1 Программное обеспечение — источник всех проблем

Итак, конечная цель взлома системы — это заставить данную систему "молить о пощаде", когда уже раскрыты все секреты установленных программ и хакеру предоставлен доступ к командному интерпретатору с неограниченными правами. Взлом компьютера практически всегда выполняется с помощью установленного программного обеспечения. И чаще всего атакуемый компьютер не является рядовой системой[1]. Практически у всех современных компьютерных систем есть своя "ахиллесова пята" в виде программного обеспечения. Благодаря этой книге вы узнаете, как взламывать программное обеспечение, и научитесь использовать уязвимые места программ, чтобы получить контроль над компьютером.

На сегодняшний день выпущено уже немало хороших книг по сетевой безопасности. Например, книга Брюса Шнейера (Bruce Schneier) *Secrets and Lies* (2000) предоставляет читателям прекрасный обзор ситуации в области защиты информации, причем эта книга содержит огромное количество великолепных примеров и мудрых советов. Книга *Hacking Exposed* (автор Мак-Клур) является прекрасным руководством для тех, кто хочет понять элементарные атаки (например, чтобы организовать защиту). Умение противодействовать таким атакам тоже важно, но является только первым шагом в правильном направлении. Вернуться на уровень элементарных программ атаки не помешает для организации правильной защиты (или проведения нужной атаки). С помощью сведений, изложенных в книге *The Whitehat Security Arsenal*, можно защитить свою сеть от огромного количества атак. Книга *Security Engineering* (автор Росс Андерсон, 2001) дает подробный аналитический обзор проблемы безопасности. Так зачем нужна *еще* одна книга по безопасности?

Шнейер в предисловии к своей книге *Building Secure Software* написал: "Мы бы не тратили так много времени, денег и усилий на обеспечение безопасной работы в сети, если бы у нас было более надежное программное обеспечение". Затем он написал следующее:

> *"Вспомните последние уязвимые места в программном обеспечении, о которых вы узнали. Возможно, это был вредоносный пакет, который позволяет*

---

[1] *Безусловно, большинство программ атаки предназначено для взлома стандартного программного обеспечения, запущенного на стандартном компьютере, которые ежедневно используются в коммерческих организациях. — Прим. авт.*

хакеру взломать какой-то сервер. Возможно, это была одна из бесчисленных атак на переполнение буфера, используя которую злоумышленник получает контроль над чужим компьютером, отправив специальное вредоносное сообщение. А возможно, это было уязвимое место в протоколе шифрования, благодаря чему хакер может читать зашифрованные сообщения или обойти систему аутентификации. Все это проблемы программного обеспечения".

Среди того огромного количества материала, который был опубликован по теме компьютерной безопасности, в центре внимания только очень небольшой части трудов была первопричина всех проблем — ошибки в программном обеспечении. Мы исследуем этот безбрежный океан ошибок в программах и научим вас прокладывать правильный маршрут в этих бескрайних просторах.

Краткая история программного обеспечения

Современные компьютеры больше не являются громоздкими агрегатами размером с комнату, при обслуживании которых оператору приходилось заходить внутрь компьютера. Теперь пользователи скорее носят компьютеры, чем заходят в них. Среди тех революционных средств, которые позволили совершить это коренное преобразование, можно назвать кинескоп, транзистор и чип на силиконовой подложке, но самым главным все же является программное обеспечение.

Именно благодаря программному обеспечению компьютеры выделяются на фоне других технологических новшеств. Блестящая идея перенастройки машины для выполнения практически неограниченного количества задач одновременно проста и гениальна. Эта идея очень долго оставалась просто теорией, до того как удалось получить осязаемые результаты ее воплощения. При работе над своей концепцией счетной машины в 1842 году Чарльз Бэббидж (Charles Babbage) воспользовался помощью переводчика — леди Ады Лавлейс (Ada Lovelace)[2]. Ада, которая сама себя называла "аналитиком (и метафизиком)" разбиралась в идее устройства не хуже, чем сам Бэббидж, но лучше выражала на словах те преимущества, которые принесет создание этого устройства. Особенно это касается пояснений к оригинальной работе. Она поняла, что счетная машина — это то, что теперь мы называем компьютером общего назначения. По ее словам, счетная машина была предназначена для "подсчета и составления таблиц для выполнения любого действия ... механизм дает возможность подсчитать значение любой неопределенной функции любой степени сложности". Уже тогда ей удалось выразить всю мощь идеи программного обеспечения.

Согласно словарю университета Вебстера, слово *software* (программное обеспечение) получило широкое распространение в 1960 году и поясняется следующим образом:

> "...что-то используемое или связанное с аппаратными средствами: например, полный набор программ, процедур и пояснительной документации, связанной с системой и особенно с компьютерной системой, в частности, компьютерные программы..."

---

[2] *Более подробную информацию об Аде Лавлейс можно прочесть по адресу* `http://www.sdsc.edu/ScienceWomen/lovelace.html`.

В 60-х годах прошлого века появление "современных, высокоуровневых" языков программирования, например Fortran, Pascal и C, позволило программному обеспечению выполнять все более сложные операции. Компьютеры стали больше определяться по тому, какое программное обеспечение на них запущено, а не по тому, какими аппаратными средствами управляют программы. Появились и начали развиваться операционные системы. Были созданы первые сети, которые стали увеличиваться стремительными темпами. Причем этот рост был связан прежде всего с развитием программного обеспечения[3]. Программное обеспечение стало *необходимым*.

Забавная вещь случилась с появлением Internet. Использование программного обеспечения, которое было задумано для упрощения жизни отдельного человека, стало противоречить правилам морали и этики. И совершенно права оказалась леди Лавлейс, когда говорила, что оно позволяет выполнять "любые действия", в том числе и вредоносные действия, потенциально опасные действия и просто ошибочные функции.

В процессе своего развития программное обеспечение стало выходить за рамки технических устройств и стало проникать в различные сферы человеческой деятельности. Использование программного обеспечения в бизнесе и военной сфере стало практически обыденным.

При ошибках в программах деловой мир несет колоссальные убытки. Программное обеспечение управляет каналами снабжения, предоставляет доступ к глобальной информации, позволяет управлять заводами и фабриками и используется для взаимодействия с заказчиками. Любая ошибка в таком программном обеспечении может привести к тяжелым последствиям:

- предоставление конфиденциальных данных неавторизованным пользователям (включая и хакеров);
- выход из строя и "зависание" систем вследствие предоставления неправильных данных;
- возможность для хакера внедрять и выполнять программный код;
- выполнение привилегированных команд со стороны хакера.

Распространение компьютерных сетей оказало огромное (и в основном негативное) влияние на использование программного обеспечения. По сравнению с в начала 1970-х годов сетью под названием ARPANET, глобальная сеть Internet ворвалась в жизнь людей просто с непредсказуемой скоростью, гораздо быстрее, чем многие другие технологии, включая электричество и телефон (см. рис. 1.1). Если Internet — это машина, то программное обеспечение — это ее двигатель.

Объединение компьютеров в сети позволяет пользователям совместно использовать данные, программы и вычислительные ресурсы. При подключении компьютера к сети он становится доступным с других удаленных компьютеров, что позволяет географически удаленным пользователям получать данные или использовать ресурсы этого компьютера. Программное обеспечение в реализации этих задач является сравнительно новым и работает нестабильно. В современной быстроменяющейся

---

[3] *Существует теснейшая взаимосвязь между развитием аппаратных средств и программного обеспечения. Тот факт, что современные аппаратные средства обладают огромными возможностями и емкостью при небольших размерах, неразрывно связан с параллельным развитием программного обеспечения. — Прим. авт.*

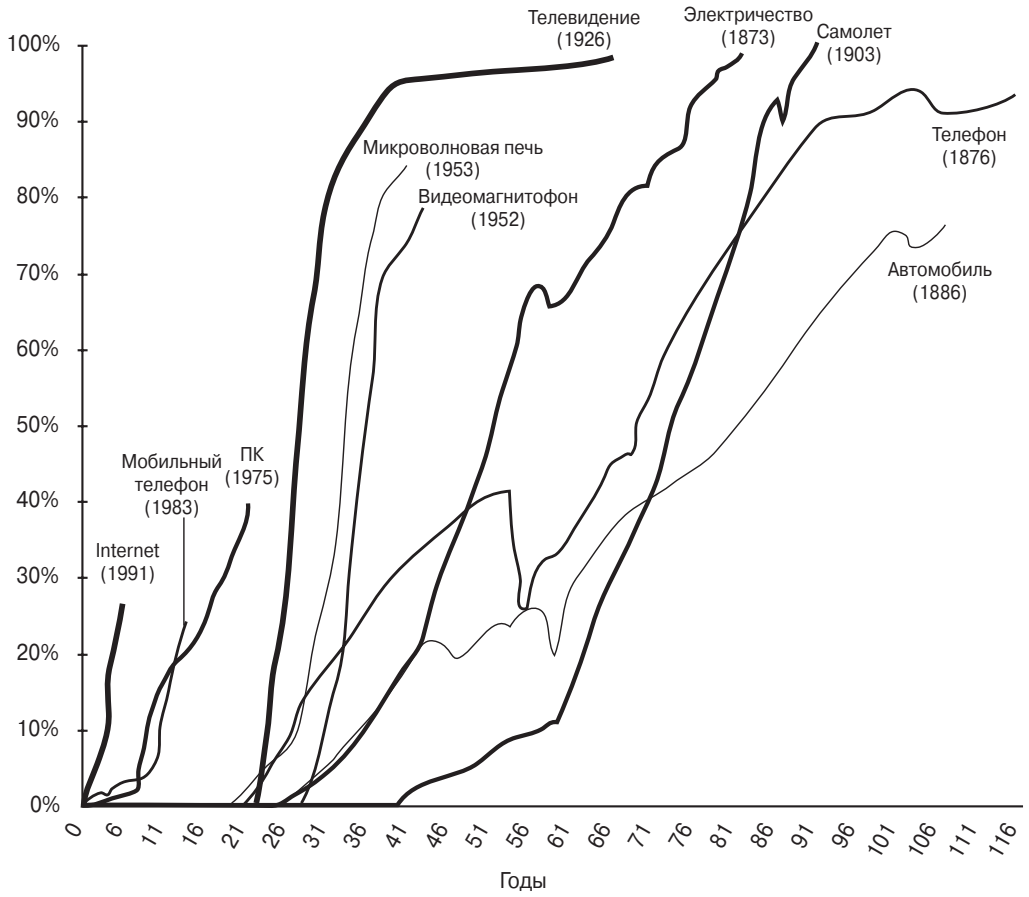

Рис. 1.1. Скорость внедрения различных технологий в годах. На оси x представлены годы (за точку отсчета принят год внедрения или изобретения), а на оси y изображено проникновение на рынок (в процентном отношении). О скорости внедрения можно судить по наклону различных кривых. Очевидно, что Internet вошел в обиход намного быстрее (и оказал значительно большее влияние на культуру), чем другие революционные технологии в истории человечества

экономике на компании, занимающиеся разработкой программного обеспечения, оказывается сильное рыночное давление, в результате чего появляются все новые и новые технологии. “Время выпуска на рынок” становится основным движущим фактором, а главный лозунг — “сделать на вчера”. Чем дольше времени занимает подготовка технологии к выпуску на рынок, тем выше вероятность коммерческого провала. Поскольку тщательное создание технологий требует больших финансовых вложений и времени, то зачастую программное обеспечение создается в спешке и без надлежащего тестирования. Неаккуратность при разработке программного обеспечения привела к тому, что сегодня в глобальной сети существуют миллиарды ошибок, которыми можно воспользоваться для взлома систем.

В большинство программ, предназначенных для взаимодействия по сети, добавлены функции безопасности. Хотя киношный стереотип о легко угадываемом пароле в принципе соответствует действительности, но пароли только иногда останавлива-

ют хакеров. Это относится только к тем злоумышленникам, которые "пытаются войти через парадную дверь". Проблема в том, что многие механизмы, призванные защитить программное обеспечение от взлома, *сами* являются программами и могут оказаться целью более сложной атаки. Поскольку большинство средств безопасности представляют собой только часть программного обеспечения, эти проверки можно обойти. И хотя все мы видели фильмы, в которых хакеры отгадывают пароли, в реальной жизни хакеры "работают" с более сложными функциями безопасности. Среди усовершенствованных систем безопасности и связанных с ними атак можно назвать следующие:

- контроль над тем, кому разрешено подключаться к конкретному компьютеру;
- выявление подложных аутентификационных данных;
- определение того, кто имеет право доступа к ресурсам совместно используемого компьютера;
- защита данных (особенно при передаче) с помощью шифрования;
- место и способ хранения данных регистрационных журналов.

В течение 1990-х годов были обнаружены и представлены широкой общественности десятки тысяч ошибок в программном обеспечении, которые были связаны с безопасностью. Наличие этих ошибок привело к распространению программ атаки. К настоящему времени десятки тысяч так называемых "потайных ходов" созданы в компьютерных сетях по всей планете. Это последствия взрыва хакинга в конце XX века. Исходя из нынешнего состояния дел, прояснить ситуацию полностью кажется практически невозможным, но мы предпримем попытку сделать это. Одна из причин создания этой книги — вызвать конструктивные дискуссии по поводу истинных проблем, которые привели к появлению программ атаки, и оставить в прошлом "интересные", но лишь поверхностные разговоры.

Программное обеспечение и информационные войны

Второй наидревнейшей профессией является война. Но даже такое древнее занятие преобразуется в современном кибер-мире. Умение выстоять в информационной войне предельно важно для каждого государства и корпорации, которые планируют свое развитие (и выживание) в современном мире. Даже если нация не готовится к информационной войне, то это делают ее противники, и неподготовленное государство будет иметь проигрышную позицию в будущих войнах.

Сбор информации критически важен для ведения войны. Поскольку информационная война ведется и за информацию в том числе, то она неразрывно связана со сбором информации[4]. Классическая разведка преследует четыре основные цели:

- защита государства (и государственная безопасность);
- помощь в военных действиях;
- расширение политического влияния на мировом рынке;
- увеличение экономической мощи.

---

[4] *Более подробную информацию по этой теме можно почерпнуть из книги Дороти Дэннинг (Doroty Denning) Information Warfare & Security.*

Хорошим шпионом всегда был тот, кто умел собирать и правильно распоряжаться большими объемами секретной информации. В современную эпоху распределенных вычислений это особенно справедливо. Если к секретной информации можно получить доступ через сеть, для шпиона вовсе необязательно личное присутствие, т.е. остается меньше шансов на то, что тебя обнаружат. Это также означает, что возможности для сбора информации обходятся дешевле, чем традиционные средства шпионажа.

Поскольку война неразрывно связана с экономикой, электронная война во многих случаях касается электронных счетов. Современные деньги — это просто набор электронов, которые находятся в нужное время в нужном месте. Триллионы электронных долларов ежедневно передаются между государствами. Управление глобальными сетями означает управление глобальной экономикой. Это и есть основная цель информационной войны.

Электронный шпионаж

Некоторые аспекты информационных войн можно смело назвать *электронным шпионажем* (digital tradecraft). В словаре этот термин поясняется как одно из "средств и методов шпионажа...".

Современный шпионаж осуществляется с помощью программного обеспечения. При информационных атаках на компьютерные системы для доступа к нужной информации используются уязвимые места в существующем программном обеспечении или потайные ходы, внедренные в программу до ее установки. Уязвимые места в программах различаются от неправильной конфигурации до ошибок, допущенных на стадии программирования и разработки проекта. Иногда злоумышленник может просто запросить нужную информацию от атакуемой программы и получить требуемый результат. В других случаях в систему должен быть внедрен вредоносный код. Некоторые авторы разделяют вредоносный код на логические бомбы, программы для перехвата данных, "троянских коней" и т.д. Правда в том, что вредоносный код способен осуществить практически любые действия. Поэтому классификация этого кода становится бесполезной, если пытаться ее выполнить исходя из конечного результата применения программы атаки. В некоторых случаях классификация все же помогает пользователям и аналитикам сетевого трафика различать различные категории атак. Если рассматривать ситуацию на высшем уровне, то с помощью вредоносного кода можно выполнять любую комбинацию следующих действий.

1. Сбор данных:
   - перехват пакетов;
   - отслеживание нажатий клавиш;
   - извлечение информации базы данных.
2. Использование "невидимости":
   - сокрытие данных (например, сокрытие файлов журналов и т.д.);
   - сокрытие запущенных процессов;
   - сокрытие пользователей, работающих в системе.
3. Скрытые соединения:
   - организация "невидимого" удаленного доступа;

- извлечение секретных данных из системы;
- создание скрытых каналов.

4. Подчинение и управление:
   - создание условий для удаленного управления системой;
   - вредительство (как вариант подчинения и управления);
   - блокирование управления системой (атаки отказа в обслуживании).

Эта книга в основном сфокусирована на технических деталях использования программного обеспечения в целях создания и внедрения вредоносного кода. Знания и методы, изложенные в этой книге, не являются чем-то принципиально новым и применялись небольшой (но растущей) группой людей уже почти 20 лет. Многие методы атак были по несколько раз изобретены группами хакеров, которые работали независимо друг от друга.

Только сравнительно недавно методы взлома программного обеспечения были объединены и отнесены к "отдельной науке". Исторически так сложилось, что для достижения одинаковых целей использовались различные методы. Зачастую методы восстановления исходного кода программы разрабатывались как побочный продукт в процессе взлома программ. Методы создания вредоносного кода подобны методам для взлома защиты программного обеспечения (например защиты с помощью заплат). Естественно, что разработчики вирусов используют приблизительно одинаковые основополагающие идеи. В 1980-х годах было несложно найти программный код вируса или код для взлома программы. С другой стороны, идеи хакинга программ зародились среди администраторов UNIX-систем. Многие люди, знакомые с классическими способами хакинга в сетях, думали прежде всего о получении паролей и создании лазеек в программном обеспечении и совершенно забывали о вредоносном коде. В начале 1990-х годов обе "дисциплины" начали сливаться в единое движение хакеров и по Internet стали распространяться первые программы атаки для получения удаленного доступа к командному интерпретатору.

Существует множество книг по компьютерной безопасности, но ни в одной из них не описывается атакующий аспект с точки зрения программиста[5]. Все книги по хакингу, включая и популярную серию *Секреты хакеров*, можно назвать кратким изложением сценариев атак хакеров и существующих программ атаки. При этом основное внимание уделяется проблемам безопасности при атаках по сети и практически ничего не говорится о поиске новых программ атаки. Такой подход нельзя назвать правильным, поскольку специалисты, которые создают системы защиты, плохо осознают, против чего же они в действительности борются. Если мы будем продолжать защищаться только против плохо вооруженных новичков в деле хакинга, то наша защита никогда не позволит нам выдержать более сложные атаки настоящих хакеров, которых становится все больше.

Зачем писать книгу, полную потенциально опасных сведений, которыми могут воспользоваться злоумышленники? В основном мы хотим рассеять распространенное заблуждение о возможностях программ атаки. Многие люди не понимают, на-

---

[5] *В связи с ростом популярности книг, подобных этой, следует отметить, что не за горами появление отдельной дисциплины по изучению программ и методов атаки на компьютеры. — Прим. авт.*

сколько опасными могут быть хакеры и что только несколько современных технологий по обеспечению защиты в сетях могут остановить этих хакеров. Возможно, дело в том, что программное обеспечение представляется неким волшебством для большинства людей, а возможно, дело в неправильной информации, распространяемой недобросовестными (или просто несведущими) поставщиками программ по обеспечению безопасности.

Хвастливые заявления некоторых хакеров можно считать важным сигналом тревоги, который мы больше не вправе игнорировать.

Как думают некоторые хакеры

"Дайте человеку взломанную программу и завтра ему понадобится новая программа, научите его взламывать программы — и он никогда не обратится к вам снова".
+ORC

Во что верят злоумышленники, которые взламывают программы? Как они узнали о возможности создания программы атаки? Какое учебное заведение окончили? Ответы на эти вопросы будут важны, если мы хотим найти верный подход к решению проблемы создания безопасных компьютерных систем.

В некотором смысле осведомленный хакер является одним из самых могущественных людей в современном мире. Хакеры часто перечисляют бесконечное количество удивительных фактов об атаках на программное обеспечение и их результатах. Интересный вопрос заключается в том, являются ли эти факты правдивыми. Многие из заявлений имеют под собой реальную основу, и даже если они преувеличены, то все равно дают возможность оценить ход мыслей хакеров.

Обычно злоумышленники распространяют следующие заявления.

- Хакеры проникли в большинство из 2000 глобальных корпораций. Каждое из главных финансовых учреждений не только было взломано, но хакеры продолжают активно использовать доступ к этим учреждениям.
- Большая часть программного обеспечения, созданного сторонними разработчиками (по контрактам), содержит многочисленные потайные ходы, и крайне сложно провести независимую оценку этих программ. Компании, которые заказывают программное обеспечение, зачастую вообще не уделяют никакого внимания безопасности программ.
- Каждое мощное государство на планете тратит крупные суммы на создание средств как для защиты программ, так и для атаки на программное обеспечение.
- Брандмауэры, антивирусные программы и системы обнаружения вторжений *работают недостаточно надежно*. Поставщики программного обеспечения, предназначенного для защиты информационных систем, дают невыполнимые обещания и реализуют защиту только от стандартных атак по сети. Не уделяется достаточного внимания проблемам безопасности программного обеспечения.

Далее приведены наиболее распространенные "постулаты" хакеров. "Знающий" человек обычно верит в эти "постулаты" по проблемам безопасности программного обеспечения.

- Защиты от копирования программного обеспечения никогда не было и никогда не будет. Это невозможно даже теоретически.
- Владение исполняемым программным кодом в двоичной форме не менее привлекательно (если не более), чем владение исходным кодом программы.

- Не существует коммерческих тайн программного обеспечения. Принцип "безопасность с помощью неизвестности" (или "о чем не знают, того не украдут"), работает только на руку потенциальным злоумышленникам, особенно если неизвестность призвана скрыть недостатки программы.
- Существуют *сотни* невыявленных программ атаки, которые используются *прямо сейчас*, и они останутся невыявленными еще многие годы.
- Никто не может надеяться на безопасность своих систем, используя только заплаты для программ и "полные" списки рассылки по проблемам безопасности. Информация этих источников обычно слишком запаздывает и появляется значительно позже появления программы атаки.
- Большинство подключенных к Internet компьютеров (за очень редким исключением) могут быть удаленно взломаны *прямо сейчас*, включая и те, на которых запущено самые современные, полностью обновленные версии Microsoft Windows, Linux, BSD и Solaris. То же самое касается и популярных приложений от других производителей, например Oracle, IBM, SAP, PeopleSoft, Tivoli и HP.
- Многие "аппаратные" устройства, подключенные к Internet (за редкими исключениями), могут быть удаленно взломаны *прямо сейчас*, включая коммутаторы 3COM, маршрутизаторы Cisco и их программы IOS, брандмауэры Checkpoint и распределители нагрузки F5.
- С помощью уязвимых мест в программном обеспечении SCADA, большинство критически важных систем обеспечения, которые управляют подачей воды, газа, нефти и электрической энергии, могут быть взломаны и управляться удаленно.
- Если квалифицированный хакер захочет взломать какую-то конкретную машину, он добьется успеха. Переустановка операционной системы или загрузка нового образа системы после ее компрометации не поможет защититься от атаки, поскольку опытный хакер способен переписать встроенные программы для микрочипов системы.
- Спутниковые системы были взломаны и продолжают использоваться хакерами.

Если верить хакерам, все это происходит в настоящее время. И даже если только некоторые из этих заявлений соответствуют действительности, то пришла пора всем нам вынуть голову из песка и понять, что происходит на самом деле. Воображать, что информации, изложенной в этой книге, не существует и что она не имеет особого значения, просто глупо.

Ошибки в программах есть всегда

Безопасность программного обеспечения обычно рассматривается только как проблема работы в Internet, но это далеко не единственная проблема. Хотя коммерческие структуры широко используют Internet, но многие системы работают изолированно в локальных сетях или вообще на отдельных компьютерах. Очевидно, что программное обеспечение отвечает за нечто большее, нежели за возможности для игр в сети или за обмен сообщениями электронной почты и электронными таблицами. При ошибках в программном обеспечении убытки исчисляются миллионами долларов, что иногда приводит даже к смерти людей. В этом разделе мы напомним о широко известных случаях ошибок в программах.

Данная информация имеет прямое отношение ко взлому программного обеспечения, поскольку "спонтанные" (т.е. без предумышленного вмешательства хакера)

ошибки в программах демонстрируют, что может произойти *даже без специально продуманных действий злоумышленника*. Читая об этих случаях, представьте, на что способен опытный хакер!

Марсоход NASA

Одна простая ошибка в программном обеспечении стоила Соединенным Штатам около 165 млн. долл., когда спускаемый модуль NASA разбился о поверхность Марса. Проблема оказалась в неправильном переводе английских единиц измерения в международные единицы измерения. В результате ошибки была выбрана неправильная траектория спуска при приближении к поверхности Марса. Двигатели выключились преждевременно, в результате чего произошла авария.

Система выдачи багажа в аэропорту Денвера

В современном международном порту города Денвер была создана автоматизированная система выдачи багажа, в которой использовались автоматические тележки, двигающиеся по фиксированному маршруту. При этом все управлялось программным обеспечением. При тестировании тележки стали двигаться неправильно из-за многочисленных ошибок в программе управления. Они двигались асинхронно, появлялись то пустые, то перегруженные тележки. Даже груды упавших чемоданов не останавливали тележек. Эти ошибки в программном обеспечении привели к задержке открытия аэропорта на 11 месяцев, что стоило аэропорту по крайней мере 1 млн. долл. в сутки.

MV-22 Osprey

Военный вертолет MV-22 Osprey (рис. 1.2) представляет собой нечто среднее между вертолетом вертикального взлета и обычным аэропланом. Сам вертолет и его аэродинамические характеристики очень сложные, поэтому полет вертолета контролируется с помощью различных систем сложнейшего программного обеспечения. Как и в большинстве машин подобного класса, в этом вертолете предусмотрено несколько аварийных систем на случай ошибки. Однажды, во время рокового взлета, один из гидравлических механизмов загорелся. Это была серьезная проблема, но ее обычно можно было исправить. Однако в данном случае ошибка в программном обеспечении привела к неисправности аварийной системы. Вертолет разбился и четыре солдата морской пехоты погибли.

Система US Viceness

В 1988 году корабль военно-морских сил США запустил реактивную ракету и поразил цель, выявленную бортовым радаром и определенную системой слежения как вражеский военный самолет (рис. 1.3). В действительности "целью" оказался коммерческий самолет Airbus A320 (рис. 1.4), на котором летели ничего не подозревающие люди. В результате попадания ракеты погибли 290 человек. В официальном извинении военно-воздушных сил США причиной ошибки были названы неправильные данные, выведенные на экран системы слежения.

Компания Microsoft и вирус любви

Распространение вируса "I LOVE YOU" оказалось возможным из-за ошибки в клиенте электронной почты Microsoft Outlook, благодаря которой выполнялись программы, полученные от неизвестных отправителей. Очевидно, никто из команды

Рис. 1.2. MV-22 Osprey в воздухе. Сложное программное обеспечение критически важно для управления полетом

Рис. 1.3. Истребитель, аналогичный тому, который был определен системой слежения US Viceness как "вражеский"

программистов Microsoft не подумал о том, на что может быть способен вирус, использующий встроенные возможности при выполнении сценариев. Согласно исследованиям, ущерб, нанесенный вирусом "I LOVE YOU", оценивается миллиардами долларов. Обратите внимание, что эта цена была заплачена пользователями Outlook, а не компанией Microsoft. Появление этого вируса продемонстрировало, как Internet-вирус может нанести значительный финансовый ущерб коммерческим организациям.

Рис. 1.4. Самолет Airbus A320, сбитый ракетой из-за ошибки системы слежения US Viceness

Сравнительно недавно по просторам Internet пронесся еще один широкомасштабный вирус под названием Blaster, ущерб от которого тоже исчисляется миллиардами. Распространение этого вируса также является результатом ошибок в программном обеспечении.

Итак, вывод очевиден: ошибки в программном обеспечении являются наиболее серьезным уязвимым местом в компьютерных системах. Эти недостатки программ приводят к огромным финансовым потерям. Подобным образом ошибки позволяют злоумышленникам преднамеренно наносить ущерб и красть важную информацию. В конечном счете ошибки в программном обеспечении приводят к появлению программ атаки.

Три основные проблемы

Почему же так трудно контролировать работу программного обеспечения? Три основных фактора превращают управление рисками при использовании программ в основную задачу современности. Этими факторами являются сложность, расширяемость и возможность взаимодействия.

Сложность

Современное программное обеспечение довольно сложное, и есть все предпосылки считать, что оно станет еще сложнее в ближайшем будущем. Например, в 1983 году программа Microsoft Word состояла только из 27000 строк кода, но, согласно данным Натана Мирвольда (Nathan Myhrvold)[6], к 1995 году эта программа увеличилась уже до 2 млн. строк кода! Программисты потратили годы на то, чтобы придумать единицы измерения для программного обеспечения. Целые книги посвящены существующим системам измерения размера программ. Но только одна единица измерения позволяет установить соотношение с числом ошибок — количество строк кода (LOC). И действительно, в некоторых кругах специалистов по программированию число строк кода стало единственным приемлемым средством измерения объема программ.

Количество ошибок на тысячу строк кода (KLOC) изменяется для каждой конкретной системы. Достоверное значение варьируется от 5 до 50 ошибок на 1000

[6] *В журнале Wired Magazine есть статья по этой теме, доступная по адресу* `http://www.wired.com/wired/archive/3.09/myhrvold.html?person=gordon_moore&topic_set=wiredpeople`.

строк кода. Даже в системах, которые прошли строгий контроль качества (Quality Assurance — QA) все равно содержатся ошибки — приблизительно 5 ошибок на 1000 строк кода. В программной системе, которая прошла тестирование только на предмет работоспособности функциональных возможностей, что справедливо для большей части коммерческого программного обеспечения, присутствует намного больше ошибок — около 50 ошибок на 1000 строк кода. Большая часть программ попадает в последнюю категорию. Многие поставщики программного обеспечения неверно предполагают, что они выполняют строгий контроль качества QA, хотя в действительности их методы тестирования являются весьма поверхностными. Строгий контроль качества программного обеспечения заключается не только в тестировании возможностей программ, но и должен включать в себя тестовое внесение неисправностей и анализ ошибок.

Чтобы оценить всю сложность современного программного обеспечения, проанализируйте следующую информацию.

Количество строк кода	***Система***
400000	Solaris 7
17 млн	Netscape
40 млн	Космическая станция
10 млн	Космический челнок
7 млн	Boeing 777
35 млн	NT5
1,5 млн	Linux
менее 5 млн	Windows 95
40 млн	Windows XP

Как мы указывали ранее, для приведенных здесь систем характерна частота ошибок от 5 до 50 на 1000 строк кода.

Для демонстрации постоянного роста количества строк кода рассмотрим его на примере операционных систем компании Microsoft. На рис. 1.5 показано, как вырос объем операционной системы Windows, начиная с ее появления в 1990 году в виде версии Windows 3.1 (3 млн. строк программного кода) и заканчивая ее последней версией Windows XP, вышедшей в 2002 году (40 млн. строк программного кода). Один простой, но не слишком радостный факт справедлив для всего программного обеспечения: *чем больше строк, тем больше ошибок*. Если эта тенденция сохранилась, то в Windows XP должно быть достаточно много ошибок[7]. Здесь возникает очевидный вопрос: какое количество таких ошибок приводит к проблемам системы безопасности? И какая существует связь между ошибками и другими уязвимыми местами и разработкой хакерами программ атаки?

Настольный компьютер под управлением Windows XP и приложения для этой системы зависят от нормальной работы ядра. Это касается и приложений, обеспечивающих защиту от атак хакеров. Однако сама система Windows XP состоит из при-

[7] *Что и подтвердилось после выявления нескольких серьезных уязвимых мест в течение нескольких месяцев после выпуска этой операционной системы. — Прим. авт.*

близительно 40 млн. строк кода, и приложения тоже соответственно становятся сложнее. Когда система становится такой сложной, ошибки просто неизбежны.

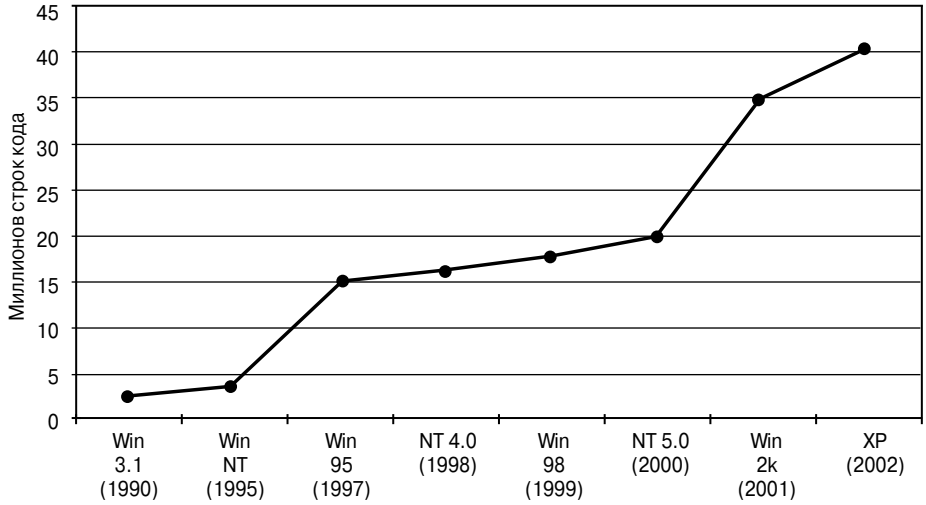

Рис. 1.5. Чем больше строк кода, тем больше ошибок и недостатков

Кроме того, усложняет проблему широкое использование низкоуровневых языков программирования, таких как C и C++, которые не способны защитить от простейших атак, например от атак на переполнение буфера (которые мы рассмотрим в этой книге). Кроме дополнительных возможностей для взлома вследствие ошибок и других просчетов, вообще в сложных системах как таковых легче скрыть вредоносный код. Теоретически мы можем проверить и доказать, что в простой программе нет ошибок, влияющих на безопасность, но в действительности это невозможно, даже если речь идет о самой простой современной настольной системе, и уж точно невероятно в отношении систем масштаба предприятия.

Больше строк, больше ошибок

Рассмотрим сеть из 30000 узлов (типичный размер сети средней корпорации). На каждой рабочей станции сети есть программное обеспечение в виде исполняемых файлов (EXE) и библиотек, а также около 3000 исполняемых модулей. Средний размер каждого модуля около 100 Кбайт. Предполагая, что в каждой строке кода содержится по 10 байт кода, и задав минимальное количество ошибок (5) на 1000 строк кода, получим, что в каждом исполняемом модуле содержится около 50 ошибок.

$$\frac{\sim 100 \text{ Кбайт}}{\text{исполняемый файл}} = \frac{10 \text{ тыс. строк кода}}{\text{исполняемый файл}}$$

$$\frac{5 \text{ ошибок}}{1000 \text{ строк кода}} = \frac{50 \text{ ошибок}}{\text{исполняемый файл}}$$

Теперь вспомним, что на каждом хосте содержится около 3000 исполняемых файлов. Это означает, что на каждый компьютер сети приходится около 150000 уникальных ошибок в программном коде.

$$\frac{50 \text{ ошибок}}{\text{исполняемый файл}} \times \frac{3000 \text{ исполняемых файлов}}{\text{хост}} = \frac{150000 \text{ ошибок}}{\text{хост}}$$

Конечно, это огромное количество ошибок. Но проблемы только начинаются. Определим количество возможных целей атаки и количество копий однотипных ошибок, которые доступны как цели для атаки. Поскольку эти же 150000 ошибок повторяются множество раз на 30 тыс. хостов, то число потенциальных целей для хакера поистине огромно. В сети из 30 тыс. хостов насчитывается около 4,5 млрд. ошибок, благоприятствующих проведению атак (согласно нашим оценкам, только 150 тыс. из этих ошибок уникальны, но это число не точно).

$$\frac{150000 \text{ ошибок}}{\text{хост}} \times \frac{30000 \text{ хостов}}{\text{сеть}} = 4,5 \text{ млрд. ошибок в сети}$$

Если представить, что 10 % всех ошибок приводят к проблемам в системе безопасности и что только 10 % из них могут быть использованы при удаленных атаках (по сети), то согласно нашим данным, в этой небольшой локальной сети будет 5 млн. уязвимых мест в программном обеспечении, доступных для удаленной атаки. Устранение 5 млн. уязвимых мест является серьезной задачей, а правильное управление заплатами для 5 млн. уязвимых мест, разбросанных по 30000 хостам, еще сложнее.

$$4,5 \text{ млрд} \times 10\% = 500 \text{ млн. ошибок по безопасности}$$

$$500 \text{ млн} \times 10\% = 5 \text{ млн. доступных удаленно уязвимых мест}$$

Согласно этим цифрам, хакер заранее находится в выигрышном положении. И неудивительно, что при использовании одинаковых операционных систем и приложений (что усугубляет значение этих цифр) вирус Blaster смог так успешно распространиться[8].

Расширяемость

Современные системы, которые строятся на основе виртуальных машин (VM), обеспечивают безопасность типов (type safety) и выполняют динамические проверки прав доступа (таким образом разрешается выполнение непроверенного переносимого кода), называют *расширяемыми системами* (extensible systems). Наиболее известные примеры — Java и .NET. Поскольку на расширяемую систему можно добавлять обновления или расширения, которые еще называют *переносимым кодом* (mobile code), то функциональные возможности такой системы могут постоянно увеличиваться. Например, виртуальная машина Java (Java Virtual Machine — JVM) может

[8] *Многие специалисты в области безопасности предполагают, что решению этой проблемы может способствовать разнообразие операционных систем и приложений, однако эксперименты показывают, что реализация этой идеи на практике намного сложнее, нежели на словах. — Прим. авт.*

задавать класс в пространстве имен и потенциально разрешать другим классам взаимодействовать с ним.

Большинство современных операционных систем поддерживают расширяемость с помощью динамически загружаемых драйверов устройств и динамически загружаемых модулей. В современных приложениях, таких как текстовые процессоры, клиенты электронной почты, программы для работы с электронными таблицами и Web-браузеры, расширяемость поддерживается с помощью сценариев, элементов управления, компонентов, динамически загружаемых библиотек и аплетов. Ни одно из этих средств не является новым. И действительно, программное обеспечение является основным способом расширения возможностей компьютеров, предназначенных для решения стандартных задач. Именно программы определяют работу компьютера и оригинальным способом расширяют его базовые возможности.

К сожалению, истинная сущность современных расширяемых систем усложняет задачу безопасности. Например, очень трудно предотвратить проникновение на компьютер вредоносного кода, замаскированного под расширение, т.е. расширение, которое позволяет расширить функциональные возможности системы (например механизм для загрузки классов Java), должно создаваться с учетом требований безопасности. Более того, исследование безопасности расширяемой системы должны проводиться намного тщательней, нежели цельной системы, не подверженной изменениям. Как можно проверить код, который только что доставлен? Эти и другие проблемы безопасности расширяемого кода подробно рассмотрены в книге *Securing Java* (Мак-Гроу и Фелтен, 1999).

Компания Microsoft резко перешла на переносимый код с внедрением своей платформы .NET Framework. Как показано на рис. 1.6, архитектура .NET имеет много общего с Java. Главное отличие заключается в наличии многоплатформенной поддержки. Но в любом случае, расширяемые системы будут использоваться и дальше. Вскоре сам термин *переносимый код* будет излишним, поскольку весь код станет переносимым.

Но, к сожалению, у переносимого кода, который предназначен для расширения возможностей программ, есть оборотная сторона. В некотором смысле вирусы и “черви” тоже можно назвать переносимым кодом. Вот почему исполняемые вложения в сообщения электронной почты и виртуальные машины, которые запускают код, внедренный на Web-страницы, становятся ночным кошмаром для специалистов по безопасности. Классические методы атак, включая распространение вирусов через дискеты и передачу инфицированных исполняемых файлов с помощью модемов, сегодня заменены электронной почтой и содержимым Web-страниц. Современные хакеры широко используют атаки с помощью переносимого кода. Вирусы и “черви” не просто распространяются по сети, они устанавливают потайные ходы, определяют тип системы и реализуют компрометацию компьютера, т.е. хакер получает зараженный компьютер в свое распоряжение.

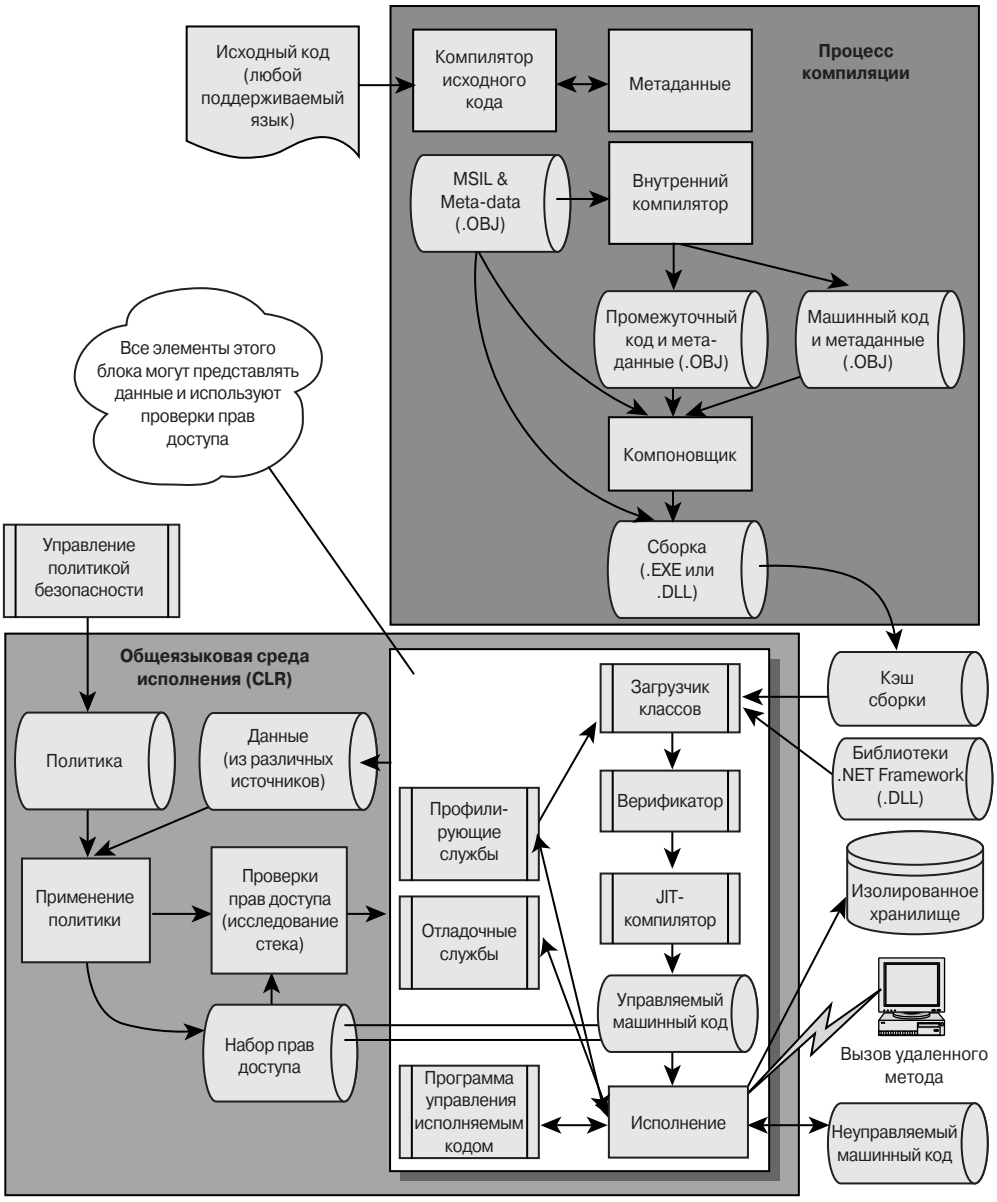

Рис. 1.6. Архитектура .NET Framework. Обратите внимание на архитектурное подобие с платформой Java: верификация, оперативная компиляция (JIT-компиляция), загрузка классов, подписание кода и виртуальная машина

"Золотая эра" вирусов наступила в начале 1990-х годов, когда они распространялись с помощью исполняемых файлов и кочевали с одного компьютера на другой с помощью дискет. "Червь" представляет собой особую форму вируса, который самостоятельно распространяется по сети. "Черви" являются очень опасной модификацией вирусов, особенно с учетом современного размаха использования сетей.

"Черви" стали активно распространяться по сетям в конце прошлого века, хотя многие опасные "черви" (разработанные хакерами-энтузиастами) так никогда и не будут запущены в сеть, а значит, никогда не будут исследованы. С момента появления первых "червей" технология создания этих программ сделала значительный шаг вперед. "Черви" позволяют хакеру провести "ковровое бомбометание" по всей глобальной сети, а также атаку на конкретное уязвимое место в максимальном масштабе. Это значительно усиливает суммарный эффект атаки и позволяет получить результаты, которые никогда не были бы достигнуты при поочередных атаках хакера вручную на разные компьютеры. В результате атак "червей" в большинство компьютерных сетей 1000 глобальных корпораций были внедрены потайные ходы. В сообществе хакеров ходят слухи о существовании так называемого *Списка успеха 500* (*Fortune 500 List*) — списка действующих потайных ходов в компьютерные сети 500 крупнейших компаний мира.

Один из первых вредоносных "червей", которому удалось поразить глобальную сеть и который широко использовался как средство взлома, был создан группой хакеров-единомышленников, которые называли себя *ADM* (*Association De Malfaiteurs*). "Червь" *ADM w0rm*[9] использует ошибку переполнения буфера в DNS-серверах[10]. После заражения "червем", взломанный компьютер начинает сканирование сети в поисках других уязвимых серверов. Этот "червь" заразил десятки тысяч компьютеров, в прессе появилось немало заметок о его распространении. Некоторые из первых жертв атаки "червя" ADM остались зараженными по сей день. Особое беспокойство вызывает тот факт, что уязвимое место, использованное для распространения "червя" ADM, было изучено весьма поверхностно. При этом структура "червя" позволяет без труда добавить в него другой код для проведения атаки. Таким образом, "червь" сам по себе является расширяемой системой. Остается только догадываться, сколько версий этого "червя" сейчас "разгуливают" по Internet.

В 2001 году сообщение о знаменитом сетевом "черве" под названием Code Red вышло на первых полосах газет. Этот "червь" поразил сотни тысяч серверов, в том числе компьютеры, на которых был запущен ISS-сервер от компании Microsoft. При этом использовалась очень простая, но, к сожалению, широко распространенная ошибка программного обеспечения[11]. Как это бывает обычно при успешной и разрекламированной атаке "червя", появилось несколько его модификаций. "Червь" Code Red заражает сервер, который затем начинает сканирование сети в поисках новых целей атаки. В оригинальной версии Code Red преимущественно осуществлялось сканирование систем, которые находились поблизости от пораженного хоста. Этим ограничивалась скорость распространения этого "червя".

Совсем немного времени прошло после дебюта Code Red в глобальной сети, как появилась его усовершенствованная версия, в которой использовался улучшенный алгоритм сканирования сети. Это еще больше ускорило темпы распространения Code Red. Успех "червя" Code Red основан на очень простом недостатке в про-

---

[9] *Архивный файл* `ADMw0rm-v1.tar` *можно найти на различных Web-сайтах в Internet. В этом файле содержится исходный код "червя" ADM w0rm, впервые появившегося весной 1998 года.*

[10] *Более подробно об ошибках в пакете BIND можно прочесть по адресу* `http://www.cert.org/advisories/CA-98.05.bind_problems.html`.

[11] *"Червь" Code Red использует ошибку переполнения буфера в библиотеке* `idq.dll`, *компонента ISAPI.*

граммном обеспечении, которое широко использовалось более 20 лет. Повсеместное наличие этого недостатка на Windows-системах, безусловно, помогло быстрому распространению Code Red.

Подобные результаты были достигнуты еще несколькими "червями", включая Blaster и Slammer. Мы более подробно рассмотрим вредоносный код и его влияние на использование программного обеспечения далее в этой книге. Мы также изучим средства хакеров, с помощью которых они взламывают чужие программы.

Взаимодействие по сети

Рост масштаба сетей и глобальное подключение компьютеров к Internet привело к одновременному увеличению количества возможных атак, причем методы организации этих атак максимально упростились. Даже небольшие ошибки в программах стали очень быстро распространяться по всей сети и выводить из строя огромное количество компьютеров. Например, множество таких фактов приводится в списке рассылки COMP.RISKS и в книге *Computer-Related Risks* (автор Нойманн, 1995).

Поскольку доступ по сети не требует непосредственного присутствия человека, то запуск автоматических атак осуществляется довольно просто. Автоматически запускаемые атаки изменили сам характер угроз в информационном мире. Рассмотрим первые формы хакинга. В 1975 году щелающие выполнять бесплатные телефонные звонки могли воспользоваться устройством под названием Blue Box. Такие устройства продавались в студенческих городках, но еще нужно было найти продавца. Кроме того, эти устройства тоже стоили денег. Таким образом, только ограниченный круг людей владел устройствами Blue Box и, поэтому угроза распространялась крайне медленно. Сравним это с нашим временем. Если обнаруживается уязвимое место, которое позволяет злоумышленникам получить доступ, например, к платным телепрограммам, то информация немедленно опубликовывается в Internet и за несколько часов миллионы людей могут загрузить себе программу атаки.

Рис. 1.7. Это сложный мобильный телефон от компании Nokia. После того как стало возможным обмениваться сообщениями электронной почты и работать в Web с помощью мобильных телефонов, последние стали более вероятной целью для атак хакеров

Постоянно разрабатываются новые протоколы и среды обмена информацией. Как результат этого процесса появляется код, который никогда не будет проверен. Например, в ваш мобильный телефон встроены операционная и файловая системы. На рис. 1.7 показан новый усовершенствованный мобильный телефон. Представьте, что будет, если вирус поразит сеть связи мобильных телефонов.

Сети, которые соединяют множество компьютеров, особенно уязвимы для атак вирусов и простоев в работе, возникающих в результате этих атак. Парадокс в том, что увеличение количества соединений является классическим методом повышения доступности и надежности, но увеличение количества путей доступа непременно способствует "живучести" вируса.

Наконец, наиболее важным аспектом для глобальной сети является экономика. Экономика любого государства связана с экономикой других стран. Миллиарды электронных долларов передаются по сетям каждую секунду, триллионы — каждый день. Одна только сеть SWIFT, которая объединяет 7000 международных коммерческих организаций, ежедневно осуществляет переводы триллионов долларов. В этой взаимосвязанной системе соединено огромное количество программных систем, которые постоянно обмениваются потоками цифр. Государства и транснациональные корпорации зависят от этой современной "фабрики" информации. Ошибка в этой системе способна привести к огромной катастрофе, дестабилизировать экономику целых государств за несколько секунд. А если произойдет серия последовательных ошибок, то весь виртуальный мир может оказаться в состоянии коллапса. Вероятно, одной из целей террористического акта 11 сентября 2001 года было разрушение мировой банковской системы. Это реальная современная угроза, с которой нам предстоит столкнуться лицом к лицу.

Широкая общественность никогда не узнает, сколько атак на программное обеспечение проводится ежедневно на компьютерные сети финансовых организаций. Банки очень хорошо охраняют свои секреты. Учитывая тот факт, что многие компьютеры, которые принадлежали арестованным хакерам и террористам, были конфискованы, можно смело предположить, что в круг их деятельности наверняка входили компьютерные сети банковских учреждений.

Выводы

Как видим, три тенденции к увеличению сложности систем, к добавлению возможностей расширения и повсеместного объединения компьютеров в сети, делают проблему обеспечения безопасной работы компьютерных сетей более острой, чем когда-либо ранее. К сожалению специалистов по безопасности, эти три проблемы программного обеспечения значительно упрощают его взлом!

В марте 2003 года Computer Security Institute выпустил свой восьмой ежегодный обзор, в котором сообщалось, что 56% из 524 крупных компаний и организаций понесли финансовые потери в результате взломов программного обеспечения, которые произошли за прошедший год. Большая часть этих взломов была проведена через Internet. Убытки, понесенные 251 компанией в результате действий хакеров, в сумме составили 202 млн. долл. Подобные факты неоспоримо свидетельствуют о растущей важности проблемы безопасности программного обеспечения.

Десять перспективных направлений

Попытаемся определить десять наиболее перспективных направлений в развитии программного обеспечения.

1. Исчезновение операционных систем.
2. Широкое распространение беспроводных сетей.
3. Появление встроенных систем и специализированных вычислительных устройств.
4. Действительно распределенные вычисления.
5. Эволюция "объектов" и компонентов.
6. Фабрика информации (возможность повсеместного доступа к вычислительным ресурсам).
7. Искусственный интеллект, управление знаниями и оперативные вычисления.
8. Оплата за отдельный байт (или цикл работы процессора, или функцию).
9. Высокоуровневое проектирование/средства программирования.
10. Запуск программ в зависимости от места их выполнения.

Из-за короткого жизненного цикла программного обеспечения его взлом выполняется довольно просто. Очевидно, что эволюция программного обеспечения не будет замедляться. Уже одно только это обстоятельство делает чрезвычайно сложным создание безошибочно работающих программ, а злоумышленникам предоставляется в результате широкое поле деятельности.

Что такое безопасность программного обеспечения?

Определение принципов работы программного обеспечения является процессом, который включает в себя установку и систематизацию правил политики, а затем реализацию этой политики с помощью соответствующей технологии. Не существует никаких волшебных или универсальных средств для обеспечения безопасной работы программ. Усовершенствованные методы проверки кода очень полезны для выявления ошибок на этапе реализации, но они не заменяют проверки на практике. Усовершенствованные методы обеспечения безопасности приложений незаменимы, когда нужно проверить, что выполняется только разрешенный код, но совсем не годятся для выявления ошибок в исполняемых файлах.

В конце прошлого века на рынке средств по обеспечению безопасности произошел резкий всплеск, когда появилось множество "решений по безопасности". Пользователями были потрачены крупные суммы. Однако после многих лет использования брандмауэров, антивирусных программ и средств криптографии, количество программ атаки продолжает увеличиваться. При этом растет и количество уязвимых мест (рис. 1.8).

На самом деле брандмауэры довольно слабо защищают компьютерные сети. Системы обнаружения вторжений пронизаны ошибками, что приводит к большому числу ложных тревог. Потрачены годы труда огромного количества специалистов, но программный код по-прежнему взламывают. Почему так происходит? На что же мы тратили деньги все это время?

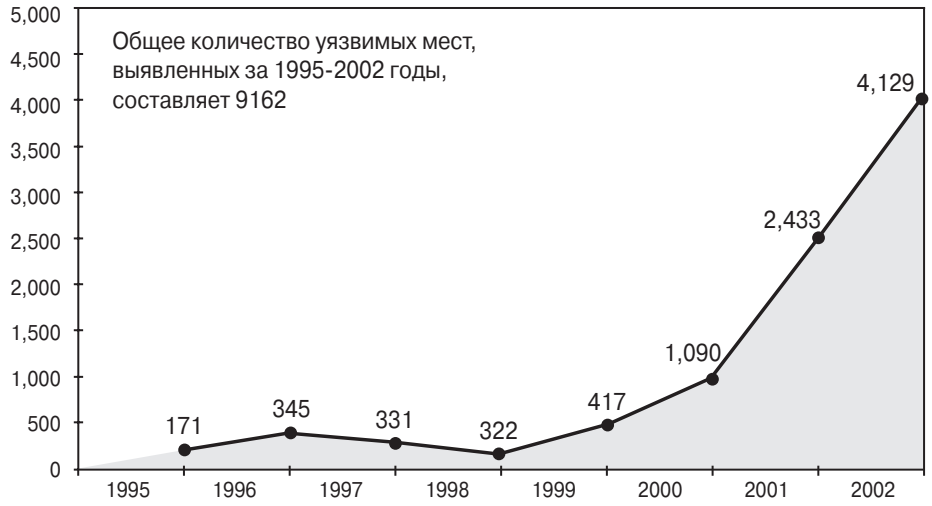

Рис. 1.8. Количество уязвимых мест, согласно сведениям CERT/CC, продолжает увеличиваться

Основной фактор, благодаря которому продаются программы безопасности, заключается в хорошей рекламе, например: "Просто купите этот продукт, и мы возьмем на себя все ваши заботы". Итак, вы купили программу, установили ее и... Большинство защитных механизмов не имеют никакого отношения к корню проблемы — ошибкам в программном обеспечении. Вместо этого они работают в режиме ответа. *Запретить прохождение пакетов к этому или другому порту. Отслеживать файлы, которые содержат заданный шаблон. Отбрасывать фрагменты пакетов и пакеты, размер которых превышает предельное значение, без просмотра.* К сожалению, фильтрация сетевого трафика — не самый лучший способ решения проблем. Основная проблема заключается в программном обеспечении, которое обрабатывает пропущенные пакеты.

Мы можем предположить, что существуют ошибки в ежедневно используемом программном обеспечении. И действительно, программное обеспечение играет объединяющую роль в большинстве коммерческих организаций. Конечно, мы пытаемся лишить злоумышленников доступа к уязвимому программному обеспечению, но проблема глубже и ее не решить с помощью традиционных барьеров. Чтобы работать быстрее в эпоху Internet, приходится быстрее обмениваться данными. Это означает появление большего количества Web-служб и внешних интерфейсов, т.е. еще больше приложений будут доступны удаленно (в том числе и для хакеров). Открытыми для атак становятся даже обычные пользователи посредством программного обеспечения, работающего в их домах, машинах и даже в карманах. Под угрозой атаки находится почти каждый из нас.

Резюме

Взлом программного обеспечения был возведен в ранг искусства, это действительно непростая задача. Сначала нужно понять, какую задачу решает фрагмент кода. Часто это можно сделать только по результатам работы. Иногда программный

код можно разделить на несколько фрагментов и изучить их отдельно. Иногда предназначение программного кода определяется с помощью некорректных входных данных. Этот код можно дизассемблировать или декомпилировать. Иногда (особенно если вы специалист по безопасности, а не хакер) с его помощью можно изучить проект программы и архитектурные проблемы.

Эта книга посвящена искусству взлома программного обеспечения. В некотором смысле она вкладывает оружие в руки читателей. В частности, хакерам[12]. Для неопытных пользователей или же лентяев, которые хотят украсть что-то ценное, эта книга будет бесполезной, потому что авторы не описывают способов атак наподобие "просто добавь воды"[13]. Эта книга не для тех, кто хочет просто взломать чужую сеть, не задумываясь о том, как это происходит. Эта книга о том, как научиться взламывать программные системы, или, следуя нашей аналогии, как делать оружие своими руками.

Большая часть программных систем является частной собственностью. Это сложные системы, созданные по оригинальным проектам. По этой причине взлом программного обеспечения далеко непростая задача. Вот почему мы считаем, что эта книга необходима, хотя, вероятно, мы тоже только немного приоткроем завесу тайны хакинга.

Это опасная книга, но окружающий мир — тоже опасное место. Осведомлен — значит вооружен. Многие могут нас раскритиковать за разглашение изложенной информации, но мы придерживаемся мнения, что сохранение правды в тайне только приведет к ухудшению ситуации. Мы надеемся, что, попав в хорошие руки, эта книга реально поможет нашим читателям устранить многочисленные проблемы, связанные с защитой программного обеспечения.

---

[12] *Мы используем термин хакер в традиционном смысле, который определен в словаре хакера.*

Хакер: (в оригинальном смысле — тот, кто делает мебель с помощью топора). сущ. 1. Человек, который старается разобраться в деталях работы программного обеспечения и ищет способы расширения возможностей программ, в отличие от большинства пользователей, которые удовлетворяются минимальными возможностями. 2. Фанат программирования, которому больше нравится писать программы, чем теоретизировать о возможностях программирования. 3. Способный человек. 4. Человек, который быстро научился программировать. 5. Эксперт по конкретной программе или тот, кто часто работает с этой программой. 6. Эксперт или специалист в любой области. 7. Человек, который получает интеллектуальное удовольствие от преодоления проблем или сложностей. 8. Зловредный человек, который стремится украсть ценную информацию.

Словарь хакера на английском языке доступен по адресу `http://www.mcs.kent.edu/docs/general/hackerdict`. *— Прим. авт.*

[13] *Термин* script kiddies*, или хакер-новичок, используется для обозначения людей, которые пытаются взломать чужие компьютеры с помощью заранее подготовленных сценариев, созданных и распространяемых другими злоумышленниками. Большинство новичков не знают, как работают программы взлома, они только знают, что эти программы работают. Эти люди не обладают реальными знаниями или навыками, а пользуются программами взлома опытных хакеров так, как ребенок пользуется заряженным пистолетом. — Прим. авт.*