую систему обнаружения вторжений, которая реализует выдачу сообщений по протоколу SNMP. При внимательном рассмотрении такой метод может оказаться не очень удачным, и подобные автоматические ответные действия часто приводят к отказам в обслуживании, вызванным своими же компьютерами. Поэтому организовывать подобные ответные действия нужно только в самых опасных ситуациях.

Выдача SYN/ACK-пакетов

Предположим, что система обнаружения вторжений имеет сведения о портах, доступ к которым заблокирован с помощью брандмауэра или фильтрующего маршрутизатора. В ответ на получение SYN-пакета, предназначенного одному из таких портов, система организует отправку подложного пакета с установленными флагами SYN и ACK. Злоумышленники могут обрадоваться массе обнаруженных целей для атаки, но это будет ошибкой. Последнее поколение программ для сканирования благодаря своим возможностям маскирования доставило массу проблем для специалистов по безопасности. Применение данного метода будет достойным ответом хакерам. Недавно я увидел его в действии. Несколько моих друзей установили брандмауэр Raptor. Он работает просто отлично. Нарушитель завершает процедуру установки соединения и даже пытается отправить данные в последнем ACK-пакете, и можно узнать, чего он хотел.

Отправка RST-пакетов

Этот метод еще называют *Reset-убийством* (Reset kill) или "завершением сеанса". Я должен высказать серьезные предупреждения по поводу его использования. Этот метод может обрывать абсолютно нормальные TCP-соединения, и я видел тому примеры в нескольких коммерческих системах обнаружения взлома при появлении ложных тревог. Смысл метода заключается в том, что при обнаружении в трафике установленного TCP-соединения сигнатуры, требующей выполнения ответного действия, для разрыва соединения отправляется по одному пакету Reset каждой стороне этого соединения. Этот метод предназначен для воздействия на хост-инициатор соединения, но хакеры быстро понимают необходимость игнорировать RST-пакеты. Нельзя сказать, что метод широко распространен, но он доступен в Snort и еще в нескольких коммерческих системах обнаружения вторжений.

Ловушка для хакера

На некоторых узлах в дополнение к увеличению задержки на запросы хакера маршрутизатор может направлять получаемый от него трафик на специально оборудованную систему, которую называют *ловушкой* (honeypot). Ловушка может использоваться как "дублер" настоящей цели атаки хакера. Мы также использовали ловушки со статическими IP-адресами как замену для локальных хостов, подвергавшихся атаке.

Иногда защищаемый хост вдруг становится предметом особого внимания со стороны хакеров. В такой ситуации удачным решением будет изменить его имя и IP-адрес и установить вместо него ловушку. Согласно информации на сайте `www.incidents.org` ловушки создаются в основном для определения целей и методов атак хакеров. Известны ловушки трех типов: система-приманка (proxy system), набор средств DTK (Deception Tool Kit) и "пустой" компьютер (метод альянса Honeynet).

Система-приманка

В 1996 и 1997 годах проводилось исследование используемых хакерами технологий. Целью проекта было собрать максимально возможное количество программ атак. Я взял компьютер на платформе Sun, работающий под управлением SunOS 4.1.3, добавил все доступные заплаты и установил набор средств TIS, назвав систему `cray3`. Скопировав файл `/etc/motd` из операционной системы Unicos, я сделал все возможное, чтобы он стал похож на `cray`. Слава Богу, что в то время не было метода определения операционной системы по протоколу TCP.

Для имитации работы служб FTP, telnet, SMTP и других использовался набор средств TIS. Скомпилировав программу IRC (Internet Relay Chat), я хотел подключиться к каналам IRC, в которых друг с другом общаются хакеры, обменяться опытом, вызвать атаки на свою систему и собрать используемые при этом методы. Была только одна небольшая проблема. Я никогда не общался в IRC! А ведь если делать что-то неправильно, то тебя просто засмеют. Поэтому я решил начать с канала `#thirtysomething`. Флирт никогда не был моим любимым занятием, и очень скоро я прекратил бессмысленно смотреть на пробегающие по экрану слова.

Следующей попыткой был канал `#Jesus`. Я думал, что религиозно настроенные люди будут добры ко мне. Как бы не так! Они выбросили меня из чата через 10 минут.

Наконец, в полном разочаровании я зарегистрировался на канале `#abortion`. Там были замечательные ребята, хотя их взгляды часто не совпадали. Самое главное, они были готовы обучить новичка. После недельной практики я решился зайти в канал `#hack`, но тут возникла еще одна заминка. Мы условились, что не должно быть никакого намека на проводимую провокацию, а так как я работал в Министерстве обороны, то мой исходящий адрес оказался из домена `.mil`. Это снова напомнило мне школьные годы и листок на спине с надписью "Ударь меня". Хакеры не стали ждать дополнительного приглашения.

Я выдержал несколько дуэлей по TCP, и немного времени спустя дела пошли на лад. Это было очень интересно, а хакеры просто не могли отказать себе в удовольствии атаки на систему из домена `.mil`, поэтому нам удалось собрать большой объем полезной информации.

DTK

Автором набора инструментальных средств DTK (Deception Tool Kit) является Фред Коэн (Fred Cohen). Получить этот набор можно по адресу `http://all.net/dtk`.

Набор средств DTK написан на языках Perl и C и позволяет эмулировать работу практически любой службы. Компиляция и установка не представляют никакой сложности. Однако после его улучшения с целью приблизить к реальности настройка стала вызывать определенные проблемы.

Метод работы DTK сильно напоминает использование средства BackOfficer Friendly.

"Пустой" компьютер

Ничто так не похоже на UNIX-систему, как другая UNIX-система. То же самое можно сказать и про системы под управлением Windows NT. Поэтому лучшей приманкой для хакера будет подобная система, которая немного старее и медленнее, а жесткий диск имеет меньший объем (чем меньше, тем лучше, если атакуемая система будет потеряна). После подмены можно установить на такую систему необходимые программы для сбора информации о проводимых атаках. Этот метод был разработан на научной основе командой альянса Honeynet. Компания Incidents.org входит в альянс Honeynet и использует средство `vmware` (`www.vmware.org`), а также брандмауэр, систему обнаружения вторжений и несколько операционных систем, которые запущены на одном компьютере. Возможности программы `vmware` поистине поразительны. Не так давно поступили тревожные сигналы, что хакеры обнаружили некоторые ловушки, знают их IP-адреса и стараются не атаковать эти системы.

Резюме по ловушкам для хакеров

Использование ловушек довольно эффективно, но может не оправдать вложенных средств. С другой стороны, при блокировании атак с помощью брандмауэра или фильтрующего маршрутизатора нельзя ничего узнать о методах хакеров. Если в сети нет особо важных систем, то лучше всего использовать ловушку для подмены DNS-сервера, Web-сервера или сервера электронной почты. Эти системы всегда входят в перечень целей хакеров. Хорошо, что можно изучать методы атак, но плохо, что очень часто повторяются одни и те же атаки.

Атака: ответные действия аналитика

Аналитики сетевого трафика нередко выполняют еще и обязанности специалиста по устранению последствий взлома. Запомните одну вещь: компьютер все равно будет взломан. Внешние атаки из Internet, атаки локальных пользователей и угрозы со стороны вредоносных программ рано или поздно приведут к компрометации. Аналитики часто считают, что они несут полную ответственность за абсолютную защиту организации. Это не так! Не могут спасатели отвечать за аварии. Мы только можем просить их о помощи после случившегося. Задумайтесь над моими словами. Я руководил большой группой по обнаружению вторжений, которая контролировала доступ ко многим узлам, и видел специалистов, убежденных в том, что они несут персональную ответственность за обеспечение защиты от абсолютно всех атак.

Если мы готовы принять удар, то компрометация системы не станет чем-то наподобие конца света. Главное выполнять свою работу максимально рационально и эффективно. Поскольку компрометация связана со стрессовой ситуацией, то план устранения последствий взлома должен быть максимально простым и четким. У реаниматологов используется весьма содержательная аббревиатура ABC. Она расшифровывается следующим образом.

- Airway (доступ воздуха). Убедиться, что воздух поступает.
- Breathing (дыхание). Есть или нет?
- Cardiac (сердечная деятельность). Сердце бьется или нет?

Приведенный ниже план действий по устранению последствий взлома я нашел в одном правительственном документе в 1995 году. С тех пор я работаю над усовершенствованием этой модели. План состоит из шести основных этапов:

- подготовка;
- идентификация проблемы;
- ограничение распространения атаки;
- устранение ошибок;
- восстановление;
- выводы из случившегося.

В этой главе не описываются подготовка и идентификация. В конце концов, большая часть этой книги посвящена этой теме.

Ограничение распространения атаки

Во время устранения последствий взлома следует соблюдать разумный темп действий, при спешке можно допустить ошибки, которые обойдутся очень дорого. В этом правиле есть одно исключение — нужно немедленно ограничить распространение атаки. Лучше восстанавливать два компьютера, чем четыре, и лучше разбираться с одной скомпрометированной рабочей группой, чем со всем доменом Windows. Опытные команды по устранению последствий взлома могут работать параллельно. Это особенно важно, когда атака затронула несколько компьютеров. Как только поступают данные об успешной атаке, я делаю копию, определяю круг компьютеров, который она могла затронуть, записываю все на бумагу и без лишних слов передаю эту бумагу человеку, занимающемуся устранением последствий взлома.

Первым делом следует прекратить возможность связи по сети со взломанными компьютерами.

Остановка атаки

Сначала я звоню человеку, который находится ближе всего к консоли взломанной системы. Следующие фразы являются результатом многих лет работы в области безопасности. Вы технический специалист, но тот, кому вы звоните, может ничего не смыслить в компьютерах. Кроме того, он может видеть наличие проблемы и быть в стрессовом состоянии. Конечно, у каждого свои методы, но я, когда звоню человеку, у которого возникла проблема со взломом компьютера, говорю:

— Пожалуйста, уберите руки с клавиатуры и отойдите от компьютера.

— Спасибо. На задней стороне системного блока есть кабель подключения к сети. Пожалуйста, найдите его и отсоедините от компьютера.

— Меня зовут Стивен Норткат, а как ваше имя?

— Рад познакомиться. Где находится ваш офис?

— Понятно. Вы можете сказать мне свой номер телефона и другие номера телефонов в офисе?

— Замечательно. Мы скоро будем. У вас есть факс? Отлично, пока наша команда в пути, я отправлю вам несколько инструкций. Нам очень нужна ваша помощь, и я буду крайне признателен, если вы будете действовать в соответствии с полученным руководством по обработке взлома. Вы можете сказать, какая операционная система установлена на вашем компьютере?

Это самые важные слова. Нужно сказать как можно меньше слов и при этом точно передать свою мысль. Тем не менее вежливость необходима, и такие слова, как "спасибо", "пожалуйста" и "отлично", никогда не помешают. Несмотря на присутствие злоумышленников, я продолжаю верить, что основная опасность заключается в наших собственных действиях. Я также работаю над интонацией. У меня отсутствует командный голос, поэтому я стараюсь говорить с уверенностью, немного медленнее, чем обычно, пытаясь добавить нотки доброты и симпатии.

Пример бланка для отправки по факсу

Офис отдела безопасности организации _____

Форма последовательности действий в ответ на взлом локального компьютера

Редакция 2.1.1

Дата: Время: Полное имя печатными буквами:

Благодарим Вас за то, что Вы уведомили отдел безопасности о случившемся инциденте. Пожалуйста, не прикасайтесь к скомпрометированным компьютерам без разрешения члена группы по устранению последствий взлома. Не оставляйте компьютер без надзора, пока прибывший специалист по безопасности не убедится, что к компьютеру никто не прикасался. Пожалуйста, опишите все подробности инцидента. Нам нужен список людей, которые были свидетелями случившегося. Напишите их имена ниже. Если нужно больше места, продолжите список на отдельном листе бумаги.

Свидетели:

1)

2)

3)

По каким признакам вы определили, что что-то произошло? Пожалуйста, опишите все как можно точнее и подробнее.

Признаки происшествия:

Следующий раздел очень важен. Пожалуйста, постарайтесь вспомнить все команды, которые вы выполняли, и все файлы, к которым вы обращались, за время от начала замеченных неполадок и до момента звонка в отдел безопасности или группу реагирования на происшествия.

Введенные команды и файлы, к которым осуществлялся доступ:

Подпись: _____

Ситуация в локальной сети

При устранении последствий взлома лучше использовать двух человек: один из них будет проверять работоспособность локальной сети, а второй, более опытный, должен следить за устранением последствий атаки на взломанном компьютере.

Проверка локальной сети

Специалист, который будет проверять работоспособность локальной сети, должен иметь при себе диктофон и описывать сложившуюся ситуацию. Запишите имена всех, кто находится поблизости. Попросите прекратить работу всех, кто не был на месте во время происшествия. В то время, пока делаются резервные копии данных взломанного хоста, задайте несколько вопросов человеку, который сообщил об инциденте. Выясните признаки произошедшего. Обратитесь к другим сотрудникам организации и узнайте, не наблюдались ли подобные признаки на их системах. Записывайте или на магнитофон, или на бумагу все, что вы видите. Каждые несколько минут подходите к специалисту, занимающемуся устранением последствий взлома, и регистрируйте его действия с указанием времени их выполнения.

Работа на взломанном хосте

Специалист, который выполняет восстановление системы после взлома, должен пригласить администратора этой системы и попросить его внимательно наблюдать за его действиями. Попросите его составить письменный отчет. Одной из основных задач является создание резервной копии данных взломанной системы.

У опытных специалистов есть свои приложения в двоичном коде, в число которых входят утилиты для выполнения резервного копирования. Если нет резервной копии для проведения сравнительного анализа и не установлены средства для контроля за важнейшими файлами, например `safeback`, то будет правильным перед выполнением каких-либо действий скопировать все файлы истории и файлы системных журналов на сменные носители. Кроме того, на сменных носителях часто сохраняют содержимое оперативной памяти (это легко сказать, но иногда трудно сделать). Лучшими резервными копиями являются побитовые резервные копии. Если создание таких копий невозможно, нужно определить важность этого компьютера и узнать, сколько есть времени на восстановление системы после взлома. Если есть подозрения в незаконных действиях и основания считать происшествие действительным взломом, желательно выполнить следующие операции:

- выключить компьютер;
- вынуть дисковод;
- положить в пакет дисковод и сделанные записи, а также отчет того человека, который сообщил о происшествии;
- положить дисковод в сейф для хранения в качестве улики.

Быстрое восстановление

Если система является критически важной, а преследование преступника — не основная цель, то нужно быстро найти источник проблем. Здесь пригодятся средства наподобие `encase` или TCT (The Coroner's Toolkit). Оба эти средства доступны как

для устаревших систем Windows (с файловыми системами FAT), так и для современных Windows-систем (файловая система NTFS). Перед запуском любого средства я обычно запускаю программу Tripwire для сохранения контрольных сумм файлов на проверяемом диске и в моей операционной системе. Это позволяет мне быстро выяснить источник проблем, если что-то пойдет не так. Раньше я использовал программу Expert Witness, но теперь ее применение запрещено решением суда.

В любом случае основная цель – определить, имеет ли система признаки, о которых сообщалось. Это называется *подтверждением происшествия*, и исследование не ограничивается проверкой данных жесткого диска. Хорошая команда по восстановлению после взлома работает сообща, и, пока один из специалистов занимается скомпрометированным компьютером, второй проверяет отчеты системы обнаружения вторжений и другие источники информации.

Устранение ошибок

Иногда после анализа ситуации оказывается, что проблему можно устранить полностью, иногда это невозможно. Здесь я хочу остановиться на превратностях судьбы специалиста по устранению последствий взлома. Сотрудники организации, в которой были взломаны компьютеры, обычно напуганы. Если прибывшая группа по устранению последствий взлома четко и быстро выполняет свою работу, то сотрудники будут весьма признательны и в их глазах вы увидите благодарность. Это хорошо. Значит, вы – герой.

Я работал на чем-то подобной работе, когда служил в ВМС США. Если какой-нибудь самолет падал в воду, мы зависали над спасшимися летчиками на вертолете, я выпрыгивал, подплывал к ним, прицеплял спасательный трос и мы вытягивали их из океана. Всякий раз, когда я потом встречал их на корабле, они мне приветливо кивали и благодарили за спасение.

Но! Если по завершению работы, на следующий день проблемы появляются снова, то вы уже не герой, вы становитесь некомпетентным идиотом. Очень важно устранить все ошибки, даже если для этого придется переставить операционную систему.

Специалист по устранению последствий взлома должен обладать полномочиями на удаление или сохранение данных в целях спасения организации. Пожалуйста, отнеситесь к смыслу предыдущего предложения со всей серьезностью. Необходимо, чтобы все действия группы восстановления после взлома были заранее одобрены одним из главных должностных лиц организации. Специалист по устранению взлома должен не бояться посмотреть в глаза очень молодому, очень преуспевающему руководителю, ответственному за программное обеспечение, который по своей прихоти уволил многих ценных сотрудников, и сказать: "Да, я знаю, насколько важен этот компьютер. Мы восстановим все данные, которые были сохранены на резервных копиях, но саму операционную систему спасти не удастся". Очень часто для устранения проблемы приходится быть максимально жесткими с подобными юнцами.

И еще. Когда я подплывал к армейским летчикам, они всегда хотели знать, а что же случилось. Было очевидно, что они хотели понять, не было ли в случившемся их вины. Точно так же и при взломе: пользователь думает, что он сделал что-то не так. Почему каждый считает, что виноват может быть кто угодно, толь-

ко не он? В разговоре будьте вежливы и спокойны. Не делайте поспешных выводов. Во многих случаях для выяснения причины инцидента приходится действовать очень аккуратно, как будто чистить капусту листик за листиком. Даже если вы уверены в вине пользователя, оставайтесь доброжелательными. Время, когда придется выяснять причины произошедшего, наступит очень скоро!

Восстановление

Целью процесса устранения последствий взлома является восстановление работоспособности системы. В ходе этого процесса нужно постараться сохранить как можно больше данных, даже если резервное копирование выполнялось очень давно. Часто можно смонтировать потенциально поврежденный диск как диск данных и скопировать с него нужные файлы. И здесь снова пригодится программа Tripwire. Перед монтированием подозрительного диска на переносной компьютер убедитесь, что запущена последняя версия Tripwire. Это гарантирует, что вредоносный программный код не проникнет на ваш компьютер.

Инструкторы по оказанию первой медицинской помощи для наглядности описываемых ситуаций используют несколько примеров. Одна из основных задач — не стать самому жертвой атаки. Например, если кто-то лежит ничком на земле, обернутый высоковольтным кабелем, к нему лучше не приближаться. Вдруг кабель и является причиной смерти этого человека? Что случится с вами, если вы возьметесь за этот кабель? Поэтому нужно сначала оценить ситуацию и продумать, какие обстоятельства сопутствовали компрометации компьютера. Будет глупо восстановить работоспособность системы, не устранив проблемы, которая привела к его взлому.

Это важный момент, так как система, скорее всего, в какой-то степени изменится. Довольно часто владельцы системы пользуются моментом и проводят модернизацию компьютера или устанавливают дополнительные заплаты. Весьма забавно, когда тот же руководитель, который в начале говорил о необходимости сохранения системы в прежнем виде, предлагает модернизировать операционную систему восстанавливаемого компьютера.

Операционную систему вполне стоит обновить. Но что происходит, когда кто-то со стороны вносит изменения в локальную сеть организации? Однажды я руководил установкой брандмауэра в сети, в которой его раньше никогда не было. В течение следующих пяти лет, если кто-то не мог установить соединение, или не работали какие-то программы, мне или звонили, или отправляли письма по электронной почте с вопросом, а не брандмауэр ли является причиной проблем. В этом и заключается сложность работы специалиста по безопасности. Вспомните нашего молодого преуспевающего руководителя. Если что-то будет работать не так, он постарается перенести внимание на эту проблему и отвлечь от факта компрометации его собственного компьютера. Как поступить в этой ситуации?

В процессе устранения последствий взлома старайтесь держать владельцев компьютеров в курсе всех событий. Пока они в опасности, они являются заинтересованными лицами. Увидев, что все становится на свои места, они переключаются на что-то другое. Очень важно в начале процесса восстановления, когда уровень адреналина в крови еще достаточно высок, отвести владельца в сторонку и сказать ему нечто подобное:

"Сэр, наша основная задача — восстановить работоспособность вашего компьютера за минимально короткий срок и с минимальными затратами. Я надеюсь, вы понимаете, что так как система была скомпрометирована, то нам придется внести определенные изменения, или взлом может повториться. Чтобы гарантировать, что внесенные нами изменения не повлияют на выполнение ваших задач, нам нужна копия технической документации компьютера, особенно сведения о конфигурации, руководства по настройке программ и, самое главное, план тестирования вашей системы. Перед тем как мы уйдем, мы будем рады вместе с вашими сотрудниками протестировать работу системы".

Известно, что едва ли найдется пять компьютеров на всей планете, для которых составлен отвечающий всем требованиям текущего момента, исчерпывающий план тестирования. Едва ли наш пронырливый молодой руководитель способен его составить. Настало время для бумажной работы. Используем заранее заготовленный бланк выполнения работ. На этом бланке есть место для подписей системного администратора, заказчика работ и владельца системы, которыми они должны заверить, что протестировали восстановленную систему, и она полностью работоспособна. Поэтому можно сказать следующее:

"Нет плана тестирования? Да, но я не могу закончить работу, не удостоверившись в работоспособности системы. Говорите, что хотите, но если ваши люди не запустят тесты, которыми они проверяют работоспособность компьютеров, и не распишутся на бланке, то мы не сможем уйти. Я готов оставаться здесь сколько потребуется, так как вы знаете, что начальник отдела безопасности не должен допускать простоя системы более чем на одни сутки, а мы не можем считать инцидент исчерпанным, пока система не будет протестирована".

Я неспроста посвятил несколько абзацев подобным заявлениям. Обидно, когда молодого многообещающего специалиста по устранению последствий взлома (особенно если это девушка) винят во всех проблемах, когда он (или она) вложил душу в восстановление системы.

Зная о риске, можно без страха начинать процедуру устранения последствий взлома. После компрометации система может перейти в полную собственность хакера. Хакеры используют различные методы для маскировки своих действий. Я знаю несколько случаев, когда скомпрометированная система была восстановлена, возвращена в эксплуатацию, а хакеры очень быстро возобновили над ней контроль. Используйте возможности своих средств обнаружения взлома для мониторинга восстановленной системы. Возможно, стоит изменить имя системы и IP-адрес, а также установить ловушку на несколько недель.

Выводы из случившегося

Сначала происшествие вызывает огромный интерес, и практически каждый хочет принять участие в его исследовании. В поисках злоумышленника изучается цепь событий, которые привели ко взлому. Затем наступает этап медленного процесса восстановления и тестирования. Это уже не так интересно, и все любопытные быстро расходятся со словами: "Ребята, я думаю, дальше вы справитесь сами". Так и происходит. Мы находим проблему, устраняем ее и восстанавливаем систему. Это стоит больших усилий. И очень трудно заставить себя оформить все бумаги.

Профессионала от любителя можно отличить по двум признакам. Во-первых, профессионал аккуратно и полностью записывает каждое свое действие, а во-вторых, он всегда доделывает работу до конца. Это не природные качества, а результат дисциплинированности. При устранении последствий любого взлома случаются ошибки. По этому поводу не следует слишком переживать. Идеал недостижим. Главное не повторять одних и тех же ошибок снова.

Выводы из случившегося — важнейшая часть процесса устранения последствий взлома, если посмотреть на него с правильной точки зрения. Нет ничего зазорного в том, что что-то можно было сделать лучше.

Специалист по устранению последствий взлома составляет черновой вариант отчета о происшествии. Машинистка перепечатывает этот отчет, размножает его и отправляет всем свидетелям, заказчику работ и владельцу системы. Каждый из них может внести свои дополнения, которые войдут в окончательный вариант отчета. При этом специалист по обработке взломов должен быть уведомлен о внесенных изменениях. В течение недели после взлома участники процесса восстановления системы должны встретиться все вместе. Единственным вопросом повестки дня должен стать анализ рекомендаций в окончательном отчете по поводу улучшения процесса устранения последствий взлома. Вряд ли удастся прийти к единому решению, поэтому лучше голосовать по каждому вопросу.

Основной частью отчета о происшествии является резюме для руководства. В нем приводятся доводы, согласно которым команда по устранению последствий взлома сэкономила для организации значительную сумму денег.

Резюме

Риск возникает при добавлении в систему каждого нового пользователя и запуске каждой новой службы (разрешенной на брандмауэре). Эффективность снижения этого риска заключается в эффективности ответных действий как автоматических, так и выполняемых вручную. Это позволяет организации быть на шаг впереди в нашем быстро меняющемся мире. Среди автоматических ответных действий можно назвать увеличение задержки ответа для замедления атаки, обрыв соединения, блокирование хакера при повторных попытках установить соединение, отключение от Internet в особо опасных ситуациях, разные хитрости с использованием протоколов (например, отправка пакетов SYN/ACK для незапущенных служб).

В каждой организации есть группа людей, занимающихся устранением последствий при чрезвычайных ситуациях (возможно, неофициальная). Профессиональная группа по устранению последствий взлома компьютеров должна следовать стандартной последовательности действий: подготовка; идентификация проблемы; ограничение распространения атаки; устранение ошибок; восстановление; подведение итогов. Аналитик сетевого трафика всегда должен входить в состав такой группы, и лучше, если в качестве ее руководителя.

Согласно одной из моделей безопасности период защищенности зависит от времени, требующегося на обнаружение атаки и выполнение ответных действий. При улучшении методов автоматических ответных действий и действий вручную мы учимся действовать быстрее и эффективнее, тем самым повышая защиту сетей контролируемых организаций.

19

Бизнес-план обнаружения взлома

С чего начать? Какое средство обнаружения взлома лучше использовать? Студенты часто задают мне эти вопросы, прослушав наш углубленный курс практического применения навыков обнаружения вторжений. Такие вопросы меня всегда удивляют. Мы провели шесть дней и вечеров, в подробностях изучая тайные ходы хакеров, вредоносные пакеты и методы определения операционных систем на удаленных хостах. Мы раз за разом повторяли, что нет никакого универсального решения, поэтому любая система обнаружения вторжений должна использоваться совместно со средством перехвата всего трафика в сети. Я ответил на заданный вопрос своим вопросом: "Если в вашей организации сейчас не используется никакое средство обнаружения взлома, то зачем оно вообще нужно? Что изменилось?". Борьба за установку средств обнаружения взлома является весьма непростой задачей, и инерционность мышления руководителей бывает очень сложно преодолеть.

Мы подходим к концу нашего повествования, поэтому я бы хотел успеть уделить внимание составлению бизнес-плана обнаружения взлома. Это очень важный этап работы. В предыдущих главах было показано, насколько сложны и интересны знания, позволяющие работать аналитиком сетевого трафика. Все мои коллеги в области безопасности компьютеров считают свою работу очень захватывающей, но это еще не причина, чтобы устанавливать средства обнаружения вторжений в своей организации. Тот, кто внимательно изучил материал первой половины этой книги, обладает достаточным объемом технических сведений для обнаружения взлома. Но этого недостаточно. Я равняюсь на трех ведущих специалистов в сфере информационной безопасности: Рона Гула (Ron Gula), Маркуса Ранума (Marcus Ranum) и Мэри Росч (Marry Roesch), каждый из которых может начать разговор фразой: "Я, как бизнесмен...". Каждый из нас является предпринимателем в той или иной степени. Это справедливо и для тех, кто работает на правительство, в университете или некоммерческой организации. Если возникла хоть мысль о необ-

ходимости защиты, значит, ваша организация владеет немалым капиталом. Мы потратили много сил на разработку технических и архитектурных методов решения проблемы безопасности, а также обсудили вопросы, связанные с управлением риском. Если средство для обнаружения взлома не вписывается в бизнес-модель вашей организации, оно станет причиной разногласий. Давайте попытаемся вместе разработать стратегию и продумать действия для интеграции возможностей обнаружения взлома в работу организации.

Материал этой главы в первую очередь предназначен для профессионалов в области информационной безопасности, которые:

- не имеют средств обнаружения вторжений и в настоящее время обдумывают преимущества и недостатки того средства, которое хотят приобрести;
- обладают устаревшими возможностями защиты и планируют приобретение новых или модернизацию старых средств;
- используют средства обнаружения взлома, но хотят знать, сохранятся ли возможности защиты при текущей реструктуризации их организации.

В нашу задачу не входит цель поразить всех новыми технологиями. Призывов к оружию и громких криков об исходящих от хакеров угрозах будет недостаточно для длительного финансирования проекта, хотя может помочь выбить деньги на дополнительные приобретения.

В этой главе рассмотрен состоящий из трех частей план, который демонстрирует важность обнаружения взлома. В первой части плана рассмотрены проблемы, связанные с руководством организации. В ней не рассматриваются технические вопросы — она предназначена для использования в качестве шпаргалки при убеждении начальства в необходимости установить средства информационной безопасности.

Вторая часть плана посвящена ответу на вопрос: "Зачем нужно обнаруживать взлом?". В ней рассмотрены угрозы и уязвимые места, и это тот этап, на котором нужно применить полученные знания о риске.

В третьей части предлагаются различные решения. Целью является создание письменного доклада, который бы служил планом проекта и его обоснованием. Я постарался сделать описание максимально наглядным, чтобы его можно было использовать на презентации (основной метод работы с руководством в наши дни). Каждый вопрос подан как схема, которую можно представить в виде слайда PowerPoint. Для дополнительного впечатления в слайдах PowerPoint можно использовать фрагменты из текстового отчета.

Все презентации и отчеты должны начинаться с введения, называемого *резюме для руководства* (executive summery), в котором должен быть показан экономический эффект проекта. При выступлении перед начальством всегда будьте готовы к рассказу в очень сжатые сроки. Нередко выдвигается жестокое условие "Уложитесь в пять минут?". В этом случае лучше показать всего три слайда: эффект внедрения, смета и график выполнения работ. Описав экономический эффект, следует раскрыть существующую проблему, устранение которой и является основной целью проекта. Возможно, здесь стоит воспользоваться информацией из второй части вашего бизнес-плана. Третий слайд — это сетевой график, где дается описание структуры презентации.

Часть первая: работа с руководством

Наша цель – продемонстрировать руководителю, что предлагаемый проект является частью общей стратегии по защите информации и дает неоспоримые преимущества для организации. При этом отличительными характеристиками предлагаемого решения должны быть:

- реальный результат;
- точная смета расходов;
- технология, не нарушающая работу организации.

Кроме того, проект должен быть частью общей стратегии защиты информации.

Реальный результат

Нужно быть реалистами. Возможность обнаружения взлома – недешевое удовольствие. Нужно иметь два быстродействующих компьютера (по 2,5 тыс. долл. каждый). Для коммерческой системы обнаружения взлома нужно купить лицензию на право ее использования (10 тыс. долл.). Таким образом, только для начала требуется 15 тыс. долл. Возможно, придется вносить изменения в локальную сеть, а это расходы на зарплату специалисту. Кроме того, нужно платить жалование аналитику сетевого трафика. Тут уже можно говорить о сумме в 100 тыс. долл. Но и это еще не все. Добавьте 100 тыс. долл. на увеличение пропускной способности, установку шести датчиков и высококачественного коммутатора для контроля за серверами. Нужна база данных для отслеживания низкоскоростных методов сканирования, RAID-матрица жестких дисков для хранения данных. Если окружающие начинают стараться снизить расходы, необходимо ясно продемонстрировать выгоды, оправдывающие все затраты.

И это не так сложно сделать. В любом бизнесе всегда присутствует риск. Вы не пропустили главу, посвященную риску? Каким образом использование системы обнаружения вторжений позволяет снизить среднегодовую вероятность ущерба (annualized loss expectancy – ALE)? Наблюдая за атаками на локальную сеть организации, специалист по защите информации может настроить брандмауэр и другие средства защиты так, чтобы эти атаки не причиняли никакого ущерба. Окупит ли это затраты в 100–350 тыс. долл.? Если нет, то с помощью систем обнаружения вторжений можно снизить риск и другим способом. С коммерческой точки зрения может оказаться, что для работы определенных приложений необходимо оставить в системах некоторые уязвимые места. Например, при добавлении в систему новых заплат безопасности некоторые приложения перестают работать. Можно настроить системы обнаружения вторжений на отслеживание попыток атак с использованием этого уязвимого места. Вообще, применять системы обнаружения вторжений выгодно. Это можно показать и доказать на примерах.

Обнаружение взлома с помощью брандмауэров

Одним из самых значительных событий на рынке продуктов для обеспечения информационной безопасности стало появление брандмауэров с возможностью регистрации всех двоичных данных. Брандмауэр операционной системы OpenBSD (IPFilter) и коммерческий брандмауэр Raptor могут сохранять данные в формате BPF. Сохранение данных в двоичном формате позволяет использовать для их анализа программы Snort или TCPdump. Это про-

сто невероятно! Я уже упоминал устройства для брандмауэров hogwash и UnityOne, в которые встроены возможности обнаружения взлома. Лично я предпочитаю использовать оба устройства: если одно выйдет из строя, его подменит второе.

Кроме того, для обнаружения взлома по-прежнему можно использовать и другие брандмауэры без возможности сохранения данных в двоичном формате. В качестве примера назовем Dshield (www.dshield.org) — технологию, применяемую компанией Incidents.org, которая позволяет использовать данные, сохраненные брандмауэром для общей возможности выявления взлома. Брандмауэры могут использоваться как датчики. Разумеется, они не фиксируют большинство значений полей TCP-заголовков, например TCP-флаги, что значительно ограничивает возможности анализа трафика. Однако брандмауэры Linux (например, программа iptables) позволяют использовать большинство методов анализа трафика, описанных в этой книге.

Если приобретение системы обнаружения вторжений невозможно, то при исследовании журналов своих брандмауэров можно и нужно использовать сведения, которые изложены в этой книге. Безусловно, сначала нужно получить разрешение и не стоит поднимать тревогу раньше времени!

Предел расходов

Недавно в газете "Sunday" была опубликована статья под заголовком "Десять советов о том, как повысить свое благосостояние". В статье рекомендовалось не покупать лодку, а если она есть, продать ее. Я не настолько нуждаюсь в деньгах, чтобы выбрасывать свои лодки, тем более, что они так и не упали в цене.

Я расскажу еще одну историю, которая поможет понять озабоченность руководителя при бесконечной трате денежных средств. Однажды я сообразил, что вся моя работа заключается в использовании нескольких переносных компьютеров и расходов на мобильный телефон. Тогда я понял, что могу жить где угодно, лишь бы там была сотовая связь и высококачественное соединение с Internet. Во время отпуска на Гавайях меня попросили выполнить срочную работу по установке системы обнаружения взлома на Оаху. Через две недели мы с женой купили дом нашей мечты на Гавайях. Через месяц после переселения дом мечты превратился в дом кошмаров, так как оказалось, что он построен на проседающей почве. Начались бесконечные консультации с агентами страхования, которые заявляли, что в страховке этот пункт не оговорен, за ними пришли инженеры, утверждавшие, что раньше не видели ничего подобного. Наконец, мне рассказали о лучшем подрядчике на острове, Луисе Солтрене, но к тому времени дом уже почти развалился. Услуги Луиса, как и любого профессионала, стоили недешево. Деньги начали улетать, как в трубу. Мы находили новые и новые проблемы. Я не новичок в строительстве, поэтому понимал, что все расходы были необходимыми. Общая смета становилась все больше и больше. В конечном итоге она составила (без преувеличения) 200 тыс. долл. Я заработал язву, а расходы все увеличивались. Когда перестройка дома была закончена, я получил один важный урок. Фраза "общая стоимость владения" в наше время очень широко используется в сфере информационных технологий, но я никогда над ней не задумывался, пока не занялся строительством своего дома, когда конечную сумму расходов знать заранее было невозможно.

Теперь применим этот опыт по отношению к обнаружению взлома и руководству организации. Не забывайте, что хороший руководитель считает каждый доллар, и неконтролируемые расходы вызывают у него плохое настроение.

Возможно, что руководитель будет готов заплатить даже 100 тыс. долл., но нужно доказать, что предлагаемые цифры расходов верны, и что вы не будете приходить за деньгами снова и снова. Например, классической ошибкой является планирование использования для системы обнаружения вторжений купленных ранее компьютеров прошлого поколения. Лучше сделать наоборот: купить компьютеры последнего поколения и через полгода или год перевести их в разряд настольных компьютеров.

Руководство наверняка оценит такой честный и вдумчивый подход. Это позволит организации получить все возможности обнаружения взлома и сэкономить деньги на последующей модернизации парка компьютеров.

Дестабилизация положения дел

Система обнаружения вторжений как бы "открывает глаза" организации на истинное положение дел. Студенты, прошедшие курс обучения в SANS, по возвращению на рабочие места часто осознают, что должны в корне пересмотреть методы работы в своей организации. И это правильно! Но это перемена, а люди по своей натуре подозрительны и консервативны. Предлагая добавить возможности обнаружения взлома, будьте очень внимательны по отношению к возможным изменениям в рабочем процессе организации и сделайте все возможное, чтобы убедить сотрудников в том, что система обнаружения вторжений "впишется" в этот процесс. Сложности могут возникнуть в связи с конфигурацией сети, режимом работы некоторых сотрудников и необходимостью добавить новые правила безопасности.

Проблемы с локальной сетью

Мы уже рассказывали о сложностях работы в коммутируемых сетях. Перед установкой системы обнаружения вторжений следует обязательно рассмотреть вопрос координации действий с пользователями локальной сети. Для этого лучше всего использовать связующий порт коммутатора, к которому подключается анализатор пакетов. Если поддержка работоспособности одного связующего порта слишком трудна для системного администратора, стоит рассмотреть возможность создания ответвлений от сети для прослушивания портов системой обнаружения взлома. Датчик системы обнаружения вторжений часто оснащают несколькими сетевыми адаптерами для следующих целей:

- прослушивание портов в неразборчивом режиме (promiscuous mode), но без IP-адреса, что усложняет злоумышленникам задачу выявления датчика;
- создание интерфейса с IP-адресом для связи с датчиком.

Установка системы обнаружения вторжений практически всегда требует изменений в работе брандмауэра. Похоже, что все поставщики коммерческих систем считают, что создание собственного протокола для взаимодействия консоли аналитика, датчиков и баз данных будет выделять их среди конкурентов. И, конечно, только их система работает абсолютно правильно. Придется потрудиться и самостоятельно выяснить, какие порты должны оставаться открытыми для нормальной работы конкретной системы обнаружения вторжений. Хорошо, если эту систему можно изменить так, чтобы использовать уже открытые порты брандмауэра. Даже в брандмау-

эрах, работающих на proxy-серверах, часто имеются возможности пропуска пакетов только по паролю, без каких-либо сведений об используемом протоколе. Этот метод применяется для поддержки некоторых приложений. Будет просто здорово, когда группа IDWC (Intrusion Detection Working Group) закончит свою работу над стандартным транспортным протоколом на основе тонально-модулированного сигнала (`www.beepcore.org/beepcore/docs/profile-idxp`) для всех систем обнаружения вторжений.

Изменение стиля работы сотрудников

Необходимость изменения стиля работы является еще одним следствием установки системы обнаружения вторжений. Система обнаружения вторжений не защищает от всех напастей. Она представляет собой всего лишь средство для сбора и анализа данных, поэтому при желании все равно можно нарваться на неприятности. Вы должны подготовить сотрудников организации к тому, что теперь весь передающийся по сети трафик будет просматриваться. В качестве примера рассмотрим использование сервера IRC.

После установки системы обнаружения взлома вы обнаруживаете, что молодой человек из отдела по обслуживанию компьютеров использует один из локальных компьютеров как IRC-сервер. Как можно об этом узнать, если служба IRC не контролируется? Известно, что DNS-серверы, Web-серверы и серверы электронной почты вызывают интерес со стороны злоумышленников. Однако этот интерес нельзя и близко сравнить с заинтересованностью хакеров к серверу IRC! Аналитик сетевого трафика сразу же увидит бесчисленное количество атак и попыток сканирования этого сервера. Дальнейшее расследование позволит выявить цель любопытства хакеров. Очевидно, что для организации будет лучше устранить этот источник проблем. Опытный аналитик сначала определяет политику безопасности и только потом запускает систему обнаружения вторжений.

Политика безопасности

В первое время в организации может использоваться политика всепрощения. Приблизительно к первым десяти нарушениям установленной политики можно отнестись со снисхождением. Затем можно разослать уведомление, в котором будут перечислены примеры нарушений и указано, что в дальнейшем такие действия прощаться не будут. Я видел несколько организаций, в которых включили новенькие системы обнаружения вторжений и сразу же начали внимательно исследовать передающийся трафик. Представьте себе их удивление, когда они увидели массу того, что было запрещено для получения или отправки. В этот момент организация должна принять важное решение. Если виновные будут немедленно наказаны или даже уволены, то система обнаружения вторжений всегда будет вызывать недоверие и враждебность. Особенно осторожно нужно общаться с системными администраторами и администраторами сетей. Они привыкли к тому, что могут делать все, что угодно. Если кто-то из группы по обслуживанию компьютеров или локальной сети будет уволен по причине нарушения политики безопасности, то, вполне вероятно, что система обнаружения вторжений будет часто выходить из строя из-за умышленного обрыва кабелей.

Руководители организаций обычно осведомлены о взаимоотношениях в организации и особенно о проблемах между начальниками и подчиненными. Они сами сталкиваются с ними ежедневно. Они будут более радушны, если в вашем бизнес-плане будет продемонстрирована забота о других сотрудниках организации, и вы гарантируете, что система обнаружения вторжений не станет причиной разногласий.

Часть общей стратегии

Эта книга в основном предназначена для помощи специалистам по обслуживанию сетевых систем обнаружения вторжений. Однако мы также затронули темы безопасности отдельных компьютеров, риска, программ сканирования уязвимых мест, несанкционированного доступа, восстановления после взлома, а теперь рассмотрим методы ведения бизнеса. Вы должны быть готовы продемонстрировать, как система обнаружения вторжений вписывается в общую стратегию обеспечения информационной безопасности организации.

Если честно, то раньше я так не думал. Я считал, что моя работа заключается в установке наиболее эффективного средства за минимальную цену, что поможет моей организации достичь "мирового уровня". Намерение неплохое. Однако когда я подходил к своему начальнику и предлагал реализовать новую технологию, он отвечал: "Это все хорошо, но покажи мне общую картину". Это меня выводило из себя. Я думал, что он полный идиот и какой-то "компьютероненавистник". Только через 15 лет я начал его понимать. Нельзя сыграть на арфе с одной струной. Любая технология, даже наилучшая, бесполезна, если она не соответствует методам ведения бизнеса в организации. Информируя руководство о великолепной системе обнаружения вторжений, которую нужно приобрести, не забывайте осветить вопросы защиты систем, риска, сканирования уязвимых мест, несанкционированного доступа, устранения последствий взлома и влияния на ведение всей коммерческой деятельности. Позвольте мне еще раз напомнить семь основных принципов обеспечения безопасности.

1. Установить политику безопасности (с учетом ведения бизнеса).
2. Проанализировать возможные риски; исследовать уязвимые места.
3. Организовать инфраструктуру безопасности.
4. Разработать методы контроля и установить стандарты для каждой технологии.
5. Определить, какие ресурсы будут доступны пользователям, задать приоритеты для мер противодействия незаконным операциям и реализовать наиболее строгие меры противодействия, какие только можно себе позволить.
6. Выполнять периодические проверки и по возможности тестирование.
7. Установить систему обнаружения вторжений и определить порядок ответных действий в случае атаки.

Если предложение о покупке системы обнаружения вторжений будет составлено с учетом этих принципов, то руководству сразу будет ясно, что оно является частью общей стратегии информационной безопасности. У руководителя нет времени на изучение отдельных частей информации, он должен руководить всем процессом. Потратьте немного своего времени, чтобы облегчить его работу.

Таким образом, мы рассмотрели четыре основные темы, которые должны быть изложены руководству в бизнес-плане обнаружения взлома. Если эти темы не будут освещены, то бизнес-план так и останется только на бумаге. Повторим еще раз:

- реальный результат;
- точная смета расходов;
- технология, не нарушающая работу организации;
- проект, являющийся частью общей стратегии защиты информации.

Теперь можно перейти к технической стороне вопроса, которой отведена вторая часть вашего плана или предложения.

Часть вторая: угрозы и уязвимые места

Во второй части вашего плана должны быть раскрыты угрозы в сопоставлении с существующими уязвимыми местами и стоимостью имущества. Все это делается, чтобы ответить на вопрос о необходимости дополнительных мер безопасности. Я считаю, что главным предназначением сетевой системы обнаружения вторжений, установленной перед брандмауэром, является оценка опасности атак, направленных против хостов организации. Благодаря такой системе можно организовать надежную защиту от выявленных атак. Но как можно оценить угрозы, когда система обнаружения вторжений еще не куплена? Необходимо исследовать проблему, угрозы и уязвимые места до того, как предлагать средство обнаружения взлома в качестве решения. Базовые сведения о риске, необходимые для написания этой части плана, были изложены в главе 17, "Безопасность в организации". Вторую часть плана можно разбить на несколько пунктов:

- оценка и анализ угроз;
- оценка имущества;
- анализ уязвимых мест;
- оценка риска.

Оценка и анализ угроз

С формальной точки зрения сначала нужно взять словарь, выписать все возможные угрозы, а затем приступить к их анализу. Это плохая идея для плана внедрения системы обнаружения вторжений, который будет читать руководитель. Ваша цель – не показать все возможные угрозы, а выбрать из них наиболее важные. Основное внимание должно быть уделено тому, что может произойти и по какой причине. Ниже перечислены основные угрозы в порядке убывания степени их опасности:

- внешняя атака по сетевому кабелю;
- внешняя атака по телефонному соединению;
- атака локального пользователя по локальной сети;
- атака локального пользователя той же системы;
- атаки с помощью вредоносных программ.

Источники атак

В туалетах ресторанов часто висит табличка "Не забудьте вымыть руки перед едой". Общеизвестно, что несоблюдение этого правила гигиены является причиной (источником) многих болезней.

Сетевые системы обнаружения вторжений позволяют выявлять внешние атаки по сети, локальные атаки по сети и, возможно, атаки с использованием специальных программ. Системы обнаружения вторежений для защиты отдельного хоста способны выявлять все типы атак.

Поиск угрозы

В бизнес-плане обнаружения взлома из всех существующих угроз должны быть выделены хорошо известные, наиболее вероятные угрозы. Как их определить? Для этого можно использовать информацию из следующих источников:

- газетные публикации или статьи на Web-сайтах об атаках в других местах (если это случилось один раз, то может повториться еще);
- журналы брандмауэров или средств обнаружения взлома с описанием конкретных атак;
- системные журналы;
- сведения, полученные в результате демонстрационного использования системы обнаружения вторжений.

Многие поставщики коммерческих систем обнаружения вторжений разрешают тестировать свои системы в течение примерно 30 дней. Если вы действительно хотите реализовать качественное обнаружение атак, то такую возможность нельзя переоценить. Это позволяет получить список действительных атак против вашей сети, т.е. определить реальные угрозы. В итоге мы получаем обоснование для третьей части нашего плана, в которой приводятся аргументы в пользу установки именно системы обнаружения вторжений, а не, скажем, еще одного брандмауэра. Кроме того, можно узнать о справедливости заявлений поставщика коммерческой системы. Слишком часто люди принимают решение или на основе где-то вычитанной информации или по совету продавца. Временное использование системы обнаружения вторжений позволяет принять правильное решение при выборе той или иной системы в третьей части плана.

Если на это есть время, то побеседуйте с сотрудниками разных отделов вашей организации. Они могут указать на уязвимые места, которые вы могли не заметить. Иногда на вопрос о самой большой угрозе для защиты информации в организации я получал ответы, в которых люди рассказывали о поразительно неосторожных действиях. Однако по собственной инициативе сотрудники ничего не говорят. Как мне сказали однажды в Алабаме: "А вы никогда не спрашивали".

Оценка имущества

Об оценке имущества говорилось в главе 17, "Безопасность в организации". Рассмотрим эту тему подробнее. Данные представляют огромную ценность. Если большинство сотрудников используют компьютеры на протяжении большей части рабочего дня, то стоимость данных, хранящихся на жестком диске их компью-

теров, может быть определена как затраты на заработную плату конкретного владельца. Угроза для данных заключается в возможности их копирования, уничтожения или изменения.

Эта тема не раз затрагивается на страницах данной книги. Повторю еще: основной угрозой для информации является ее уничтожение самим владельцем системы. Таким образом, есть опасность как действия, так и бездействия. В первом случае это явное действие: случайное или умышленное удаление данных без возможности их восстановления. Под бездействием понимаются ошибки при резервном копировании данных, в том числе тех копий, которые хранятся вне локальной сети.

Полностью защитить все данные от копирования, уничтожения или изменения практически невозможно. То же самое можно сказать и о возможности резервного копирования всей выполненной работы, которой не обладает ни одна из известных мне организаций. Подобное вступление часто используется в текстах по защите информации при описании *критически важных данных*.

Стоимость данных

Разные данные имеют разную ценность. На самом деле только небольшая часть информации организации позволяет ей выделяться среди конкурентов. Именно эти данные должны быть защищены лучше всего.

Этот аспект должен найти свое отражение в вашем плане, а именно в разделе, посвященном угрозам. Рассмотрите источники атак и наиболее известные угрозы и используйте эти примеры описания угроз и уязвимых мест во второй части вашего бизнес-плана обнаружения взлома.

В третьей части плана будут рассмотрены меры противодействия описанным угрозам целостности наиболее ценных данных. К этим мерам относятся:

- установка систем обнаружения вторжений на критически важных системах;
- создание контролируемых файлов. Если в вашей организации есть особо важные документы, то в них можно добавить специальные строки символов. Появление в трафике этих строк будет отслеживаться системой обнаружения вторжений с помощью заданного правила. Теперь вы будете знать, когда важный документ поступает в локальную сеть или отправляется за ее пределы;
- оснащение локальных компьютеров персональными брандмауэрами;
- установка сетевых систем обнаружения вторжений в особо важных точках сети.

Поиск уязвимых мест

Уязвимые места — это ворота, через которые действуют угрозы. Если уязвимые места отсутствуют, то никакая угроза не страшна.

Вы разочарованы, что я не привожу длинный перечень всех уязвимых мест? Но этот список постоянно меняется, и точное перечисление просто невозможно. Лично мне очень по душе проект CVE (`cve.mitre.org`), так как там даются ссылки на многие списки уязвимых мест, например `bugtraq` или X-Force. Не следует находить эти уязвимые места без использования специальных программ. Со-

ставить список основных угроз и оценить главные уязвимые места совсем не сложно. Для этого существует немало программных средств. С их помощью можно выявить потенциально уязвимые места и проверить реальность конкретной угрозы. Рассмотрим три типа таких программ: программы сканирования уязвимых мест на хосте, программы сканирования уязвимых мест сети и программы сканирования телефонных линий.

Пакеты COPS, SPI, Tiger и STAT используются для сканирования уязвимых мест на локальных компьютерах. Они позволяют выявить отсутствие заплат, некорректно установленные права доступа к файлам и другие похожие проблемы.

Для поиска уязвимых мест в сети используются программы сканирования nmap, `nessus`, `saint`, Internet Scanner и NetRecon. Они работают быстро и эффективно и позволяют выявить незащищенные порты или службы на хостах всей сети.

При оценке уязвимых мест также можно оценить риск сканирования злоумышленниками телефонных линий в поисках работающих модемов. Наиболее популярной программой для этой цели является Toneloc, которая доступна практически на всех сайтах хакеров. Коммерческая программа Phonesweep (`http://www.sandstorm.net`) обладает расширенными возможностями сканирования телефонных линий.

Оценка уязвимого места должна состоять из трех компонентов:

- **со стороны системы** (определяется с помощью программ сканирования систем);
- **со стороны сети** (выполняется внутреннее сканирование локальной сети);
- **со стороны Internet** (сканирование локальной сети и (желательно) телефонных линий выполняется из точки перед брандмауэром).

Не забудьте отдельно просканировать свой брандмауэр, DNS-сервер, Web-сервер и сервер электронной почты, а также наиболее важные системы. От надежности защиты этих систем зависит безопасность вашей организации.

Очевидно, что работы будет много. Кроме того, оценка уязвимых мест не выполняется один раз и навсегда. Зачем аналитику сетевого трафика участвовать в поиске уязвимых мест? Это позволит ему выявить конкретные недостатки, а также уязвимые места в локальной сети организации.

Оценка риска

Мы получили большой объем сведений. Что с ними делать? Полученную информацию совсем не нужно включать в отчет, даже в виде приложений. С другой стороны, следует упорядочить эту информацию и обеспечить к ней оперативный доступ. При докладе руководителю все-таки нужно оперировать какими-то реальными данными. Это значит, что если есть 12 компьютеров, на которых хранится критически важная информация, то вы должны иметь под рукой список этих систем и рациональное пояснение, почему были выбраны именно они.

Итак, мы разобрались, что не нужно включать во вторую часть отчета. А что должно присутствовать обязательно?

- Стоимость имущества организации. Предположим, есть 100 компьютеров, рассчитанных на пятилетний срок эксплуатации. Аппаратные средства,

программное обеспечение и обслуживание стоят 200 тыс. долл. в год. Таким образом, стоимость информации составляет 1 млн. долл.

- Схема сети, на которой будут показаны границы защищаемой зоны.
- Краткое описание основных источников угроз.
- Выводы относительно обнаруженных уязвимых мест.
- Описание критически важных систем: где они находятся, их стоимость и т.д. (включая брандмауэр, сервер электронной почты, DNS- и Web-сервер).
- Конкретные угрозы для критически важных систем.
- Конкретные уязвимые места критически важных систем.

Этот доклад должен существовать в письменном виде, а также в виде графической презентации. При создании презентации с помощью PowerPoint представьте каждый пункт приведенного выше списка в виде отдельного слайда с тремя-пятью пунктами.

Часть третья: альтернативы и рекомендуемое решение

Наконец-то вы установили свою систему обнаружения вторжений и не можете дождаться того момента, когда сядете за новую консоль. Подождите еще немного. Необходимо предложить ряд альтернативных вариантов, не забывая при этом, что все должно выглядеть как единое целое. Обнаружение взлома является сложной задачей. Следующим этапом оценки риска должно стать определение критериев его допустимости. Это значит, что нужно дать руководству возможность самостоятельно определить допустимый уровень риска. Затем снова приходит ваш черед — нужно разработать эффективные меры противодействия всем рискам, уровень которых будет выше допустимого. Для этого нужно сделать следующее:

- описать архитектуру управления риском согласно концепции защиты информации;
- определить, что уже сделано;
- указать неотложные действия;
- описать рекомендуемые меры противодействия;
- провести анализ по типу "затраты-выгода";
- составить график выполнения проекта;
- описать следующие действия, чтобы определить будущее направление деятельности.

Описание архитектуры управления риском

Задача может показаться сложной, но на самом деле все просто. Основные угрозы уже определены. Меры противодействия тоже известны и заключаются в применении следующих аппаратно-программных решений:

- установка брандмауэра на линии соединения с Internet;
- внедрение сетевой системы обнаружения вторжений перед брандмауэром;

- применение локальных брандмауэров для критически важных систем;
- использование сетевых систем обнаружения вторжений для защиты группы наиболее ценных компьютеров;
- установка системы обнаружения вторжений на критически важных системах;
- добавление подложных файлов в файловые системы хостов, играющих ключевую роль;
- базовая защита всех компьютеров (антивирусные программы, заплаты, контроль за установкой корректных прав доступа к файлам);
- выполнение резервного копирования информации на накопитель, установленный за пределами локальной сети;
- ежеквартальное сканирование локальной сети для выявления уязвимых мест;
- шифрование на основе цифровых подписей при обмене данными по Internet с клиентами, заказчиками, а также работающими дома сотрудниками;
- строгая аутентификация пользователей при входных соединениях;
- шифрование данных на дисках и персональные брандмауэры для переносных компьютеров.

Этот список может быть дополнен для конкретной организации, но он дает общую картину правил обеспечения защиты информации.

Что уже сделано

В каждом докладе руководителю должно указываться текущее состояние дел. Если после описания архитектуры управления риском при защите информации дать определение текущей ситуации, то сразу становятся очевидными ваши последующие действия.

Рекомендации

Вы определились с системой обнаружения вторжений своей мечты. Хорошо бы к ней в дополнение получить программу сканирования уязвимых мест (или что-то подходящее для вашей организации). В полном плане по внедрению системы защиты должны быть перечислены возможности, предоставляемые в результате реализации этого плана, а также указана стоимость и определен график выполнения работ.

Я очень расстраиваюсь, когда вижу, как кто-то забирает час времени руководителя на "краткое" описание проблемы и предлагает свое решение этой проблемы, не имея сметы и дополнительных данных. Если руководителю покажется недостаточным объем предоставленной информации, то проект будет просто отложен. Здесь нужно учитывать психологию человека. Когда мы в первый раз слышим о возникшей проблеме, мы пугаемся и пытаемся что-то сделать для ее устранения. Однако, если ничего так и не будет сделано, то проблема теряет свою остроту, и повторное ее описание не вызывает желания немедленно все исправить. Поэтому необходимо, чтобы ваш проект был принят с первого раза!

Описание мер противодействия

Ненавижу это! Я знаю, чего я хочу! Я изучил рынок. Почему я должен описывать выбранный мною продукт? Если вы не знали, то поставщики коммерческих систем обнаружения вторжений совсем не глупы. Они стараются выйти на руководящих лиц вашей организации и рассказать им о своей системе в самых общих чертах (ориентируясь на предоставляемые преимущества решения проблем), не углубляясь в технические детали. Например, во многих организациях прежде всего беспокоятся об атаках со стороны локальных пользователей. Поэтому, обратившись к руководителю такой организации с проектом, в котором не будут упомянуты локальные угрозы или этому не будет уделено должного внимания, вы можете навсегда потерять свою репутацию.

Персональные брандмауэры

Если при встрече с руководством встает вопрос об атаках локальных пользователей, то пригодятся данные, сохраненные локальными и персональными брандмауэрами. Они могут предупреждать специалиста по безопасности о возникновении проблем в локальной сети. Перед тем как спрашивать руководителя о том, кто в организации несет ответственность за управление риском, финансирование и поддержку, следует как можно больше узнать о потенциальном риске.

Оставьте время на описание в докладе по крайней мере еще одного варианта решения проблемы и альтернативного продукта, приобретение которого вы рекомендуете. На эту тему не нужно создавать слайды. Этот вариант оставьте только на случай возникновения проблем. При этом нужно следить за согласованностью всего предлагаемого решения. Следует четко сформулировать свои цели: вам нужна игрушка, чтобы получить опыт работы и усовершенствовать свои знания, или вы хотите защитить свою организацию? Зачем было нужно изучать журналы брандмауэров? Если они работают правильно и хорошо, то, возможно, стоит потратить время и деньги на другие проблемы системы защиты информации.

Сравнительный анализ затрат и выгод

Сторона затрат в этом разделе важнее получаемых выгод. Здесь вы намекаете своему руководству, что осведомлены о том, сколько будут стоить рекомендуемые меры противодействия. Когда я как руководитель проекта, понимаю, что мне что-то нужно, я не хочу знать никакой лишней информации — только скажите мне, сколько это будет стоить, и когда я его получу. Выше мы говорили о ситуации, когда весь доклад должен уложиться в пять минут. В этой ситуации нужно показать только три слайда: резюме для руководства, смета и график выполнения работ.

Важность анализа затрат и выгод

Аналитикам сетевого трафика совершенно не хочется заниматься подсчетом затрат и выгод, но написание даже одной страницы финансового анализа стоит потраченного времени. Когда-то у меня работала одна очень способная сотрудница, которая имела талант выдвигать абсолютно неосуществимые проекты. Поскольку вообще она была очень сообразительна, вначале я рассматривал ее предложения всерьез. Потом я понял, что каждый раз буду даром тратить свое время, потерплю поражение и буду выглядеть полным глупцом в глазах начальства. Как вы думаете, что было потом? Она снова приходила ко мне со своей обычной фразой "Я думаю, мы должны...". Мое сердце замирало, мысли путались, это был очередной стресс. Более опытный начальник сел бы с ней рядом и научил ее подсчитывать стоимость, риск и потенциальные выгоды предлагаемых действий. Это очень просто, стоит

только один раз научиться. Я просто напоминал ей о предыдущих ошибках, отказываясь выслушать, возможно замечательное предложение хорошего программиста.

Не все выгоды очевидны, и это очень важно. Настал момент для подсчета затрат. В письменном докладе нужно перечислить все затраты, в презентации можно ограничиться основными.

У вас когда-нибудь возникали разногласия с руководством? Допустим, одно из высокопоставленных лиц заявляет: "Не думаю, что это будет работать". Он даже не дает никаких объяснений. Возможно, его интеллектуальные способности очень высоки, но ведь вы в центре внимания. Здесь поможет предварительная подготовка. Представьте себе следующий диалог. Руководители часто задают вопросы, а вы отвечаете. Будет ли система обнаружения вторжений:

- блокировать все атаки? Нет;
- обнаруживать все попытки атак? Нет;
- требовать значительных расходов на покупку оборудования и зарплату? Да.

Как видите, подготовка действительно нужна! В качестве дополнительного материала я рекомендую подсчитать значение параметра ALE или SLE для наиболее опасных (по вашему мнению) угроз для организации. По возможности лучше знать пару примеров конкретных угроз для критически важных систем. Выбирайте эти угрозы с осторожностью: так, чтобы они соответствовали рекомендуемым мерам противодействия. Если эти слайды понадобятся, а их не будет, то ваш доклад будет иметь "бледный вид". Поэтому постарайтесь подготовиться как следует.

Бизнес-план

Я откровенный, бесхитростный человек и хочу быть честным до конца. Я физически не способен создать документ, который можно было бы назвать "анализ затрат и выгод". Конечно, 9 из 10 консультантов согласятся с подобным названием для документа, который предоставляется руководству для окончательного одобрения проекта. Возможно, именно этого и ждет руководитель, принимающий решение. Позвольте мне рассказать, что делаю я. Я пишу несколько страниц бизнес-плана, в котором основное внимание уделяется существующим проблемам. Его содержимое во многом схоже с анализом затрат и выгод. Я описываю затраты, преимущества, очевидные и неочевидные выгоды и то, как это поможет в продвижении бизнеса. По моему мнению, все действия должны служить двум целям: решению проблем и улучшению бизнеса. Прилагаемые усилия и материальные средства должны помочь организации занять лидирующие позиции в области ее деятельности. Вы можете возразить, что обнаружение взлома только требует средств, и делать деньги на защите информации невозможно. Хотите поспорить? Обнаружение взлома позволило мне заработать очень, очень много денег, как и многим моим друзьям. Не лишайте себя заработка и не пропускайте материал этой главы. Учитесь писать бизнес-планы и выполнять анализ затрат и выгод. Полученные навыки окупятся в прямом смысле этого слова.

График выполнения проекта

Я создавал программы около 15 лет, а кроме того, я руководил группой квалифицированных программистов. Я стараюсь от них получить сведения о сроках выполнения проекта. В некоторых случаях я увеличиваю указанный срок в два или три раза. Все программисты думают, что достаточно добавить несколько строк кода и все будет в порядке, пока не сталкиваются с настоящими сложностями.

Мне иногда кажется, что руководители обладают каким-то шестым чувством, позволяющим им сразу определять нереальные сроки. Сейчас мы рассматриваем

предпоследний слайд вашей презентации или предпоследний раздел вашего отчета. Не хотелось бы допускать здесь решающих ошибок.

Для тех, кто не имеет опыта руководства проектами, приведем несколько примеров ситуаций с вымышленными коэффициентами, на которые нужно умножить первоначально установленные сроки.

- Розыск всего необходимого (2x).
- Компиляция и запуск всех бесплатных программ (2x).
- Получение разрешения руководства на определение нужной политики (5x).
- Установка программного обеспечения и его тестирование (2x).
- Введение в эксплуатацию датчика на коммутируемой сети (5x).
- Подключение консоли аналитика к сенсору через брандмауэр (3x).
- Получение разрешения на установку программного обеспечения системы обнаружения вторжений на отдельных компьютерах (5x).
- Проверка уязвимых мест по телефонным линиям (5x).
- Устранение обнаруженных уязвимых мест (5x).

Этот список носит немного шутливый характер, но есть в нем и доля истины. Если эти пункты присутствуют в вашем графике, то, возможно, их стоит проверить еще раз.

Следующие действия

К этому моменту сделано все для формирования готового решения. Определены и оценены как проблема, так и ее решение с возможными вариантами. Что могло быть упущено? Устранит ли предложенное решение все проблемы организации? Если нет, нужно определить несколько следующих действий. Если, например, рекомендуется установить сетевую систему обнаружения вторжений, то следующими действиями могут быть:

- защита по периметру сети критически важных систем;
- для проведения анализа тенденций создание базы данных, которая позволяет исследовать сведения, полученные от систем обнаружения вторжений, брандмауэров, маршрутизаторов и из файлов системных журналов;
- установка сетевых систем обнаружения вторжений для особо важных подсетей;
- повсеместная защита отдельных хостов.

Каждое из этих действий должно быть определено по срокам исполнения и требующимся затратам. Это означает, что вы представляете в целом весь проект, и ваш план действительно хорошо продуман. Кроме того, бюджет на капитальные вложения рассчитывается на несколько лет вперед, и разработчики бюджета смогут учесть ваши запросы.

Резюме

Надеюсь, что эта глава была вам полезна. Она написана для профессионалов в области защиты информации, у которых нет средства для обнаружения взлома, и

которые хотят усовершенствовать возможности защиты или повысить свою квалификацию. В этой главе мы показали, как общаться с руководством и как согласовать свои действия с коммерческой деятельностью организации.

Самое главное — не забывать (как и самому, так и при докладе руководству), что обнаружение взлома должно стать неотъемлемой частью общей стратегии защиты информации. Более того, обнаружение вторжений должно стать частью информационной безопасности каждого государства. Это доказали недавние события начиная с распределенных атак отказа в обслуживании, осуществленных с помощью IRC, и заканчивая запуском программ атаки с использованием возможностей SNMP и ASN.1. Для выявления распределенной атаки отказа в обслуживании не обязательно использовать систему обнаружения вторжений, но эта система позволяет найти скомпрометированные хосты до того, как они будут использованы хакерами. В последней главе нашей книги рассматриваются перспективные направления деятельности по обнаружению взлома.

20

Прогнозы и тенденции

Составление прогнозов — дело неблагодарное. Вы когда-нибудь читали точные прогнозы в газетных статьях? Как тут быть? Попытаемся отразить только самые основные вопросы: появляющиеся новые средства и тенденции в области обнаружения взлома. Меня приглашают рассказать о моих прогнозах на будущее по нескольку раз в год, поэтому я всегда стараюсь оставаться в курсе всех последних событий и слежу за развивающимися тенденциями.

Что касается этой главы, то мы обсудим новые угрозы, компьютерный терроризм, установку и запуск хакерами на чужом компьютере вредоносных программ, усовершенствованные методы проведения разведывательных действий и поиска цели атаки, многоуровневую систему защиты и глобальные методы обнаружения взлома. Завершит главу небольшой раздел об актуальных тенденциях в сфере информационной безопасности.

Возрастание угроз

Одна из причин постоянного интереса к проблемам обнаружения взлома заключается в постоянном росте числа угроз со стороны злоумышленников. Прогресс средств атаки за последний год поистине невероятен. Я не говорю о вирусах наподобие Code Red или Leaves и автоматических программах атаки на службу RPC, которые достигли значительного уровня сложности в середине 2001 года. Злоумышленники способны вывести из строя практически любой узел Internet. Скоординированные действия при сканировании позволяют проверить половину компьютеров сети класса B приблизительно за пять минут. При этом число сканирующих хостов достигает 2500 компьютеров, которые установлены в разных сетях. Сканирования могут быть и очень медленными, практически незаметными. При этом многие из злоумышленников днем работают на должностях специалистов по защите информации, что не может не вызывать беспокойства. Эти люди совершенно не планируют прекращать создание программ атаки.

Компьютерный терроризм

"Ты не заметил увеличения количества атак, чего-то необычного?" Не менее пяти моих друзей, которые работают на правительство, задали мне этот вопрос к полудню 11 сентября 2001 года. Тогда мы впервые услышали о компьютерном терроризме и после выхода директивы 13231 поняли, что правительство США готовит защитные меры против таких террористов. Лично мне не удалось найти никаких доказательств реальных атак со стороны террористов. Ходят анекдоты и разные слухи, но основными средствами террористов по-прежнему остаются бомбы и пистолеты. Является ли компьютерный терроризм реальной угрозой? В некотором смысле, да. Многие учреждения управляются компьютерами, а компьютеры уязвимы. Однако основным сдерживающим моментом для такого терроризма является то, что у злоумышленников отсутствуют необходимые знания и мотивация. Другими словами, террористы пока предпочитают бомбы переносным компьютерам.

Теперь давайте обсудим последствия крупных атак на компьютерные сети, проведенных в прошлом году. Одна из причин, по которой эти атаки не нанесли большого ущерба, заключалась в том, что многие из нарушителей, участвовавших в создании этих атак, были не слишком злобными.

Любопытная тенденция, которая продолжает оставаться справедливой с 1997 года, заключается в том, что основную часть этих атак составляют атаки отказа в обслуживании на IRC. Группы хакеров сражаются за контроль над комнатами чатов IRC и для этой цели создают средства отказа в обслуживании. Интересно, что многие из таких атак могут использоваться не только для атаки на службу IRC. Если бы группе террористов удалось захватить одну из сетей для использования в своих целях, они могли бы нанести значительный ущерб, особенно экономике. Каким будет финансовый ущерб, если десять крупнейших служб электронной коммерции в Internet выйдут из строя на неделю? Будет потерян доход, ослаблены позиции некоторых компаний и нанесен удар по фондовым биржам.

Основной мерой защиты против любого вида терроризма является разработка плана действий в чрезвычайной ситуации. После 11 сентября казалось, что все решили заняться созданием таких планов, но энтузиазм быстро прошел. Основная задача этого плана — обеспечить альтернативный способ ведения бизнеса, если обмен данными по компьютерным сетям будет сильно затруднен.

Компрометация многих компьютеров

Программы типа "троянский конь", почтовые бомбы и уязвимые места в программном обеспечении предоставляют хакерам широкое поле деятельности. Практически невозможно полностью защитить современную операционную систему. Одной из причин этого является их сложность. Посмотрите на список запущенных программ на вашем компьютере (команда `ps -ax` или `ps -ef` для UNIX-компьютеров и <Ctrl+Alt+Del> для компьютеров под управлением Windows). Как вы думаете, заметите ли вы что-нибудь при каких-либо изменениях в работе запущенных приложений? Это перечень приложений высокого уровня, а не вызовов функций или библиотек DLL. Если хакер сможет запустить вредоносную программу на вашем компьютере, то вам, вероятно, не удастся выявить ее без

специального антивирусного приложения. Как эти потайные ходы оказались в вашей системе?

В течение двух прошлых лет основными проблемами систем Windows были проблемы с Web-броузерами. Получили известность уязвимые места в программном коде Internet Explorer, которые позволяли хакерам запускать на атакуемом хосте произвольные программы при загрузке Web-броузером Web-страниц со специально подготовленными строками. Подобные атаки стали даже популярнее атак, основанных на использовании уязвимых мест Outlook, которые удерживали пальму первенства ранее. К несчастью, эти атаки направлены как на обычные персональные компьютеры, для которых используются коммутируемые соединения (здесь ущерб не так уж велик), так и на хосты правительственных учреждений, корпораций, учебных заведений, для которых используются линии соединения с высокой пропускной способностью. Запуск хакером любых программ на UNIX-системах позволяет осуществить многочисленные атаки на переполнение буфера. Кроме усовершенствования методов взлома, хакеры добились и серьезного прогресса в способах выявления подходящих целей атаки.

Улучшенный поиск цели атаки

В этой книге подробно рассмотрены различные методы проведения разведывательных действий. Составлением схем сетей в Internet занимаются целые организации. Среди усовершенствований выявления целей атаки можно выделить следующие.

- Усовершенствованные методы получения результатов при широковещательных запросах. Если на узле не блокируются входящие широковещательные пакеты, то хакер получает хорошие результаты сканирования с помощью небольшого количества подготовленных пакетов. В последнее время обычное сканирование выполняется крайне медленно, поэтому сначала стараются использовать программу nmap или другое подобное средство для получения эхо-ответа. Возвращение этого ответа расценивается хакерами как достаточный повод для сканирования открытых портов компьютера и используемых протоколов.
- Хакеры стали исключать из сканируемых целей опасные для себя диапазоны IP-адресов, определяемые ими на основе списков ловушек и узлов, которые активно выявляют атаки и уведомляют о них группу CIRT.
- Обмен полученной разведывательной информацией между нарушителями. Когда две группы хакеров используют различные методы разведки и предоставляют друг другу полученные сведения, выявить подобные действия значительно сложнее.

Поскольку разведка чужих сетей выполняется уже давно, в настоящее время очевидны результаты долгой работы хакеров. Выявление нескольких пакетов сканирования может означать, что хакер просто проверяет имеющуюся схему сети. При обнаружении новых уязвимых мест он будет иметь под рукой готовые цели атаки.

Причины роста угроз

Тот факт, что системы являются уязвимыми, и хакеры улучшают методы поиска уязвимых систем, ни для кого не будет открытием. Что изменилось, так это масштаб атак. Проведенные в начале 2002 года успешные атаки с помощью Leaves, SubSeven, вируса Code Red против IIS-серверов, атаки по протоколам SNMP и ASN.1 и PHP-атаки против серверов Apache позволили хакерам завладеть (и даже управлять) целыми сетями с тысячами и тысячами скомпрометированных систем.

Такие события имели несколько последствий. Например, некоторые теперь опасаются атак со стороны любого компьютера в сети Internet. К февралю 2003 года, спустя три года с момента проведения масштабных распределенных атак отказа в обслуживании против Web-сайтов CNN и Yahoo, злоумышленники продолжают атаковать не менее успешно, чем защищаются провайдеры услуг Internet. Когда институт SANS в феврале 2002 года с помощью технологии WebCast проводил по Internet презентацию нового средства безопасности для настройки маршрутизаторов Cisco, компания Digital Island, один из провайдеров, транслирующих эту презентацию, сообщил об успешной атаке отказа в обслуживании, которая позволила хакерам блокировать работу WebCast-трансляций. В марте 2002 года были проведены пробные атаки блокирования работы сайтов, при которых использовалась программа `traceroute`, с ее помощью определялся маршрутизатор, используемый сайтом для подключения к Internet, а затем блокировался этот маршрутизатор. Кроме того, хакеры начали экспериментировать с TCP-портом 179 (служба BGP). Конечно, я не знаю будущего, но могу предположить следующие варианты развития событий.

- Хакеры не прекратят своих попыток. Многие из них весьма корыстны и постараются получить деньги посредством вымогательства при блокировании работы Web-сайтов служб электронной коммерции, таких как eBay или Amazon. Другие хакеры действуют ради интереса и могут постараться нарушить работу двух самых популярных служб в Internet: маршрутизации и службы DNS, а если будет заблокирована маршрутизация, то DNS отключится сама собой. Наши лучшие аналитики считают, что хакерам не удастся заблокировать работу всей сети Internet, так как она состоит из многих независимых подсетей.
- Правительство делает только то, что способно сделать: установлена серьезная криминальная ответственность за проведение подобных атак.

Нельзя сказать, что все так уж плохо. Угрозы могут возрасти еще больше, но есть и определенный предел этому росту.

Защита от угрозы

Существуют объективные факторы и меры противодействия, которые ограничивают рост угроз со стороны хакеров. Мы сначала рассмотрим сдерживающие факторы, а затем перейдем к навыкам и средствам, используемым специалистами по информационной безопасности. Кроме того, пользователи Internet быстро учатся и сами реализуют методы защиты. Мы также рассмотрим обнаружение взлома с общей точки зрения, чтобы продемонстрировать наиболее важные тенденции. Каковы же ограничения, сдерживающие хакеров?

- Для современных средств, предназначенных для проведения распределенных атак отказа в обслуживании, наподобие Leaves или litmus команды передаются с помощью службы IRC. В этом их сила и слабость. С одной стороны, люди становятся разумней и все чаще блокируют такой трафик. Хакеры предпринимают ответные действия, но все равно присутствует сдерживающий фактор.
- Большое количество попыток сканирования основано на использовании открытых IP-адресов. С переходом каждой новой организации на технологию NAT (Network Address Translation –трансляция сетевых адресов) и использование частных адресов задача хакеров становится все труднее.
- Некоторые компьютеры были скомпрометированы в результате использования вирусных атак Leaves, Code Red и атак по протоколам SNMP и ASN.1. Многие из взломанных систем являлись системами Windows и тому есть веские доказательства. Я думаю, компанию Microsoft заинтересовало, что за несколько месяцев после появления вируса Code Red 180000 Web-серверов IIS были преобразованы (в большинстве случаев) в серверы Apache.
- Следите за тем, куда вкладываются деньги! В первую очередь, они направляются на защиту. Хакеры проявляют чудеса изобретательности, но наличие на рынке бесплатных, простых в настройке средств безопасности, а также проводимое обучение принципам защиты позволяет сделать компьютеры не такими доступными. Если средства защиты будут применяться повсеместно, то атаки перестанут быть увлекательным развлечением, и будет проще выявить узлы, которые не контролируют действия своих пользователей.

Вопрос финансирования имеет большое значение. Кажется логичным, что если вкладывать деньги в безопасность, то обязательно будет какой-то результат. Однако не все так просто. В отчете Министерства финансов США от 25 мая 2002 года (`http://www.whitehouse.gov/omb/inforeg/fy01securityactreport.pdf`) были приведены вполне прогнозируемые недостатки в области защиты информации:

- недостаточное внимание к безопасности со стороны руководителей;
- неэффективное обучение принципам защиты информации;
- неправильное с точки зрения безопасности взаимодействие с поставщиками;
- недостаточное выявление уязвимых мест.

Но самым интересным моментом этого отчета является вывод о том, что не было обнаружено явной зависимости между количеством денег, вложенных в защиту информации, и конечным результатом. Более того, они даже не учитывали важность наличия хороших средств обнаружения вторжений. Я считаю, что в ближайшем будущем будут иметь преимущество те, кто сможет разумно потратить деньги на обучение персонала и покупку нужных средств обеспечения защиты.

Знания против программ

Интерес к теме обнаружения взлома продолжает расти. Институт SANS объявил набор на первые курсы по обучению методам обнаружения взлома в марте

2000 года, и они заинтересовали только технических специалистов, которые хотели получить дополнительные знания в области социальной инженерии. В настоящее время подобные курсы читаются по шесть дней в неделю по всему миру. Спрос рождает предложение. Люди стремятся научиться искусству выявления взлома. Будущие аналитики сетевого трафика изучают все темы, изложенные в данной книге: побитовая маскировка, основные навыки аналитика, правила создания фильтров и специальные навыки, необходимые конкретному аналитику.

В то же время компании работают над созданием все более эффективных средств. Надеемся, что к этому моменту нашим читателям стало очевидно, что невозможно создать систему обнаружения вторжений, которая бы работала без помощи квалифицированного специалиста.

Навыки аналитиков

В системах обнаружения вторжений присутствует та же проблема, что и в антивирусном программном обеспечении: новые атаки нельзя обнаружить из-за отсутствия сигнатур для их выявления. Проблема даже серьезней. Ведь для сетевых систем обнаружения вторжений создано только небольшое количество сигнатур, до 2000, что очень мало по сравнению с 30000 сигнатур для антивирусных программ. Есть естественные ограничения для сетевых систем обнаружения вторжений на основе сигнатур. Для эффективного использования этих систем я рекомендую дополнительно регистрировать весь проходящий трафик. Это позволит после поднятой тревоги исследовать причины, которые вызвали ее появление. Я также стараюсь сохранять данные в двоичном формате по крайней мере за несколько дней. Поэтому, если мне повезет и удастся обнаружить что-либо необычное, то я смогу просмотреть данные предшествующих пакетов. Современные аналитики обязаны уметь поставить фильтры для поиска интересующей информации в сохраненных данных. В будущем возможности подобного поиска будут интегрированы в системы обнаружения вторжений и даже в реляционные базы данных, так что аналитик сможет описать свой запрос на SQL.

В компаниях понимают необходимость использования квалифицированных работников. Даже в момент экономического спада 2001 года институт SANS выпускал класс за классом, и большинство из них были полностью укомплектованы. Нанимая сотрудников, компании все чаще выдвигают требования наличия сертификатов. Вначале это было похвально, но несколько тоскливо: "Требуется аналитик системы обнаружения взлома, который умеет писать правила, интерпретировать данные в шестнадцатеричном формате и имеет сертификат CISSP". Не воспринимайте это как критику сертификата CISSP (Certified Information System Security Professions), но сам по себе сертификат не означает, что человек с таким сертификатом может работать с системой обнаружения вторжений или умеет настраивать брандмауэр, или способен решить другую техническую задачу. Компании быстро осознали этот факт и согласно последним исследованиям наибольшей популярностью у работодателей пользуются специалисты с сертификатом GIAC. Программные средства постоянно совершенствуются, но по крайней мере в следующие несколько лет (и, как я надеюсь, навсегда) ничто не заменит навыков квалифицированного аналитика.

Быстрый рост количества персональных брандмауэров уже привел к тому, что они стали основным средством защиты данных. Диапазон этих средств весьма велик (начиная от брандмауэра Internet Security компании Symantec, в котором объединяются антивирусные возможности с простейшими возможностями защиты и выявления взлома, и заканчивая брандмауэром BlackIce, который позволяет регистрировать проходящие пакеты). Создатели этих средств решили одну из важнейших проблем, организовав защиту на конкретном хосте! Сознательные сотрудники устанавливают их и на рабочих местах, и дома. Более того, эти сотрудники могут сообщать о выявленных атаках, что будет весьма полезно. Существуют даже автоматические средства наподобие Dshield, которые позволяют собирать данные от нескольких персональных брандмауэров и анализировать общие тенденции. Сетевые системы обнаружения вторжений по-прежнему находят широкое применение. Намного проще установить в сети пару машин, чем добавлять непроизводительный, ресурсоемкий программный уровень на каждом из хостов локальной сети. Если разобраться в эффективности всего процесса обнаружения взлома, то персональные брандмауэры на компьютерах сотрудников обеспечивают огромные преимущества и очень удачно дополняют сетевые средства защиты. Поэтому нет ничего удивительного в том, что наступает время консольных систем, взаимодействующих с базами данных, когда информация системы обнаружения вторжений дополняется данными брандмауэров, персональных брандмауэров, антивирусных программ, а также другими данными, например из системных журналов. Это позволяет лучше, чем когда-либо ранее, понять, что же происходит в конкретной сети.

Улучшенные средства

Создано много различных видов консолей аналитиков сетевого трафика. Некоторые из них обладают усовершенствованными возможностями просмотра записей журналов, например Big Brother (`www.bb4.com`) или Brother NetIQ (`www.netiq.com`). Другие позволяют анализировать содержимое проходящего трафика, например SilentRunner (`www.silentrunner.com`), или являются средствами, учитывающими сведения из различных источников информации, что можно сказать про netForensics (`www.netforensics.com`), ISS SiteProtector (`www.iss.net`) и Intellitactics NSM (`www.intellitactics.com`). И это только вершина айсберга. Я знаю много компаний, которые разрабатывают собственные продукты, как, например, консоль Sourcefire OpenSnort (`www.sourcefire.com`), в которой используется высокопроизводительное средство для работы с базами данных `barnyard`, созданное Эндрю Бэйкером (Andrew Baker). Когда все эти компании выпустят свои средства и пройдет время их обсуждения и проверки, мы получим еще несколько очень полезных программ.

Компании покупают средства защиты информации, но их качество не всегда соответствует стоимости. Мы уже рассказывали про коммерческие системы обнаружения вторжений, которые приходится заменять через год после приобретения. Возможно, следующие версии этих программ будут лучше, но специалисты должны рекомендовать приобретать только те средства, которые действительно работают.

Соревнование на рынке средств обнаружения взлома протекает весьма увлекательно. Не нужно быть опытным аналитиком, чтобы понять, что Рон Гула (Ron

Gula) с его средством Dragon, Роберт Грехэм (Robert Graham) с его BlackIce (теперь RealSecure) и Марти Роч (Marty Roesch) со своим Snort не просто выдающиеся программисты, они вложили душу и сердце в свои продукты. Нельзя не упомянуть и о Кэвине Зейсе (Kevin Zeise) из команды сотрудников Cisco. Возможно, его деятельность не столь заметна, но он пробегает четыре мили с утра, съедает два куска лимонного торта на завтрак, разрабатывает новую линию товаров к обеду и перед сном успевает спасти планету от катастрофы в киберпространстве. Он использует все перечисленные системы обнаружения вторжений. Дискуссии в списках рассылки и виртуальных конференциях тоже служат общей цели определения лучшего средства. В мире лжи, где все поставлено на продажу, эти три человека потратили все свои усилия на создание оптимального средства защиты. Кто из них победит? Я думаю, победа будет не за отдельной личностью, а за командой. Поэтому в духе пророчества я могу сказать следующее.

- У компании Enterasys в данное время есть небольшие финансовые затруднения, в частности проблемы с Комиссией по ценным бумагам (SEC). Мне нравится их система Dragon и некоторые средства для работы в сети, но я бы не стал приобретать больше ста акций этой компании (слишком вероятно, что их можно будет использовать как обои). Поэтому, я считаю, что система Dragon могла бы стать достойным конкурентом на рынке, если бы не финансовые недоразумения.
- В начале 2002 года компания ISS и Роберт Грехэм были явными фаворитами на рынке средств защиты информации. Отличная руководящая группа, прекрасный маркетинг обеспечили репутацию этой компании. Мне очень многое нравилось в системе BlackIce, и многие из этих возможностей были встроены в новую версию под названием RealSecure. Я не сомневаюсь, что Роберт скоро создаст нужный датчик и завершит создание этого средства. Вопрос только в том, удастся ли его интегрировать с основной консолью аналитика. На момент написания этих строк я не успел оценить по достоинству программу SiteProtector, но, судя по заявлениям ISS, они рассчитывают с помощью этого средства прорваться на рынок управляемых служб. Я предполагаю, что здесь основным аргументом будут знания, а не сама программа. Если консоль позволит опытному специалисту проявить все свои лучшие качества, то ISS, пожалуй, обойдет любую компанию, кроме Cisco. Если же консоль будет создана на основе принципа: "Сидите и смотрите. Если появится красный треугольник, звоните на фирму", то я думаю, что все шансы на рынке будут потеряны.
- Компания Cisco еще много лет назад разработала стратегию, согласно которой обнаружение взлома должно осуществляться на уровне локальной сети. Объединение возможностей Catalyst 6000 и Policy Feature Card привело к созданию системы названием TopLayer, позволяющей выполнять анализ при обнаружении взлома. Она стоит недешево. Тем не менее на самых крупных узлах с самым дорогим оборудованием используют продукты компании Cisco, а это говорит само за себя.
- Возглавляемая Марти Рочем компания Sourcefire получила два миллиона долларов на развитие своего бизнеса. Система Snort является популярней-

шим средством выявления взлома, а набор правил Snort используется чаще всего. Здесь все ясно. Но все это касается бесплатной программы Snort, а я наблюдал, как мой друг Джин Ким (Gene Kim) пытался превратить в источник дохода свою бесплатную программу Tripwire, и понял, что это не такая простая задача. Более того, Марти не единственный, кто хочет перевести систему Snort на коммерческие рейки. По моему мнению, он очень удачно выбрал время для предложения товара на рынке. В тот момент, когда все труднее и труднее найти действительно качественный товар, когда компании ISS и Enterasys находятся на спаде, есть прекрасная возможность для усиления позиций Sourcefire.

Как мне кажется, когда наши читатели будут держать в руках эту книгу, наиболее мощными компаниями в области обнаружения взлома будут Cisco и Sourcefire, ISS сохранит свои позиции, а в Enterasys, скорее всего, продолжится спад. Будет ли выпущена программа Tippingpoint — новое универсальное средство обеспечения информационной безопасности? Вероятно, нет. Скорее всего, пройдет еще год или два до того, как пользователи будут готовы использовать брандмауэр, интегрированный с системой обнаружения вторжений. Нельзя отрицать тот факт, что соревнование на рынке приводит к появлению большого количества новшеств, от чего пользователи только выигрывают. Одна из причин, по которым я так подробно рассказал о новом поколении консолей, заключается в том, что они являются базовыми средствами аналитика, предоставляющими сведения о ситуации и позволяющими организовать активную эшелонированную защиту.

Эшелонированная защита

Военная история учит, что никогда не следует полагаться на одну линию обороны или на один метод оборонительных действий. Мы тоже указываем на недопустимость полного доверия только системе обнаружения вторжений. Если фильтры не "срабатывают", то необходимо определить, почему это случилось, и какие события в сети предшествовали этому. Необходимо уметь самостоятельно декодировать подозрительные пакеты. Это только небольшой пример эшелонированной защиты.

Брандмауэр служит как эффективное средство для удаления лишних пакетов и блокирования многих атак на входе в локальную сеть. В пределах локальной сети маршрутизатор (или коммутатор) может быть настроен на поиск признаков взлома. При выявлении значимого события коммутатор может или заблокировать сеанс с хостом злоумышленника, или отправить уведомление об опасности. Эту модель можно усовершенствовать, добавив защиту на уровне хоста. Таким образом, можно будет выявить попытки незаконного доступа к чужим файлам локальных пользователей, обладающих легитимными именами пользователей (своими или других пользователей). Добавив еще несколько сетевых систем обнаружения вторжений, включая и несколько скрытых, мы получим архитектуру, достаточную для противодействия возрастающим угрозам. К сожалению, такая архитектура практически неосуществима в реальной жизни. Так как же возможно сегодня и в ближайшем будущем организовать эшелонированную защиту?

В наше время следует применять пять следующих правил организации защиты по периметру локальной сети. При этом не стоит думать, что защита должна быть

только на линии подключения к Internet и больше нигде. Все эти пять правил раскрываются на страницах глав и приложений этой книги. Так как это последняя глава, попытаемся сделать некоторые выводы.

- **Блокировать все исходящие ICMP-сообщения о недостижимости.** Можно также блокировать другие исходящие ICMP-сообщения об ошибке. Это позволит защитить свой узел от проведения разведывательных действий.
- **Ограничить базу данных DNS-сервера.** Это правило можно сформулировать и по-другому, но сама идея очень проста. Доступный по Internet DNS-сервер должен иметь сведения только о нескольких хостах локальной сети, включая сервер электронной почты, Web-сервер и так далее. В противном случае DNS-сервер может быть использован хакером для получения разведывательной информации.
- **Использовать максимальное количество proxy-систем.** Не следует ограничиваться proxy-серверами на брандмауэрах, их следует добавлять также между точкой подключения к Internet и устройствами фильтрации входящего трафика.
- **Использовать трансляцию сетевых адресов (NAT).** Если появится возможность отказаться в локальной сети от общедоступных IP-адресов и перейти на частные адреса, то качество защиты от атак повысится на порядок.
- **Реализовать автоматические ответные действия.** Этот метод уже много лет применяется для блокирования рекламы, поступающей по электронной почте. Активные методы защиты, используемые брандмауэром Raptor и BackOfficer Friendly, не привели к серьезным проблемам в Internet. Всегда найдется возможность для использования автоматического ответа, и лучше быть максимально защищенным.

Разумеется, эшелонированная защита не должна ограничиваться периметром сети. Она включает в себя: работы по администрированию компьютеров, применение персональных брандмауэров, антивирусных приложений, сканирование содержимого входящих и исходящих пакетов, установку заплат для операционных систем и использование программ поиска уязвимых мест.

Масштабное выявление попыток взлома

Одним из самых интересных событий 2001 года в области защиты информации стало проведение трех исследований по масштабному обнаружению взлома: Aris (SecurityFocus.com), MyNetWatchman (`www.mynetwatchman.com`) и Dshield (`www.dshield.com`). В каждом их этих проектов сотням или тысячам клиентов предоставлялось программное обеспечение для создания отчетов об атаках. В качестве клиентов использовались различные устройства от брандмауэров Check Point и маршрутизаторов кабельных сетей Linksys до персональных брандмауэров. Все зарегистрированные данные отправлялись на центральный узел для анализа и выявления общих тенденций атак.

Набор этих данных, собранных со всего мира, позволяет сделать очень важные выводы. Например, с помощью Dshield сохранялось около шести миллионов новых записей в неделю. Несмотря на то, что в первый год использования Dshield были серьезные проблемы с обработкой полученных данных, эта технология по-

логия позволила обнаружить появление червей Ramen, Lion и Leaves. На рис. 20.1 можно проследить, как увеличилось число проводимых сканирований широко распространенных уязвимых мест в работе протоколов SNMP и ASN.1 после выхода рекомендации CERT от 12 февраля 2002 года.

Рис. 20.1. Статистические данные, собранные с помощью Dshield

Это новые проекты, и сообщество пользователей еще только пытается найти наилучшее применение этим средствам. В распределенных системах обнаружения вторжений использована настолько простая и в то же время эффективная схема, что многие специалисты, включая и автора этих строк, не понимают, почему этого не делалось раньше. Возможно, с наступлением нового века в умах людей что-то изменилось, и теперь они хотят совместно использовать накопленную информацию.

Обмен информацией

Я спросил сотрудников Incidents.org, хочет ли кто-нибудь написать дополнения к этой книге. Автором следующего дополнения является Ричард Бэйтлих (Richard Bejtlich) — опытный аналитик. Обратите особое внимание на четвертый вопрос.

При оптимизации своей сетевой системы обнаружения вторжений я задаю четыре вопроса.

- Что может вызвать появление подозрительного трафика?
- Какие события может пропустить моя система обнаружения вторжений?
- Чем реальные события в Internet отличаются от теоретического описания в учебниках?
- Должен ли я рассказывать о выявленных событиях другим специалистам по защите информации?

В первом вопросе предполагается, что кроме обычных пакетов могут появляться пакеты с подложными адресами, специально подготовленные пакеты или незапрошенные ответные пакеты. Второй вопрос требует от меня вспомнить ограничения возможностей моей системы обнаружения вторжений и не забывать, что я не могу правильно интерпретировать или просто перехватить абсолютно все проходящие пакеты. Третий вопрос означает, что не всегда трафик, который не соответствует требованиям RFC и технических руководств, является вредоносным. Последний вопрос стимулирует аналитиков сетевого трафика делиться накопленными знаниями друг с другом то ли посредством сайта `www.sans.org`, то ли на форумах, подобных `securityfocus.com`.

Сети компьютеров с возможностями обнаружения взлома играют ключевую роль при организации ответных действий. Кроме сбора данных и отправки сохраненной информации для централизованного анализа, может быть организована автоматическая отправка отчетов о нарушениях лицам, ответственным за корректное использование IP-адресов. Например, 28 февраля 2002 года было обнаружено, что с хоста с IP-адресом `217.128.207.17` из домена `abo.wanadoo.fr` отправлено 33995 пакетов. Это послужило основанием к отправке уведомления администраторам этого домена, хотя такое уведомление о прекращении сканирования по службе FTP напоминает обращение к Бен Ладену с просьбой прекратить террористические акты – в лучшем случае они просто не реагируют. Однако многие люди более порядочные, и уведомление от системы Dshield может послужить первым сигналом системному администратору о возникшей проблеме. Ниже приводится пример отзыва от другого благодарного пользователя.

"Благодарю за уведомление о незаконных операциях, осуществляемых с компьютера нашего университета. При помощи программы поиска вирусов `eSafe` мы обнаружили на этом компьютере опасный вирус. Еще раз спасибо за предупреждение. Если вдруг повторится нечто подобное, сообщите нам, пожалуйста."

Я немного сомневаюсь по поводу указанных выше причин проводимых атак, но даже если уведомление позволило выявить недобросовестного пользователя, то это еще одна победа новой системы защиты. К настоящему времени модель развития бизнеса создана только для системы Aris, поэтому не ясно, будет ли долговечной первая реализация распределенного выявления взлома. Я искренне надеюсь, что это не случайность, а тенденция. Завершить эту книгу я бы хотел несколькими разделами, посвященными антивирусным программам, обсуждению систем обнаружения вторжений на основе аппаратных или программных средств и, наконец, некоторым изменениям в проведении контрольных проверок системы безопасности.

Новые технологии защиты

Возможности современных систем обнаружения взлома довольно ограничены. Они не годятся для выявления атак со стороны локальных пользователей, атак с помощью изменяющегося кода, интеллектуальных вирусов для сбора информации, атак с помощью модемов. Системы обнаружения вторжений для защиты хоста способны выявлять эти атаки, но у них есть два больших недостатка: стоимость реализации и непроизводительные расходы ресурсов системы. Компания, которая сможет создать более эффективное средства, заработает много денег. Рассмотренные нами консоли системных аналитиков для обеспечения защиты на уровне корпорации предназначены для получения части этих прибылей. На сегодняшний день наиболее развит сектор рынка, занимающийся продажей антивирусных средств.

Еще раз об антивирусных программах

Я с изумлением наблюдал, как две компании NAI и Symantec, которые смогли точно найти баланс между нарастающими угрозами и ответной реакцией, не смогли получить полный контроль над сектором рынка средств по обнаружению взлома. Если

производители программного обеспечения для защиты от вирусов еще не приступили к созданию специализированных средств для выявления взлома, они, по крайней мере, уже частично присутствуют и в этом секторе рынка. Для всех троянских программ Subseven и NetBus могут быть созданы сигнатуры. Антивирусные приложения позволяют выявить и удалить подобные программы. Но не все так просто. Впервые я понял, что антивирусную программу можно "обойти", когда один из моих студентов случайно загрузил троянскую программу (он увидел запуск команды `notepad.exe` после щелчка на загруженной Web-странице). После небольшого исследования оказалось, что была загружена троянская программа QAZ. Тем не менее антивирусная программа не выявила ничего подозрительного. Но как это возможно по отношению к хорошо известной троянской программе? Дело в том, что хакеры могут упаковывать троянские программы в другие средства. Более подробную информацию по этой теме можно найти по адресу `http://rr.sans.org/malicious/trojan_war.php`.

Тем не менее антивирусные программы нельзя сбрасывать со счетов. Они позволяют выявить пакеты наиболее популярных программ атаки и, конечно, (при использовании совместно с персональным брандмауэром) активные троянские программы, запущенные в сети. Все это будет справедливо, если программа атаки одним из своих первых действий не отключит брандмауэр или антивирусное приложение, но производители программного обеспечения работают и над этой проблемой.

Средство Internet Security компании Symantec объединяет в себе антивирусное приложение и персональный брандмауэр, но действует уж слишком "грубо". Компании, занимающиеся разработкой антивирусных средств, могут иметь успех в области обнаружения взлома по следующим причинам:

- ни одна из программ защиты не может обеспечить такого глубокого уровня анализа работы компьютера, как антивирусное приложение;
- средствами обнаружения взлома обычно используются до 500 сигнатур, а для антивирусных программ их создано более 20000;
- антивирусные программы входят в комплект программного пакета брандмауэров, операционных систем серверов или настольных компьютеров;
- антивирусное программное обеспечение позволяет выполнять обнаружение, блокирование и устранение вредоносных программ, а также восстановление после взлома при минимальном участии пользователя;
- компании-производители антивирусных средств полностью решили задачу обновления пользовательских сигнатур наиболее простыми способами;
- многие крупные организации приобрели лицензии на использование программного обеспечения этих компаний;
- производители антивирусных средств готовы к быстрому изменению пользовательских сигнатур при обнаружении новой программы атаки;
- часто совместно с антивирусными приложениями поставляются другие продукты с функциями обеспечения защиты.

Аналогия настолько очевидна, что я не понимаю, почему антивирусные средства не доминируют на рынке программного обеспечения для обнаружения взлома. Требуется внести очень небольшие изменения в персональные брандмауэры компаний NAI или Symantec, чтобы превратить их в сетевые системы обнаружения

вторжений, но вместо этого с выходом каждой новой версии эти средства все больше отдаляются от средств корпоративной защиты.

Теперь рассмотрим аппаратные средства для обнаружения взлома. Компания Cisco больше других занималась внедрением средств обнаружения вторжений в сетевое оборудование, благодаря чему и было создано несколько соответствующих аппаратных решений.

Обнаружение взлома с помощью аппаратных средств

При использовании сетевых систем обнаружения вторжений приходится решать три серьезные проблемы:

- наличие зашифрованных пакетов, в которых невозможно выявить заданные строки символов;
- пропускная способность сети выше пропускной способности датчика;
- сложность анализа трафика в коммутируемых сетях.

Обнаружение взлома в коммутируемых сетях рассмотрим чуть позже. Более серьезной проблемой является шифрование. Надежное шифрование при сохранении в безопасности ключа — очень хороший метод защиты данных. Но шифрование превращается в проблему, когда его используют хакеры для обмана системы обнаружения вторжений. Как узнать, что поток данных зашифрован? Конечно, проверив наличие случайных повторений. Сделать это довольно просто, но накладно с точки зрения использования центрального процессора. Это аргумент в пользу осуществления такой проверки с помощью специальных аппаратных средств. Я не уверен в правильности такого подхода, современные компьютеры очень быстро становятся все более производительными. Но есть и области, в которых применение специальных аппаратных средств имеет смысл, например в высокопроизводительных сетях.

Идеальным местом для размещения системы SecureIDS компании Cisco является сетевой адаптер, установленный в маршрутизаторе или коммутаторе Cisco. Однако по-настоящему хорошие результаты достигаются, когда часть атак выявляется посредством программного процесса в машрутизаторе или коммутаторе. Такой подход является насущной необходимостью, а его основное преимущество заключается в достижении скорости обработки пакетов в реальном времени. Во всех других случаях (за исключением обнаружения взлома на брандмауэре) обнаружение взлома происходит непосредственно после прохождения пакетов. В данном случае можно остановить вредоносный пакет или направить его на хост-ловушку. Такая возможность пригодится при использовании модуля Policy Feature Card, доступного для коммутаторов Catalyst 6000. Я не уверен в том, что нужно было создавать отдельную плату. Вероятно, это сделано для того, чтобы этот продукт мог соперничать с TopLayer — коммутатором уровня приложений, который используется аналитиками сетевого трафика, когда им требуется высокая скорость обработки пакетов. Возможно, этот проект имеет преимущества на рынке продуктов, ориентированных на качество обслуживания. Однако возможность фильтрации и маркировки пакетов с помощью коммутатора уровня приложений внутри сетевого устройства при интенсивности получения 5 миллионов пакетов в секунду на уровне 3 открывает широкие горизонты обнаружения атак и организа-

ции защиты. Среди преимуществ такого решения можно отметить возможности автоматических ответных действий, например, обрыва соединения, ограничения интенсивности проходящего трафика, копирования пакетов в более мощную систему обнаружения взлома или перевода соединения на хост-ловушку. Как и для распределенных систем обнаружения вторжений, требуется некоторое время, чтобы оценить возможности подобных средств, но их изучение должно стать очень интересным.

Программное обнаружение взлома

Я просто не могу привыкнуть к размерам современных программ. Когда-то у меня был компьютер Commodore 64. Цифра 64 означала объем оперативной памяти – 64 Кбайт. Предполагалось, что этого будет достаточно для запуска и работы программ. Интересно сравнить Commodore 64 с моим Pentium II с частотой 400 МГц и объемом оперативной памяти в 1024 Мбайт. Те приложения, которые использовались на Commodore 64, выполняли примерно те же функции, что и приложения пакета Microsoft Office. Но современные программы просто огромны! Если размер программ и дальше будет расти (а так оно и будет), пользователи вправе требовать обеспечения некоторого уровня защиты.

На второй конференции по проблемам обнаружения взлома SANS в 1999 году мне повезло познакомиться с Симсоном Гарфинкелем (Simson Garfinkle), программистом специализированных средств обеспечения защиты. Многие приложения обеспечения безопасности, особенно программы сканирования уязвимых мест, могут быть использованы для незаконных действий. Симсон хочет защитить свою интеллектуальную собственность от пиратов, а также гарантировать невозможность некорректного использования своих программ, не раскрывая при этом исходного кода.

Можно ли предотвратить копирование или незаконное использование приложения? В одно время это имело огромное значение для производителей компьютерных игр, по крайней мере, в отношении защиты от незаконного копирования. Сегодня эта проблема не выглядит такой актуальной. Ни одна из игр, купленных моим сыном, не требовала защитной заглушки. Такая аппаратная заглушка применяется для защиты программного кода одного из используемых мною средств, Expert Witness. Компания Microsoft использует свою стратегию защиты программ, которая заключается в наклейке странных оранжевых полосок на ее фирменные компакт-диски, в длинных серийных номерах, в методе перезвона домой и инспектировании локальных сетей на предмет наличия лицензий для своих продуктов. Симсон подошел к этому вопросу серьезнее, чем любая из компаний. Он предложил целый набор мер противодействия незаконному копированию, включая шифрование частей программ и добавление контрольных сумм.

Может ли критически важная программа сама обнаружить попытку проведения атаки? Предположим, что служба `sendmail` или пакет BIND имеют статическую библиотеку с функциями защиты. Программа могла бы выявлять несанкционированное использование, попытку доступа или введения данных в виде двоичного кода. Это могло бы заблокировать атаку и поднять тревогу. Для программ могут быть даже созданы специальные "профили" их использования таким образом, чтобы выявлялись все попытки несанкционированного доступа к фай-

лам программы и осуществлялось предопределенное ответное действие. Еще один способ обеспечения безопасности на программном уровне заключается в применении упаковщиков, что является наиболее очевидной тенденцией.

Первым упаковщиком [4] была программа Витса Винима TCP Wrapper, которая оставалась отличным средством безопасности на протяжении нескольких лет (хотя в наше время лучше упомянуть о программе `xinetd`). Но эта идея была расширена и дополнена. Более подробную информацию по этому вопросу можно получить на сайте `www.immunix.org`. Я считаю, что в скором времени все службы с доступом в Internet придут к использованию надежного метода упаковки, работы в замкнутом пространстве `chroot` или одновременному применению и того, и другого.

Сообразительные аудиторы

В первом издании этой книги я утверждал, что аудиторы станут гораздо умнее, но этого не произошло. Я повторюсь и во втором издании, так как прогресс все-таки есть, и я надеюсь, что это актуальная тенденция! Как говорил Алан Кэя (Alan Kay), лучший способ спрогнозировать будущее — организовать его. Когда вы будете держать в руках эту книгу, институт SANS должен внести свой вклад и создать средства и ресурсы для аудиторов. Аудиторы уже достаточно мудры, поэтому они занимаются аудитом в то время, пока вы сидите и потеете. Они значительно лучше стали разбираться в технологиях обеспечения безопасности. Прошли те времена, когда после утвердительного ответа на вопрос о наличии брандмауэра они разворачивались и уходили прочь.

По моему мнению, основной тенденцией в области аудита является обучение аудиторов, которые бы могли оценивать уровень защиты сетей с помощью специальных программных средств. Аудиторы могут посетить вашу локальную сеть с доступом в Internet и запустить средство оценки во время проведения интервью. Потом они могут сравнить ваши ответы с результатами, полученными с помощью своей программы.

Хотя системным администраторам это может и не понравиться, но осведомленные, хорошо подготовленные аудиторы могут стать одной из наиболее эффективных мер противодействия нарастающим угрозам. Хакеры, злоумышленники, являющиеся пользователями локальной сети, и создатели программ атаки, не настолько мудры. Просто мы ленивы, беспечны и наивны. Когда мы допускаем ошибку или небрежность, в нашей системе обороны остается открытая брешь, которой и пользуются злоумышленники. Если же мы будем кому-то подотчетны, то будем стараться сделать все как следует на благо своей организации.

Резюме

Все указывает на то, что аналитиков сетевого трафика ждет хорошее будущее. Перед нами огромный объем работы, которая должна быть хорошо оплачена. Хорошие аналитики просто нарасхват, и вряд ли что-то изменится в ближайшем буду-

[4] Программы-упаковщики позволяют системным администраторам контролировать процесс доступа программ к службам (серверам) того или иного хоста. Упаковщик перехватывает вызов и проверяет, разрешено ли подключение данному хосту.

щем. Руководство компаний начинает понимать, что знания очень важны и они требуют сертификатов GCIA или просят продемонстрировать навыки при приеме на высокооплачиваемые должности. Для противодействия угрозам создаются специальные средства, разрабатываются новые методы и проводятся курсы обучения.

Спасибо всем читателям этой книги. Мне было приятно работать в одной команде с Джуди и Марти над этим обновленным изданием и я благодарен им за понимание. У нас действительно получилась настольная книга аналитика сетевого трафика.

Еще одно последнее замечание. Информация сайта `www.incidents.org` зависит только от сообщества неравнодушных людей. Покуда есть еще энтузиасты, чтобы быть в курсе всех событий, участвуйте в форумах этого сайта, высказывайте свои мнения по тому или иному вопросу и делитесь накопленными знаниями. Это позволит оставаться в курсе всех событий Пожалуйста, будьте активны. Мы рады участию людей любой нации, любым мнениям и сообщениям о выявленных атаках с помощью любого программного обеспечения. Дисциплина обнаружения втожений находится в периоде становления и требует усовершенствования. Для этого нужны совместные усилия. До встреч на этом сайте!

V

Приложения

В этой части...

A

Программы атаки и методы сканирования

В этом приложении рассматривается несколько сетевых трассировок. У каждой из них есть своя история. Большинство трассировок приведены в формате отчетов TCPdump, который соответствует формату трассировок, использованному в книге Ричарда Стивенса *TCP/IP Illustrated, Volume 1: The Protocols.* Этот справочник всегда должен быть под рукой у каждого настоящего аналитика сетевого трафика.

Ложные тревоги

Начнем это приложение с описания наиболее распространенных ошибок аналитиков. Хотя в группы CIRT (Computer Incident Response Team – группа реагирования на угрозы безопасности компьютеров) берут только первоклассных специалистов, описанные в первом разделе ошибки могут допустить даже они. Официально многие группы CIRT заявляют о своем либеральном отношении к получаемым отчетам, даже если в них будут присутствовать сообщения о ложных тревогах. Я, в принципе, с этим согласен, но считаю, что если в чем-то не уверен, то так и нужно сказать в отчете! Аналитик имеет максимум информации о случившемся, и ложные тревоги может предотвратить именно он.

Реакция без воздействия

Следующая трассировка представляет собой стандартный пример ситуации, которую ошибочно принимают за атаку через потайной ход. В 7:17 датчик зарегистрировал получение пакета от хоста `mysystem`, при этом портом отправителя был порт службы `echo` (7). Пакет размером в 64 Кбайт был отправлен хосту `target1` на порт `24925`.

```
ВРЕМЯ            Хост-отп. Порт_отп. >  Хост-пол.  Порт_пол.  Протокол Размер
07:17:09.615279   mysystem.echo      >     target1.24925:        udp      64
```

Когда я впервые увидел этот отчет, мне стало плохо. Я был уверен, что имею дело с потайным ходом. Вы спросите, почему? Я *знал*, что в моей сети заблокированы исходящие эхо-запросы на брандмауэре, поэтому никто не должен был возвращать никаких эхо-ответов. Я решил, что имею дело с действиями какой-то программы атаки, наводнением UDP-пакетами через порт 7 или же с потайным ходом. Но этому не находилось подтверждений. Кто в здравом уме будет создавать программу атаки, использующую в качестве порта-отправителя порт 7? Это почти наверняка привлечет внимание.

Я попытался найти трафик, который привел к появлению ответного пакета, но *ничего не нашел*. Это меня *озадачило*. А *произошло следующее*. На самом деле за выходные изменился периметр моей сети, и кто-то на хосте `mysystem` действительно возвращал эхо-ответы. Почему я не увидел первоначальный пакет воздействия? Есть два наиболее вероятных варианта: асимметричная маршрутизация или неверная настройка связующего порта. Некоторые из старых реализаций коммутируемых сетей в режиме соединения позволяют передавать трафик только в одном направлении, что и может привести к возникновению ложной тревоги (как показано в следующей трассировке).

```
07:17:09.615279 mysystem.echo > target1.24925: udp 64
07:17:10.978236 mysystem.echo > irc.some.where.40809: udp 600
07:17:11.001745 mysystem.echo > irc.some.where.14643: udp 600
07:17:11.146935 mysystem.echo > irc.some.where.49911: udp 600
07:17:12.254277 mysystem.echo > irc.some.where.28480: udp 600
07:17:12.350014 mysystem.echo > irc.some.where.20683: udp 600
07:17:12.835873 mysystem.echo > target1.5134: udp 64
07:17:13.266794 mysystem.echo > irc.some.where.16911: udp 600
07:17:13.862476 mysystem.echo > target1.32542: udp 64
07:17:14.032603 mysystem.echo > irc.some.where.32193: udp 600
07:17:14.579404 mysystem.echo > irc.some.where.24455: udp 600
07:17:14.619173 mysystem.echo > irc.some.where.5120: udp 600
07:17:14.792983 mysystem.echo > irc.some.where.47466: udp 600
07:17:14.879559 mysystem.echo > target1.16878: udp 64
07:17:15.308270 mysystem.echo > irc.some.where.12234: udp 600
```

Связующие порты

Коммутируемые сети представляют собой одну из основных проблем для аналитика обнаружения взлома. Датчик с одним сетевым адаптером как для прослушивания трафика в неразборчивом режиме, так и для отправки отчетов на консоль аналитика, может нарушить работу нескольких коммутируемых сетей.

Если администратор сети хочет добавить второй адаптер для датчика, то на это следует согласиться. *Используйте один интерфейс для прослушивания трафика в неразборчивом режиме* (здесь даже не нужен IP-адрес), а второй — для взаимодействия датчика с консолью аналитика. В наше время это лучший способ использовать датчик сетевой системы обнаружения взлома, который позволяет защитить его от хакеров.

Если приведенная выше трассировка не вызвана неверной настройкой связующего порта в коммутируемой сети, то что еще может являться ее причиной? Например, соединение через потайной ход или программа атаки. Вначале стоит проверить второе предположение.

Как правило, при соединениях по протоколу IP есть как реакция, так и воздействие. Когда аналитик сталкивается с непонятной трассировкой, он должен оп-

ределить, что ее вызвало. Это позволит понять, что происходит на самом деле. Эта трассировка выявлена потому, что перехватывался весь проходящий трафик, но найти первоначальное воздействие не удалось. В данном случае требовалось найти эхо-запрос, отправленный компьютеру mysystem.

Что еще можно узнать из этой трассировки? Для начала нужно вспомнить принципы работы службы echo. Программа echo считывает строку символов и возвращает ее. Теперь можно представить, как должен выглядеть искомый трафик, если, например, неверно сконфигурирован датчик или используется соединение через потайной ход.

Предполагаемый трафик должен иметь следующий вид.

```
07:17:09.527910 target1.24925 > mysystem.echo: udp 64
07:17:09.615279 mysystem.echo > target1.24925: udp 64
07:17:10.823651 irc.some.where.40809 > mysystem.echo: udp 600
07:17:10.978236 mysystem.echo > irc.some.where.40809: udp 600
```

Что это значит? Это значит, что хосты target1 и irc.some.where отправили хосту mysystem строку символов и получили эхо-ответ. Возможно ли это? Скорее всего, нет. Даже если одной системой использовался эхо-запрос для тестирования соединения или устранения неполадок, такое совпадение одновременно для двух компьютеров практически нереально. Возможно, что это атака отказа в обслуживании, направленная против компьютеров target1 и irc.some.where. Одним из основных правил безопасности является отключение на компьютере всех неиспользуемых служб. Если бы системный администратор хоста mysystem закомментировал службу echo в файле /etc/inetd.conf, то этой трассировки никогда бы не было. Если это еще не убедило вас в необходимости отключения службы echo — не беда. Далее мы рассмотрим еще более интересные трассировки, связанные с этой службой.

У указанной трассировки есть еще одна проблема. Портами получателя являются 24925, 40809, 14643, 49911 и т.д. Поскольку перед нами эхо-ответы, можно предположить, что они использовались в качестве портов отправителя в эхо-запросах. Однако эти порты скорее случайны, что не характерно при выборе порта отправителя. Обычно после порта 24925 используется порт 24926 и т.д. Таким образом, вероятно, приходят эхо-ответы на специально подготовленные пакеты. Такую трассировку можно по ошибке принять за действия через потайной ход (когда на самом деле она вызвана неправильной настройкой коммутируемой сети), но случается это нечасто.

Рассмотрим последний пример подобной трассировки. На первый взгляд, ее также можно принять за какую-то атаку.

```
11:38:54.010000 masker.com > 192.168.133.127: icmp: address mask is
0xfffffe00
11:39:43.180000 masker.com > 172.16.33.116: icmp: address mask is 0xfffffe00
11:53:37.780000 masker.com > 192.168.58.105: icmp: address mask is 0xfffffe00
11:56:43.690000 masker.com > 172.16.178.85: icmp: address mask is 0xfffffe00
12:15:52.550000 masker.com > 172.16.121.67: icmp: address mask is 0xfffffe00
12:25:41.800000 masker.com > 172.16.247.72: icmp: address mask is 0xfffffe00
12:45:07.470000 masker.com > 172.16.110.69: icmp: address mask is 0xfffffe00
12:45:31.530000 masker.com > 172.16.167.73: icmp: address mask is 0xfffffe00
12:58:23.350000 masker.com > 192.168.214.116: icmp: address mask is
0xfffffe00
```

Помните, что такое ICMP-запрос маски подсети? В ответе хоста должна содержаться маска подсети той сети, в которой он установлен. Хотя в отчетах TCPdump не указывается слово *reply* (ответ), но есть слова *address mask* (адресная маска) и приводится ее шестнадцатеричное значение. Это и есть ответы на ICMP-запрос о маске подсети. Однако ни один из хостов, получающих ответ, реально не существует, поэтому они не могли быть инициаторами запроса.

Итак, похоже, что дело снова в подмене в запросах IP-адресов сетей `192.168` и `172.16`. Зачем это кому-то нужно? Возможно, это попытка наводнить пакетами хост `masker.com` с помощью метода доставки пакетов, отличного от эхо-запросов ICMP. На самом деле не имеет значения, какой вид трафика направляется на атакуемый хост при организации наводнения или атаки отказа в обслуживании. Рассмотрим ложную тревогу, которая заставляла ошибаться многих начинающих аналитиков.

Сканирование или реакция

Ниже приведена трассировка, полученная из ежечасного отчета системы Shadow. Эта система была настроена на отслеживание трафика, предназначенного для UDP-порта 1080, который используется для proxy-сервера службы `socks`. Существует несколько программ атаки на эту службу, поэтому желательно отслеживать несанкционированный доступ к UDP-порту 1080.

```
18:20:12.080000 dns.com.53 > myhost.com.1080: 5 NXDomain* 0/1/0 (128)
18:20:12.300000 dns.com.53 > myhost.com.1080: 6 NXDomain* 0/1/0 (119)
18:20:12.410000 dns.com.53 > myhost.com.1080: 7* 1/0/0 (48)
```

Этот отчет ничего вам не напоминает? Обратите внимание на запись после `1080`. *Являются ли пакеты окончательным ответом? Что можно сказать о порте отправителя?* Напоминает ли это пакеты службы DNS? Да, похоже, что это ответы хоста `dns.com` на несколько отправленных DNS-запросов. Идентификационные номера запросов – 5, 6 и 7, и в ответе на последний запрос содержится одна запись о ресурсах, ни одной записи о полномочиях и никаких дополнительных записей.

Поскольку это больше напоминает ответ, чем сканирование, нужно изучить исходящий трафик и проверить, были ли хостом `myhost.com` отправлены DNS-запросы. Следующий результат позволяет расставить все по своим местам.

```
18:20:11.870000 myhost.com.1080 > dns.com.53: 5+ (50)
18:20:12.090000 myhost.com.1080 > dns.com.53: 6+ (41)
18:20:12.310000 myhost.com.1080 > dns.com.53: 7+ (32)
```

Объясняется все тем, что хост `myhost.com` отправил запросы на определение имен под номерами 5, 6 и 7 DNS-серверу `dns.com`. При этом клиентом использовался временный порт 1080. Система Shadow не способна учитывать наши собственные действия, она просто сообщает о каждом совпадении с заданной сигнатурой. Это – ложная тревога. Одна из наиболее распространенных ложных тревог связана с наводнением SYN-пакетами.

SYN-наводнение

Интерпретация SYN-наводнений является весьма неприятной обязанностью аналитика сетевого трафика. Системы обнаружения вторжений очень часто

ошибаются по этому поводу и выдают ложные уведомления о тревогах. Можно отчитаться о выявлении атаки, если пакеты SYN-наводнения поступают от известного враждебного хоста, или с данным соединением связаны другие вредоносные действия, или наводнение очевидно (например, получение более 50 запросов на соединение за минуту). В противном случае приходится сидеть и ждать дальнейших событий.

Настоящее SYN-наводнение

Следующая трассировка соответствует настоящему наводнению SYN-пакетами.

```
14:18:22.5660 flooder.601 > server.login: S 1382726961:1382726961(0) win 4096
14:18:22.7447 flooder.602 > server.login: S 1382726962:1382726962(0) win 4096
14:18:22.8311 flooder.603 > server.login: S 1382726963:1382726963(0) win 4096
14:18:22.8868 flooder.604 > server.login: S 1382726964:1382726964(0) win 4096
14:18:22.9434 flooder.605 > server.login: S 1382726965:1382726965(0) win 4096
14:18:23.0025 flooder.606 > server.login: S 1382726966:1382726966(0) win 4096
14:18:23.1035 flooder.607 > server.login: S 1382726967:1382726967(0) win 4096
14:18:23.1621 flooder.608 > server.login: S 1382726968:1382726968(0) win 4096
14:18:23.2284 flooder.609 > server.login: S 1382726969:1382726969(0) win 4096
14:18:23.2825 flooder.610 > server.login: S 1382726970:1382726970(0) win 4096
14:18:23.3457 flooder.611 > server.login: S 1382726971:1382726971(0) win 4096
14:18:23.4083 flooder.612 > server.login: S 1382726972:1382726972(0) win 4096
14:18:23.9030 flooder.613 > server.login: S 1382726973:1382726973(0) win 4096
14:18:24.0052 flooder.614 > server.login: S 1382726974:1382726974(0) win 4096
```

Что-то знакомое? Может, это поможет вспомнить:

> От: tsutomu@ariel.sdsc.edu (Tsutomu Shimomura), comp.security.misc
>
> Дата: 25 января 1995 года
>
> "Примерно через шесть минут мы получили поток SYN-пакетов (попыток установки соединения) от хоста 130.92.6.97 на порт 513 (`login`) нашего сервера. Целью этого наводнения является заполнить очередь на установку соединений для порта 513 сервера, чтобы тот не мог устанавливать новые соединения. В ответ на отправку сервером SYN/ACK-пакетов не будут получены пакеты с установленным флагом RST".

Ложное SYN-наводнение

После сравнения предыдущего фрагмента из отчета об атаке Митника можно вообще не найти отличий со следующей трассировкой. Они есть, хоть и весьма незначительны. В обоих трассировках увеличиваются номера портов отправителя и порядковые номера. Размер TCP-окна остается постоянным — 4096 байт. Очевидно, что ниже показан отчет о поступлении двух наборов пакетов (по четыре пакета в каждом), отправленных для установки соединения. Обратите внимание на неизменный порт отправителя, неизменный порядковый номер и временные промежутки между получением пакетов (3, 6 и 12 секунд).

```
14:02:22.5166 host.2104 > server.25: S 1382726960:1382726960(0) win 4096
14:02:25.5669 host.2104 > server.25: S 1382726960:1382726960(0) win 4096
14:02:31.7447 host.2104 > server.25: S 1382726960:1382726960(0) win 4096
14:02:42.8311 host.2104 > server.25: S 1382726960:1382726960(0) win 4096
14:02:58.8868 host2.3311 > server.25: S 2382927964:2382927964(0) win 4096
14:03:01.9434 host2.3311 > server.25: S 2382927964:2382927964(0) win 4096
```

```
14:03:07.0025 host2.3311 > server.25: S 2382927964:2382927964(0) win 4096
14:03:19.1035 host2.3311 > server.25: S 2382927964:2382927964(0) win 4096
```

Казалось бы, такое небольшое отличие: использование электронной почты, а не какой-то другой службы, но какое оно имеет значение! Сообщение электронной почты очень важно, по крайней мере для ретрансляторов. Если ретранслятор не сможет отправить почту в первый раз, он попытается это сделать еще раз часом позже. Если обратить внимание на время, то можно догадаться, что происходит. Нет никакой атаки отказа в обслуживании, просто не работает сервер электронной почты. Почтовые ретрансляторы по всему миру каждый час (обычно в начале часа) пытаются доставить сообщение. Вот и необходимое условие для ложной тревоги.

Как еще одну причину для появления подобных ложных тревог можно назвать открытие Web-страницы с помощью Internet Explorer. Эта программа устанавливает соединение для каждого элемента страницы в форматах GIF, JPEG, HTML и так далее, пока не достигнет предела в 32 соединения. Поэтому следует отключить уведомление о тревоге для SYN-наводнений на TCP-порты 25, 80 или 443.

Еще лучше крайне недоверчиво относиться к уведомлениям своей системы обнаружения вторжений о SYN-наводнениях (по крайней мере, начинающим аналитикам). Большинство коммерческих систем обнаружения вторжений генерируют ложные тревоги о SYN-наводнениях настолько часто, что приходится устанавливать такие высокие значения, которые не позволяют выявить настоящее наводнение SYN-пакетами. Хорошо то, что многие современные операционные системы более устойчивы к SYN-наводнениям. Для воздействия на такие системы потребуется огромное количество пакетов, не заметить которые весьма сложно.

SYN-наводнения невысокой интенсивности можно игнорировать, чего нельзя сказать про троянские программы, например про Back Orifice. Эти программы позволяют злоумышленнику получить полный контроль над взломанным компьютером. По отношению к таким серьезным угрозам, как Back Orifice, никогда нельзя отключать фильтр для системы обнаружения вторжений, даже если его применение приводит к возникновению ложных тревог.

Back Orifice

Внедрение троянских программ и сканирование в поисках подходящих целей стали популярными начиная с середины 1997 года и остаются таковыми по нынешнее время. В конце 1998 и начале 1999 года лидерство удерживали программы Back Orifice и Netbus, а в конце 1999 года его перехватила программа SubSeven. Для Back Orifice по умолчанию используется UDP-порт 31337, а для Netbus – TCP-порт 12345 (а также TCP-порт 12346, хотя я никогда не видел реальных примеров его использования). Конечно, для большинства троянских программ номера портов можно изменить, что значительно усложняет их обнаружение. Следующую трассировку мы как-то наблюдали дважды за один день.

```
11:20:44.148361 ns1.com.31337 > ns2.arpa.net.53: 38787 A? arb.arpa.net. (34)
11:52:49.779731 ns1.com.31337 > ns1.arpa.net.53: 39230 ANY? hq.arpa.net. (36)
```

Самое время напомнить о пользе TCPdump. Хотя это UDP-трассировка, она не похожа на первую трассировку в этой главе. Программа TCPdump сообщает до-

полнительные сведения о пакете, так как она поддерживает работу с протоколом DNS (UDP-порт 53). Наша клиентская система выполняет поиск имени на DNS-сервере `ns*.arpa.net`. Так причем здесь все эти порты `31337`?

Именно на этот вопрос я хотел получить ответ, когда увидел данную трассировку. Мы выделили пакет, распечатали его содержимое в шестнадцатеричном формате, проверили его с помощью анализатора пакетов Tcpshow и сравнили с другими пакетами службы DNS. Он оказался вполне нормальным.

До появления программного пакета BIND 8 ожидаемым, хотя и не обязательным, методом поиска имен DNS-сервером было использование порта 53. Иногда, если клиентом была Windows-система, я видел, что портом отправителя запроса на поиск имени был порт 137. Но почему 31337?

Я забыл об этом случае, пока другой аналитик не обратил мое внимание на такой же пакет. Я взялся за телефон и нашел человека, который управлял работой DNS-сервера в организации. Я рассказал ему о случившемся.

> **Норткат**. Я наблюдаю поступление на DNS-серверы пакетов, в которых указан порт отправителя 31337.
>
> **Молодой человек**. Мы все проверили и это не Back Orifice.
>
> **Норткат**. Я тоже это знаю, но такие пакеты вызывают сообщения о тревогах на всех системах обнаружения вторжений.
>
> **Молодой человек**. Вам следует настроить свою систему обнаружения вторжений.
>
> **Норткат**. Нет. Это вы измените свой порт отправителя или на моем узле будут блокироваться все ваши запросы, а также на узлах моих друзей. Ваша компания потеряет свои контракты, а вы — свою работу.

Он переспросил, кто я такой, и мы начали искать приемлемое решение.

Таким образом, это была ложная тревога. Атаки не было. Просто один юноша вообразил себя "крутым" и решил настроить работу службы DNS. Ему оставалось доделать сущую мелочь, и все было бы в порядке. Несмотря на подобные случаи, аналитик сетевого трафика никогда не должен отключать фильтры для выявления потенциально опасных атак. Также он должен проверять, не является ли уведомление об опасности ложной тревогой. Некоторым может показаться, что я был слишком резок с молодым человеком. Представьте себе, какие последствия могут вызвать его действия для группы CIRT. Не забывайте, только опытный аналитик может отличить ложную тревогу от реальной.

Заметим, что, хотя современные DNS-серверы, на которых установлен пакет BIND 8, используют для отправки пакетов временный порт с номером больше 1024, порт 31337 исключен из диапазона допустимых портов.

Этот пример ясно продемонстрировал, что важно не только отправлять отчеты группе CIRT, но также и делиться информацией с другими дружественными организациями, обладающими средствами обнаружения взлома. Именно это позволило мне выяснить, что использование порта 31337 не было случайным. Иногда приходится блокировать целые группы IP-адресов, если владельцы компьютеров, которым принадлежат эти адреса, ведут себя вызывающе.

Блокирование в действии

Однажды нас не поддержал один из основных Internet-провайдеров, с IP-адресов которого осуществлялись атаки на наши сети. Снова и снова мы пытались привлечь эту организацию к сотрудничеству. Наконец мы заблокировали им доступ к своим сетям (электронная почта, Web и все, что только можно). Через три недели они взмолились о пощаде, поскольку начали терять деньги из-за разрыва контрактов с корпоративными клиентами. Мы получили обещание нести ответственность за действия клиентов и утроить персонал обеспечения защиты. Большего и не требовалось.

На этом мы завершаем рассказ о ложных тревогах. *Атакой* можно назвать метод или программу, позволяющую хакеру использовать уязвимое место в компьютерной системе. На практике очень трудно найти различие между разведывательным сканированием и настоящей атакой. Современное поколение программ для атаки позволяет реализовать и то, и другое.

Я не единственный, кто выступает против разделения рассмотренных трассировок по категориям. Исследователи в области обнаружения взлома работают над этой проблемой уже несколько лет, но пока так и не выработали единой системы классификации атак. Результатом этих исследований стал выпущенный в феврале 1999 года компакт-диск с проектом DOVES (Database of Vulnerabilities, Exploits, and Signatures — база данных уязвимых мест, программ атаки и сигнатур). Более подробную информацию можно получить у Мэта Бишопа (`bishop@cs.ucdavis.edu`). Компания Mitre стала организатором создания перечня уязвимых мест CVE (Common Vulnerability Enumeration). Этот проект стал одним из лучших и получил широкую поддержку производителей программного обеспечения. К настоящему времени описано более 2000 уязвимых мест и еще 1700 сообщений находятся на рассмотрении. Более подробную информацию можно узнать на сайте CVE по адресу `cve.mitre.org`.

В следующем разделе рассмотрены трассировки атак, осуществляемых посредством протокола IMAP.

Программы атаки на IMAP

Ни один из типов программ атаки не наносил такого ущерба, как атаки посредством протокола IMAP. Нет ничего необычного в атаках на переполнение буфера IMAP, подобные проблемы возникали и для службы DNS. Поскольку обе эти программы работают в привилегированном режиме, то атаки на них особенно опасны и позволяют хакеру получить доступ с правами суперпользователя (root).

Сигнатура атаки на IMAP, порт отправителя 10143

Эта атака является одной из наиболее опасных атак на переполнение буфера. Обратите внимание на то, что данное сканирование (в приведенном ниже листинге) проводится сразу для двух сетей. Важное значение имеет и временной интервал между пакетами. Такой большой интервал между получением пакетов объясняется тем, что сканирование направлено на все сети класса B в Internet. Эта трассировка была зарегистрирована в середине 1997 года и сейчас встречается крайне редко.

```
14:13:54.847401 newbie.hacker.org.10143 > 192.168.1.1.143: S
14:24:58.151128 newbie.hacker.org.10143 > 172.31.1.1.143: S
14:35:40.311513 newbie.hacker.org.10143 > 192.168.1.2.143: S
```

```
14:43:55.459380 newbie.hacker.org.10143 > 192.168.2.1.143: S
14:54:58.693768 newbie.hacker.org.10143 > 172.31.2.1.143: S
15:05:41.039905 newbie.hacker.org.10143 > 192.168.2.2.143: S
15:13:59.948065 newbie.hacker.org.10143 > 192.168.3.1.143: S
```

Сигнатура атаки на IMAP, порядковый номер 111

Следующая трассировка представляет собой еще один отчет о сканировании (программе атаки) на IMAP с четкой сигнатурой. Постоянным остается порт отправителя, а также значения в полях порядкового номера и номера подтверждения (111) и, конечно, размер окна со значением 0. С точки зрения применения сигнатур эта трассировка представляет особенный интерес. Первый раз она была замечена в конце 1998 года, после чего появились многочисленные попытки сканирования с использованием порта отправителя 0 и установленными флагами SYN и FIN (как показано в следующем разделе). В начале 1999 года сигнатура была выявлена опять. Такое впечатление, что программа атаки была потеряна на несколько месяцев.

```
00:25:09.57 prober.2666 > relay.143: S 111:111(0) win 0
00:25:09.59 prober.2666 > relay.143: S 111:111(0) win 0
00:42:50.79 prober.2666 > web.143: S 111:111(0) win 0
00:43:24.05 prober.2666 > relay.143: S 111:111(0) win 0
00:43:24.07 prober.2666 > relay.143: S 111:111(0) win 0
00:44:20.42 prober.2666 > relay2.143: S 111:111(0) win 0
00:44:42.62 prober.2666 > ns2.143: S 111:111(0) win 0
00:44:42.64 prober.2666 > ns2.143: S 111:111(0) win 0
00:44:42.67 prober.2666 > ns1.143: S 111:111(0) win 0
00:44:42.69 prober.2666 > ns1.143: S 111:111(0) win 0
```

Атака на порты с помощью SYN/FIN-пакетов

Было очень интересно наблюдать различные изменения атаки с использованием пакетов, в которых были одновременно установлены флаги SYN и FIN (часто используют аббревиатуру SF). Это один из важнейших шаблонов атак, поэтому каждый аналитик должен его знать. Мне знакома эта сигнатура с конца 1996 года (с момента появления атаки `jackal.c`), а самым последним вариантом этой атаки была атака на переполнение буфера против службы SSH, проведенная в декабре 2001 года. Установка флагов SYN и FIN позволяет пакету пройти через статический фильтр пакетов, так как такие фильтры блокируют только SYN-пакеты. Однако если такой пакет поступит на открытый порт Windows или UNIX-системы, то в ответ будет отправлен SYN/ACK-пакет. Это просто находка для хакера, ведь он может проникнуть сквозь периметр и скомпрометировать нужную систему. Рассмотрим основные варианты этой атаки.

SYN/FIN-пакет, порт отправителя 0

Впервые об этой трассировке я узнал из списка рассылки Bugtraq в марте 1998 года. Сигнатурой является использование порта отправителя 0 (что нелогично) и установка в пакете флагов SYN и FIN (тоже нелогично). Система обнаружения взлома должна уметь выявлять подобные трассировки. Обратите внимание на случайно выбираемые подсети 26, 24, 17, 16, а также на номер хоста. Возможно, это делается для того, чтобы сканирование было не таким явным. Также важна скорость сканиро-

вания. Детекторы сканирования обязаны выявить пять попыток подключения к пяти различным хостам, осуществленные приблизительно за четверть секунды.

```
13:10:33.281198 newbie.hacker.org.0 > 192.168.26.203.143: SF
↳374079488:374079488(0) win 512
13:10:33.334983 newbie.hacker.org.0 > 192.168.24.209.143: SF
↳374079488:374079488(0) win 512
13:10:33.357565 newbie.hacker.org.0 > 192.168.17.197.143: SF
↳374079488:374079488(0) win 512
13:10:33.378115 newbie.hacker.org.0 > 192.168.16.181.143: SF
↳374079488:374079488(0) win 512
13:10:33.474966 newbie.hacker.org.0 > 192.168.24.194.143: SF
↳374079488:374079488(0) win 512
```

Приведенное выше сканирование имеет несколько интересных преимуществ. FIN-пакеты могут пропускаться устройствами фильтрации пакетов, даже если SYN-пакеты блокируются. Это увеличивает вероятность получения ответа. Кроме того, флаг FIN сигнализирует о разрыве соединения, поэтому некоторые системы регистрации событий могут не уведомить о попытке соединения. SYN/FIN-пакеты являются характерным признаком средства сканирования под названием `jackal`, которое предназначено для обмана брандмауэров. Сложность этой сигнатуры заключается в том, что для ее создания, по всей видимости, используется несколько программ атаки (или средств сканирования). Более современной программой, использующей подобную сигнатуру, является программа `nmap` – наиболее эффективное средство из созданных хакерами.

SYN/FIN-пакет, порт отправителя 65535

Следующая трассировка – это любопытный вариант предыдущей. Она была зарегистрирована в ноябре 1998 года. Предполагалось, что она является результатом применения средства, которое позволяет пользователю выбрать любой порт отправителя. Хотя я ничуть не сомневаюсь, что такое средство уже существует или будет создано в ближайшее время, но это не объясняет причины появления зарегистрированных аналитиками сотен примеров с использованием в качестве портов отправителей портов 0 или 65535. До 1999 года вообще не появлялось SYN/FIN-пакетов с другими портами отправителей. Значит, номер порта отправителя был жестко закодирован в программу атаки, и программа с номером 65535 была второй версией после оригинала. Трассировка выглядит следующим образом.

```
16:11:38.13 IMAPPER.65535 > ns2.org.143: SF 3794665472:3794665472(0) win 512
16:11:38.13 IMAPPER.65535 > ns2.org.143: SF 3794665472:3794665472(0) win 512
```

Атака на службы DNS и NFS с помощью SYN/FIN-пакетов

Служба IMAP является удобной, но далеко не единственной целью атак хакеров. В следующей трассировке также используется порт отправителя 0 и установлены флаги SYN и FIN. Но в данном случае атака имеет двойную цель. Хакер пытается провести атаку на TCP-порт 53, который также используется службой DNS.

TCP-порт 53 используется вместо UDP-порта 53 при переносе зоны DNS – по существу, списка хостов локальной сети.

Как указывалось выше, порт отправителя 0 и установленные флаги SYN и FIN являются сигнатурой для обычной атаки посредством IMAP. Для проведения данной атаки против службы NFS почти наверняка используется та же программа, что и для предыдущей. Такие изменения программ позволяют отличить хакеров, которые способны создавать, изменять или компилировать программы, от злоумышленников, которые только используют уже готовые программы. В данном случае нарушитель придумал новый вид атаки, воспользовавшись старым методом доставки пакетов.

Зачем? Обнаружение взлома отчасти напоминает военное ремесло. У оружия боеголовка часто отделяется от носителя. Например:

- лучники могут применять одни наконечники стрел против пехоты, а другие – чтобы поджигать стены зданий;
- катапульты способны выбрасывать камни для разрушения стен, но также способны метать и горящие снаряды;
- современные крылатые ракеты могут нести обычные заряды и попадать точно в окно врагу, но также могут и нести ядерные боеголовки.

В каждом из этих случаев система доставки позволяет провести различные атаки. Тот же принцип используется и в информационных войнах. "Наконечником" в следующей трассировке является порт службы NFS 2049. Сигнатура механизма доставки (порт отправителя 0 и набор флагов SYN и FIN) выделена жирным шрифтом.

```
12:11:48 prober.21945 > ns1.net.53: SF 1666526414:1666526414(0) win 512
12:11:49 prober.21951 > ns2.net.53: SF 11997410:211997410(0) win 512

12:36:54 prober.0 > relay.net.2049: SF 3256287232:3256287232(0) win 512
12:37:03 prober.0 > web.net.2049: SF 3256287232:3256287232(0) win 512
12:37:05 prober.0 > relay2.net.2049: SF 3256287232:3256287232(0) win 512
```

Атаки этого шаблона продолжают происходить. Последний известный случай произошел в феврале 2000 года. В GIAC поступили уведомления об атаках с использованием пакетов, поступающих на TCP-порт 109 (служба POP2), в которых был установлен порт отправителя 0, флаги SYN и FIN. Совсем недавно атака изменилась, и теперь TCP-порт 109 указывается и как порт отправителя, и как порт получателя. И последнее замечание по поводу приведенной выше трассировки. По всей видимости, действует хакер-новичок. Если программа атаки на хост не проходит с первого раза, то благоразумнее изменить проверяемый IP-адрес перед повторной попыткой. Повторная проверка одного и того же IP-адреса увеличивает шансы на то, что скоро в дверь постучится полиция. Таким образом, мы рассмотрели первый пример модификации оригинального кода программы атаки для атаки на службу NFS. И изменения вносятся постоянно. В декабре 2001 года мы выявили атаку этого типа против службы SSH (TCP-порт 22) при использовании порта отправителя 22, порта получателя 22 и установленных флагах SYN и FIN.

Сканирование для выбора цели атаки

В этом разделе будут рассмотрены интересные шаблоны попыток сканирования, для которых (за исключением службы `discard` и атаки IP-191) используются хорошо известные порты уязвимых служб. Одна из проблем, связанных с отделением программ атаки от средств сканирования, связана с тем, что на большинстве узлов брандмауэры или фильтрующие маршрутизаторы используются для блокирования доступа к потенциально уязвимым службам, что затрудняет сбор информации. Например, для атак по протоколу TCP полная процедура установки соединения никогда не завершается, так как соединение блокируется, что не дает возможности определить цели злоумышленника.

Утилита mscan

Следующая трассировка представляет один из наиболее распространенных шаблонов атаки – использование утилиты `mscan`. Код программы атаки для мультисканирования доступен повсеместно и не является показателем опытного хакера. Эта утилита использовалась для компрометации многих систем, так как позволяет проверить уязвимые места многих систем, подключенных к Internet.

```
06:13:23.188197 bad.guy.org.6479 > target.mynetwork.com.23: S
06:13:28.071161 bad.guy.org.15799 > target.mynetwork.com.80: S
06:13:33.107599 bad.guy.org.25467 > target.mynetwork.com.143: S
06:13:38.068035 bad.guy.org.3861 > target.mynetwork.com.53: S
06:13:43.271220 bad.guy.org.14296 > target.mynetwork.com.110: S
06:13:47.831695 bad.guy.org.943 > target.mynetwork.com.111: S
```

Из отчета AL-98.01 группы AusCERT о тревоге, вызванной использованием средства multiscan (`mscan`) от 20 июля 1998 года (`ftp://ftp.auscert.org.au/pub/auscert/advisory/AL-98.01.mscan`):

> "AusCERT получила уведомления, указывающие на резкое увеличение активности осуществляемых попыток сканирования. Очевидно, что злоумышленники используют новое средство под названием "Multiscan" или `mscan`. Эта программа позволяет просканировать целые домены или полные диапазоны IP-адресов и выявить хорошо известные уязвимые места для таких служб: `statd`, NFS, программы CGI-BIN (например, "`handler`", "`phf`" и "`cgi-test`"), X, POP3, IMAP, DNS, `finger`".

А почему в главе, посвященной средствам атаки, речь зашла о программе сканирования? Просто создано настолько много программ атаки на службы `telnet`, IMAP, DNS, POP3 и `portmap`, и они настолько хорошо известны, что решено было описать что-то более интересное.

Наследник mscan

Если один хакер создал `mscan`, то другому обязательно захочется улучшить оригинальную программу. Следующая трассировка была впервые зарегистрирована в ноябре 1998 года. Из нее можно узнать много полезной информации. Интенсивность сканирования составляет 10 пакетов в секунду. Это не рекорд, но довольно высокий показатель. Мы, безусловно, надеемся, что программы выявления попыток

есть множество портов, а хакер не знает, что можно атаковать. Представим, например, сеть класса B 172.20.0.0 (в ней есть 65535 возможных IP-адресов). На каждом из хостов доступны 65536 TCP-портов и 65536 UDP-портов. Следовательно, перед хакером есть более 23 триллионов возможных целей. При сканировании с интенсивностью 18 пакетов в секунду на полную проверку всей этой сети уйдет 5 миллионов лет. Поскольку компьютеры меняются в среднем раз за 3–5 лет, то сканирование становится абсолютно бесполезным.

Конечно, сейчас злоумышленники используют более быстрые и разумные методы сканирования. Они не проверяют все возможные порты компьютеров. Для работы наиболее популярных служб используются около пятидесяти TCP- и UDP-портов, что ограничивает количество возможных целей значением около 163 миллионов (сканирование со скоростью 18 пакетов в секунду займет около 4 месяцев). Вполне подходяще для хакера. И если для сети не установлено средств обнаружения взлома, ее владельцы, возможно, никогда не узнают о сканировании своих случайных хостов и некоторых портов этих хостов.

Если злоумышленнику удастся составить точную схему хостов сети, то он сможет использовать эту информацию против специалистов по информационной безопасности. Большинство адресных пространств заполнены не целиком. Когда хакер узнает адреса работающих хостов, он получает серьезное преимущество. Допустим, что в сети класса B работают только 6000 компьютеров, и злоумышленник выяснил все их адреса. Теперь он может просканировать обнаруженные хосты (по 18 пакетов в секунду) менее чем за 10 дней, а есть и более эффективные методы сканирования. Например, если разрешено прохождение широковещательных эхо-запросов ICMP, то для составления схемы сети потребуется всего лишь 255 пакетов.

Мораль этого рассказа очевидна. Если хакеру не удастся провести разведывательных действий, ему придется гадать при выборе цели атаки. Если разрешить хакерам проводить разведку, то им не потребуется делать никаких лишних попыток.

Каким же образом злоумышленник может составить точную схему сети? На многих сетях до сих пор поддерживаются *списки хостов* (host table), доступные для скачивания по FTP. В других сетях разрешено осуществлять переносы зон DNS. Кроме того, хакер может получить эту информацию с помощью зондирования сети.

В главе 4, "Протокол ICMP", описаны устаревшие методы составления схем сетей с помощью ICMP-сообщений. Самые простые из них заключаются в отправке эхо-запросов ICMP на отдельные хосты, что, несомненно, привлекает внимание. Мы также рассмотрели широковещательные эхо-запросы, когда схема сети определяется с помощью эхо-запросов по адресам `.0` и `.255`, что делает этот процесс более эффективным и менее заметным. В этом разделе представлен еще один метод составления схемы сети с помощью эхо-запросов и более подробно исследованы широковещательные запросы.

Зондирование сети с помощью эхо-запросов UDP

В следующей трассировке хакер проверяет группу сетевых адресов. Он выявляет только два работающих хоста, но вполне вероятно, что в этой сети их гораздо больше. Разброс и чередование адресов зондируемых хостов позволяют сделать попытку незаметной для большинства программ обнаружения сканирования. Запись `udp 6` означает, что в UDP-пакете содержится 6 байт полезных данных.

Как указывается в последнем разделе этого приложения, понятие скрытого сканирования довольно неопределенно, но, по моему мнению, медленный и незаметный процесс является самым лучшим методом скрытого сканирования.

```
02:08:48.088681 slowpoke.mappem.com.3066 > 192.168.134.117.echo: udp 6
02:15:04.539055 slowpoke.mappem.com.3066 > 172.31.73.1.echo: udp 6
02:15:13.155988 slowpoke.mappem.com.3066 > 172.31.16.152.echo: udp 6
02:22:38.573703 slowpoke.mappem.com.3066 > 192.168.91.18.echo: udp 6
02:27:07.867063 slowpoke.mappem.com.3066 > 172.31.2.176.echo: udp 6
02:30:38.220795 slowpoke.mappem.com.3066 > 192.168.5.103.echo: udp 6
02:49:31.024008 slowpoke.mappem.com.3066 > 172.31.152.254.echo: udp 6
02:49:55.547694 slowpoke.mappem.com.3066 > 192.168.219.32.echo: udp 6
03:00:19.447808 slowpoke.mappem.com.3066 > 172.31.158.86.echo: udp 6
```

Вместо ожидания эхо-ответов на посланный эхо-запрос в данном сканировании проверяются ответы хостов на пакет, отправленный на порт echo. Служба echo должна возвращать любые полученные символы. Грамотные системные администраторы не оставляют этот порт открытым, а грамотные администраторы сетей не допускают поступления трафика на этот порт.

Замечание о выявлении попыток сканирования

Пока не еще нет гениального исследователя, который бы предложил другой метод, поэтому сканирование выявляется на основе определенного количества запросов к хостам за установленный промежуток времени. Система обнаружения вторжений может и должна использовать несколько временных промежутков для выявления сканирования. Например, мы наблюдали несколько попыток сканирования, когда за секунду проверяются пять целей. Используя короткий промежуток времени от 1 до 3 секунд, система обнаружения вторжений сможет выявлять высокоскоростные попытки сканирования и реагировать на них практически в реальном времени (через 3–5 секунд после начала сканирования). Подавление таких попыток сканирования в начальной стадии — один из лучших примеров применения автоматических ответных действий. Другой приемлемый промежуток для выявления попыток сканирования составляет от 1 до 5 минут. Это позволяет выявить более медленные, но не менее очевидные попытки. В системе обнаружения вторжений Shadow успешно использовался шаблон выявления сканирования при попытке доступа к пяти-семи различным хостам за один час.

Я написал программу, которую усовершенствовал Билл Ральф (Bill Ralph). Эта программа позволяет выявить попытки скрытого сканирования (с помощью SYN-пакетов) во временном промежутке 24 часа, а также сканирования со средней и низкой скоростью. Так как большинство современных систем обнаружения вторжений поставляются с готовыми базами данных сигнатур, то основное внимание приходится уделять обнаружению попыток сканирования со средней или низкой скоростью. Сканирования обнаруживались при таких низких скоростях, как пять пакетов с одного IP-адреса за 60 дней. Такие низкие скорости сканирования имеют смысл, только если они одновременно осуществляются с нескольких хостов. Мы зарегистрировали попытки сканирования, осуществленные с 2500 совместно работающих хостов, а "сеть" хостов, которые были поражены червем w32.leaves, включала около 30000 скомпрометированных хостов. Таким образом, злоумышленники могут проводить распределенные сканирования с медленными скоростями.

Широковещательные запросы по маскам подсети

Какие из эхо-запросов в приведенной ниже трассировке являются широковещательными? Все! Всем известно о широковещательных адресах 0 и 255, но при определенных условиях широковещательными могут оказаться все пакеты этой трассировки. Что это за условия? Они возникают при разбиении сети на подсети, когда используются нестандартные маски подсетей.

```
02:21:06.700002 pinger> 172.20.64.0: icmp: echo request
02:21:06.714882 pinger> 172.20.64.64: icmp: echo request
02:21:06.715229 pinger> 172.20.64.63: icmp: echo request
```

```
02:21:06.715561 pinger> 172.20.64.127: icmp: echo
request
02:21:06.716021 pinger> 172.20.64.128: icmp: echo request
02:21:06.746119 pinger> 172.20.64.191: icmp: echo request
02:21:06.746487 pinger> 172.20.64.192: icmp: echo request
02:21:06.746845 pinger> 172.20.64.255: icmp: echo request
```

Когда-то я работал в организации, ответственной за распределение сетевых адресов. Один адрес хоста стоил 50 долларов в месяц, а подсеть с маской 255.255.255.0 (другими словами, 256 возможных адресов) — 1000. Этой организации принадлежало адресное пространство сети класса B 172.29.0.0, которое было разбито на подсети. Оказалось, что если купить маршрутизатор и взять в аренду подсеть, то плату за адресное пространство можно снизить. Покажем, как это делается.

Допустим, мы взяли в аренду подсеть `172.29.15.0` за 1000 долларов в месяц. Маска подсети должна быть `255.255.255.0`, что дает нам 256 возможных адресов. Однако `0` и `255` не могут принадлежать хостам, т.е. остается 254 доступных адреса. Если брать отдельно адреса для хостов по 50 долларов в месяц, то сумма составит бы 12700 долларов в месяц, т.е. брать в аренду подсеть куда выгоднее. Однако при наличии собственного маршрутизатора мы можем назначать любую маску подсети на нашей стороне.

Предположим, мы нашли небольшие группы людей, таких же бережливых и независимых, как мы сами. Мы можем использовать два бита нашего адресного пространства для создания четырех подсетей по 6 бит адресного пространства каждая. Поскольку 2^6 равно 64, маска подсети станет `255.255.255.192`, или в шестнадцатеричном формате — `0xffffffc0`. Теперь каждый из нас может иметь собственную подсеть, мы разделим 1000 долларов на четверых и будем платить в месяц чуть больше, чем за 5 адресов отдельных хостов. Отлично, но каким будет широковещательный адрес для маски подсети `255.255.255.192`?

Выполним вычитание: 255 – 192 = 63. Это и есть значение широковещательного адреса "все единицы", а для неопределенного адреса "все нули" соответственно должны использоваться адреса `0` или `64`. Если это непонятно, то ниже дано пояснение.

```
c       0       в шестнадцатеричном формате
1100    0000    в двоичном формате
^^              два старших бита использованы для сетевой части адреса
  ^^    ^^^^    на часть адреса хостов осталось 6 бит
```

Если все 6 бит являются единицами, то 32 + 16 + 8 + 4 + 2 + 1 = 63.

Теперь посмотрим на адреса, использованные в эхо-запросах ICMP нашей трассировки: `0`, `64`, `63`, `127`, `128`, `191`, `192` и `255`.

Могут ли быть адреса 127 и 128 также широковещательными? Конечно, если нам нужно больше подсетей с меньшим количеством хостов в каждой из них. В этом случае можно позаимствовать один бит из части адреса для хостов (узловая часть адреса) и использовать его для сетевой части. Если для сетевой части адреса используется 25 бит (33554432 возможные подсети), при этом в каждой сети для адресов хостов будет использоваться по 7 бит (128 возможных адресов), то маска подсети будет `255.255.255.128`. Каким будет широковещательный адрес? Поскольку 255 – 128 = 127, адрес `127` будет широковещательным адресом "все единицы".

Могут ли быть адреса 191 и 192 также широковещательными? Да, если нужно создать множество подсетей с небольшим числом хостов в каждой из них. В этом случае для сетевой части адреса можно использовать 27 бит (134217728 возможные подсети), в каждой из подсетей для адреса конкретного хоста используется по 5 бит (32 возможных адреса). Маска подсети 255.255.255.64. Широковещательный адрес 255 – 64 = 191.

Конечно, если разрешить поступление в сеть входящих ICMP-сообщений, то хакеру достаточно будет послать только один пакет с запросом маски подсети. Полученный ответ с маской подсети избавит хакера от лишнего угадывания.

Сканирование портов

Рассмотрим более простую трассировку. Это стандартный пример сканирования портов. После того как злоумышленник нашел активный хост, он, наверняка, захочет узнать, какие службы запущены. Это сканирование выполняется по протоколу TCP, и в каждом пакете уменьшается номер порта получателя. Любопытно видеть пропуски в номерах порта отправителя. Возможно, используется компьютер, выполняющий большой объем задач, или одновременно выполняется несколько сканирований. Это пример ускоряющегося сканирования. Сравните время поступления пакетов в начале трассировки и в ее конце. На скорость может влиять множество факторов. Если сравнить эту трассировку с другими зарегистрированными трассировками, связанными с ней, и поток пакетов тоже увеличивается, то можно сделать некоторые предположения относительно хоста-отправителя. Пропущенные номера портов указывают на то, что причиной ускорения сканирования является хост-отправитель, а не промежуточная сеть. Если есть возможность сравнить указанные порты отправителя с портами в других зарегистрированных трассировках, то можно сделать вывод о том, происходит несколько сканирований одновременно, или причина заключается в использовании многопользовательского компьютера, выполняющего большой объем работы.

```
09:52:25.349706 bad.guy.org.1797 > target.mynetwork.com.12: S
09:52:25.375756 bad.guy.org.1798 > target.mynetwork.com.11: S
09:52:26.573678 bad.guy.org.1800 > target.mynetwork.com.10: S
09:52:26.603163 bad.guy.org.1802 > target.mynetwork.com.9: S
09:52:28.639922 bad.guy.org.1804 > target.mynetwork.com.8: S
09:52:28.668172 bad.guy.org.1806 > target.mynetwork.com.7: S
09:52:32.749958 bad.guy.org.1808 > target.mynetwork.com.6: S
09:52:32.772739 bad.guy.org.1809 > target.mynetwork.com.5: S
09:52:32.802331 bad.guy.org.1810 > target.mynetwork.com.4: S
09:52:32.824582 bad.guy.org.1812 > target.mynetwork.com.3: S
09:52:32.850126 bad.guy.org.1814 > target.mynetwork.com.2: S
09:52:32.871856 bad.guy.org.1816 > target.mynetwork.com.1: S
```

Сканирование конкретного порта

Какая служба запущена на TCP-порте 7306? Кто его знает. Как я говорил в приложении А, "Программы атаки и методы сканирования", никогда не вредно поискать информацию с помощью www.google.com, так как все списки портов, которые я видел, были неполными. Эта трассировка была получена в конце 1998 года, и ознаменовала начало нескольких необычных сканирований, целью которых были неизвестные порты. Сканирование было хорошо подготовлено, нет ни одной очевидной сигнатуры.

В первом и последнем пакетах этой трассировки указано имя запрашиваемого хоста, во всех промежуточных пакетах содержится IP-адрес. Это может означать, что хакер "стреляет вслепую", не имея точной схемы сети. Невозможность определить имя хоста часто означает, что этого хоста просто не существует. Обратите внимание на последний пакет трассировки. Номера портов отправителя обычно увеличиваются, а этот номер неожиданно уменьшается на 22. Поскольку начальный порядковый номер тоже имеет меньшее значение (49684211), то, вероятно, пакет был утерян на пути следования и поступил не в порядке отправления.

```
09:54:40.930504 prober.3794 > lula.arpa.net.7306: S 49684444:49684444(0)
↳ win 8192 (DF)
09:54:40.940663 prober.3795 > 192.168.21.20.7306: S 49684454:49684454(0)
↳ win 8192 (DF)
09:54:41.434196 prober.3796 > 192.168.21.21.7306: S 49684945:49684945(0)
↳ win 8192 (DF)
09:54:41.442674 prober.3797 > 192.168.21.22.7306: S 49684955:49684955(0)
↳ win 8192 (DF)
09:54:41.451029 prober.3798 > 192.168.21.23.7306: S 49684965:49684965(0)
↳ win 8192 (DF)
09:54:41.451049 prober.3776 > host.arpa.net.7306: S 49684211:49684211(0)
↳ win 8192 (DF)
```

Сложный сценарий, возможна компрометация

Следующая трассировка состоит из множества отдельных попыток зондирования и атак. Она поделена на пять частей. Попытка доступа к службе portmapper означает, что злоумышленник хочет взломать компьютер или собирает разведывательную информацию. Более того, хакер получает ответ от системы, что очень плохо. Запросы к службе преобразования портов (portmapper) должны быть заблокированы на фильтрующем маршрутизаторе или на брандмауэре, а на самом хосте следует использовать защищенный вариант этой службы. Обратите внимание на то, что атаки направлены против двух хостов (16 и 17). По номерам портов получателей я предположил, что они работают под управлением UNIX. Вполне вероятно, что между этими двумя системами установлены доверительные отношения, и, если одна из будет взломана, то же самое произойдет и со второй.

Затем мы видим попытку доступа к TCP-порту 906, который не является стандартным для какой-либо службы, но атакуемая система все равно отвечает. Это может означать, что на компьютере ранее была установлена вредоносная программа. Однако вместо отправки или получения данных, злоумышленник разрывает соединение. Через два часа он проверяет, по-прежнему ли работает система.

```
00:35:33.944789 prober.839 > 172.20.167.16.sunrpc: udp 56
00:35:33.953524 172.20.167.16.sunrpc > prober.839: udp 28
00:35:33.984029 prober.840 > 172.20.167.17.sunrpc: udp 56
00:35:33.991220 172.20.167.17.sunrpc > prober.840: udp 28

00:35:34.046598 prober.840 > 172.20.167.16.906: S 2450350587:2450350587(0)
↳ win 512
00:35:34.051510 172.20.167.16.906 > prober.840: S 1996992000:1996992000(0)
↳ ack 2450350588 win 32768 (DF)

00:35:34.083949 prober.843 > 172.20.167.17.sunrpc: udp 56
00:35:34.089272 172.20.167.17.sunrpc > prober.843: udp 28
```

```
00:35:34.279472 prober.840 > 172.20.167.16.906: F 117:117(0) ack 69 win 32120
00:35:34.284670 172.20.167.16.906 > prober.840: F 69:69(0) ack 118 win 32768
(DF)

02:40:43.977118 prober > 172.20.167.16: icmp: echo request
02:40:43.985138 172.20.167.16 > prober: icmp: echo reply
```

Эта трассировка очень важна, и, как аналитик, я бы посоветовал провести ее внимательный анализ. Обсудим ответные действия. Необходимо выполнить резервное копирование, провести исследование случившегося, предотвратить распространение атаки и удалить файлы хакера. Если бы это была одна из защищаемых мной систем, я бы посоветовал пользователю следующее:

- уберите руки с клавиатуры;
- немедленно выдерните сетевой кабель;
- после появления специалиста по восстановлению выполните резервное копирование;
- используйте магнитную ленту как доказательство проведенной атаки.

Использование порта 906 требует дальнейшего расследования. Проще всего принести с собой переносной компьютер и небольшой концентратор. С помощью концентратора соедините скомпрометированную систему с вашим переносным компьютером. Загрузите собственные копии системных утилит (например, `ls`, `ps`, `netstat`) на проверяемую систему либо перепишите их с заранее созданного компакт-диска. С переносного компьютера организуйте `telnet`-соединение с портом 906 проверяемой системы. Запустите собственные версии утилит `netstat` и `ps`, чтобы выявить активные процессы. Кроме того, проверьте установленные доверительные отношения для скомпрометированного компьютера по информации файлов `.rhosts` и `/etc/hosts.equiv`.

Альтернативный метод

Невозможно рассказать об устранении взлома в двух словах. Документ под названием *Incident Handling Step by Step* является результатом совместных усилий более чем 90 специалистов по обеспечению защиты информации. Он доступен на сайте `www.sans.org`. Одним из лучших методов при выходе системы из строя или необходимости ее перезагрузки является использование загрузочных компакт-дисков. Затем можно смонтировать системный диск как диск данных. По возможности старайтесь сохранять информацию жесткого диска в неизменном виде, чтобы ее можно было использовать в качестве доказательства атаки.

Затем обратитесь к владельцам компьютера и узнайте, когда осуществлялось последнее резервное копирование его данных. Хмыкните и неодобрительно покачайте головой, когда услышите ответ "никогда" или "два года назад". Посмотрите пристально в глаза владельцу и спросите, есть ли данные, которые абсолютно точно необходимо сохранить. Создайте резервные копии только файлов с данными, переформатируйте жесткий диск и посоветуйте установить все последние заплаты перед тем, как снова подключать систему к сети. Подключите свой переносной компьютер к локальной сети. Просканируйте хосты локальной сети и определите, не отвечают ли они на запросы на порт 906. Продолжайте проверку, пока не будут уничтожены все последствия атаки.

Мы рассказывали об атаках Loki и средствах для проведения распределенных атак отказа в обслуживании наподобие Trinoo, в которых эхо-запросы и эхо-ответы используются для других целей. Возможно, следует внимательнее изучить содержимое ping-пакетов.

Сканирование случайных портов

Данное сканирование является одним из самых высокоскоростных. Это еще один пример бессмысленного сканирования портов. Здесь нет явной сигнатуры, цель сканирования неизвестна.

```
11:48:42.413036 prober.18985 > host.arpa.net.794: S
1240987936:1240987936(0)
win 512
11:48:42.415953 prober.18987 > host.arpa.net.248:  S
909993377:909993377(0)
win 512
11:48:42.416116 prober.19031 > host.arpa.net.386: S
1712430684:1712430684(0)
win 512
11:48:42.416279 prober.19032 > host.arpa.net.828: S 323265067:323265067(0)
win 512
11:48:42.416443 prober.19033 > host.arpa.net.652: S
1333164003:1333164003(0)
win 512
11:48:42.556849 prober.19149 > host.arpa.net.145: S
2112498338:2112498338(0)
win 512
11:48:42.560124 prober.19150 > host.arpa.net.228: S
1832011492:1832011492(0)
win 512
11:48:42.560824 prober.19151 > host.arpa.net.840: S
3231869397:3231869397(0)
win 512
11:48:42.561313 prober.19152 > host.arpa.net.1003: S
2435718521:2435718521(0)
win 512
11:48:42.561437 prober.19153 > host.arpa.net.6: S 2632531476:2632531476(0)
win 512
11:48:42.561599 prober.19165 > host.arpa.net.280: S
2799050175:2799050175(0)
win 512
11:48:42.563074 prober.19166 > host.arpa.net.845: S
2065507088:2065507088(0)
win 512
11:48:42.563115 prober.19226 > host.arpa.net.653: S
1198658558:1198658558(0)
win 512
11:48:42.563238 prober.19227 > host.arpa.net.444: S
1090444266:1090444266(0)
win 512
11:48:42.565041 prober.19274 > host.arpa.net.907: S
2414364472:2414364472(0)
win 512
```

Итак, нам не известна цель сканирования, и это настораживает. Что в этом случае должен делать аналитик? Мы видим, что это сканирование выполняется

быстро, и предсказать номер порта отправителя невозможно. Зачем же кому-то сканировать так много неизвестных портов? Я не уверен, получим ли мы вообще ответ на этот вопрос. За последние несколько лет было много случаев довольно странных сканирований. Мое единственное предположение в том, что кто-то использует программы `nmap`, `hping2`, `isic`, `packetx` или другие подобные средства для проведения пробных сканирований, которые не имеют определенной цели (возможно, используются ложные IP-адреса отправителя). Это ответ на вопрос "как", но не на вопрос "зачем".

Еще одно предположение. Это делается, чтобы свести с ума аналитиков сетевого трафика, проверить, о чем они уведомляют, а о чем — нет, а может быть, чтобы услышать в новостях CNN о новой атаке хакеров. Согласен, что это маловероятно, но у меня нет лучших предположений. Как реагировать аналитику на это и другие подобные сканирования случайных портов? Я советую уведомлять об этих попытках, потому что никогда не знаешь, какая информация окажется полезной группам CIRT. Если настройки брандмауэра не позволяют проходить никаким пакетам, кроме специально разрешенных, и ни один из хостов локальной сети не отвечает, то лучше всего создать каталог под названием "Сканирования_с_Марса" и сохранить там выявленные трассировки.

Использование баз данных

Я рекомендую аналитикам сетевого трафика придерживаться принципа "сделал и забыл". При обнаружении проблемы следует уведомить о ней и продолжать свою работу. Когда мы впервые приступили к выявлению взломов для довольно крупной сети (пять узлов, около 12000 хостов), то аналитику приходилось вручную сверять показания датчиков о порте и IP-адресе отправителя подозрительных пакетов, порте и IP-адресе получателя и т.д. Если нужно провести сравнение по значению поля TTL или узнать принцип присвоения порядковых номеров, то этот процесс может продлиться несколько дней.

Жизнь слишком коротка, чтобы тратить ее на подобные глупости. После выявления шаблона атаки и уведомления о ней сравнение можно осуществлять, используя базу данных. Следующий отчет был получен с помощью военной системы сравнительного анализа Dark Shadow. Она основана на использовании базы данных Oracle. Когда аналитик обнаруживает попытку взлома и сообщает о ней, система Dark Shadow сопоставляет признаки атаки с информацией базы данных, полученной от *X* датчиков за *Y* месяцев. При выявлении совпадения аналитику отправляется соответствующий отчет. Таким образом, аналитик может не помнить о всех ранее выявленных атаках.

Обратите внимание на то, что, судя по изменению номеров порта отправителя (на порт получателя 111 пакеты поступают с портов отправителя 617–1023, а на порт получателя 25 — с портов отправителя 2294–29419), на сканирующем хосте запущено два процесса: один для сканирования службы электронной почты, а второй — службы `portmapper`. Скорее всего, для запуска этих процессов использован единый сценарий, и выполняется чтение из файла, в котором перечислены IP-адреса атакуемых хостов. По всей вероятности, одновременно сканируется большое количество хостов.

```
06/04/98 03:20:25 scanner    622      172.20.1.41     111  t
06/04/98 04:02:35 scanner  21091       172.20.1.1      25  t
06/04/98 04:02:36 scanner    890       172.20.1.1     111  t
06/04/98 04:06:04 scanner  21242    172.20.10.114      25  t
06/04/98 04:09:15 scanner    617    172.20.10.114     111  t

06/04/98 07:24:47 scanner   2295   192.168.229.18      25  t
06/04/98 07:28:06 scanner   1017   192.168.229.18     111  t
06/04/98 07:28:21 scanner   2333     172.20.1.41      25  t
06/04/98 07:31:40 scanner    729      172.20.1.41     111  t
06/04/98 12:46:21 scanner  20553    172.20.48.157      25  t
06/04/98 12:49:40 scanner   1023    172.20.48.157     111  t
06/04/98 16:05:22 scanner  29276       172.20.1.1      25  t
06/04/98 16:08:33 scanner    803       172.20.1.1     111  t
06/04/98 16:08:52 scanner  29419    172.20.10.114      25  t
06/04/98 16:08:53 scanner    900    172.20.10.114     111  t
```

SNMP/ICMP

Еще до выхода в начале 2002 года набора средств RPOTOS и последовавшего за этим использования программ атак протокол SNMP (Simple Network Management Protocol – упрощенный протокол сетевого управления) позволял хакерам получать немало информации о чужих хостах и конфигурации сетей. Согласно документации RFC для протокола SNMP выделен TCP/UDP-порт 161. Я никогда не встречался с примерами использования TCP-версии протокола SNMP, но для безопасности оба эти порта следует заблокировать для входящих пакетов из Internet.

Просто удивительно, как много устройств, например, мини-концентраторы, X-терминалы и принтеры, поддерживают работу по SNMP. По умолчанию доступ к эти устройствам предоставляется с помощью хорошо известного пароля (строки доступа), обычно слова "`public`". Во многих организациях, в которых "заботятся" о безопасности, этот пароль изменяют на один из следующих вариантов:

- `private`;
- `internal`;
- название организации.

Безусловно, выбор одного из этих слов в качестве строки доступа SNMP не будет удачным. Нужно подобрать более сложный пароль.

В следующей трассировке обратите внимание на широковещательные запросы по SNMP и ICMP. Это очень эффективный метод составления схемы сети, так как злоумышленнику не приходится отправлять большее количество пакетов, а потенциально он может получить значительный объем информации.

```
17:31:33.49 prober.1030 > 192.168.2.255.161: GetNextRequest(11)[|snmp]
17:31:33.73 prober.1030 > 255.255.255.255.161: GetNextRequest(11)[|snmp]
17:31:33.73 prober > 255.255.255.255: icmp: echo request
...
17:43:17.32 prober > 192.168.1.255: icmp: echo request
17:43:17.32 prober.1030 > 192.168.1.255.161: GetNextRequest(11)[|snmp]
```

Атака по FTP

Рассмотрим еще одну трассировку, для которой применяется сравнение с информацией базы данных. В данном случае аналитик хочет найти сведения отно-

сительно исходящих соединений через TCP-порт 20 (ftp-data) без использования порта FTP (TCP-порт 21). Подобное соединение может являться результатом сканирования портов с помощью FTP-сервера. Такая атака стала известна как *возврат по FTP* (*FTP bounce*). Преимущество этой атаки заключается в обеспечении анонимности злоумышленника (ведь сканирование выполняется от имени FTP-сервера). Это напоминает использование открытого proxy-сервера, только порт отправителя будет указан как TCP-порт 20. Чтобы добиться своей цели, хакеру достаточно подключиться к уязвимому FTP-серверу в качестве анонимного пользователя и проверить любые порты атакуемого хоста. Как правило, эта атака не несет серьезной угрозы, за исключением ситуации, когда у организации с FTP-сервером установлены доверительные отношения. Тогда хакер сможет просканировать все хосты этой организации. Возврат по FTP рассматривается в одной из рекомендаций CERT, которую можно найти по адресу www.cert.org/ftp/cert_advisories/CA-97.27.FTP_bounce.

В некоторых реализациях демонов службы FTP команда port может быть использована для установки соединения с указанным хакером портом на компьютере, с которым он бы не мог установить соединение напрямую. Эта проблема живо обсуждалась на протяжении нескольких лет, и некоторые поставщики исправили существующие ошибки.

Выявив трафик следующей трассировки, мы проверили хост prober и обнаружили, что он является анонимным FTP-сервером. При этом команду port можно было использовать как угодно. Интересно, что эта трассировка была зарегистрирована задолго до того, как попытки доступа к неизвестным портам стали популярной забавой. Ниже показаны все запросы на соединение, отправленные хостом prober на адрес защищенной сети (172.20.152).

```
дата     время      IP-адрес отпр. Порт отпр. IP-адрес получ. порт получ.
04/27/98 10:17:31   prober         20          172.20.152.2    3062 t
04/27/98 10:27:32   prober         20          172.20.152.2    4466 t
05/06/98 06:34:22   prober         20          172.20.152.2    1363 t
05/06/98 09:12:15   prober         20          172.20.152.2    4814 t
05/06/98 09:15:07   prober         20          172.20.152.2    1183 t
05/06/98 10:11:30   prober         20          172.20.152.2    1544 t
```

Использование NetBIOS

В этом разделе будет рассмотрено несколько трассировок, созданных при атаках на системы Windows. Службами NetBIOS используются TCP- и UDP-порты 135–139. Конечно, NetBIOS может применяться не только на Windows-системах (вспомним хотя бы Samba), но, чаще всего, трафик служб NetBIOS предназначается именно Windows-системам.

Пакеты от Web-сервера

Одна из характерных черт служб NetBIOS заключается в том, что трафик, поступающий на UDP-порт 137, часто отправляется с Web-сервера какого-то узла. Например, если по электронной почте отправить сообщение на узел, где запущена программа Microsoft Exchange, то часто на порт 137 приходит ответный пакет. Следующая трассировка появилась в результате нашего исследования причин поступления пакетов на порт 137. Чтобы выяснить причину, мы проверили весь

трафик для хоста jellypc и обнаружили попытку доступа к Web-серверу. То же самое было сделано для хоста jampc, и результат не отличался. Просмотр всего трафика для интересующего хоста — очень ценная возможность для проведения анализа. Если ваша система обнаружения вторжений не поддерживает такой возможности, стоит подумать о ее замене!

Замечание по поводу общественной безопасности

Хотя этот раздел в основном посвящен NetBIOS, не могу не упомянуть о том, что в Internet существуют прямо-таки вредоносные Web-серверы. Подобные Web-серверы собирают информацию о связывающихся с ними компьютерах, включая сведения об операционных системах и версиях броузеров. Если на узле не поддерживается служба NAT (Network Address Translation — трансляция сетевых адресов), то Web-сервер получит IP-адрес обратившегося к нему хоста. Часто также извлекается адрес электронной почты Web-клиента. Некоторые сайты открывают обратное соединение с клиентом и проводят то, что мы называем анализом стека протоколов TCP/IP (здесь мы даже не затрагиваем тему файлов cookie).

При получении пакетов от хоста jellypc Web-серверу оказалось недостаточно информации HTTP-заголовков. Поэтому другой компьютер из той же подсети, что и Web-сервер, обращается к хостам, которые посетили этот Web-сервер, с целью собрать информацию, предоставляемую службой NetBIOS Name Service.

```
12/02/97 08:27:18   jellypc.arpa.net 1112 -> www.com    http
12/02/97 08:27:19         0 bill.com        137 -> jellypc.arpa.net     137
12/02/97 17:06:03   jampc.arpa.net 2360 -> www.com     http
12/02/97 17:08:10         0 bill.com        137 -> jampc.arpa.net     137
```

Я долго общался по телефону с одним из администраторов сети, в которой был установлен этот Web-сервер. Оказалось, что они используют специальную коммерческую программу для маркетингового исследования, которая предназначена для составления базы данных относительно предпочтений своих клиентов.

С помощью команды nbtstat, выполненной на компьютере под управлением Windows NT, можно узнать информацию об этой Windows-системе. Если на Windows-системе запущена служба NetBIOS, она обязательно ответит на команду nbtstat. Посмотрите на следующий пример.

```
C:\>nbtstat -a goo

NetBIOS Remote Machine Name Table

   Name               Type           Status
---------------------------------------------
Registered Registered Registered
MAC Address = 00-60-97-C9-35-53

GOO            <20>  UNIQUE
600            <00>  UNIQUE
KD2            <00>  GROUP
KD2            <1C>  GROUP
KD2            <1B>  UNIQUE
GOO            <03>  UNIQUE
SRN0RTH        <03>  UNIQUE
INet-Services  <1C>  GROUP
IS~GOO         <00>  UNIQUE
KD2            <1E>  GROUP
KD2            <1D>  UNIQUE
..__MSBROWSE__.<01>  GROUP
```

Итак, мы узнали NetBIOS-имя моего компьютера Goo, а также имя моей рабочей группы (KD2). Моим именем пользователя для регистрации на этом компью-

тере является snorth. Кроме того, определено, что у меня есть файл cookie для главного броузера (master browser).

Возможно, что этот пример с приложением для сбора информации не произвел на вас впечатления. Однако с помощью запросов nbstat мне удалось определить структуру сетей всей организации, а также большинство имен пользователей.

Нулевой сеанс

Нулевые сеансы (null session) описывались аналогично использованию службы finger. По существу, нулевые сеансы – это вход в систему от имени пользователя nobody. Этот пользователь не имеет никаких полномочий на изменение данных, но может просматривать всю системную информацию. Для этого используется, например, следующая команда.

```
net use \\172.20.244.164\IPC$ "" /USER:""
```

Эта команда позволяет получить буквально целые страницы информации, часть из которой представлена ниже.

```
2/18/98 1:39 AM - Jsmith - \\192.168.4.22
UserName

Administrator
   Groups,Administrators (Local,
Members can fully administer the computer/domain)
   AccountType,User
   HomeDrive
   HomeDir
   PswdCanBeChanged,Yes
   PswdLastSetTime,Never
   PswdRequired,Yes
   PswdExpires,No
   AcctDisabled,No
   AcctLockedOut,No
   AcctExpiresTime,Never
   LastLogonTime,11/20/98 3:24 PM
   LastLogonServer,192.168.4.22
Sid,S-1-5-21-706837240-361788889-398547282-500
```

Нулевые сеансы можно заблокировать в Windows 2000, а также в Windows XP Professional. Для этого на Панели управления (Control Panel) выберите элемент Администрирование (Administrative Tools) и пиктограмму Локальная политика безопасности (Local Security Policy).

Скрытые атаки

Первый раз термин *скрытые* (stealth) был использован в статье Криса Клауса (Chris Klaus) "Stealth Scanning – Bypassing Firewalls/SATAN Detectors" ("Скрытое сканирование в обход брандмауэров и программы SATAN". Теперь подобные попытки сканирования часто называют *сканированием с незавершенным открытием TCP-сеанса* (half open). Существуют различные варианты такого сканирования, в этом разделе будут рассмотрены наиболее типичные из них. Подобное сканирование не так уж сложно обнаружить. В настоящее время некоторые авторы стали использовать термин *скрытое сканирование* для сканирований с помощью нуль-

пакетов (в которых не установлено никаких флагов). Я считаю, что этот термин допустимо использовать только для медленных и незаметных попыток сканирования или для сканирований с помощью нескольких распределенных компьютеров. С течением времени статические фильтры пакетов используются все реже и реже, и сканирования с незавершенным открытием TCP-сеанса уходят в прошлое. Такие сканирования не стоит называть скрытыми, так как они просто очевидны. На Web-сайте анализатора пакетов Snort (`www.snort.org`) приведены эффективные правила для обнаружения попыток такого сканирования.

Наступают времена применения усовершенствованных методов сканирования. Злоумышленники, способные создавать и компилировать программы, теперь используют средства, которые считаются признаком высокой квалификации хакера. Три года назад это была программа `jackal`, на рубеже нового тысячелетия – `hping` и `nmap`, а теперь – программы для распределенного сканирования.

В книге Роберта Зиглера (Robert Ziegler) *Inside Firewalls* я отмечал, что не перестаю поражаться защите, обеспечиваемой с помощью службы NAT (Network Address Translation –трансляция сетевых адресов). На моем переносном компьютере запущена виртуальная машина VMWare и под Windows XP работает Linux 7.2. Мои самые важные файлы хранятся в Linux 7.2, в которой используется служба NAT. Таким образом, если хакеру удастся проникнуть сквозь защиту по периметру, которая также включает в себя NAT, и взломать мой переносной компьютер под управлением Windows XP, ему придется разобраться с еще одной службой NAT. Поскольку современные брандмауэры доступны всего за $300, можно использовать большое количество NAT в своей организации, что помешает провести большинство из рассматриваемых сканирований. Кроме того, невозможно обойти грамотно настроенный и оснащенный proxy-сервером брандмауэр (хотя этот вопрос мы еще обсудим). Никакая вредоносная программа не способна обмануть квалифицированного аналитика, который использует систему обнаружения взлома, сохраняет весь передаваемый трафик и обновляет базу данных. При выборе более легкого пути могут возникнуть неприятности.

> Перед началом изучения некоторых методов скрытого сканирования давайте процитируем выдержки из описания исходного кода программы атаки `jackal.c`. "Программа сканирования Jackal-Stealth с помощью отправки SYN-пакетов (иногда добавляются дополнительные флаги, например FIN) позволяет выполнять сканирование компьютеров, защищенных брандмауэром. Сканирование не должно обнаруживаться, так как полная процедура установки TCP-соединения не выполняется. Разработчики: Halflife, Jeff (Phiji) Fay, Abdullah Marafhie. Проверял: Уолтер Копецки (Walter Kopecky). Результаты тестирования: некоторые брандмауэры пропускают пакеты с установленными флагами SYN и FIN. Не было ни одного случая регистрации соединения..."

Замечательная идея! Если фильтрующий маршрутизатор блокирует SYN-пакеты, его можно обойти, установив в пакете два флага: SYN и FIN. Однако утверждение, что действия программы `jackal` не были ни разу выявлены, конечно, не соответствуют действительности. В приложении А, "Программы атаки и методы сканирования", приведены трассировки трафика SYN/FIN-пакетов, выявленные с помощью системы Shadow. Хорошие системы обнаружения вторжений регистрируют и анализируют все пакеты, созданные с помощью программы `jackal` (или `hping`, или `nmap`). В действительности не составляет труда обнаружить SYN/FIN-пакеты.

Явные методы скрытого сканирования

Известны два метода явного сканирования: использование SYN/FIN-пакетов и сканирование с помощью FIN-пакетов. В обоих случаях активные хосты должны возвращать пакеты с установленным флагом RESET. Кроме того, должно возвращаться ICMP-сообщение при недостижимости хоста. Эти методы эффективнее возврата по FTP, но более очевидны.

Сканирование с помощью FIN-пакетов

Я никогда не встречался с реальными случаями сканирования с помощью FIN-пакетов и не хочу ничего моделировать. При этом сканировании отправляется большое количество FIN-пакетов, хотя никакого TCP-соединения не было установлено. Мы уже обсуждали использование баз данных при изучении атак по FTP. Хорошая система обнаружения вторжений должна выявлять подобный трафик, состоящий из FIN-пакетов, для несуществующих соединений. Лично я видел только, как для сканирования использовались SYN-пакеты, и как они проникли сквозь брандмауэр Check Point.

Метод "от обратного"

Составление схем сетей по методу "от обратного" позволяет получить списки недостижимых сетей или хостов, а затем использовать эту информацию для предположительного представления о реально работающих компьютерах. Рассмотрим пример использования запросов службы DNS для этой цели. Если в локальной сети невозможно применить службу NAT и должны использоваться общедоступные IP-адреса, то необходимо гарантировать, что будут блокироваться все исходящие ICMP-сообщения о недостижимости. Это не остановит все попытки сканирования "от обратного", но сделает бесполезными большинство из них. При изучении следующей трассировки помните о том, что основную опасность несут ответы от хоста `router.mynet.net`.

```
02:58:05.490 stealth.mappem.com.25984 > 172.30.69.23.2271:
↳ R 0:0(0) ack 674719802 win 0
02:59:11.208 stealth.mappem.com.50620 > 172.16.7.158.1050:
↳ R 0:0(0) ack 674719802 win 0
02:59:20.670 stealth.mappem.com.19801 > 192.168.184.174.1478:
↳ R 0:0(0) ack 674719802 win 0
02:59:31.056 stealth.mappem.com.7960 > 192.168.242.139.1728:
↳ R 0:0(0) ack 674719802 win 0
02:59:42.792 stealth.mappem.com.16106 > 172.16.102.105.1008:
↳ R 0:0(0) ack 674719802 win 0
03:00:50.308 stealth.mappem.com.8986 > 172.16.98.61.1456:
↳ R 0:0(0) ack 674719802 win 0
03:00:58.939 stealth.mappem.com.35124 > 192.168.182.171.1626:
↳ R 0:0(0) ack 674719802 win 0
03:00:58.940 router.mynet.net > stealth.mappem.com:
↳ icmp: host 192.168.182.171 unreachable
```

Ответы на DNS-запросы

Ниже представлен еще один вариант составления схемы сети "от обратного". Сканирующий компьютер отправляет ответы на DNS-запросы, которые к нему

никогда не поступали. Цель заключается в получении ответных ICMP-сообщений о недостижимости. Как указывалось ранее, такой метод позволяет избежать обнаружения сканирования. Программы выявления сканирования обнаружат подобную попытку, если злоумышленник будет слишком "энергичен" и захочет быстро прозондировать большое количество хостов. Также эти попытки выявляются при ретроспективном анализе или с помощью поиска в базах данных нарушений в работе приложений.

```
05:55:36.515566 stealth.com.domain > 172.29.63.63.20479: udp
06:46:18.542999 stealth.com.domain > 192.168.160.240.12793: udp
07:36:32.713298 stealth.com.domain > 172.29.185.48.54358: udp
07:57:01.634613 stealth.com.domain > 254.242.221.165.13043: udp
09:55:28.728984 stealth.com.domain > 192.168.203.163.15253: udp
10:38:53.862779 stealth.com.domain > 192.168.126.131.39915: udp
10:40:37.513176 stealth.com.domain > 192.168.151.126.19038: udp
10:44:28.462431 stealth.com.domain > 172.29.96.220.8479: udp
11:35:40.489103 stealth.com.domain > 192.168.7.246.44451: udp

11:35:40.489103 stealth.com.domain > 192.168.7.246.44451: udp
11:35:40.489523 router.mynet.net > stealth.com:
↳ icmp: host 192.168.7.246 unreachable
```

Поскольку подмена IP-адресов является, по существу, неотъемлемой частью атак отказа в обслуживании, то может возникнуть вопрос: "А не являются ли данные пакеты результатом подмены IP-адресов сетей `172.29`, `192.168` и т.д. и ответа `stealth.com` на полученные запросы?". Дело в том, что это не похоже на стандартные DNS-ответы, и нет никаких признаков того, что были отправлены какие-либо DNS-запросы.

При более углубленном анализе можно проверить, действительно ли хост `stealth.com` является DNS-сервером. Для этого воспользуйтесь командой `nslookup`, измените имя используемого DNS-сервера на `stealth.com` и попытайтесь определить имя хоста по какому-либо адресу. Если это удастся, значит, `stealth.com` действительно DNS-сервер и нет никаких загадок (к сожалению, по крайней мере на UNIX-хостах утилиту `nslookup` все чаще заменяет менее точная утилита `dig`). Если ответа не будет, значит, есть шанс, что это никакой не DNS-сервер, а хост злоумышленника (также возможно, что у вас просто нет доступа к этому DNS-серверу).

Следующая трассировка во многом напоминает предыдущую, так как в обоих случаях портом отправителя является TCP-порт 53 (`domain`). Однако в данном случае пакеты поступают от различных отправителей. Есть предположения о том, что происходит?

```
11:19:30.885069 host1.corecomm.net.53 > myhost1.com.21: S 7936:7936(0)
↳ win 1024
11:17:29.375069 host1.corecomm.net.53 > myhost1.com.139: S 7936:7936(0)
↳ win 1024
11:15:32.115069 host1.corecomm.net.53 > myhost1.com.23: S 7936:7936(0)
↳ win 1024
11:11:17.485069 host1.corecomm.net.53 > myhost1.com.43981: S 7936:7936(0)
↳ win 1024
11:09:12.945069 host1.corecomm.net.53 > myhost1.com.880: S 7936:7936(0)
↳ win 1024
12:01:05.060000 host70.corecomm.net.53 > pc112.com.880: S 1738:1738(0)
↳ win 1024
12:03:24.820000 host70.corecomm.net.53 > pc112.com.139: S 1738:1738(0)
↳ win 1024
```

```
12:06:12.620000 host70.corecomm.net.53 > pc112.com.21: S 1738:1738(0)
↳ win 1024
12:09:09.940000 host70.corecomm.net.53 > pc112.com.43981: S 1738:1738(0)
↳ win 1024
12:09:57.960000 host70.corecomm.net.53 > pc112.com.23: S 1738:1738(0)
↳ win 1024
```

Это похоже на сканирование, например, хостов `myhost1.com`, `pc112.com` и многих других хостов, которые не были показаны в этом фрагменте трассировки. Сканируются порты стандартных служб, а именно 21 (FTP), 23 (`telnet`) и 139 (NetBIOS Session Manager). Кроме того, проверяется несколько необычных номеров портов, например 43981 и 880. Можно высказывать любые предположения относительно попыток доступа к этим портам, но в данном случае основное внимание следует обратить на порт отправителя.

Трафик, поступающий с TCP-порта 53, не блокируется для многих сетей, так как этот порт используется для "больших" DNS-ответов. В главе 6, "DNS", указывалось, что для отправки DNS-ответов размером более 512 байт (вместо UDP протокола) DNS-сервер использует TCP-порт 53. Предусмотрительные системные администраторы позволяют проходить таким пакетам только тогда, когда они поступают от хоста исходной сети и только, если номер порта получателя превышает 1023 (временный порт), что обязательно для "объемных" DNS-ответов.

В предыдущем сканировании ситуация другая: злоумышленник отправляет пакеты с указанным портом отправителя 53, надеясь на то, что устройство фильтрации пакетов разрешает прохождение такого трафика без каких-либо проверок. Таким образом, часто удается обойти обычно надежное устройство фильтрации пакетов.

Интересно отметить, что в приведенном выше отчете о сканировании порядковые номера TCP-сегментов повторяются при проверке портов каждого нового хоста. Обычно они должны изменяться для каждого создаваемого TCP-сегмента. Еще один интересный момент, для обнаружения которого нужно проанализировать больше пакетов, чем приведено в этом фрагменте сканирования, в методе назначения порядковых номеров. В нашем фрагменте использованы только два порядковых номера: 7936 и 1738. Учитывая, что для хранения значений порядкового номера TCP-сегмента предназначено 32-битовое поле, эти значения довольно малы, что необычно. Мы выбрали все порядковые номера, использованные в данном сканировании, перевели их в шестнадцатеричный формат и обнаружили отгадку. Старшие 16 бит порядкового номера TCP-сегмента всегда были нулями. Это подтверждает факт манипуляции значениями порядковых номеров и становится сигнатурой для обнаружения данного сканирования.

Фрагменты, только фрагменты

Рассмотрим последний пример составления схемы сети по принципу "от обратного". Как известно, только в первом фрагменте содержится информация о протоколе. Это используют злоумышленники для проникновения в сети, защищаемые устаревшими брандмауэрами и фильтрующими маршрутизаторами. Некоторые брандмауэры ошибочно принимают фрагменты за часть трафика, который уже успешно прошел проверку. Нет необходимости говорить о том, что большинство поставщиков уже устранили подобный недостаток.

Однако в данном случае злоумышленник не стремится обойти брандмауэр. Повторим еще раз: если хоста не существует, маршрутизатор отправляет сообщение

о недостижимости этого хоста. Хакер может составить список несуществующих хостов и предположить, что все остальные хосты существуют. Вот почему метод называют "от обратного".

```
18:32:21.050033 PROBER > 192.168.5.71: (frag 9019:480@552)
18:32:21.109287 PROBER > 192.168.5.72: (frag 9275:480@552)
18:32:21.178342 PROBER > 192.168.5.73: (frag 9531:480@552)
18:32:21.295332 PROBER > 192.168.5.74: (frag 9787:480@552)
18:32:21.344322 PROBER > 192.168.5.75: @(frag 10299:480@552)
18:32:21.384284 PROBER > 192.168.5.76: (frag 10555:480@552)
18:32:21.431136 PROBER > 192.168.5.77: (frag 11067:480@552)
18:32:21.478246 PROBER > 192.168.5.78: (frag 11579:480@552)
18:32:21.522631 PROBER > 192.168.5.79: (frag 11835:480@552)
```

Оценка времени ответа

Не так давно мы получили большой объем трафика, поступающего из самых разных мест и предназначенного DNS-серверам, но это не были DNS-запросы, и не было очевидным какое-либо вредоносное содержимое пакетов. На самом деле какие-то компании создали программы, которые пытались оценить наилучшее время ответа на Web-запросы. Исследование показало, что большинство пользователей терпимо относятся ко времени задержки ответа продолжительностью до восьми секунд. Если это время будет большим, они обратятся к сайту конкурентов. Значит, от времени ответа зависит прибыльность и успех электронной коммерции, а так как необходимость – хороший стимул для изобретений, то для решения этой задачи были созданы специальные программы. В этом разделе мы обсудим шаблоны действий программы 3DNS.

Один из методов работы этой программы заключается в сопоставлении хоста пользователя с уполномоченным DNS-сервером для хоста этого пользователя. Предпринимается попытка определить время ответа DNS-сервера на запрос. При этом предполагается, что уполномоченный DNS-сервер и хост пользователя географически близки, что не всегда соответствует действительности. Почему просто не узнать расстояние до компьютера пользователя? Действительно это кажется более логичным, но многие сети хорошо защищены и доступ к хостам пользователей часто невозможен. Было решено, что вероятность доступа к DNS-серверу выше, что иногда правда, иногда – нет.

Трафик, генерируемый программой 3DNS, вызывает многочисленные жалобы аналитиков, являясь причиной тревог со стороны систем обнаружения вторжений. Громкое недовольство легко пояснить тем, что трафик предназначен "неприкасаемым" DNS-серверам, обслуживающим все хосты организации. Многие не поняли причин происходящего и решили, что задействована какая-то новая атака. В конце концов, это несанкционированный сбор информации, независимо от того, используется ли она на благо конечных пользователей.

Давайте рассмотрим несколько сигнатур, связанных с трафиком этого типа. Помните о том, что очень многие сайты используют это программное обеспечение, поэтому нет ничего удивительного в различии IP-адресов отправителя. Так как в уникальных сигнатурах обнаружения этого трафика были учтены различные IP-адреса отправителей, по ошибке его принимали за какой-то вид скоординированной атаки. Как вы убедитесь, это не так.

Эхо-запросы

Приведенный ниже отчет TCPdump демонстрирует действия, выполняемые для оценки времени ответа чужого DNS-сервера. Отправляется эхо-запрос, а время ответа оценивается по факту возвращения эхо-ответа.

```
10:25:44.070000 216.32.68.13 > mydns.com: icmp: echo request
10:25:44.070000 216.32.68.13 > mydns.com: icmp: echo request
10:25:44.070000 216.32.68.13 > mydns.com: icmp: echo request
10:30:01.530000 216.32.68.13 > mydns.com: icmp: echo request
10:30:01.530000 216.32.68.13 > mydns.com: icmp: echo request
10:30:01.550000 216.32.68.13 > mydns.com: icmp: echo request
10:30:25.660000 209.67.29.8 > mydns.com: icmp: echo request
10:30:25.660000 209.67.29.8 > mydns.com: icmp: echo request
10:30:25.670000 209.67.29.8 > mydns.com: icmp: echo request
10:32:12.520000 209.67.78.200 > mydns.com: icmp: echo request
```

Но, как известно, эхо-запросы ICMP часто блокируются, ведь с их помощью можно составлять схемы сетей. Тогда злоумышленник или даже провайдер с помощью средства типа 3DNS может сконцентрировать все внимание на DNS-сервере.

Действительные DNS-запросы

Если DNS-сервер не ответил на эхо-запрос ICMP, а сервер, использующий 3DNS, настроен на дальнейшие попытки установления связи с чужим DNS-сервером, то отправляется трафик другого типа, как показано ниже.

```
216.32.68.11.3200 > mydns.com.53: 0 [0q] Type0 (Class 0)? . (36)
mydns.com.53 > 216.32.68.11.3200: 0 FormErr [0q] 0/0/0 (12) DF
216.32.68.11.3201 > mydns.com.53: 256 [0q] Type0 (Class 0)? . (36)
mydns.com.53 > 216.32.68.11.3201: 0 FormErr [0q] 0/0/0 (12) OF
216.32.68.11.3202 > mydns.com.53: 512 [0q] Type0 (Class 0)? . (36)
mydns.com.53 > 216.32.68.11.3202: 0 FormErr [0q] 0/0/0 (12) DF
```

На самом деле DNS-запрос не посылается, просто на UDP-порт 53 отправляется "пустое" DNS-сообщение, содержащее одни нули. Программа TCPdump позволяет провести небольшое исследование DNS-сообщений и при выявлении пустых полей сообщает об этом. Например, строка `0q` означает, что в DNS-сообщении нет полезных запросов. Анализатор пакетов TCPdump уведомляет об этом поле и обо всех других полях, заполненных нулями. Получение этого DNS-сообщения вызывает ответное сообщение об ошибке от `mydns.com`, которое и будет использовано для определения времени ответа.

Запрос на UDP-порт 33434

Если предыдущие методы опроса DNS-сервера не дали результата, используется следующий.

```
209.67.78.203.3310 > mydns.com.33434: udp 36 [ttl 1]
209.67.78.203.3311 > mydns.com.33434: udp 36 (ttl 2)
216.32.68.10.3307 > mydns.com.33434: udp 36 [ttl 1]
216.32.68.10.3308 > mydns.com.33434: udp 36 (ttl 2)
216.32.68.10.3307 > mydns.com.33434: udp 36 [ttl 1]
216.32.68.10.3308 > mydns.com.33434: udp 36 (ttl 2)
209.67.78.200.3411 > mydns.com.33434: udp 36 [ttl 1]
209.67.78.200.3412 > mydns.com.33434: udp 36 (ttl 2)
```

Этот результат во многом напоминает отчет о действиях UNIX-утилиты `traceroute`. Она также использует UDP-соединения с портами с номером 33000 и выше, как и в данном случае. Но есть и небольшое отличие, так как в стандартной реализации `traceroute` номера портов постоянно увеличиваются для каждого нового пакета. Здесь используется постоянный номер UDP-порта получателя `33434`. Целью является получить ICMP-сообщение о недостижимости порта, после чего программа 3DNS проводит оценку времени ответа DNS-сервера. Еще одной сигнатурой применения утилиты `traceroute` могут служить увеличивающиеся значения поля TTL.

Отправка TCP-сегмента на порт 53

Если все предыдущие попытки не принесли успеха, осуществляется последняя попытка установить соединение с TCP-портом 53. Эта попытка отличается от отправки обычных SYN-пакетов, поскольку в полезной нагрузке содержится 64 байт данных. Обычно никакие данные не передаются до завершения полной процедуры установки TCP-соединения. Объем данных размером в 64 байт выглядит вполне нормально, это и не много, и не мало. Прогнозируемым результатом является получение SYN/ACK-пакета от сервера, который ожидает запросов на этот порт, и пакета с установленными флагами RST и ACK, если порт закрыт.

```
209.67.78.202.2202 > mydns.com.53: S 997788921: 997788985(64) win 2048
209.67.78.202.2200 > mydns.com.53: S 869896644: 869896708(64) win 2048
209.67.78.202.2201 > mydns.com.53: S 1386586413:1386586477(64) win 2048
216.32.68.11.3102 > mydns.com.53: S 765045139: 765045203(64) win 2048
216.32.68.11.3100 > mydns.com.53: S 865977968: 865978032(64) win 2048
216.32.68.11.3101 > mydns.com.53: S 565178644: 565178708(64) win 2048
```

Скорее всего, этот метод не сработает на большинстве узлов, особенно если все предыдущие попытки были неудачными по причине блокирования трафика. Дело в том, что на защищенных узлах блокируются попытки доступа к TCP-порту 53, так как этот порт используется для получения информации баз данных DNS, содержащих перечень всех хостов сети и их IP-адреса. Таким образом, если трафик блокируется, вывод о времени ответа можно сделать только по сообщению о недостижимости, отправленному маршрутизатором. А если пакеты блокируются на маршрутизаторе, настройки которого запрещают отправлять ICMP-сообщения о недостижимости? Тогда попытка будет бесполезной, как и все предыдущие.

Сбор информации с помощью вирусов

Все пользователи локальной сети совместно используют общий почтовый сервер, который настроен на поиск известных вирусов, что позволяет блокировать их распространение до того, как они поразят хост пользователя. Но не все пользователи работают с общими почтовыми серверами, не все используют собственные антивирусные программы, а те, кто используют, не всегда их вовремя обновляют. Это увеличивает риск заражения.

Обычно вирусы и черви не рассматриваются в качестве средств сбора информации. Но теперь появился новый класс червей, действующих как агенты

по сбору информации. В их возможности входят попытки установить соединения с другими хостами со стороны зараженного хоста. Установленная система обнаружения вторжений, осуществляющая фильтрацию исходящего трафика, позволяет выявить подобные действия. Мы рассмотрим два таких червя: Pretty Park и RingZero.

Червь Pretty Park

При появлении сообщения о тревоге в связи с блокированием исходящего трафика на одном из контролируемых узлов оказалось, что локальный хост пытался установить многочисленные соединения с портом получателя 6667 (служба IRC). На этом узле блокировался исходящий трафик для портов службы ICR только потому, что требовалось обеспечить приемлемое качество работы других служб. Я уверен, что существует множество полезных, достойных внимания чат-комнат, но слишком часто пользователи выбирают те чаты, которые не имеют никакого отношения к их работе. И каждое лето, когда на этом узле появлялась новая группа студентов, несколько из них пытались получить доступ к каналам IRC.

Сообщение о тревоге пришло в пятницу вечером в конце февраля. Я сообщил об этом ответственному лицу, и он сказал, что проинформировал администратора о выявленных событиях. Кроме того, я предоставил соответствующие записи из журнала об исходящих пакетах, но не провел глубокого анализа. При более внимательном исследовании я бы обнаружил, что хост пытался установить соединение с IRC-серверами около пяти раз в минуту. Это указывает либо на навязчивое желание пользователя, либо на автоматические действия программы.

В следующий понедельник пришло новое сообщение о тревоге. Я подумал, что это тот же хост, исходящие пакеты которого я уже выявлял. Но, когда я просмотрел журналы, оказалось, что те же действия предпринимаются с четырех других хостов. Особенно настораживал факт попыток соединения с теми же удаленными хостами, многие из которых находились в других странах. У меня возникло ужасное подозрение: были скомпрометированы несколько наших хостов благодаря использованию общего уязвимого места, а теперь взломщик пытается установить соединение со своей базой, чтобы уведомить о триумфе. Другая, более успокаивающая, мысль (по сравнению с компрометацией) заключалась в предположении о заражении этих хостов каким-то червем.

И действительно, когда мой сослуживец (специалист по Windows-системам) проверил один из зараженных хостов, он обнаружил несколько запущенных странных программ (`files32.vxd` и `pretty park.exe`). Таким образом был выявлен червь Pretty Park. С помощью утилиты `netstat` мой коллега установил, что хост отправлял SYN-пакеты на порт получателя 6667. Очевидно, что червь Pretty Park распространяется по электронной почте в виде приложения к сообщению. Одна из его функций заключается в подключении к IRC-сайтам для отправки информации о зараженных хостах, например паролей. Более подробную информацию о Pretty Park можно получить по адресу `http://vil.nai.com/vil/wm98500.asp`.

Ниже представлен фрагмент отчета о действиях, зарегистрированных с помощью TCPdump. Хосты-получатели меняются, но порт получателя остается постоянным (6667).

```
09:30:34.470000 infected.com.1218 > ircnet.grolier.net.6667: S
↳ 662405:662405(0) Bwin 8192 (DF)
09:30:37.370000 infected.com.1218 > ircnet.grolier.net.6667: S
↳ 662405:662405(0) Bwin 8192 (DF)
09:30:43.370000 infected.com.1218 > ircnet.grolier.net.6667: S
↳ 662405:662405(0) Bwin 8192 (DF)
09:30:55.370000 infected.com.1218 > ircnet.grolier.net.6667: S
↳ 662405:662405(0) Bwin 8192 (DF)
09:31:04.050000 infected.com.1220 > irc.ncal.verio.net.6667: S
↳ 691990:691990(0) Bwin 8192 (DF)
09:31:06.970000 infected.com.1220 > irc.ncal.verio.net.6667: S
↳ 691990:691990(0) Bwin 8192 (DF)
09:31:12.970000 infected.com.1220 > irc.ncal.verio.net.6667: S
↳ 691990:691990(0) Bwin 8192 (DF)
09:31:24.970000 infected.com.1220 > ire.ncal.verio.net.6667: S
↳ 691990:691990(0) Bwin 8192 (DF)
09:32:34.130000 infected.com.1222 > mist.cifnet.com.6667: F
↳ 722101:722101(0) ack 1426589426 win 8680 (DF)
09:32:43.070000 infected.com.1224 > krameria.skybel.net.6667: S
↳ 782083:782083(0) Bwin 8192 (DF)
09:32:55.070000 infected.com.1224 > krameria.skybel.net.6667: S
↳ 782083:782083(0) Bwin 8192 (DF)
09:33:04.170000 infected.com.1226 > zafira.eurecom.fr.6667: S
↳ 812112:812112(0) Bwin 8192 (DF)
```

Из этого примера можно сделать вывод, что наилучший результат дает одновременное применение средств защиты отдельных хостов и сети.

На хосте проникновению червя должны препятствовать антивирусные программы, но так происходит далеко не на каждом хосте. В данном случае проблема была обнаружена за счет сетевых средств безопасности благодаря тому, что регистрировался весь заблокированный трафик.

RingZero

В 1999 году появился еще один червь, троянская программа под названием RingZero. Первый шаблон трафика, связанного с действиями этого червя, был выявлен системой Shadow и заключался в обнаружении сканирования различных хостов по TCP-порту получателя 3128 (порт, используемый Squid-сервером). Ниже представлен фрагмент трафика, выявленного системой Shadow.

```
12:29:48.230000 4.3.2.1.1049 > 172.16.54.171.3128: S 9779697:9779697(0)
↳ win 8192 B<mss 1460> (DF) (ttl 19, id 9072)
12:29:58.070000 4.3.2.1.1049 > 172.16.54.171.3128: S 9779697:9779697(0)
↳ win 8192 B<mss 1460> (DF) (ttl 19, id 29552)
12:30:10.960000 4.3.2.1.1049 > 172.16.54.171.3128: S 9779697:9779697(0)
↳ win 8192 B<mss 1460> (DF) (ttl 19, id 39792)
12:44:54.9600001.2.3.4.3243 > 172.16.187.212.3128: S 356330349:356330349(0)
↳ win B8192 <mss 1460> (DF) (ttl 242, id 962)
12:44:57.930000 1.2.3.4.3243 > 172.16.187.212.3128: S 356330349:356330349(0)
↳ win B8192 <mss 1460> (DF) (ttl 242, id 11714)
12:45:03.930000 1.2.3.4.3243 > 172.16.187.212.3128: S 356330349:356330349(0)
↳ win B8192 <mss 1460> (DF) (ttl 242, id 22466)
12:45:15.930000 1.2.3.4.3243 > 172.16.187.212.3128: S 356330349:356330349(0)
↳ win B8192 <mss 1460> (DF) (ttl 242, id 33218)
12:46:13.070000 1.1.1.1.2262 > 172.16.99.110.3128: S 20315949:20315949(0)
↳ win B8192 <mss 1460,nop,nop,sackOK> (DF) (ttl 116, id 35676)
12:46:16.080000 1.1.1.1.2262 > 172.16.99.110.3128: S 20315949:20315949(0)
↳ win B8192 <mss 1460,nop,nop,sackOK> (DF) (ttl 116, id 46428)
```

```
12:46:22.070000 1.1.1.1.2262 > 172.16.99.110.3128: S 20315949:20315949(0)
↳ win B8192 <mss 1460,nop,nop,sackOK> (DF) (ttl 116, id 57180)
12:46:34.080000 1.1.1.1.2262 > 172.16.99.110.3128: S 20315949:20315949(0)
↳ win B8192 <mss 1460,nop,nop,sackOK> (DF) (ttl 116, id 2397)
```

Три атакующих хоста (`1.1.1.1`, `1.2.3.4` и `4.3.2.1`) сканируют различные хосты сети `172.16` (порт получателя `3128`). При дальнейшем исследовании было выявлено, что сканирующие хосты также пытались установить соединения с портами `80` (HTTP) и `8080` (альтернативный порт HTTP). Фильтры системы Shadow не отслеживали трафик, предназначенный этим портам, так как это бы вызвало массу ложных тревог. На многих узлах были выявлены подобные попытки, сканирование проводилось многими хостами-отправителями из разных регионов планеты с интенсивностью около шести сканирований в час. Большинство пакетов сканирования были отправлены по адресам несуществующих хостов, т.е. никакой предварительной разведки проведено не было, или она была проведена очень плохо.

Появилось предположение, что сканирование выполняется с одного хоста с помощью подмены IP-адресов. Приведенный выше отчет TCPdump был получен благодаря использованию параметра командной строки `-vv` (это позволяет вывести информацию в скобках) и предоставляет сведения о значении поля TTL и идентификатора (`id`). Если бы применялась подмена IP-адресов, то значения поля TTL оставались бы примерно одинаковыми (если только не подменялись еще и значения этого поля).

С помощью `traceroute` мы проверили число переходов до многих IP-адресов отправителей. Проверка показала, что указанные в пакетах значения поля TTL заслуживают доверия. Предположив, какое начальное значение TTL установлено на хосте-отправителе, и вычислив разность между этим значением и значением этого поля в полученном пакете, получим приблизительное число переходов между отправителем и получателем. Вся сложность заключается в определении начального значения поля TTL. Решить эту проблему поможет таблица начальных значений поля TTL, доступная по адресу `www.honeynet.org/papers/finger/traces.txt`.

Сходились не только значения количества переходов, но и все IP-адреса отправителей принадлежали работающим хостам, что практически нереально при случайном выборе подменяемых IP-адресов. Наконец, было обнаружено, что в последней попытке сканирования (осуществляемой хостом `1.1.1.1`) использовался отличный от других попыток набор TCP-параметров (`nop,nop,sackOK`). Значит, намного вероятнее использование различных компьютеров, чем специальное создание хакером подобных различий.

Мы обратились к SANS за помощью в определении причин этих попыток сканирования. В университете Вандербильта работал очень проницательный администратор сетей Рон Маркум. Он обнаружил в своих сетях компьютер, осуществляющий сканирования по портам 80, 8080 и 3128. Виновником оказалась троянская программа RingZero. Она собирала сведения о компьютерах с открытыми портами proxy-серверов (80, 8080 и 3128) и передавала эти сведения на FTP-сайт. Для хакеров очень полезна информация об адресах открытых proxy-серверов. Это позволяет им избежать обнаружения действительных IP-адресов своих компьютеров, используя вместо них адреса proxy-серверов. Относительно RingZero остается еще ряд нерешенных вопросов. Например, неизвестно, каким образом происходит заражение отдельного хоста, а также, какие IP-адреса сканируются после установки этой троянской программы.

Резюме

Хакеры тратят невероятные усилия на сканирование чужих хостов по Internet. Очень важно заблокировать на своем узле входящие эхо-запросы ICMP. К разведывательным действиям хакеров нужно относиться очень серьезно. Если злоумышленникам удастся узнать IP-адреса работающих хостов, то им не потребуется много времени, чтобы выяснить, какие службы запущены на этих хостах. В противном случае злоумышленникам приходится действовать наугад. Прекрасным средством защиты является использование частного адресного пространства вместо общедоступных IP-адресов. При использовании общеизвестного адресного пространства и отсутствии деления на зоны DNS хакеры могут получить искомую информацию о работающих хостах с помощью запросов к DNS-серверу. Отличным средством защиты также является служба NAT. Я рекомендую применять несколько уровней защиты NAT. И, наконец, заблокируйте на периметре локальной сети прохождение исходящих ICMP-сообщений о недостижимости.

С появлением нового класса вирусов и червей могут возникнуть внутренние угрозы даже в хорошо защищенной локальной сети. Это пример естественной эволюции методов сбора информации, так как во многих сетях стали умело пресекаться очевидные попытки проведения разведывательных действий извне.

Предметный указатель

И

К

Л

М

Н

О

П

P

C

Т

У

Ф

Х

Ч

Ш

Э

Научно-популярное издание

Стивен Норткат, Джуди Новак

Обнаружение нарушений безопасности в сетях, 3-е издание

Литературный редактор *О.Ю. Белозовская*
Верстка *О.В. Линник*
Художественный редактор *С.А. Чернокозинский*
Корректор *Л.А. Гордиенко, Л.В. Коровкина, Л.В. Чернокозинская*

Издательский дом "Вильямс"
101509, г. Москва, ул. Лесная, д. 43, стр. 1
Изд. лиц. ЛР № 090230 от 23.06.99
Госкомитета РФ по печати

Подписано в печать 25.09.2003. Формат 70x100/16.
Гарнитура Times. Печать офсетная.
Усл. печ. л. 36,1. Уч.-изд. л. 27,46.
Тираж 2500 экз. Заказ № 207 .

Отпечатано с диапозитивов ОАО "САНКТ-ПЕТЕРБУРГСКАЯ ТИПОГРАФИЯ №6"
191144, Санкт-Петербург, ул. Моисеенко, д. 10

Обнаружение нарушений безопасности в сетях, третье издание

Несмотря на усилия специалистов по обеспечению защиты информации, компьютерные сети продолжают подвергаться тщательно продуманным атакам хакеров. И число таких атак растет. Третье издание книги *Обнаружение нарушений безопасности в сетях* дополнит знания, полученные нашими читателями из двух первых изданий, и позволит повысить их квалификацию как аналитиков сетевого трафика.

Эксперты в области безопасности сетей Стивен Норткат и Джуди Новак поделятся своим опытом по проблемам обнаружения взлома, что позволит встретить хакеров во всеоружии. Читателям будут изложены все базовые концепции. Например, каким образом пакеты поступают в сеть; как узнать, является трассировка результатом воздействия или реакции; как выполнить анализ трафика и информации, содержащейся в полях IP-заголовка и полях других вложенных пакетов; как выявить характерные атаки по сигнатурам и как интегрировать методы обнаружения взлома в нормальную работу коммерческой организации.

Стивен Норткат, один из создателей системы обнаружения вторжений Shadow, был первым начальником группы обнаружения вторжений с помощью Shadow при Министерстве обороны США. Затем он стал начальником ведомства информационной безопасности департамента защиты от нанесения удара баллистическими ракетами. В данное время занимает пост руководителя по обучению и сертификации в институте SANS.

Джуди Новак в данное время является старшим аналитиком безопасности в компании Jacob and Sundstrom, Inc. Большую часть своего времени она проводит в Лаборатории прикладной физики Университета Джона Хопкинса, где занимается исследованием проблем обнаружения взлома, мониторингом трафика и изучением процессов обмена информацией. Джуди — один из основателей группы по реагированию на происшествия исследовательских лабораторий вооруженных сил США. Она принимала участие в создании курса по изучению TCP/IP и собственного курса "Анализ сетевого трафика с помощью TCPdump", которые изучаются студентами в институте SANS.

Эта книга станет настольной для специалистов в области защиты информации. По ее материалам вы научитесь:

- Понимать основные принципы работы протоколов семейства TCP/IP в реальных условиях
- Выполнять анализ сетевого трафика
- Извлекать информацию об атаке из полей IP-заголовка и заголовков других протоколов
- Распознавать стандартные методы нападения и быть готовым к использованию нестандартных атак
- Использовать анализаторы TCPdump и Snort для выявления вредоносного трафика
- Настраивать свои системы с целью обеспечения максимальной безопасности работы в сети
- Отвечать на выявленные попытки взлома
- Интегрировать возможности обнаружения взлома в бизнес-модель своей организации
- Определять основные признаки разведывательных действий, атак отказа в обслуживании и сканирований с помощью специальных программ

Сети / Безопасность

ISBN 5-8459-0526-5

ВИЛЬЯМС
www.williamspublishing.com

New Riders
www.newriders.com