**Лекция №10:** **Подготовка и укрепление веб-сервера**

**Цель: н**аучить студентов принципам безопасной настройки веб-сервера, включая подходы к защите учетных записей, предотвращению использования слабых или несуществующих паролей, а также методам укрепления веб-сервера для минимизации рисков и уязвимостей.

##### Учетные записи со слабыми или несуществующими паролями

Большая часть операционных систем изначально настроена на использование паролей, что является первой и единственной преградой на пути злоумышленника. Если злоумышленник определит имя и пароль учетной записи, он сможет войти в сеть.

Во многих системах содержатся встроенные учетные записи, учетные записи по умолчанию с паролями по умолчанию. Такие учетные записи имеют один и тот же пароль во всех установленных копиях программного продукта. Злоумышленники, как правило, проверяют их наличие, так как они хорошо известны в хакерских кругах. Удивительно большое число администраторов допускают существование этих учетных записей в своих системах.

Легко угадываемые пароли представляют собой большую проблему, так как злоумышленники используют "грубые" атаки (т.е. атаки, заключающиеся в попытках угадать пароль) или атаки с применением словарей паролей, генерирующих слова и комбинации символов. Если длина пароля мала, и он состоит только из букв алфавита, то его взлом для хакера – не проблема.

Для предотвращения таких атак администраторам и пользователям нужно избегать использования в паролях имен людей, домашних животных или названий мест. Сложность угадывания пароля зависит от количества символов в пароле: чем больше символов, тем труднее угадать пароль. Используйте при работе так называемые сильные пароли, состоящие, как минимум, из восьми символов и являющихся комбинациями чисел, специальных знаков пунктуации и букв алфавита в верхнем и нижнем регистрах (если система позволяет указывать пароли с разным регистром символов).

##### Отсутствие или незавершенность резервных копий

При нарушении защиты система восстановления требует наличия "свежих" резервных копий и проверенных способов восстановления данных. В одних компаниях ежедневно резервируются данные, но их работоспособность не проверяется. В других – создается политика резервирования данных, но не предусматривается политика восстановления. Подобные ошибки проявляются уже после того, как хакер проник в сеть и уничтожил или повредил данные. Всегда следует разрабатывать политику резервирования и восстановления, проверять ее и улучшать!

##### Большое количество открытых портов

Как легальные пользователи, так и злоумышленники подключаются к системам через открытые порты. Чем больше портов открыто, тем вероятнее, что кто-нибудь сможет подключиться к компьютеру. Следует иметь несколько открытых портов на брандмауэре, а именно: порт 80 для HTTP на веб-сервере, порты 25 и 110 для SMTP и POP. Из-за этих портов возникают проблемы, если не знать о том, что они существуют и находятся в открытом состоянии. Если оставить открытый порт незащищенным, злоумышленнику понадобится лишь средство сканирования для его обнаружения, после чего он выполнит через этот порт эксплоит, входящий в комплект хакерского программного обеспечения.

Наилучшим способом избавления от ненужных портов является закрытие всех портов до единого с последующим открытием только необходимых портов по очереди при настройке системы.

##### Отсутствие фильтрации пакетов для проверки корректности адресов входящих и исходящих пакетов

Хакеры, как правило, маскируют свой IP-адрес адресом узла внутренней сети, так как в этом случае он действует под видом полноправного члена сети и его проникновение трудно распознать. Внутренние пользователи обычно имеют привилегированный доступ к информации и системам, недоступным для внешних пользователей. Поэтому внешний пользователь, работающий под видом внутреннего, сможет использовать эти привилегии для выполнения злоумышленных действий. Такой тип атак называется спуфингом (*spoofing*) или подменой пользователя. Данный тип атаки предотвращается с помощью IP-фильтрации, о которой пойдет речь в ["Особенности процесса разработки"](https://www.intuit.ru/studies/courses/1002/122/lecture/3527).

##### Несуществующий или неправильно настроенный журнал

Одной из аксиом безопасности является высказывание: "Идеальный случай – предотвращение всех атак, но обнаружение злоумышленника жизненно необходимо". Если вы подверглись атаке, то без журналов вряд ли сможете узнать, что злоумышленники сделали с системой. Без журнала вашей организации придется либо перезагружать операционные системы с исходных носителей в надежде, что зарезервированные данные не повреждены, либо продолжать использование системы, опасаясь, что злоумышленник все еще контролирует ее. Нельзя обнаружить атаку, если не знать, что происходит в сети. Журналы содержат подробные сведения о том, что произошло, какие системы были атакованы, а какие подверглись несанкционированному доступу. О журналах будет идти речь в ["Аудит и журналы безопасности"](https://www.intuit.ru/studies/courses/1002/122/lecture/3525).

##### Уязвимая программа CGI

Большая часть серверов, включая Microsoft IIS, поддерживает программы общего шлюзового интерфейса (Common Gateway Interface, CGI), обеспечивающие работу с веб-страницами и возможность поиска и сбора данных. К сожалению, несанкционированный доступ к этим программам позволит получить прямой доступ к операционной системе веб-сервера и запустить программу с привилегиями программного обеспечения самого веб-сервера. В атаках на взлом CGI используются их уязвимые места для повреждения веб-страниц, хищения информации о кредитных картах, установки "черных ходов" для дальнейших вторжений.

В зависимости от использования сценариев и/или CGI предпринимаются меры по защите. Надстройки безопасности Microsoft устраняют некоторые уязвимые места IIS, возникающие при использовании сценариев. В ["Безопасность активного содержимого"](https://www.intuit.ru/studies/courses/1002/122/lecture/3537)рассказывается об этих мерах.

##### Злоумышленный или вредоносный код

Веб-сайты уязвимы для атак, заключающихся в распространении вирусов, червей или "троянских коней". Злоумышленные коды могут поступать из самых различных *источников: сообщений* электронной почты, при загрузке файлов, модифицированных адресов URL и др. Несмотря на то, что данная проблема является актуальной для всех систем, ее решение, как правило, специфично для каждой программной платформы. Например, многие вирусы воздействуют только на платформы Windows. Далее мы расскажем о вредоносных кодах.

##### Уязвимые места, зависящие от платформы

В дополнение к общим уязвимым местам каждая вычислительная платформа имеет свой собственный уникальный набор уязвимых мест. Из-за своей широкой популярности Windows 2000 и IIS подвергаются более частым атакам, и их уязвимые места представляют большую опасность. Далее мы расскажем обо всех уязвимых местах, известных на момент написания книги. Все они устраняются с помощью установки Microsoft Service Pack 2 для Windows 2000, однако появляются новые опасности, поэтому регулярно проверяйте перечень уязвимых мест на веб-сайте SANS по адресу [http://www.sans.org/top20.htm](http://www.sans.org/top20.htm" \t "_blank).

##### Угроза универсального кода Unicode (прохождение папок веб-сервера)

Универсальный код Unicode представляет собой производственный стандарт программного обеспечения, облегчающий адаптацию программных продуктов для различных операционных систем и переводящий содержимое веб-сайта (программы) на различные языки. В данном стандарте каждому символу присвоен уникальный номер независимо от платформы, программы или языка. Стандарт Unicode одобрен большинством производителей программного обеспечения, включая Microsoft, а исходный код Unicode является частью Windows 2000 и IIS.

Использование стандарта Unicode в продуктах Microsoft послужило причиной возникновения уязвимого места. Отправив на сервер IIS особым образом написанный адреса URL, содержащий некорректную последовательность Unicode UTF-8, злоумышленник может вызвать беспорядочный проход системы по каталогам и выполнение случайных сценариев. Эта атака также известна под названием *атаки на прохождение каталогов* (directory *traversal* *attack*). Если хакер осуществит эту атаку, то сможет запустить свою собственную программу, внедренную на сервер.

##### Переполнение буфера

В Windows 2000 и IIS обнаружено большое число уязвимых мест, через которые осуществляются атаки на переполнение буфера. "Прорехи" в защите также позволяют проводить атаки, вызывающие сбой в работе веб-сервера IIS, выполнение произвольного кода, использование целевого веб-сервера в качестве инструмента для атаки на стороннюю систему. Ниже приведен краткий обзор таких уязвимых мест.

* При установке IIS инсталлируются некоторые дополнения для Internet Services Application Programming Interface (*ISAPI*) (Интерфейс прикладного программирования интернет-сервера), позволяющие расширять возможности IIS при помощи динамически подсоединяемых библиотек (DLL). Выявлено, что некоторые DLL-библиотеки (например, idq.dll) содержат ошибки программирования, позволяющие проводить атаки на переполнение буфера и получать полный контроль над веб-сервером IIS. В Windows 2000 файл с именем idq.dll определен как уязвимое место, позволяющее произвести переполнение буфера в Microsoft Index Server 2.0 (Сервер индексов) и Indexing Service (Служба индексации). Переполнение буфера возможно на Windows 2000 Server, Advanced Server и Server *Data Center* Edition с установленным компонентом IIS 5.0. Уязвимая библиотека DLL имеется и в Windows 2000 Professional, но не устанавливается по умолчанию.
* Устанавливаемое вместе с IIS 5.0 расширение ASP *ISAPI* допускает переполнение буфера. С его помощью удаленный взломщик может вызвать сбой сервера или, при соответствующей настройке, выполнить любой код, выбранный им самим. В установках по умолчанию код будет выполняться с привилегиями IWAM\_ *machinename*, присущими локальному непривилегированному пользователю (он также называется гостевой учетной записью интернета, о которой пойдет речь в ["Учетные записи, аутентификация и политика безопасности"](https://www.intuit.ru/studies/courses/1002/122/lecture/3523)).
* Вторая возможность переполнения буфера выявлена в механизме передачи данных ASP *ISAPI*. Оно приводит к таким же последствиям, как и первая, и так же влияет на приложение IIS Web Server.
* Третья возможность переполнения буфера заключается в способе проверки безопасности разделителей перед анализом полей заголовков HTTP в IIS 5.0 и 5.1. Возможно прослушивание этой проверки с выдачей информации о существовании корректных разделителей, в то время как на самом деле их не существует. Таким образом, удаленный взломщик может отправить на сервер специальный HTTP-запрос, который вызовет переполнение свободных буферов. В случае успеха это вызовет сбой в работе сервера или выполнение произвольного кода. В конфигурации по умолчанию IIS 5.0 и 5.1 код выполнится с привилегиями, соответствующими IWAM\_ *machinename*.
* Уязвимое место возникает при проверке корректности имени файла и его размера на сервере. С помощью особого пользовательского *запроса удаленный* злоумышленник может инициировать принятие слишком длинного имени файла, избежать проверки безопасности и переполнить статический буфер, что вызовет отказ в обслуживании или выполнение произвольного кода с привилегиями IWAM\_ *machinename* на атакуемом сервере.
* Уязвимое место имеется в компоненте IIS 5.0, поддерживающем входящие запросы .htr. При инсталляции IIS по умолчанию устанавливается расширение HTR *ISAPI*, представляющее собой программный интерфейс приложения для работы со сценариями. Функцией HTR *ISAPI* является обработка паролей пользователей на базе сети. С помощью нескольких особым образом созданных файловых запросов .htr удаленный злоумышленник может вызвать переполнение буфера и выполнить произвольный код на атакуемой системе. Эта атака вызовет отказ в обслуживании или, в некоторых случаях, позволит ему получить привилегии IWAM\_ *machinename* на атакуемом сервере.
* Отказ в обслуживании (DoS) может вызвать уязвимое место, содержащееся в методе поддержки ошибок некоторых фильтров *ISAPI*. Посредством отправки запроса URL, превышающего установленную в фильтре *ISAPI* длину буфера, злоумышленник может вызвать нарушение доступа в службах IIS.
* В IIS имеется уязвимое место, возникающее при работе служб IIS. Посредством определенного запроса о состоянии создается условие для возникновения ошибки, в результате чего нарушается доступ в службе FTP и службах сети. Это произойдет только в том случае, если на веб-сервере включены службы FTP.
* Веб-серверы IIS подвержены трем отдельным угрозам действия межсайтовых сценариев (Cross-Site Scripting, CSS). CSS представляет собой *пассивную атаку*, основанную на социальном инжиниринге. Она выполнится, если злоумышленник уговорит пользователя посетить веб-страницу и перейти по соответствующей ссылке, осуществляющей злоумышленные действия атакующего, или открыть вредоносное сообщение электронной почты в формате HTML. CSS позволяетразместить вредоносный код внутри веб-запроса, отправляемого с компьютера пользователя и передаваемого на сторонний веб-сайт. Вредоносный код запускается и выполняется с настройками безопасности и в зоне безопасности, применяемой к стороннему веб-сайту. Такие действия позволят злоумышленнику получить данные с веб-сайта, если иначе это сделать невозможно.

##### NetBIOS: незащищенные общие сетевые ресурсы Windows

Возможность *File Sharing* (Общий доступ к файлам), имеющаяся в Windows, позволяет открывать общий доступ к файлам в сети. *File Sharing* не защищен от доступа из интернета и представляет возможность несанкционированного доступа к системным файлам или ко всей файловой системе.

*File Sharing* использует протокол Server Message Block (SMB) (Блок серверных сообщений), позволяющий применить общий доступ к файлам Windows для получения секретной информации о системе, т.е. данных о пользователях и группах (имена пользователей, даты последнего входа, политика паролей, информация о сервисе удаленного доступа), данных о системе и ключах реестра. Доступ выполняется через соединение NULL-сеанса ("недействительного сеанса") со службой NetBIOS Session Service. Данная информация очень важна для хакеров при угадывании паролей или при осуществлении атак "грубой силы" на систему Windows.

##### Утечка информации через соединения недействительной сессии

В системах Windows NT и Windows 2000 многие локальные службы работают в учетной записи SYSTEM или LocalSystem (для Windows 2000). Эта учетная запись используется в критических системных операциях. При получении одним компьютером системных данных с другого компьютера учетная запись SYSTEM открывает недействительную сессию соединения для этого компьютера.

Учетная запись SYSTEM имеет неограниченные привилегии и не требует пароля при доступе, поэтому пользователь не сможет войти в систему под этой учетной записью. Учетной записи SYSTEM иногда требуется доступ к информации, расположенной на других компьютерах и представляющей собой общие файлы SMB, имена пользователей и т.д. Так как она не может войти в систему с помощью идентификатора пользователя и пароля, то использует для получения доступа недействительный сеанс соединения. К сожалению, с помощью недействительного сеанса осуществляют вход в систему и злоумышленники. Это относится как к Windows 2000, так и к Windows NT.

##### Уязвимость зашифрованных паролей в Security Accounts Manager (SAM)

Несмотря на то, что большинству пользователей Windows не требуется поддержка *LAN Manager* (Диспетчер локальной сети), в системах Windows NT и Windows 2000 в SAM (Диспетчер безопасности учетных записей) наряду с другими паролями по умолчанию хранятся зашифрованные пароли *LAN Manager*. Так как *LAN Manager* использует более уязвимую схему шифрования, чем остальные продукты Microsoft, пароли *LAN Manager* можно взломать за относительно короткий промежуток времени. Даже пароли, зашифрованные с помощью мощных алгоритмов, взламываются менее чем за месяц.

Уязвимость паролей в *LAN Manager* заключается в том, что пароли всегда дополняются или урезаются до 14 символов и разбиваются на два семисимвольных отрезка; эта схема и упрощает их взлом. Программе по взлому паролей нужно взломать только два семисимвольных пароля даже без проверки символов нижнего регистра. *LAN Manager* уязвим и к перехвату зашифрованных паролей. Это относится как к Microsoft Windows NT, так и к Windows 2000.

**Важно**. Не откладывайте установку обновлений Microsoft Service Pack и Security Update, которые устраняют эти уязвимые места, если действующий веб-сайт от них не защищен. Перечень уязвимых мест открыт для общего пользования, поэтому можно предположить, что хакеры изучили его и имеют дополнительную информацию, доступную в хакерских кругах. Не позволяйте хакерам опережать вас!

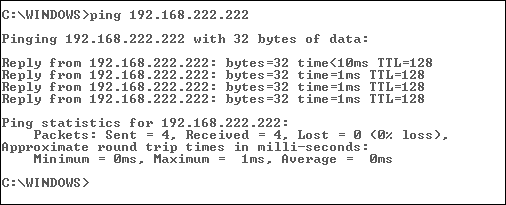
### Поиск уязвимых мест

Хакеры знают об уязвимости атакуемых ими систем. Большую часть времени при подготовке и осуществлении атаки хакеры отводят под составление перечня систематических процедур, которые помогут им выяснить эти уязвимые места с наименьшими затратами. Наверняка злоумышленники сначала проверят систему-жертву на наличие открытых портов NetBIOS. После этого они просмотрят учетные записи и пароли по умолчанию, которые не были изменены с момента установки системы. Следовательно, если вы не умеете эффективно устранять "дыры", *хакер* рано или поздно обнаружит их посредством рекогносцировки.

#### Возможно, за вами следят...

В ["Угрозы безопасности в интернете"](https://www.intuit.ru/studies/courses/1002/122/lecture/3517)говорилось, что одни хакеры находят свои цели случайным образом, а другие концентрируют внимание на конкретных целях, используя в качестве "орудия" отправку массовых ping-запросов. Координаты ответивших компьютеров помещаются в список потенциальных жертв. Если хакер идентифицировал сервер и "нацелился" на него, он выполнит ping-запрос (см. [рис. 2.3](https://www.intuit.ru/studies/courses/1002/122/lecture/3519?page=5" \l "image.2.3)), чтобы убедиться в его работе, перед дальнейшей разведкой или перед непосредственной атакой.

Сканирование является следующим этапом сбора информации и осуществляется изнутри или извне компьютерной сети. Это эффективный метод рекогносцировки, так как виртуально все приложения и службы, использующие интернет-коммуникации, осуществляют прием и отправку данных через один и тот же набор стандартных *коммуникационных протоколов*. Сканирование помогает хакеру определить, какие порты на целевом компьютере открыты, и какие приложения и службы запущены.

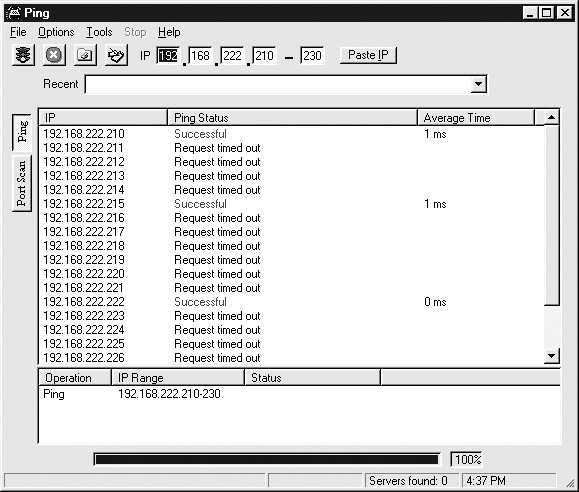


**Рис. 2.3.**Перед выполнением атаки хакер осуществляет ping-запрос сервера-жертвы

#### Принципы работы ping-запросов и сканирования

Ping-запросы, отдельные или массовые, используют атрибуты протокола интернета для получения информации. Операционные системы всех компьютеров имеют встроенную утилиту для осуществления ping-запросов. Утилита по отправке ping-запросов отправляет пакет запроса ICMP ECHO указанной системе и ожидает ответа ICMP ECHO. Если ответ получен, ping-утилита выводит сообщение о том, что система-жертва работает. Если ответ не получен, выводится соответствующее сообщение. Для осуществления массовых ping-запросов необходима программа, осуществляющая их отправку по адресам указанного диапазона.

Некоторые средства отправки ping-запросов и адреса URL, необходимые для их загрузки, указаны в [табл. 1.2](https://www.intuit.ru/studies/courses/1002/122/lecture/3519?page=5). Системы, приславшие ответ, отображаются в виде "живых" адресов (см. [рис. 2.4](https://www.intuit.ru/studies/courses/1002/122/lecture/3519?page=5" \l "image.2.4)).



**Рис. 2.4.**Данные, полученные путем массового ping-запроса (скриншот программы Grims Ping для Windows 2000)

Сканирование посредством ping-запросов можно обнаружить. По этой причине осторожные хакеры, как правило, избегают использования случайного ping-сканирования. Если есть такая возможность, они выполняют ping-запросы избирательно в надежде, что отдельный ping-запрос не будет замечен, или используют другие формы рекогносцировки.

Простым способом защиты от внешних ping-запросов является блокирование ping-сканирования посредством фильтрации ICMP на сетевых экранах и маршрутизаторах. Администраторы, для которых проблема безопасности является одной из главных, обычно настраивают эти устройства на отклонение всех ping-запросов, поступающих извне. Но такая защита работает до тех пор, пока сетевой экран не будет взломан.

При сканировании портов также используются атрибуты TCP, UDP и ICMP, но выполняются другие действия. Сканирование портов осуществляется программой для сбора информации о состоянии портов TCP и UDP системы-жертвы. Сканер отправляет пакет данных (зонд) на порт с последующим ожиданием ответа. По полученному ответу определяется, является ли порт открытым, занятым или закрытым, он содержит косвенные данные о том, какое приложение использует данный порт. Ниже приведен пример сканирования, направленного на отдельный компьютер, с помощью утилиты Nmap.

Interesting ports on (xxx.xxx.x.xx):

Port State Protocol Service

21 open tcp ftp

80 open tcp http

В зависимости от используемой программы сканируются отдельные порты отдельного компьютера, выполняется массовое сканирование портов и компьютеров. В интернете доступно множество бесплатных средств сканирования. Самой популярной является программа Nmap, доступная как для Microsoft Windows, так и для UNIX. Часто используется программа Nessus. Обе программы содержат широкий набор возможностей сканирования для зондирования больших и малых компьютерных сетей.

Сканирование порта позволяет с определенной степенью конфиденциальности выяснить, какие приложения и службы выполняются на компьютере-жертве. Например, если открыты порты 21 и 80, хакер может предположить, что на компьютере работают службы протоколов FTP и HTTP соответственно. Это достаточно весомая информация, так как каждый порт, ожидающий приема данных, является потенциальным каналом связи с компьютером-жертвой, а каждое приложение или служба представляет собой точку атаки, если хакер знает об уязвимых местах системы (или имеет соответствующую программу).

##### Некоторые методы сканирования портов

Существует большое количество методов сканирования, однако в каждом конкретном случае требуется свой метод. Хакер ищет компромисс между затрачиваемым временем и уровнем секретности своих действий. В зависимости от обстоятельств хакер либо осуществляет действия безнаказанно и не боится быть обнаруженным, либо действует осторожно с применением каких-либо хитростей. Сканирование всесторонне использует возможности интернет-протокола.

**Совет**. При составлении следующего раздела использовались материалы *The Art of Port Scanning* ([http://www.insecure.org/nmap/nmap\_doc.html](http://www.insecure.org/nmap/nmap_doc.html" \t "_blank)) и *Remote OS Detection via TCP/IP Stack FingerPrinting* ([http://www.insecure.org/nmap/nmap-fingerprinting-article.html](http://www.insecure.org/nmap/nmap-fingerprinting-article.html" \t "_blank)), автором которых является Fyodor. Для получения более подробной информации посетите эти веб-сайты.

Ниже приведены описания нескольких методов сканирования TCP. Некоторые из этих методов предотвращают обнаружение и работают извне сети, если периметровая защита ограничена фильтрацией пакетов. В разделе "Проблема. TCP-соединения" приведена информация, с помощью которой вы разберетесь в терминах, если еще не знакомы с TCP/IP.

* Метод сканирования TCP ACK используется для идентификации активных веб-сайтов, которые не отвечают на стандартные ping-запросы ICMP. Программа сканирования отправляет пакеты ACK. Если порт открыт, то компьютер-жертва отправляет ответ в виде *RST*. Если ответ не получен, это значит, что рассматриваемый порт закрыт. Такой метод сканирования является прямым, он обнаруживается и блокируется *пакетной фильтрацией* (см. ["Особенности процесса разработки"](https://www.intuit.ru/studies/courses/1002/122/lecture/3527)).
* В методе сканирования TCP Connect используется *системный вызов* *connect* операционной системы для открытия подключения к каждому порту, находящемуся в режиме ожидания. Этот метод реализуется быстро, однако выполняемые действия обнаруживаются, если на компьютере-жертве ведется журнал.
* При использовании метода сканирования TCP SYN компьютеру-жертве отправляется пакет SYN. Если порт открыт, потенциально атакуемый компьютер отправит пакет SYN|ACK. В легальном запросе следующим шагом является отправка запроса на открытие соединения тем компьютером, который отправил пакет ACK. В данном случае, как только становится известно, что порт открыт, программа разрывает соединение. Такой метод сканирования зачастую не обнаруживается, так как соединение не устанавливается, и многие системы не заносят в журнал данные о несостоявшихся TCP-соединениях. Одни средства периметровой защиты блокируют такие пакеты, другие – нет.
* Метод сканирования TCP FIN заключается в отправке пакетов FIN по адресу системы-жертвы. На компьютере с системой UNIX закрытые порты ответят отправкой пакетов *RST*. Открытые порты проигнорируют пакеты, поэтому если система не отвечает, то, вероятно, рассматриваемый порт открыт. Тем не менее, порты Microsoft отправляют пакеты *RST*, даже если они являются открытыми. Таким образом, метод сканирования FIN не очень полезен при сканировании портов, однако позволит определить, установлена ли на удаленном компьютере система Windows. Некоторые сканеры операционных систем используют этот принцип в своей работе.
* В методе сканирования FTP *Bounce* используется протокол FTP для установки FTP-соединений. Если служба, используемая в этом методе, открыта, злоумышленник сможет использовать удаленный компьютер в качестве прокси-сервера для сканирования другой цели без обнаружения и выявления своего адреса.

###### ПРОБЛЕМА. TCP-соединения

Если вы знакомы с принципами установки соединений TCP, то вам будет легче разобраться в способах реализации атак с использованием протоколов. Для надежной транспортировки данных узлы TCP устанавливают между собой сеансы соединения посредством механизма "трехстороннего рукопожатия".

Механизм подключения использует проверочные биты, имеющиеся в передающихся между узлами пакетах. Ниже приведен перечень проверочных битов с указанием назначения каждого из них.

* **ACK** – подтверждает успешный запрос на установку соединения.
* **RST** – вызывает принудительный сброс соединения.
* **SYN** – осуществляет синхронизацию последовательных счетчиков соединения.
* **FIN** – сообщает об отсутствии данных и закрывает соединение.

"Трехстороннее рукопожатие" проводит синхронизацию соединения с обоих концов, что позволяет обеим сторонам согласовываться по начальным номерам последовательности. Этот механизм гарантирует готовность обеих сторон к передаче данных и то, что стороны знают о своей готовности. Необходимо, чтобы пакеты не передавались вообще или не передавались повторно во время установки сессии или после ее завершения.

Каждый узел *случайным методом* выбирает номер последовательности, используемый для учета байтов в потоке отправляемых и принимаемых данных. После этого вступает в действие механизм "трехсторонего рукопожатия", работа которого заключается в следующем.

1. Первый узел (узел А) инициирует соединение посредством отправки пакета с начальным *номером последовательности* (Х) и битом SYN для обозначения запроса на соединение.
2. Второй узел (узел B) принимает SYN, записывает номер последовательности X и отправляет ответ посредством подтверждения SYN (пакет c битом ACK = X + 1). Узел B указывает свой собственный начальный номер последовательности (SEQ = Y). Если ACK = 20, это значит, что узел принял байты с 0 по 19 и ожидает приема байта 20. Такой подход называется предварительным подтверждением.
3. Узел A подтверждает все отправленные узлом B байты с помощью предварительного подтверждения, что означает ожидание узлом A следующего байта (ACK = Y + 1).
4. Начинается обмен данными.

Методы сканирования UDP менее эффективны, чем сканирование TCP, так как атрибуты и возможности этих протоколов различны. В процессе передачи данных через UDP происходит мало "рукопожатий" и установок соединения, поэтому средство сканирования не сможет собрать достаточно информации. Тем не менее, сканирование UDP используется хакерами; многие программы типа "троянский конь", такие как Back Orifice, используют протокол UDP, так как трафик UDP сложнее обнаружить и остановить. В результате средство сканирования UDP может найти "черный ход" через уже имеющуюся на жертве программу "троянский конь".

Средства сканирования UDP отправляют на порт пустые пакеты, ответ на которые и будет информативным результатом сканирования. Если порт закрыт, операционная система ответит, что данный порт недоступен. Если порт открыт, то пустой пакет, скорее всего, будет проигнорирован. Следовательно, отсутствие ответа обычно означает, что порт открыт, если только порт не защищен устройством периметровой защиты, блокирующим весь трафик UDP.

#### Определение веб-сервера или операционной системы

Перечень уязвимых мест SANS/*FBI* показывает, что большая их часть относится к конкретным платформам. Поэтому следующим шагом хакера будет выяснение операционной системы компьютера-жертвы, производителя и версии приложения или службы. Эти данные помогут ему подобрать нужный эксплоит.

Иногда хакер использует программное обеспечение компьютера для выяснения версии веб-сервера или операционной системы. Простой и часто используемый метод заключается в получении клиента службы Telnet и попытке входа в систему. На веб-сервере это действие осуществляется с помощью подключения через Telnet к порту 80 сервера (порт служб HTTP). На уязвимых системах при попытке входа в систему сервер отображает сообщение с указанием информации о производителе и версии (независимо от того, был ли вход в систему успешным).

Сканирование операционных систем осуществляется с помощью таких программ, как Nmap и Nessus. Они используют утилиту сбора данных, которая выясняет вид стека TCP/IP жертвы посредством отправляемых пакетов-зондов и сопоставляет полученные данные с информацией в базе данных, чтобы выяснить производителя и версию программного обеспечения исследуемого компьютера. Как правило, для успешной идентификации недостаточно одного зонда, однако средства сканирования являются автоматизированными, что обеспечивает высокую скорость и простоту работы.

Для обмана хакеров администраторы часто изменяют сообщения веб-сервера с указанием в нем ложного имени производителя и версии программного обеспечения. Тем не менее, этот способ не очень эффективен против хакеров, имеющих большой опыт работы с Microsoft IIS, так как успешное сканирование операционной системы быстро сужает круг возможных версий IIS.

#### Методы защиты от обнаружения сканирования

Одним из способов защиты от обнаружения сканирования является использование средства FTP *Bounce*, о котором уже шла речь в этой лекции. Данный метод применяет обманную тактику, используя прокси-сервер для маскировки истинного сетевого адреса сканера. Другой метод заключается в уменьшении частоты сканирования. Достаточно низкая частота подачи сканирующих запросов не попадает в диапазон наблюдения за сетевой активностью, определенный в системе обнаружения вторжений (IDS). Систему IDS можно настроить на пониженный уровень частоты. Но, если этот уровень будет слишком низким, IDS сгенерирует большое количество ложных отчетов о сканировании, относящихся на самом деле к совершенно другим процессам, и действия, осуществляемые программой-сканером, просто затеряются в этой массе.

Некоторые методы сканирования позволяют сканирующим пакетам преодолевать периметровую защиту. Сетевые экраны с фильтрацией пакетов иногда не обнаруживают сканирующие пакеты, разбивающие дейтаграммы в заголовках TCP, так как фильтр не распознает полный заголовок TCP, соответствующий правилам фильтрации, и пакет не блокируется.

### Эсплоиты, использующие уязвимые места программного обеспечения

Существует много эксплоитов, наиболее известными из которых являются следующие.

* **Эксплоиты ошибок конфигурации**. Используют недостатки в системе управления привилегиями и/или уязвимые или раскрытые идентификаторы и пароли, позволяющие осуществлять прямую атаку, вход и дальнейшие действия. Если хакер добыл нужную информацию с помощью разведки, он выполнит направленный эксплоит и проникнет в систему.
* **Вредоносные эксплоиты**. Размещают на сервере сторонний код, позволяющий похитить, изменить, осуществить или перехватить управление системой с помощью следующих подходов.
  + **Заражение**. Представляет собой атаку вредоносным кодом в форме вируса или червя, который проникает в сеть системы-жертвы с помощью внутренних пользователей (через исполняемый файл на диске, сообщение электронной почты или браузер) и быстро распространяется на большое число компьютеров произвольным образом.
  + **Дефект (баг)**. Ошибки в программном обеспечении жертвы, демонстрационные сценарии или апплеты CGI/Java, предоставляющие возможность переполнения буфера, с помощью которых выполняется вредоносная программа с необходимыми привилегиями для несанкционированного доступа или повреждения системы.
  + **Троянский конь**. Программа, преодолевающая периметровую систему безопасности и защиту веб-сервера, как правило, через электронную почту (или через обмен файлами между пользователями), создающая "дыру" в безопасности системы для беспрепятственного доступа к системе.
* **Распределенные эксплоиты**. Действуют на одном или нескольких компьютерах, используемых хакерами для осуществления атак на стороннюю цель. Эти эксплоиты сильно затрудняют идентификацию истинного *источника атаки*.

Эксплоиты, использующие ошибки в конфигурации, требуют от хакера много времени и усилий для поиска ошибок и последующего проникновения в систему. Эксплоиты других категорий требуют от него большей сообразительности и находчивости. За исключением случая, когда *хакер* просто повторно использует существующую программу-эксплоит, ему требуется написать некоторый код и разработать схему его доставки. Но, опять-таки, с помощью набора хакерских средств программист довольно просто скомпилирует атакующую программу и выполнит эксплоит. Другое дело – распределенные эскплоиты. Их использование требует тщательнейшего планирования и подготовки, но это компенсируется тем, что таким атакам, как правило, очень трудно противостоять.

#### Эксплоиты, использующие ошибки в конфигурации

В зависимости от полученных данных хакер может иметь достаточно информации для осуществления направленной атаки. Узнав об открытых портах на компьютере-жертве, о приложении или службе, использующей этот порт, выяснив поставщика и номер версии программного обеспечения, хакер может начинать атакующие действия. Имея на руках указанную информацию, взломщик обратится за помощью к своим собственным ресурсам (среди которых непременно будет перечень уязвимых мест SANS/*FBI*) и определит, соответствуют ли полученные данные условиям для атаки.

Иногда хакер находит незащищенный порт при подключении с помощью программы Telnet. В случае неудачи злоумышленник попытается войти в систему под учетной записью пользователя, найти домашний каталог со списком файлов и подкаталогов и т.д. Атака может быть направлена и на использование других служб IIS, таких как FTP и NNTP.

#### Злоумышленный или вредоносный код

Одной из серьезных опасностей, угрожающих сайтам в новом тысячелетии, является угроза новейших злоумышленных и вредоносных атак. Вирусы, черви, "троянские кони", умышленно размещенные синтаксические ошибки на интерактивных веб-страницах и программы, написанные самими хакерами, представляют собой разновидности вредоносного кода. За последние два года в сети интернет появились довольно изощренные атаки, и их становится все больше и больше.

Атаки с помощью вредоносного кода бывают как направленными, так и ненаправленными. Сайт является частью интернета, поэтому даже если у вас нет врагов, и вы не выделяетесь из общей массы пользователей, он является потенциальной целью для таких атак. Атаки с вредоносным кодом выполняют такие действия, как DoS, искажение графической информации, удаление файлов и *перехват данных*. При выполнении атаки DoS одним из способов переполнения буфера является отправка веб-серверу адреса URL или программы, выполнение которой вызовет его сбой. В некоторых случаях такая атака предшествует другой атаке. Например, хакер размещает на веб-сервере программу типа "троянский конь", которая поможет ему проникнуть в систему, являющуюся частью внутренней сети.

Вредоносный код, расположенный на веб-страницах, является сравнительно новой угрозой, о которой будет кратко рассказано в разделе "Сценарии и Java-апплеты" данной лекции.

##### Вирусы, черви и "троянские кони"

Различия между вирусами, червями и "троянскими конями" совсем незначительны. Эти вредоносные программы отличаются друг от друга по своей форме и способу воздействия. Ниже приведены определения этих программ, взятые из глоссария терминов Агентства национальной безопасности США.

* **Вирус** – программа, которая может "заражать" другие программы, зачастую скрываясь в них, посредством внедрения в их тело копии самой себя (возможно, эволюционировавшей). Вирусы часто используются для выполнения различных злоумышленных действий. Для распространения вирусу необходима базовая программа.
* **Червь** – независимая, самодостаточная программа или алгоритм, распространяющая себя среди компьютеров через сетевые соединения, как правило, выполняющая вредоносные действия, которые заключаются в использовании ресурсов компьютеров, засорении сетей и информационных систем вследствие самораспространения программы.
* **"Троянский конь"** – внешне полезная и безобидная программа, содержащая скрытый код, который может осуществлять несанкционированный сбор, использование, фальсифицирование и уничтожение данных. В отличие от вируса и червя "троянский конь" не распространяет сам себя, однако не менее вредоносен.

На практике некоторые атаки с использованием вредоносного кода реализуются посредством комбинации каждого из трех типов в некий гибрид. По этой причине границы между типами вредоносного программного обеспечения размыты, а классические определения – некорректны. Тем не менее, с помощью определений можно составить некоторую классификацию. Большую опасность представляет то, что атаки с использованием вредоносных программ осуществляются множеством различных способов с разнообразными методами маскировки и всевозможными хитроумными уловками. Вы можете не заметить скрытой угрозы, но нужно находиться в постоянной готовности, чтобы противостоять ей.

При атаке с применением вредоносного кода обычно выполняется одно (в худшем случае, все) из перечисленных действий.

* *Выполнение вредоносного кода*. Это код в любой из своих форм представляет исполняемые программные инструкции. Независимо от того, является ли он вирусом, червем или "троянским конем", после загрузки инструкций программы в *память компьютера* может быть перехвачено управление системными ресурсами, вплоть до выполнения любых действий, соответствующих уровню безопасности текущего пользователя в операционной системе. Во вредоносном коде содержатся любые инструкции, начиная от отображения сообщений на экране и заканчивая удалением всех файлов с жесткого диска компьютера.
* *Сокрытие следов действия*. Чтобы не быть обнаруженной, вредоносная программа скрывает следы своего присутствия в системе. Например, вирус, заразивший загрузочный сектор диска, будет загружаться при запуске операционной системы, что поможет ему "затеряться" в перечне запускаемых файлов.
* *Самораспространение*. После загрузки в память вредоносный код использует различные методы заражения или проникновения в другие файлы, программы и компьютеры, с которыми он вступает во взаимодействие. Если вредоносный код изначально доставлен и внедрен через сообщение электронной почты, он попытается использовать электронную почту своего несущего компьютера для распространения на другие компьютеры. Если вредоносным кодом является макровирус, он загружается при открытии несущего документа и заражает другие документы, создаваемые в данном приложении.

**Особенности вирусов**. Определение вируса является слишком упрощенным, но все же полезным. Существует множество вирусов, каждый из которых имеет свои индивидуальные особенности. Вирусы обычно попадают на компьютер при помощи пользователей, после чего распространяются на другие компьютеры, с которыми он взаимодействует. Например, вирусы *загрузочных секторов* заражают файлы на дисках и распространяются при обмене дискетами. *Макровирусы* инфицируют объекты и распространяются через обмен файлами широко используемого приложения. Почтовые вирусы распространяются через сообщения электронной почты, отправляемые почтовым клиентом. Вирусы скрываются в веб-страницах и заражают веб-сайты через браузеры посетивших их пользователей. Существуют также гибриды вирусов.

Постоянно увеличивающаяся сложность и изощренность вирусов представляет собой серьезную проблему для профессионалов в области информационной безопасности. Сетевые экраны не способны серьезно противостоять вирусам, если политика безопасности позволяет обмениваться сообщениями электронной почты, а брандмауэр настроен на беспрепятственное прохождение соответствующих пакетов данных. Программное обеспечение, предназначенное для обнаружения вирусов, приносит ощутимую пользу, но это относится только к известным вирусам, в то время как без помех распространяются новые вирусы, не распознаваемые программой защиты.

Все вирусы похожи друг на друга тем, что они являются программами, содержащими инструкции для выполнения, и написаны на языках программирования, включая C, Java, Visual Basic и языки макросов Microsoft Office. Вирусы вставляют самих себя в несущие файлы, что способствует их скорейшей загрузке в *память компьютера*. Они маскируются в загрузочном секторе диска или имеют вид файла операционной системы, добавляют себя в виде строк кода к телу программы, представляют макрос в файле данных или приложении к сообщению электронной почты, скрываются в сценарии JavaScript, Java-апплете, VB Script или элементе Active X на веб-странице. В любом случае после загрузки в память и выполнения файла с вредоносным кодом загружается и выполняется программа-вирус.

Некоторые вирусы, называемые полиморфными, мутируют и изменяются, преобразуясь в другие разновидности вирусов во избежание обнаружения и пресечения своих действий. Мутирующие вирусы осложняют работу программных систем защиты, осуществляющих обнаружение вирусов, которые работают по принципу нахождения характерных признаков вирусов, являющихся их ключевыми программными инструкциями. После изменения этих признаков в процессе *мутации* вирус нельзя будет обнаружить до тех пор, пока не обновится антивирусное программное обеспечение.

###### ПРОБЛЕМА. Исследование вируса W32.Simile

Данное исследование базируется на данных компании Symantec Corp., (© 2001). Вирус W32.Simile заражал исполняемые фалы в папках на всех жестких дисках компьютера-жертвы и на любых удаленных устройствах, с которыми взаимодействовал этот компьютер. Вирус не представлял серьезной опасности, однако в определенные дни осуществлялось отображение сообщений.

Вирус содержал строку " *Metaphor v1 by The Mental Driller/29A* ". Эта строка отображалась 17 мая, июня, сентября и декабря после запуска инфицированного исполняемого файла. 14 мая на всех компьютерах с поддержкой шрифта *Hebrew* отображалось сообщение " *Free Palestine*!".

Предполагалось, что W32.Simile изначально распространился через электронную почту. После заражения компьютера в интрасети W32.Simile заражал и веб-сервер IIS, если зараженный компьютер содержал диск, связанный с веб-сервером. Вирус W32.Simile мог проникнуть на веб-сервер через интернет, если зараженная программа выполнялась на сервере после ее внедрения через переполнение буфера.

Данный вирус являлся полиморфным и использовал в своих действиях методы скрытия точки входа. Другими словами, вирус записывал себя в случайное место и осуществлял шифрование строки с целью повышения сложности обнаружения. При своем запуске вирус перестраивался для изменения характерных признаков. Происходило как сокращение, так и увеличение объема кода вируса для предотвращения неконтролируемого роста, присущего другим метаморфическим вирусам. Вирус содержал множество тестов для предотвращения заражения файлов, предназначенных для захвата вируса.

После модификации вирус осуществлял поиск файлов .exe в текущей папке, а затем во всех папках на фиксированных и удаленных дисках. Как правило, он заражал исполняемые файлы Win32, имеющиеся в системе и написанные на языке C, и воздействовал, в основном, на файлы операционной системы.

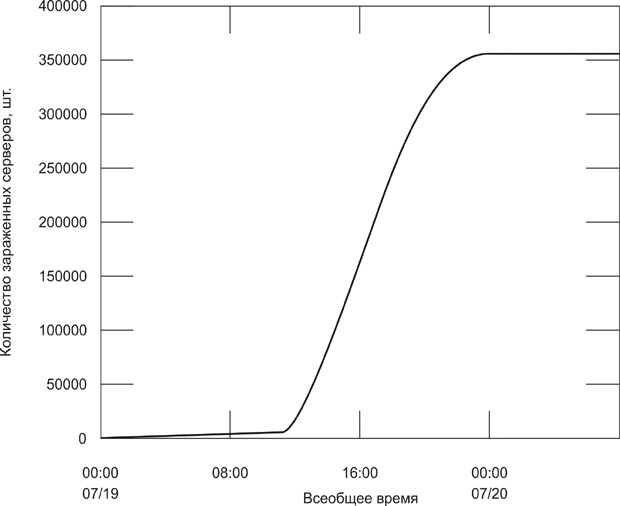
**Особенности червей**. Червь по своей природе аналогичен вирусу. Иногда термин "червь" используется по отношению к вирусу, так как некоторые вирусы используют механизмы распространения, присущие червям. Тем не менее, с технической точки зрения червь отличается от вируса, так как первый представляет собой самодостаточную программу, а не программу, присоединенную к другой программе. Кроме этого, в отличие от вирусов, которые скрываются во вложениях электронной почты и запускаются пользователем при открытии вложения, атака червей представляет собой полностью автоматизированную процедуру. После успешного проникновения в систему они полностью устанавливают и загружают самих себя, выполняя эти действия в рамках атаки, а затем осуществляют автоматическое самораспространение.

Полностью автоматизированные черви наносят огромный ущерб за малый промежуток времени. Червь Code Red, чей механизм распространения осуществлял рассеянное сканирование для определения уязвимых серверов IIS, стал примером того, насколько разрушительны последствия успешных действий червя. Черви распространяются экспоненциально, так как одна жертва распространяет вирус на несколько других, каждая из которых, в свою очередь, выполняет то же самое. На [рисунке 2.5](https://www.intuit.ru/studies/courses/1002/122/lecture/3519?page=8" \l "image.2.5) показан график, предоставленный Ассоциацией анализа данных интернета (CAIDA.org), на котором показано распространение червя Code Red более чем на 350 000 серверов за 13 часов.

Червь является независимой программой, и ему свойственны действия по самосохранению. Эффективным методом самосохранения является скрытие. Разработчики червей пытаются маскировать черви под легальные процессы, выполняющиеся в системе компьютера. По этой причине файлы, добавляемые червями в систему компьютера, имеют обычные, не "режущие" взгляд имена (например, MS Windows содержит огромное количество файлов с расширениями .vxb, .*vxd* и т.д.). В иных случаях файлы червей заменяют собой легальные исполняемые файлы, что вовсе создает впечатление о неизменности набора файлов в системе.

Создатель червя усложняет процесс его устранения требованием более высоких привилегий, нежели уровень пользователя компьютера, на котором червь был установлен.

Как и вирус, червь может мутировать и изменять свой внешний вид, чтобы всегда быть на шаг впереди своих противников. Такие программы, как VBS *Worm* Generator (доступна в интернете), значительно упрощают создание мутированных версий ранее созданных червей. Ущерб, наносимый мутирующим червем, огромен, о чем свидетельствуют факты, приведенные ниже.

[](https://www.intuit.ru/EDI/27_04_15_5/1430086749-14056/tutorial/221/objects/2/files/02_05.gif)

**Рис. 2.5.**Червь Code Red распространился более чем на 350 000 серверов за 13 часов

###### ПРОБЛЕМА. Исследование червя Nimda

Это исследование основано на данных отчета "Nimda *Incident* Report" института SANS (© 2001). В сентябре 2001 г. червь под названием Nimda атаковал веб-сайты Microsoft IIS и вызвал отказ в обслуживании на тысячах сайтов.

Nimda серьезно нарушил целостность и безопасность атакованных систем, что позволило злоумышленникам завладеть полными привилегиями администраторов компьютеров-жертв, получить доступ ко всей файловой системе и внести бесчисленные изменения в системные файлы и настройки реестра.

Червь распространял сам себя на новые компьютеры посредством четырех отдельных механизмов.

* Червь сканировал интернет на предмет веб-серверов и использовал их уязвимые места для получения контроля над жертвой, включая уязвимые места "IIS/Personal Web Services Extended Unicode Directory *Traversal* *Vulnerability*", "IIS/*PWS* *Escaped Character* *Decoding* Command Execution *Vulnerability*" и "черные ходы", оставленных червем Code Red II (см. далее). После получения контроля над жертвой – сервером IIS/*PWS* – червь через протокол FTP передавал свой код с атакующего компьютера на следующую жертву.
* Червь осуществлял сбор адресов электронной почты из адресной книги Windows, папок с входящими и исходящими сообщениями, а также из локальных файлов HTML/HTM и отправлял себя по всем адресам во вложении к сообщению электронной почты в виде файла readme.exe.
* Если червю удавалось заразить веб-сервер, он использовал службу HTTP для распространения себя на клиентов, просматривающих страницы этого веб-сервера. После инфицирования сервера-жертвы червь создавал свою копию в виде файла REAMDE.*EML* и осуществлял прохождение по дереву каталогов в поиске связанных с интернетом файлов с расширением .HTML, .HTM и .ASP. Каждый раз, когда червь находил файл с веб-содержимым, он присоединял к файлу фрагмент кода JavaScript. Код JavaScript вызывал принудительную загрузку файла README.*EML* на компьютер любого пользователя, просматривающего файл через браузер.
* Червь поддерживал работу в сети и распространялся через открытые общие файловые ресурсы. Он копировал себя во все каталоги, включая каталоги в общем сетевом расположении, для которых пользователь имел разрешение на запись. Червь осуществлял поиск исполняемых файлов на устройствах общего доступа и присоединял себя к каждому из них. Зараженным становился всякий узел, с которого осуществлялся доступ к общему сетевому расположению и загружался один из этих файлов.

Механизмы распространения на серверы IIS, описанные выше, требуют сканирования интернета инфицированной системой на предмет поиска уязвимых серверов IIS. Данный червь осуществлял заражение своих соседей по IP-адресу. Он выбирал узлы с идентичной первой или второй частью (имеется в виду адрес с идентичным значением в крайних правых полях, например, xxx.xxx.654.3210). Это действие приводило к массовой сетевой активности на сайтах с несколькими зараженными системами. В частности, можно было наблюдать эффект переполнения *протокола ARP* в зависимости от топологии сети жертвы.

Червь вносил бесчисленное множество изменений в файловую систему жертвы, включая создание большого числа копий самого себя под различными именами. В некоторых случаях создавалось столько копий червя, что оказывалось занятым все пространство жесткого диска. Именами файлов червя являлись Admin.dll, Load.exe, MMC.EXE, readme.exe, Riched20.dll и MEP\*.TMP.EXE.

Действие червя зависело от операционной системы жертвы, от имени файла, под видом которого выполнялся червь, а также от *параметров командной строки*, использованных при выполнении программы. При заражении жертвы червь Nimda открывал общий доступ через сеть к каждому локальному диску. В операционных системах Windows 95/98/Me доступ открывался без всякого пароля. В системах Windows NT/2000 пользователю Guest присваивалось разрешение на доступ ко всем общим ресурсам, причем он добавлялся в группу пользователей Администратор. Червь сводил к нулю всю защиту общих сетевых ресурсов в системах Windows NT/2000.

Для сканирования сети червь запускал до двухсот нитей (подпроцессов). Это действие вызывало значительную загрузку зараженного компьютера и всей сети. ARP, генерируемые сканирующей системой, или запросы DNS, генерируемые системой, отправляющей червь Nimda в сообщениях электронной почты, вызывали ошибки, выглядевшие как явные *DoS-атаки*. Дальнейший анализ с помощью *дизассемблера* показал, что червь Nimda повторно входил в фазу распространения через электронную почту каждые десять дней.

**Особенности "троянских коней"**. "Троянский конь" представляет собой программу, размещаемую на компьютере, который предполагается использовать в дальнейшем. Термин, взятый из греческой мифологии, означает подаренного городу Троя гигантского коня, внутри которого находились греческие солдаты, атаковавшие город. Такая аналогия очень удачна, так как "троянские кони", в основном, используются для осуществления атаки на взлом с целью оставить "открытые двери" в систему, либо устанавливаются после успешного взлома в качестве "черных ходов", которые позволят хакеру вновь проникнуть в систему.

"Троянское кони" могут физически устанавливаться изнутри сети (опять мы имеем дело с внутренней угрозой), через электронную почту или веб-страницу, легкомысленно открытую получателем. Все это похоже на атаки с использованием вирусов, не правда ли? Другими словами, сообщение электронной почты или веб-страница, внедряющая "троянского коня", содержит вирус или червь с определенным типом действия (троянскую программу). Однако вирус наносит ущерб, как правило, сразу после первого запуска программы. Что касается "троянских коней",то они скрытно находятся в системе на протяжении некоторого времени, пока не представится возможность для нанесения удара.

Независимо от способа доставки, если не обнаружить и не удалить "троянского коня", он будет оставаться на компьютере до тех пор, пока не удалит себя сам, или пока хакер не решит, что "троянский конь" ему больше не нужен. Пока "троянский конь" присутствует в системе жертвы, он находится в полном распоряжении хакера.

"Троянские кони" используются самыми различными способами. Некоторые из них являются одноразовыми вредоносными программами, настроенными на удаление в определенное время. Другие выполняют функции, упрощающие выполнение задуманных действий хакера, такие программы называются агентами.

Одним из распространенных типов агентов является *прослушивающая программа* (*sniffer*), отслеживающая сообщения электронной почты, сетевые пакеты, клавиатуры, *память компьютера* и другие компоненты системы. Другой распространенный агент носит название *Remote Access* *Trojan* (*RAT*). *RAT* позволяет хакеру захватить управление компьютером или сервером и удаленно выполнять команды. Многие из программ "троянский конь", известные в хакерских кругах, комбинируют элементы снифферов и возможностей *RAT*.

При осуществлении атаки с помощью "троянского коня" такая комбинация используется довольно часто. Роль сниффера заключается в выявлении идентификаторов и паролей входа в систему. Он записывает нажимаемые пользователями клавиши или пакеты, отправляемые компьютером в сеть. Механизм записи зависит от типа прослушивающей программы. Сниффер может записывать данные в журнал, который потом просматривается хакером. В этот момент используется компонент атаки, реализующий удаленный контроль. *RAT* будет служить "черным ходом" для входа и просмотра файла журнала. Если на компьютере жертвы нет "черного хода", прослушивающая программа отправит журнал по электронной почте. В любом случае, просмотрев журнал, хакер найдет нужную для себя информацию.

При использовании инструментов, принятых в хакерском сообществе, суровой реальностью атаки с помощью "троянского коня" является то, что хакер, ответственный за конкретную атаку, является не единственным человеком, который использует "троянского коня" после его загрузки на компьютер-жертву. Хакерские средства настроены на работу с определенными портами. Из этого следует, что сканер портов может обнаружить уже размещенных в системах удаленно контролируемых "троянских коней". Всякий пользователь, выявивший присутствие "троянского коня", сможет получить доступ ко всем ресурсам, раскрываемым данной программой, даже если она была размещена на компьютере другим хакером.

###### ПРОБЛЕМА. Исследование вируса W32.Simile

Данное исследование базируется на данных компании Symantec Corp., (© 2001). Вирус W32.Simile заражал исполняемые фалы в папках на всех жестких дисках компьютера-жертвы и на любых удаленных устройствах, с которыми взаимодействовал этот компьютер. Вирус не представлял серьезной опасности, однако в определенные дни осуществлялось отображение сообщений.

Вирус содержал строку " *Metaphor v1 by The Mental Driller/29A* ". Эта строка отображалась 17 мая, июня, сентября и декабря после запуска инфицированного исполняемого файла. 14 мая на всех компьютерах с поддержкой шрифта *Hebrew* отображалось сообщение " *Free Palestine*!".

Предполагалось, что W32.Simile изначально распространился через электронную почту. После заражения компьютера в интрасети W32.Simile заражал и веб-сервер IIS, если зараженный компьютер содержал диск, связанный с веб-сервером. Вирус W32.Simile мог проникнуть на веб-сервер через интернет, если зараженная программа выполнялась на сервере после ее внедрения через переполнение буфера.

Данный вирус являлся полиморфным и использовал в своих действиях методы скрытия точки входа. Другими словами, вирус записывал себя в случайное место и осуществлял шифрование строки с целью повышения сложности обнаружения. При своем запуске вирус перестраивался для изменения характерных признаков. Происходило как сокращение, так и увеличение объема кода вируса для предотвращения неконтролируемого роста, присущего другим метаморфическим вирусам. Вирус содержал множество тестов для предотвращения заражения файлов, предназначенных для захвата вируса.

После модификации вирус осуществлял поиск файлов .exe в текущей папке, а затем во всех папках на фиксированных и удаленных дисках. Как правило, он заражал исполняемые файлы Win32, имеющиеся в системе и написанные на языке C, и воздействовал, в основном, на файлы операционной системы.

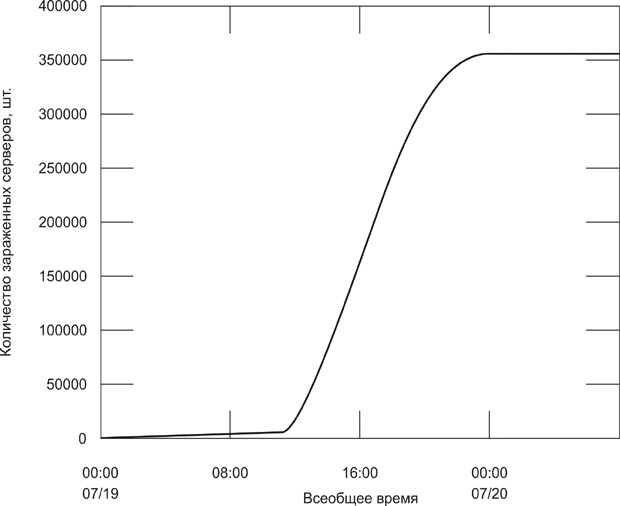
**Особенности червей**. Червь по своей природе аналогичен вирусу. Иногда термин "червь" используется по отношению к вирусу, так как некоторые вирусы используют механизмы распространения, присущие червям. Тем не менее, с технической точки зрения червь отличается от вируса, так как первый представляет собой самодостаточную программу, а не программу, присоединенную к другой программе. Кроме этого, в отличие от вирусов, которые скрываются во вложениях электронной почты и запускаются пользователем при открытии вложения, атака червей представляет собой полностью автоматизированную процедуру. После успешного проникновения в систему они полностью устанавливают и загружают самих себя, выполняя эти действия в рамках атаки, а затем осуществляют автоматическое самораспространение.

Полностью автоматизированные черви наносят огромный ущерб за малый промежуток времени. Червь Code Red, чей механизм распространения осуществлял рассеянное сканирование для определения уязвимых серверов IIS, стал примером того, насколько разрушительны последствия успешных действий червя. Черви распространяются экспоненциально, так как одна жертва распространяет вирус на несколько других, каждая из которых, в свою очередь, выполняет то же самое. На [рисунке 2.5](https://www.intuit.ru/studies/courses/1002/122/lecture/3519?page=8" \l "image.2.5) показан график, предоставленный Ассоциацией анализа данных интернета (CAIDA.org), на котором показано распространение червя Code Red более чем на 350 000 серверов за 13 часов.

Червь является независимой программой, и ему свойственны действия по самосохранению. Эффективным методом самосохранения является скрытие. Разработчики червей пытаются маскировать черви под легальные процессы, выполняющиеся в системе компьютера. По этой причине файлы, добавляемые червями в систему компьютера, имеют обычные, не "режущие" взгляд имена (например, MS Windows содержит огромное количество файлов с расширениями .vxb, .*vxd* и т.д.). В иных случаях файлы червей заменяют собой легальные исполняемые файлы, что вовсе создает впечатление о неизменности набора файлов в системе.

Создатель червя усложняет процесс его устранения требованием более высоких привилегий, нежели уровень пользователя компьютера, на котором червь был установлен.

Как и вирус, червь может мутировать и изменять свой внешний вид, чтобы всегда быть на шаг впереди своих противников. Такие программы, как VBS *Worm* Generator (доступна в интернете), значительно упрощают создание мутированных версий ранее созданных червей. Ущерб, наносимый мутирующим червем, огромен, о чем свидетельствуют факты, приведенные ниже.

[](https://www.intuit.ru/EDI/27_04_15_5/1430086749-14056/tutorial/221/objects/2/files/02_05.gif)

**Рис. 2.5.**Червь Code Red распространился более чем на 350 000 серверов за 13 часов

###### ПРОБЛЕМА. Исследование червя Nimda

Это исследование основано на данных отчета "Nimda *Incident* Report" института SANS (© 2001). В сентябре 2001 г. червь под названием Nimda атаковал веб-сайты Microsoft IIS и вызвал отказ в обслуживании на тысячах сайтов.

Nimda серьезно нарушил целостность и безопасность атакованных систем, что позволило злоумышленникам завладеть полными привилегиями администраторов компьютеров-жертв, получить доступ ко всей файловой системе и внести бесчисленные изменения в системные файлы и настройки реестра.

Червь распространял сам себя на новые компьютеры посредством четырех отдельных механизмов.

* Червь сканировал интернет на предмет веб-серверов и использовал их уязвимые места для получения контроля над жертвой, включая уязвимые места "IIS/Personal Web Services Extended Unicode Directory *Traversal* *Vulnerability*", "IIS/*PWS* *Escaped Character* *Decoding* Command Execution *Vulnerability*" и "черные ходы", оставленных червем Code Red II (см. далее). После получения контроля над жертвой – сервером IIS/*PWS* – червь через протокол FTP передавал свой код с атакующего компьютера на следующую жертву.
* Червь осуществлял сбор адресов электронной почты из адресной книги Windows, папок с входящими и исходящими сообщениями, а также из локальных файлов HTML/HTM и отправлял себя по всем адресам во вложении к сообщению электронной почты в виде файла readme.exe.
* Если червю удавалось заразить веб-сервер, он использовал службу HTTP для распространения себя на клиентов, просматривающих страницы этого веб-сервера. После инфицирования сервера-жертвы червь создавал свою копию в виде файла REAMDE.*EML* и осуществлял прохождение по дереву каталогов в поиске связанных с интернетом файлов с расширением .HTML, .HTM и .ASP. Каждый раз, когда червь находил файл с веб-содержимым, он присоединял к файлу фрагмент кода JavaScript. Код JavaScript вызывал принудительную загрузку файла README.*EML* на компьютер любого пользователя, просматривающего файл через браузер.
* Червь поддерживал работу в сети и распространялся через открытые общие файловые ресурсы. Он копировал себя во все каталоги, включая каталоги в общем сетевом расположении, для которых пользователь имел разрешение на запись. Червь осуществлял поиск исполняемых файлов на устройствах общего доступа и присоединял себя к каждому из них. Зараженным становился всякий узел, с которого осуществлялся доступ к общему сетевому расположению и загружался один из этих файлов.

Механизмы распространения на серверы IIS, описанные выше, требуют сканирования интернета инфицированной системой на предмет поиска уязвимых серверов IIS. Данный червь осуществлял заражение своих соседей по IP-адресу. Он выбирал узлы с идентичной первой или второй частью (имеется в виду адрес с идентичным значением в крайних правых полях, например, xxx.xxx.654.3210). Это действие приводило к массовой сетевой активности на сайтах с несколькими зараженными системами. В частности, можно было наблюдать эффект переполнения *протокола ARP* в зависимости от топологии сети жертвы.

Червь вносил бесчисленное множество изменений в файловую систему жертвы, включая создание большого числа копий самого себя под различными именами. В некоторых случаях создавалось столько копий червя, что оказывалось занятым все пространство жесткого диска. Именами файлов червя являлись Admin.dll, Load.exe, MMC.EXE, readme.exe, Riched20.dll и MEP\*.TMP.EXE.

Действие червя зависело от операционной системы жертвы, от имени файла, под видом которого выполнялся червь, а также от *параметров командной строки*, использованных при выполнении программы. При заражении жертвы червь Nimda открывал общий доступ через сеть к каждому локальному диску. В операционных системах Windows 95/98/Me доступ открывался без всякого пароля. В системах Windows NT/2000 пользователю Guest присваивалось разрешение на доступ ко всем общим ресурсам, причем он добавлялся в группу пользователей Администратор. Червь сводил к нулю всю защиту общих сетевых ресурсов в системах Windows NT/2000.

Для сканирования сети червь запускал до двухсот нитей (подпроцессов). Это действие вызывало значительную загрузку зараженного компьютера и всей сети. ARP, генерируемые сканирующей системой, или запросы DNS, генерируемые системой, отправляющей червь Nimda в сообщениях электронной почты, вызывали ошибки, выглядевшие как явные *DoS-атаки*. Дальнейший анализ с помощью *дизассемблера* показал, что червь Nimda повторно входил в фазу распространения через электронную почту каждые десять дней.

**Особенности "троянских коней"**. "Троянский конь" представляет собой программу, размещаемую на компьютере, который предполагается использовать в дальнейшем. Термин, взятый из греческой мифологии, означает подаренного городу Троя гигантского коня, внутри которого находились греческие солдаты, атаковавшие город. Такая аналогия очень удачна, так как "троянские кони", в основном, используются для осуществления атаки на взлом с целью оставить "открытые двери" в систему, либо устанавливаются после успешного взлома в качестве "черных ходов", которые позволят хакеру вновь проникнуть в систему.

"Троянское кони" могут физически устанавливаться изнутри сети (опять мы имеем дело с внутренней угрозой), через электронную почту или веб-страницу, легкомысленно открытую получателем. Все это похоже на атаки с использованием вирусов, не правда ли? Другими словами, сообщение электронной почты или веб-страница, внедряющая "троянского коня", содержит вирус или червь с определенным типом действия (троянскую программу). Однако вирус наносит ущерб, как правило, сразу после первого запуска программы. Что касается "троянских коней",то они скрытно находятся в системе на протяжении некоторого времени, пока не представится возможность для нанесения удара.

Независимо от способа доставки, если не обнаружить и не удалить "троянского коня", он будет оставаться на компьютере до тех пор, пока не удалит себя сам, или пока хакер не решит, что "троянский конь" ему больше не нужен. Пока "троянский конь" присутствует в системе жертвы, он находится в полном распоряжении хакера.

"Троянские кони" используются самыми различными способами. Некоторые из них являются одноразовыми вредоносными программами, настроенными на удаление в определенное время. Другие выполняют функции, упрощающие выполнение задуманных действий хакера, такие программы называются агентами.

Одним из распространенных типов агентов является *прослушивающая программа* (*sniffer*), отслеживающая сообщения электронной почты, сетевые пакеты, клавиатуры, *память компьютера* и другие компоненты системы. Другой распространенный агент носит название *Remote Access* *Trojan* (*RAT*). *RAT* позволяет хакеру захватить управление компьютером или сервером и удаленно выполнять команды. Многие из программ "троянский конь", известные в хакерских кругах, комбинируют элементы снифферов и возможностей *RAT*.

При осуществлении атаки с помощью "троянского коня" такая комбинация используется довольно часто. Роль сниффера заключается в выявлении идентификаторов и паролей входа в систему. Он записывает нажимаемые пользователями клавиши или пакеты, отправляемые компьютером в сеть. Механизм записи зависит от типа прослушивающей программы. Сниффер может записывать данные в журнал, который потом просматривается хакером. В этот момент используется компонент атаки, реализующий удаленный контроль. *RAT* будет служить "черным ходом" для входа и просмотра файла журнала. Если на компьютере жертвы нет "черного хода", прослушивающая программа отправит журнал по электронной почте. В любом случае, просмотрев журнал, хакер найдет нужную для себя информацию.

При использовании инструментов, принятых в хакерском сообществе, суровой реальностью атаки с помощью "троянского коня" является то, что хакер, ответственный за конкретную атаку, является не единственным человеком, который использует "троянского коня" после его загрузки на компьютер-жертву. Хакерские средства настроены на работу с определенными портами. Из этого следует, что сканер портов может обнаружить уже размещенных в системах удаленно контролируемых "троянских коней". Всякий пользователь, выявивший присутствие "троянского коня", сможет получить доступ ко всем ресурсам, раскрываемым данной программой, даже если она была размещена на компьютере другим хакером.

###### ПРОБЛЕМА. Исследование: новый анализ червя Code Red II

Данное исследование проведено компанией CNET New.com (© 2001). В июле 2001 г. червь под названием Code Red атаковал и заразил тысячи серверов Windows 2000. Несколькими неделями спустя, в августе 2001 г., мутировавшая форма червя под названием Code Red II использовала тот же механизм, что и Code Red, для заражения уязвимых серверов IIS, на которых не была устранена прямая опасность переполнения буфера в idq.dll, или не были удалены отображения сценариев *ISS* *ISAPI*.

**Совет**. О том, как устранить уязвимое место, связанное с возможностью переполнения буфера, вы узнаете в ["Подготовка и укрепление веб-сервера"](https://www.intuit.ru/studies/courses/1002/122/lecture/3521).

За исключением механизма переполнения буфера для выполнения кода червя на уязвимом сервере IIS червь Code Red II коренным образом отличался от исходных вариантов червя Code Red CRv1 и CRv2.

Червь Code Red II имел несколько опасных особенностей. Самой серьезной из них являлось то, что червь создавал "черный ход" посредством размещения "троянского коня" в файле CMD.EXE на зараженном сервере, что делало систему абсолютно открытой для любого хакера.

Червь копировал файл %windir%\CMD.EXE в следующие места:

* c:\inetpub\scripts\root.exe;
* c:\progra~1\common~1\system\MSADC\root.exe;
* d:\inetpub\scripts\root.exe;
* d:\progra~1\common~1\system\MSADC\root.exe.

"Черный ход" позволял хакеру выполнять любые команды на взломанном сервере.

Кроме того, червь размещал на системе-жертве второго "троянского коня", являвшегося модифицированной версией файла explorer.exe (Диспетчер рабочего стола), который давал удаленному хакеру беспрепятственный доступ к корневым каталогам C: и D: после следующего входа пользователя в систему (если в системе не было устранено уязвимое место "Relative Shell Path"); это стало возможным благодаря способу, которым Windows по умолчанию осуществляет поиск исполняемых файлов.

**Совет**. В ["Подготовка и укрепление веб-сервера"](https://www.intuit.ru/studies/courses/1002/122/lecture/3521)будет рассказываться о том, как устранить уязвимое место "Relative Shell Path", связанное с *относительным путем* оболочки.

По прошествии времени распространения инфекции система принудительно перезагружалась. При перезагрузке из памяти удалялся резидентный червь, а "черные ходы" и explorer.exe оставались на месте.

При первом попадании червя на компьютер-жертву и его выполнении осуществлялась проверка заражения данного сервера, в этом случае червь отключал сам себя. После создания "троянских коней" в виде файлов explorer.exe процессы создания файлов приостанавливались. Каждые 10 минут они повторялись с выполнением всех своих процедур, поэтому даже если администратор обнаруживал в реестре параметры, открывающие доступ к C: и D: и удалял их, через несколько минут "троянский конь" прописывал настройки заново.

Червь выбирал жертву, автоматически сканируя системы и определяя, устранены ли на них уязвимые места, связанные с файлом idq.dll и с отображениями сценариев *ISS* *ISAPI*, которые использовались для переполнения буфера. После успешного подключения к цели подпроцесс червя отгружал весь код червя на удаленный компьютер, ожидал подтверждения, после чего продолжал поиск и заражение других узлов.

##### Сценарии и Java-апплеты

Интерактивные веб-страницы, уязвимые для вредоносных сценариев и апплетов, являются довольно новым аспектом веб-безопасности. Это уязвимое место связано с использованием CGI на веб-серверах. Оно также зависит от возможности веб-серверов изменять содержимое сайтов с помощью предоставленного сервером кода (Java или Visual Basic), выполняемого в веб-браузере. В данную категорию уязвимых мест попадают сценарии JavaScript и VBscript, имеющиеся на HTML-страницах веб-сервера.

Вредоносный код, поступающий с веб-серверов, более разрушителен, чем вирусы и черви, особенно код Java, так как на клиентской части он практически не зависит от платформы и влияет на компьютеры Windows, UNIX, Mac, Palm или на любой другой с Java-совместимым браузером. Для выполнения апплетов или сценариев JavaScript в браузере достаточно лишь перейти на несущую страницу апплета.

**Межсайтовые сценарии (CSS)**. CSS относится к интерактивным веб-страницам. Вредоносный код проникает на интерактивную веб-страницу с помощью метода, вызывающего выполнение кода браузером, считывающим данную страницу. В результате вредоносная программа выполняет любое действие, допускаемое привилегиями текущего пользователя, включая считывание информации, удаление файлов, отправку сообщений электронной почты и т.д.

Ниже приведено исследование, являющееся примером реальной атаки CSS, которая в данном случае не была вредоносной, однако вполне могла бы таковой стать.

**ПРОБЛЕМА. Исследование: эксплоит Zkey**

Данный материал заимствован из статьи SANS автора David Rothermel. В августе 2000 г. хакер использовал уязвимость портала хранилища данных .com под названием Zkey посредством выполнения вредоносного кода JavaScript с целью перехвата имен и паролей пользователей электронной почты Zkey. Данный эксплоит является примером атаки CSS.

Атака инициировалась посредством внедрения вредоносного кода JavaScript в сообщение электронной почты и отправки этого сообщения с учетной записи Zkey, использованной хакером, на другую учетную запись пользователя Zkey. Этот код вынуждал пользователей осуществлять вход в систему, посредством чего осуществлялось раскрытие имен пользователей и их паролей.

Сообщение электронной почты имело формат HTML, и код располагался в теге <textarea>. Поверх сообщения размещался прозрачный рисунок GIF, использовавший команду onMouseOver для запуска вложенного кода.

Данный код обманным способом перехватывал имена пользователей и пароли, отображая окна входа с сообщением "You timed out of your session, please log-in" ("Период ожидания вашей сессии истек, пожалуйста, войдите заново"). Когда пользователь вводил свои имя и пароль, они передавались в базу данных на сервер хакера, а не на сервер Zkey. Чтобы пользователь ничего не заподозрил, код направлял пользователя обратно на сервер Zkey и осуществлял его вход в систему для продолжения сеанса.

Потенциальный *размер ущерба* от этой атаки был огромен. Получив полный контроль над учетной записью, хакер имел возможность выполнять следующие действия.

* Загружать файлы с диска Z жертвы.
* Удалять/заменять файлы на диске Z.
* Осуществлять доступ и изменять контактную информацию жертвы.
* Осуществлять доступ и изменять информацию, хранимую в календаре или расписании жертвы.
* Изменять имя и пароль пользователя для запрета доступа пользователя к своей учетной записи.
* Осуществлять доступ к любым общим устройствам диска Z из вторичных учетных записей.
* Читать и удалять сообщения электронной почты Zkey или отправлять сообщения почты Zkey от имени жертвы.
* Осуществлять доступ к электронной почте с любых вторичных учетных записей электронной почты, настроенных на проверку почты.

После демонстрации своих возможностей хакер поставил руководство портала Zkey в неудобное положение, придав огласке наличие уязвимого места. В качестве рекламы самого себя он заявил, что, несмотря на использование сайтом протокола защищенных сокетов SSL и наличие фильтров у службы электронной почты, он смог преодолеть оба барьера защиты.

#### Распределенный отказ в обслуживании

Уязвимые места DoS существовали на компьютерах с самого начала появления ЭВМ. Хакеры вызывали отказ в обслуживании систем посредством физической атаки (отключая электроэнергию) или посредством других атак с вторжением. Злоумышленники освоили атаки как с внутренней, так и с наружной стороны системы.

Внешние атаки на отказ в обслуживании осуществляются через сеть интернет. Например, при использовании одного известного эксплоита, называемого "атакой Smurf" (Smurf – автоматизированное средство для выполнения этой атаки) хакер отправляет большой ping-трафика ICMP по широкому рассеянному диапазону IP-адресов. Эта атака срабатывает, потому что большая часть подключенных к интернету маршрутизаторов и узлов отвечают на ping-запросы рассеянными ECHO-пакетами. Таким образом, в сети интернет с экспоненциальной скоростью растет количество пакетов, что и вызывает отказ в обслуживании.

В течение последних двух лет выявлена новая DoS-уязвимость. С помощью метода, называемого распределенным отказом в обслуживании (*DDoS*), можно задействовать большое число компьютеров сети для перегрузки и выхода из строя одного веб-сайта. Атака *DDoS* в течение долгого времени обосновывалась лишь теоретически. Однако 7-го и 8-го февраля 2000 г. она была успешно реализована на сайтах Yahoo!, Amazon, eBay, Buy.com, *CNN*, E\**TRADE* и ZDNet.

Как и атаки с применением вредоносного кода, распределенные атаки *DDoS* попадают в категорию эксплоитов, которым трудно противостоять, так как злоумышленникам не требуется проникновение в систему для успешной реализации атаки. Скорее всего, такие атаки будут еще долгое время будоражить сознание специалистов в области информационной безопасности.

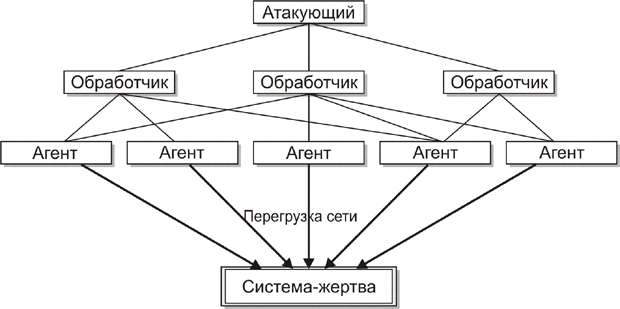
##### Стратегия атак DDoS

Известные на сегодняшний день атаки *DDoS* попадают либо в категорию атак типа "*flood*" (перегрузка сети огромным числом пакетов), либо в категорию атак, использующих предумышленно некорректные пакеты. Атаки на перегрузку выводят сайты из строя посредством использования протоколов TCP для перегрузки маршрутизаторов и серверов, обеспечивающих работу сайтов. Принцип этих атак заключается в том, что после их осуществления пропускная способность каналов связи сводится к нулю, и становится невозможным обслуживание легальных пользователей, пытающихся открыть сайт в своих браузерах.

Атаки с применением некорректных пакетов осуществляются через выявление уязвимых серверов, не поддерживающих ошибки. Отправка пакетов, размер которых превышает допустимую норму для IP-пакетов, или пакетов с одним и тем же IP-адресом источника и пункта назначения, может вывести из строя такие системы.

Для успешного осуществления атака *DDoS* должна производиться с набора систем, совместные действия которых помогут вывести из строя компьютер-жертву. Иногда атака планируется группой хакеров. Каждая система в такой атаке выполняет свою собственную роль. В модели атаки, приведенной на [рис. 2.6](https://www.intuit.ru/studies/courses/1002/122/lecture/3519?page=10" \l "image.2.6), используются системы, выполняющих два различных набора функций и совместно атакующие систему-цель.

С главной атакующей системы осуществляется управление вспомогательными системами, каждая из которых, в свою очередь, управляет группой агентов, генерирующих сетевой трафик. Если хакеру необходимо привлечь произвольные системы в качестве вспомогательных узлов, он с помощью различных методов сканирования захватит управление необходимыми системами, после чего установит на них средства, необходимые для реализации атаки.

[](https://www.intuit.ru/EDI/27_04_15_5/1430086749-14056/tutorial/221/objects/2/files/02_06.gif)

**Рис. 2.6.**Сокрушение системы-жертвы с помощью атаки DDoS

##### Защита от атак DDoS

Защита от атак *DDoS* требует не меньших усилий, чем сама организация данной атаки. В сетях организаций, подключенных к интернету, должны использоваться сетевые экраны, настроенные на обнаружение массового потока ICMP-пакетов и на соответствующие ответные меры. Поставщики услуг интернета должны принимать все меры для защиты от возможных атак данного рода.

Начинающим пользователям следует удостовериться в отсутствии на их компьютерах средств для осуществления атак *DdoS*. Установите надстройки Microsoft Service Pack и "заплатки", устраняющие уязвимые места, перечисленные в перечне SANS/*FBI* Top 20. Это исключит возможность использования хакером системы в качестве принудительного агента атаки *DDoS*.

### Перечень известных уязвимых мест

* Конфигурация операционных систем и приложений по умолчанию.
* Учетные записи с уязвимыми или несуществующими паролями.
* Большое число открытых портов.
* Отсутствие фильтрации адресов.
* Неправильное ведение журнала или его отсутствие.
* Программные "дыры" в CGI.
* Злоумышленный или вредоносный код.
* Проход по папкам веб-сервера.
* Переполнение буфера.
* Доступ к общим ресурсам SMB посредством соединения с NetBIOS с помощью NULL-сеанса.
* Microsoft Remote Networking (Удаленная работа с сетью) и Data Services (Службы данных).

### Вопросы для закрепления

1. В чём заключается основная цель подготовки и укрепления веб-сервера?
2. Какие риски связаны с использованием учетных записей с паролями по умолчанию?
3. Что такое "грубая сила" (brute force) и как она используется для взлома паролей?
4. Почему пароли, состоящие только из букв алфавита, считаются слабыми?
5. Что представляет собой атака со словарём паролей?
6. Какие меры следует предпринять для защиты учетных записей на веб-сервере?
7. Почему важно удалять или переименовывать встроенные и неиспользуемые учетные записи?
8. Какова роль длины и сложности пароля в обеспечении безопасности?
9. Что такое принцип наименьших привилегий и как он применяется к учетным записям?
10. Как администратор может отслеживать и предотвращать несанкционированный доступ к веб-серверу?
11. В чём разница между паролем по умолчанию и слабым паролем?
12. Как часто рекомендуется менять пароли и почему?
13. Что такое двухфакторная аутентификация, и почему её следует использовать?
14. Какие действия входят в процесс "усиления" веб-сервера?
15. Почему обновление и установка патчей также являются частью укрепления сервера?