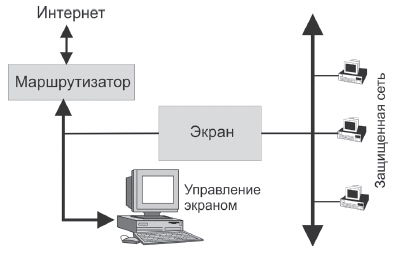
Учитывая важность проблемы защиты, разработана специальная система *firewall* (см. также <http://book.itep.ru/6/fwal_63.htm>). Первые *Firewall* появились в конце 80-х годов, в 1991 году *фирма* *DEC* предложила устройство **SEAL** (*Secure* *External* *Access Link*), устройства же современного типа появились в 1993 году ( **TIS** – Trusted *Information* *System*). Система *firewall* в некоторых случаях может заменять *маршрутизатор* или внешний *порт* сети (*gateway*). Защищенная часть сети размещается за ним. Пакеты, адресованные *Firewall*, обрабатываются локально, а не просто переадресуются. Пакеты же, которые адресованы объектам, расположенным за *Firewall*, не доставляются. *По* этой причине *хакер* вынужден иметь дело с системой защиты *Firewall*. Схема взаимодействия *Firewall* с локальной сетью и внешним *Интернет* показана на [Рис 13.1](https://intuit.ru/studies/courses/1124/201/lecture/5223?page=1#image.13.1).

Такая схема проще и надежнее, так как нужно заботиться о защите одной машины, а не многих. Экран, *маршрутизатор* и ЭВМ управления экраном объединены небольшой, незащищенной локальной сетью. Основные *операции* *по* защите осуществляются здесь на IP-уровне. Эту схему можно реализовать и на одной ЭВМ, снабженной двумя интерфейсами, через один *интерфейс* осуществляется *связь* с *Интернет*, а через второй — с защищенной сетью. Такая ЭВМ совмещает функции маршрутизаторашлюза, экрана и управления экраном. Возможна реализация *Firewall*, показанная на [Рис 13.2](https://intuit.ru/studies/courses/1124/201/lecture/5223?page=1#image.13.2). Здесь *функция* экрана выполняется маршрутизатором, но и прокси может выполнять некоторые защитные функции. Возможен вариант прокси-сервера и с одним сетевым интерфейсом.



**Рис. 13.1.**Схема Firewall

В этой *схеме доступ* из *Интернет* возможен только к прокси-серверу, ЭВМ из защищенной сети могут получить *доступ* к *Интернет* тоже только через *прокси-сервер*. Ни один пакет посланный из защищенной ЭВМ, не может попасть в *Интернет* и, аналогично, ни один пакет из *Интернет* не может попасть непосредственно защищенной ЭВМ. Возможны и другие более изощренные схемы, например, со вторым "внутренним" *Firewall* для защиты от внутренних угроз.



**Рис. 13.2.**Схема Firewall, где функцию экрана выполняет маршрутизатор

Вне зависимости от того, насколько надежен ваш сетевой экран, не следует снижать требования к безопасному конфигурированию рабочих станций и серверов.

Следует также помнить, что сетевой экран не способен защитить от *вирусов*, *сетевых червей*, троянских коней, **атак из локальной сети** и различных мошеннических трюков ("социальная инженерия"). Но самой серьезной угрозой являются "новые", не известные доселе атаки.

Существует несколько разновидностей *Firewall*.

**Фильтрующие Firewall**

Фильтрующая разновидность сетевых экранов производит отбор пакетов по содержимому заголовков (адреса и номера портов). Но такие экраны не могут различить команду get от put, так как для этого надо просматривать поле данных. Функции таких сетевых экранов могут быть реализованы практически любым маршрутизатором.

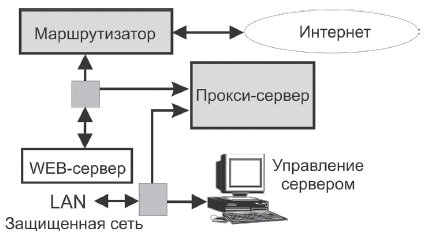
**Прокси Firewall**

Сетевые экраны на основе прокси-серверов способны анализировать не только заголовки, но и пересылаемые данные. Такие серверы исключают прямое соединение клиента и удаленного сервера. От имени клиента запрос посылает сетевой экран. Для большой сети целесообразно иметь отдельные машины для прокси-серверов, обеспечивающих разные виды услуг (WWW, FTP и пр.), см. [рис. 13.3](https://intuit.ru/studies/courses/1124/201/lecture/5223?page=1#image.13.3) и [рис. 13.3а](https://intuit.ru/studies/courses/1124/201/lecture/5223?page=1#image.13.3%D0%B0).

*Разные серверы для разных услуг, помимо безопасности, пропорционально увеличивают быстродействие системы в целом*.

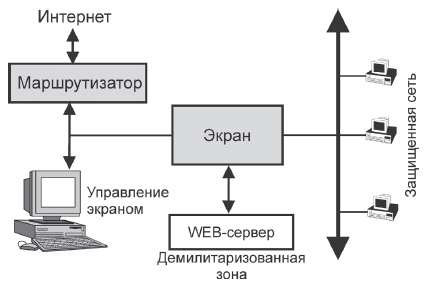
На рисунках показаны два (но не единственных) варианта защиты сети с помощью сетевого экрана, построенного на основе прокси-сервера и имеющего "демилитаризованную" зону. В такой зоне обычно размещаются серверы, доступные из Интернет непосредственно (вне защищаемого контура). Такие серверы бывают нужны для рекламирования изделий фирмы, обслуживания клиентов или для демонстрации достижений учреждения или научного центра. Такие схемы сохраняют доступность вашего WEB-сервера со стороны поисковых систем, что позволит узнать о вашей компании или учреждении большему количеству людей. Вариант [рис. 13.3а](https://intuit.ru/studies/courses/1124/201/lecture/5223?page=1#image.13.3%D0%B0) несколько более гибкий, но требует трех сетевых интерфейсов для прокси-сервера. Существуют аппаратные решения, совмещающие многие из описанных здесь возможностей.

*Следует помнить, что подключение через модем должно производиться через специальный защищенный сервер с поддержкой протоколов типа Radius. Наилучшее для него место размещения – демилитаризованная зона, ведь безопасность сети определяется ее самым уязвимым элементом*.



**Рис. 13.3.**Размещение WEB-сервера в демилитаризованной зоне

При выборе политики и правил отбора пакетов важно ответить на следующие вопросы:



**Рис. 13.3а.**Вариант построения демилитаризованной зоны с помощью сетевого экрана с 3-мя интерфейсами

* Какие сетевые услуги вы желаете реализовать и для каких направлений (изнутри и снаружи)?
* Следует ли ограничивать "внутренних" клиентов в части подключения к серверам в Интернет?
* Имеются ли узлы в Интернет, для которых вы бы хотели создать особые условия доступа (например, филиалы вашего учреждения)?

Обычно сетевым экраном реализуется одна из двух политик безопасности:

* разрешено все, что не запрещено правилами;
* запрещено все, что не разрешено правилами.

На первый взгляд, может показаться, что эти две политики неотличимы, но это совсем не так. В первом варианте при появлении нового приложения или протокола придется вводить новые ограничительные правила. Второй вариант политики консервативнее, и здесь менять правила приходится реже.

Прокси-сервер обычно привязан к конкретному типу приложений и способен разрешать или запрещать выполнение определенных операций. Рассмотрим, как внутренним клиентом через такого рода сервер может быть загружена определенная WEB-страница с сервера в Интернет.

Прямое соединение внутреннего клиента с внешним сервером через сетевой экран невозможно. Прокси-сервер посылает HTTP-запрос внешнему серверу от своего имени с IP-адреса своего внешнего интерфейса, подключенного к Интернет. Понятно, что и отклик от внешне го WEB-сервера придет через указанный интерфейс в прокси-сервер. После получения запрошенной WEB-страницы прокси-сервер производит определенные администратором проверки, после чего открывает сессию передачи полученных данных через свой интерфейс LAN клиенту-заказчику. На фазе проверок возможна сортировка данных и блокировка доступа клиента к определенному типу данных, например, порнографического вида.

Вообще фильтрация данных стала в последнее время широко обсуждаемой проблемой. Отбор может производиться по URL, хотя этот способ селекции мало эффективен, во-первых, потому что URL часто меняются, во-вторых, из-за того, что клиент может всегда заменить URL на IP-адрес. Надежда на то, что в ограничительный список можно включить и IP-адреса, напрасна, так как некоторые программы могут воспринимать IP-адрес в десятичном (а не в десятично-точечном) представлении. Существуют и более изощренные фильтры, где содержимое анализируется по ключевым словам или по характеру вложений (Java или ActiveX).

Выбирая правила фильтрации, нужно сначала сформулировать общую стратегию:

* Какие сервисы в сети нужно обеспечить и в каком направлении (изнутри вовне или извне внутрь).
* Какие ограничения на выход в Интернет для внутренних машин вы готовы установить.
* Имеются ли внешние машины, для которых вы готовы предоставить какие-то формы доступа в вашу сеть.

В разных системах межсетевых экранов правила фильтрации формулируются различным образом, но, как правило, требуются следующие данные:

В каком направлении и через какой интерфейс передаются пакеты.

* IP-адреса отправителя и получателя.
* Допустимые опции IP
* Используемые протоколы высокого уровня (UDP, TCP, ICMP)
* Допустимые типы ICMP-сообщений
* Номера портов отправителя и получателя (для протоколов (UDP и TCP)

***NCSA*** (National *Computer Security* Association) выработала рекомендации по правилам фильтрации для сетевых экранов.

1. Порт RPC (Remote Procedure Call) 111 может использоваться для получения доступа к системной информации (операции чтения-записи).
2. Порты 512, 513 и 514 (*rlogin*, Rsh и rexec) при неправильном конфигурировании могут привести к несанкционированному доступу к учетным записям.
3. Через порты 6000 и 2000 (XWindow и Open Windows) можно получить информацию, отображенную в окне, включая все нажатия клавиш.
4. Порт 69 (FTP), используемый для загрузки бездисковых терминалов, рабочих станций и маршрутизаторов, при неправильном конфигурировании может быть использован для чтения файлов в системе.

Некоторые протоколы желательно отфильтровывать из-за их потенциальной опасности. Работа с ними допускается лишь на специально выделенных машинах.

1. Порт 23 (Telnet), не должен использоваться вообще или использоваться для исследовательских целей на одной ЭВМ.
2. Порты 20 и 21 (FTP) лучше закрыть везде, но, в крайнем случае, их можно открыть при необходимости на одном FTP-сервере.
3. Порт 25 (SMTP) обычно разрешается только на центральном почтовом сервере.
4. Порт 53 (DNS) открывается только для серверов имен (первичного и вторичного).
5. Порт 520 (RIP) может быть использован для *перенаправления потока* данных. Если можно обойтись без протокола маршрутизации RIP, это следует сделать.
6. Порты 70 (*Gopher*) и 80 (WWW) должны быть открыты только для шлюзов соответствующих приложений.
7. Порт 119 (NNTP – служба новостей) должен использоваться только сервером новостей.
8. Порт 79 (Finger) желательно закрыть, так как через него может быть получена полезная для хакера персональная информация.
9. Порт 69 (TFTP) безоговорочно должен быть закрыт для любых внешних пользователей. Открытие для внутренних пользователей должно осуществляться в случае крайней необходимости для выбранных IP-адресов.
10. 10. Порт 540 (*UUCP*) лучше заблокировать из-за его уязвимости (сама услуга устарела и ее безопасность не совершенствуется).

В принципе, прокси-серверы могут работать в обоих направлениях, т.е. разрешать внешним клиентам доступ к внутренним серверам. Обеспечивая *сокрытие данных* о клиентах и структуре внутренней сети, прокси-серверы становятся критической точкой системы в целом. Взлом или выход из строя такого устройства может парализовать работу большого числа людей.

Существуют так называемые прозрачные прокси-серверы. Такие серверы создают больший комфорт клиентам, ведь не нужно авторизоваться сначала в прокси или как-то адаптировать свое программное обеспечение. Для клиента создается впечатление, что он работает непосредственно с внешним сетевым объектом, он может даже не знать о существовании прокси. Запросы внутреннего клиента к внешнему серверу перехватываются прокси-сервером и после анализа блокируются или передаются во внешнюю сеть. Пользователь авторизуется на внешнем сервере, а не в прокси. Но у прокси остается возможность блокировки некоторых операций, например, GET или PUT. Прокси может вести журнал операций или предоставлять различные права разным клиентам. Следует заметить, что прозрачный сервер не скрывает IP-адреса внутренних клиентов. Если прокси организует подключение внешнего клиента к внутреннему серверу, для внутреннего сервера клиентом является прокси, а не внешний объект. Но внешний объект знает IP-адрес внутреннего сервера. Ситуацию можно несколько улучшить, если поместить все внутренние серверы в демилитаризованную зону, — тогда хакер не получит никаких данных об остальных объектах внутренней сети. Внутренние серверы в этом случае должны быть дополнительно защищены, например, локальными программами типа *Firewall*.

В случае классического прокси клиенту нужно знать имя прокси и ЭВМ, с которой он хочет взаимодействовать. Имя прокси клиент должен преобразовать в его IP-адрес, послав запрос в DNS, а имя ЭВМ — передать прокси-серверу в качестве параметра запроса. DNS в этом случае сообщает внутренние IP-адреса только внутренним сетевым объектам. Для работы с внешними IP-адресами служит другой DNS-сервер. В случае же прозрачного прокси схема взаимодействия с DNS идентична той, которая существует при отсутствии сетевого экрана.

Для прокси-серверов достаточно типично использование трансляции сетевых адресов NAT (Network Address Translation). Такая схема, среди прочего, позволяет обойтись меньшим числом реальных IP-адресов. При реализации запросов во внешнюю сеть программа NAT подставляет в поле адрес отправителя IP сервера NAT. При получении отклика программа NAT заносит в поле адреса получателя адрес клиента источника запроса.

Существуют сетевые услуги, которые следует блокировать сетевым экраном.

* **NFS (Network File System)**. В случае разрешения доступа к этой услуге через сетевой экран, на удаленной машине можно будет смонтировать файловую систему вашей сети и делать с ней все что угодно…
* **NIS** (Network Information System – сетевая информационная система). Эта система позволяет хакерам узнать нужные им имена узлов и пользователей в вашей сети.
* **X-windows**. По уязвимости эта услуга сравнима с Telnet, допускает удаленный запуск процессов.

Не следует оставлять без внимания безобидный на первый взгляд протокол ICMP, так как он позволяет получить много разнообразной и полезной для хакера информации. По этой причине целесообразно блокировать прохождение ICMP-пакетов следующих типов:

* входящие *echo request* и исходящие *echo replay*. Это сохранит возможность для внутренних клиентов тестировать доступность узлов Интернет, но не даст зондировать ваши внутренние сетевые объекты;
* входящие сообщения *redirect*. Такие сообщения позволяют модифицировать таблицу маршрутизации;
* входящие сообщения *service unavailable* и исходящие *destination unreachable*. Это блокирует хакеру зондирование вашей сети. Если хакер может узнать, доступны ли нужные ему узлы или службы, это существенно упростит его задачу.

Если вы поставили и сконфигурировали *Firewall*, не следует расслабляться. Во-первых, ваш сетевой экран защитит вашу сеть не от всех потенциальных угроз (например, *вирусы*, *сетевые черви*, троянские кони, доставляемые по почте, не в его сфере возможностей). Во-вторых, если за экраном не вы один, то вас могут обслужить по части сетевых проблем ваши соседи по локальной сети. По этой причине нужно позаботиться об уязвимости ваших внутренних серверов и рабочих станций.

Расстаньтесь с услугами *rlogin*, *rcp*, rexec, telnet (замените на ssh), так как с ними работать удобно не только вам, но и хакерам. По возможности замените услуги FTP на *SFTP* или *scp*. Загляните в конфигурационный файл /etc/inetd.conf, а также в /etc/rc.\* и удалите все ненужные и потенциально опасные услуги и приложения, например, finger. Не оставляйте без внимания и такие файлы, как /etc/networks, /etc/protocols, /etc/services, /etc/hosts.

*Если сетевой экран работает на ЭВМ с ОС Windows, там не должно быть файловой системы FAT, так как она не обеспечивает защиты*.

В последнее время появился новый вид услуг – **MFWS** (Managed *Firewall* Service). Компании, предоставляющие такую услугу, берутся за настройку и управление сетевыми экранами клиента. Сами сетевые экраны могут располагаться у клиента или сервис-провайдера. (О настройке межсетевых экранов смотри <http://lib.ru/SECURITY/firewallbyhand.txt>.)

Недостатки системы *Firewall* происходят от ее преимуществ, — осложняя доступ извне, система делает трудным и доступ наружу. По этой причине система *Firewall* должна выполнять функции DNS (сервера имен) для внешнего мира, не выдавая никакой информации об именах или адресах внутренних объектов, и функции почтового сервера, поддерживая систему псевдонимов для своих клиентов. Псевдонимы не раскрываются при посылке почтовых сообщений во внешний мир. Служба FTP в системе может и отсутствовать, но если она есть, доступ возможен только в сервер *Firewall* и из него. Внутренние ЭВМ не могут установить прямую FTP-связь ни с какой ЭВМ из внешнего мира. Процедуры telnet и *rlogin* возможны только путем входа в сервер *Firewall*. Ни одна из ЭВМ в защищенной сети не может быть обнаружена с помощью PING (ICMP) извне. И даже внутри сети будут возможны только определенные виды трафика между строго определенными машинами. Понятно, что в целях безопасности защищенная сеть не может иметь выходов во внешний мир помимо системы *экран*, в том числе и через модемы. Экран конфигурируется так, чтобы маршрут по умолчанию указывал на защищенную сеть. Экран не принимает и не обрабатывает пакеты внутренних протоколов маршрутизации (например, RIP). ЭВМ из защищенной сети может адресоваться к экрану, но при попытке направить пакет с адресом из внешней сети будет выдан сигнал ошибки, так как маршрут по умолчанию указывает назад в защищенную сеть. Для пользователей защищенной сети создаются специальные входы для FTP/*scp*, telnet/ssh и других услуг. При этом не вводится каких-либо ограничений по транспортировке файлов в защищенную сеть и блокируется передача любых файлов из этой сети, даже в случае, когда инициатором FTP-сессии является клиент защищенной сети. Единственные протоколы, которым всегда позволен доступ к ЭВМ *Firewall*, — это SMTP (электронная почта) и NNTP (служба новостей). Внешние клиенты Интернет не могут получить доступа ни к одной из защищенных ЭВМ ни через один из протоколов. Если нужно обеспечить доступ внешним пользователям к каким-то данным или услугам, для этого можно использовать сервер, подключенный к незащищенной части сети (или воспользоваться услугами ЭВМ управления экраном, что нежелательно, так как снижает безопасность). ЭВМ управления экраном может быть сконфигурирована так, чтобы не воспринимать внешние (приходящие не из защищенной сети) запросы типа FTP/*scp*, telnet/ssh и пр., это дополнительно повысит безопасность. Стандартная система защиты здесь часто дополняется программой *wrapper*. Немалую пользу может оказать и хорошая система регистрации всех сетевых запросов. Системы *Firewall* часто используются и в корпоративных сетях, где отдельные части сети удалены друг от друга. В этом случае в качестве дополнительной меры безопасности применяется шифрование пакетов. Система *Firewall* требует специального программного обеспечения. Следует иметь в виду, что сложная и дорогостоящая система *Firewall* не защитит от "внутренних" злоумышленников. Нужно тщательно продумать систему защиты модемных каналов (сама система *Firewall* на них не распространяется, так как это не внешняя часть сети, а просто *удаленный терминал*). Хороший результат можно получить, совместно обрабатывая журнальные файлы *IDS* и *Firewall*.

Если требуется дополнительная степень защиты, при авторизации пользователей в защищенной части сети могут использоваться аппаратные средства идентификации, а также шифрование имен и паролей.

В последнее время появилось большое число аппаратных решений для межсетевых экранов. Это, прежде всего, CISCO PIX *Firewall*. Но крайне интересное предложение поступило от компании 3СОМ (см. <http://www.3com.com/products/>), где *Firewall* встроен в сетевой интерфейс и снабжен программным обеспечением, позволяющим мониторировать состояние группы таких интерфейсов.

При выборе той или иной системы *Firewall* следует учитывать ряд обстоятельств.

1. **Операционная система**. Существуют версии *Firewall*, работающие с UNIX и Windows NT. Некоторые производители модифицируют ОС с целью усиления безопасности. Выбирать следует ту ОС, которую вы знаете лучше.
2. **Рабочие протоколы**. Все *Firewall* могут работать с FTP (порт 21), e-mail (порт 25), HTTP (порт 80), NNTP (порт 119), Telnet/ssh (порт 23/22), *Gopher* (порт 70), SSL (порт 443) и некоторыми другими известными протоколами. Как правило, они не поддерживают SNMP.
3. **Типы фильтров**. Сетевые фильтры, работающие на прикладном уровне прокси-сервера, предоставляют администратору сети возможность контролировать информационные потоки, проходящие через *Firewall*, но они обладают не слишком высоким быстродействием. Аппаратные решения могут пропускать большие потоки, но они менее гибки. Существует также "схемный" уровень прокси, который рассматривает сетевые пакеты как черные ящики и определяет, пропускать их или нет. Отбор при этом осуществляется по адресам отправителя, получателя, номерам портов, типам интерфейсов и некоторым полям заголовка пакета.
4. **Система регистрации операций**. Практически все системы *Firewall* имеют встроенную систему регистрации всех операций. Но здесь бывает важно также наличие средств для обработки файлов с такого рода записями.
5. **Администрирование**. Некоторые системы *Firewall* снабжены графическими интерфейсами пользователя. Другие используют текстовые конфигурационные файлы. Большинство из них допускают удаленное управление.
6. **Простота**. Хорошая система *Firewall* должна быть простой. Прокси-сервер (экран) должен иметь понятную структуру и удобную систему проверки. Желательно иметь тексты программ этой части, так как это прибавит ей доверия.
7. **Туннелирование**. Некоторые системы *Firewall* позволяют организовывать туннели через Интернет для связи с удаленными филиалами фирмы или организации (системы Интранет). Естественно, что информация по этим туннелям передается в зашифрованном виде.

К средствам мониторинга сетевых атак относятся такие программные продукты, как SNORT ( *IDS* ); для предотвращения атак используются различные системы типа *Firewall*.

Интересные возможности предоставляет программный пакет **TCP Wrapper** — его применяет демон tcpd, запускаемый вместо сетевых служб, которые указаны в файле inetd.conf. TCP *wrapper* позволяет разрешить доступ только с определенных узлов, находящихся в вашей сети. Кроме того, эта программа регистрирует запросы к сервисам и имена (адреса) узлов, откуда они поступили. Это ее свойство может быть крайне полезной при анализе, который нужно проводить при подозрении вторжения.

Хорошего результата можно достичь, грамотно конфигурируя программное обеспечение ЭВМ и контролируя качество паролей. Пример фрагмента журнального файла ZoneAlarm (разновидность *Firewall* ) представлен ниже:

FWIN,2005/08/19,14:25:04 +4:00 GMT,61.235.154.103:44666,194.85.70.31:1

027,UDP

FWIN,2005/08/19,14:39:36 +4:00 GMT,220.168.156.70:37740,194.85.70.31:1

026,UDP

FWIN,2005/08/19,14:39:36 +4:00 GMT,220.168.156.70:37740,194.85.70.31:1

027,UDP

FWIN,2005/08/19,14:44:34 +4:00 GMT,222.241.95.69:32875,194.85.70.31:10

27,UDP

Эта распечатка демонстрирует попытки прощупывания ЭВМ с IP-адресом 194.85.70.31 на предмет откликов со стороны портов 1026 и 1027 (протоколы cap и exosee). Зондирование производится с нескольких разных адресов (61.235.154.103, 220.168.156.70 и 222.241.95.69). Объектом атаки в данном случае является рабочая станция, которая не поддерживает эти протоколы.

После проникновения хакер старается ликвидировать следы своей работы и в то же время оставить для себя "калитку". Сделать это он может, заведя новую учетную запись или загрузив троянского коня. По этой причине нужно регулярно проверять список учетных записей и сканировать машину на предмет наличия троянских коней.

**Использование журнальных файлов при расследовании сетевых инцидентов**

Для отслеживания работы ОС и приложений обычно предусматривается система журнальных файлов, которая фиксирует все события (приход запросов, соответствие запросов определенным критериям и т.д.)

Рассмотрим использование журнальных файлов на примере анализа успешной атаки вторжение через приложение SSH.

Если возникло подозрение относительно возможного вторжения, надо начинать с просмотра файлов **secure** и **messages** (каталог / **var** / **log** / ОС LINUX). В нашем случае атака началась в пятницу вечером (8-го июля 2005 года). Ниже представлены фрагменты журнальных файлов, иллюстрирующие характер атаки.

Jul 8 18:23:18 fender sshd[15017]:Illegal user anonymous from 207.232.63.45

Jul 8 18:23:20 fender sshd[15019]: Illegal user bruce from 207.232.63.45 (Нью-Йорк, США)

Jul 8 18:23:22 fender sshd[15021]: Illegal user chuck from 207.232.63.45

Jul 8 18:23:23 fender sshd[15023]: Illegal user darkman from 207.232.63.45

…

Jul 9 13:15:13 fender sshd[16764]: Illegal user bruce from 129.237.101.171

Jul 9 13:15:14 fender sshd[16766]: Illegal user chuck from 129.237.101.171

Jul 9 13:15:16 fender sshd[16768]: Illegal user darkman from 129.237.101.171

Jul 9 13:15:17 fender sshd[16770]: Illegal user hostmaster from 129.237.101.171

….

Jul 10 15:25:34 fender sshd[28450]: Did not receive identification string from 80.18.87.243\par

Jul 10 16:56:16 fender sshd[28457]: Illegal user lynx from 80.18.87.243\par

Jul 10 16:56:17 fender sshd[28459]: Illegal user monkey from 80.18.87.243\par

Jul 10 16:56:18 fender sshd[28461]: Illegal user lion from 80.18.87.243\par

….

Jul 10 02:42:02 fender sshd[18064]: Did not receive identification string from 166.70.74.35

Jul 10 03:09:13 fender sshd[18067]: Illegal user admin from 166.70.74.35 (Солт Лейк Сити, США)

Jul 10 03:09:14 fender sshd[18069]: Illegal user admin from 166.70.74.35

Jul 10 03:09:16 fender sshd[18071]: Illegal user admin from 166.70.74.35

….

Jul 10 16:56:16 fender sshd[28457]: Illegal user lynx from 80.18.87.243 (Венеция, Италия)

Jul 10 16:56:17 fender sshd[28459]: Illegal user monkey from 80.18.87.243

Jul 10 16:56:18 fender sshd[28461]: Illegal user lion from 80.18.87.243

….

Jul 10 16:56:40 fender sshd[28509]: Failed password for root from 80.18.87.243 port 45208 ssh2

Jul 10 16:56:41 fender sshd[28511]: Accepted password for root from 80.18.87.243 port 45298 ssh2

….

Jul 10 19:13:49 fender sshd[31152]: Accepted password for root from 81.181.128.181 port 4943 ssh2

Листинг 13.1.

Из записей видно, что машина была атакована из 7 точек. Четыре расположены в США ( IP=166.70.74.35; 207.232.63.45; 129.79.240.86 и 129.237.101.171 ), по одной — в Италии, Румынии и Венгрии ( IP=80.98.194.185 ). Производится подбор параметров доступа имя-пароль. Подбор продолжался около двух суток.

Успешный вариант был найден машиной из Италии ( IP-адрес=80.18.87.243 Венеция). Практически сразу атака со стороны всех ЭВМ была прервана и хакер вошел на атакуемую ЭВМ ( имя\_ЭВМ=fender ) из машины с IP=81.181.128.181 ( **Румыния** ).

Для дальнейшего анализа событий нами были использованы данные из файла **.bash\_history**, куда записываются все команды, исполняемые пользователем в терминальном режиме. Записи этого файла и результаты работы демона **syslogd** говорят о том, что через 3 минуты после успешного вторжения хакер заблокировал работу **syslog**.

Далее хакер заблокировал доступ к системе других пользователей, загрузил туда файл **pass\_file** (объем 696057 байт), содержащий комбинации имя-пароль (небольшие фрагменты содержимого файла представлены ниже).

lynx lynx

monkey monkey

lion lion

heart heart

michel michel

alibaba alibaba

…..

root 123456

root 1234567

….

root 1234567890

root rootroot

root rootrootroot

root 123root123

root 987654321

….

root 4321

root 321

root root!

root root!@

root root!@#

….

Хакер рассчитывает на то, что пользователь ЭВМ ленив и выбирает простой пароль (легче запомнить — легче подобрать). Кроме того, хакер скопировал на взломанную ЭВМ несколько скриптов и файл со списком адресов — кандидатов на взлом. После этого машина включилась в работу по подбору паролей на других ЭВМ.

К сожалению, факт атаки был установлен лишь утром в понедельник. Сначала была проведена частичная блокировка. Хакер почувствовал неладное и выдал команды last и ps, пытаясь понять, что происходит; дальнейшая его работа была полностью блокирована.

Какие выводы из этой истории можно сделать? На атакованной рабочей станции была установлена SSH устаревшей версии (имевшей уязвимость) и использован достаточно простой пароль. По этой причине нужно своевременно обновлять ОС и версии приложений. (Про выбор паролей смотри RFC-2196.) Особо опасными с точки зрения атак является ночь и выходные дни. Если нет насущной необходимости, лучше на это время блокировать доступ к ЭВМ или даже выключать ее.

Помимо журнальных файлов ОС надо просматривать и соответствующие файлы приложений, например, *Firewall* (BlackIce Defender, ZoneAlarm и т.д.), Apache, баз данных и пр. Если даже в вашей *зоне ответственности* только один компьютер, просмотр всех важных файлов достаточно трудоемок. По этой причине следует рассмотреть возможность использования специализированных скриптов, которые возьмут эту работу на себя, информируя вас в случае выявления тревожных событий. Результаты работы скриптов должны накапливаться в базе данных. Эти данные могут использоваться для получения данных об атакерах и формирования ACL (списков управления доступом).

Хакер может попытаться уничтожить следы своего пребывания, стерев или очистив определенные журнальные файлы. Вполне возможно, в нашем случае хакер так бы и поступил, восстановив перед уходом и доступ по SSH. По этой причине следует заранее побеспокоиться о периодическом копировании журнальных файлов на недоступное для хакера устройство или сохранении их в зашифрованном виде. Но хакер может поступить и более жестоко, например, разметив заново системный диск. Хорошая схема защиты должна предотвращать такого рода действия или позволять хотя бы быстро восстанавливать разрушенную конфигурацию системы.

Следует учитывать, что сами журнальные файлы могут стать объектом атаки типа DoS. Большой поток запросов, поступающих с нескольких ЭВМ и обращенных к одному или нескольким ресурсам машины, могут привести к быстрому росту журнальных файлов, переполнить дисковое запоминающее устройство и блокировать работу сервера.

Широкое внедрение ИНТРАНЕТ, где группы разбросанных по сети пользователей локальных сетей объединяются друг с другом с помощью виртуальных каналов ***VLAN*** (Virtual Local Area Network; <http://www.3com.com/nsc/200374.html>), потребовало разработки новых протоколов. Архитектура *VLAN* позволяет эффективно разделять трафик, лучше использовать полосу канала, гарантировать успешную совместную работу сетевого оборудования различных производителей и обеспечить высокую степень безопасности. При этом пакеты следуют между портами в пределах локальной сети. В последнее время для задач построения *VLAN* разработан стандартный протокол **IEEE 802.10** (3-ий сетевой уровень). Этот протокол предполагает, что пакеты *VLAN* имеют свои идентификаторы, которые и используются для их переключения. Протокол может поддерживать работу 500 пользователей и более. Полное название стандарта - IEEE *802.10* *Interoperable* LAN/MAN Security (MAN — *Metropolitan Area Network* — региональная или муниципальная сеть). Стандарт принят в конце 1992 года. Количество *VLAN* в пределах одной сети практически не ограничено. Протокол позволяет шифровать часть заголовка и информационное *поле пакетов*.

Стандарт IEEE *802.10* определяет один *протокольный блок данных* (*PDU*), который носит название **SDE** (Secure *Data Exchange*) *PDU*. Заголовок пакета IEEE *802.10* имеет внутреннюю и внешнюю секции и показан на [рис. 13.4](https://intuit.ru/studies/courses/1124/201/lecture/5223?page=3#image.13.4).



**Рис. 13.4.**Формат пакета IEEE 802.10

Поле чистый заголовок включает в себя три субполя. MDF (Management Defined Field) является *опционным* и содержит информацию о способе обработки *PDU*. Четырехбайтовое субполе said (Security Association Identifier) — идентификатор сетевого объекта ( *VLAN* ID). Субполе *802.10* LSAP (Link Service Access Point) представляет собой код, указывающий принадлежность пакета к протоколу *vlan*. Предусматривается режим, когда используется только этот заголовок.

Защищенный заголовок копирует себе адрес отправителя из mac-заголовка (MAC — Media Access Control), что повышает надежность.

Поле ICV (*Integrity Check* Value) служит для защиты пакета от несанкционированной модификации. Для управления *VLAN* используется защищенная управляющая база данных **SMIB** (*security management* information base).

Наличие *VLAN* ID (said) в пакете выделяет его из общего потока и переправляет на опорную магистраль, через которую и осуществляется доставка конечному адресату. Размер поля data определяется физической сетевой средой. Благодаря наличию mac-заголовка *VLAN* -пакеты обрабатываются как обычные сетевые кадры. По этой причине *VLAN* может работать в сетях TCP/IP (*Appletalk* менее удобна). В среде типа Netbios работа практически невозможна. Сети ATM прозрачны для *VLAN*. Протокол *VLAN* поддерживается корпорацией Cisco, *3com* и др. Хотя *VLAN* ориентирован на локальные сети, он может работать и в WAN, но заметно менее эффективно. В последнее время разработано большое число специальных программных средств сетевой безопасности.

Одним из наиболее эффективных способов защиты информации является шифрование сообщений, что, к сожалению, заметно увеличивает время отклика (шифрование-дешифрование). Задержки при большом входном трафике могут привести к блокировке сервера.

Традиционные *Firewall* постепенно замещаются программами, способными анализировать не только заголовки, но и данные, например, XML- *firewall* или WEB-сервис *firewall*. Следует иметь в виду, что наличие *Firewall* сети или отдельной ЭВМ не является гарантией безопасности, — в частности, потому, что сама эта программа может стать объектом атаки. XML- *firewall* анализирует содержимое (поле данных пакетов) сообщений и контролирует аутентификацию, авторизацию и акаунтинг. Различие между XML- *firewall* и WEB-сервис *firewall* заключается в том, что последний не поддерживает открытые стандарты. В настоящее время *Firewall* должны контролировать четыре аспекта:

1. Целостность сообщений
2. Предотвращение DoS атак
3. Защита с учетом **анализа данных**
4. Аутентификация и авторизация.

Смотри:

* <http://www.nwc.com/showitem.jhtml>
* [http://www.oasis-open.org](http://www.oasis-open.org/)
* [http://vulcan.forumsys.com](http://vulcan.forumsys.com/)
* http://nwc.securitypipline.com (Network Computing USA's Security Pipeline — 'XML Gateways' статья Lori McVittie).

До недавнего времени компания CISCO хранила пароли в конфигурационных файлах в виде открытого текста, теперь они собираются использовать хешированное хранение паролей.

Но современные ЭВМ легко позволяют подобрать пароль при наличии хэша (2ГГц Intel может проверить 5000 паролей в сек). По этой причине нужно ограничить доступ к файлу хэшей паролей. Табличный метод (rainbow tables) может ускорить подбор на порядок.

Многофакторная аутентификация является, похоже, единственной альтернативой современной системе паролей. Здесь имеется в виду использование сертификатов, ID-карт пользователей, и контроль их биометрических данных (отпечатков пальцев, голоса или радужной оболочки глаза).

Многие администраторы, установив *Firewall*, *IDS* и антивирусную защиту, считают задачу обеспечения сетевой безопасности выполненной. К сожалению, это лишь небольшая часть мер обеспечения безопасности. При проектировании системы безопасности полезно иметь в виду пирамиду Maslow'а (смотри [рис. 13.5](https://intuit.ru/studies/courses/1124/201/lecture/5223?page=3#image.13.5)). И начинать надо с проектирования основания пирамиды, где расположено управление обновлением программных продуктов (Patches) и организация основных процедур. Это логично, так как хакеры обычно фокусируют свои усилия на известных уязвимостях ОС или приложений, и своевременное их обновление, блокирующее выявленные слабости, крайне важно. Такие обновления должны сначала тестироваться с целью детектирования возможных негативных последствий, прежде чем они будут рекомендованы или установлены на всех ЭВМ организации. Источник обновления должен также проверяться всеми возможными средствами. Выполнение обновлений должно поручаться квалифицированному персоналу. Должны быть разработаны инструкции для базовых операций администрирования ЭВМ и сети в целом. Сюда входят операции аутентификации, авторизации и контроля качества паролей, шифрование административного трафика, обслуживание журнальных файлов и т.д.

Несколько увеличить безопасность может применение *VLAN* или VPN. Эти технологии не дают абсолютной защиты, но заметно поднимают уровень безопасности. Здесь нужно разделять истинные VPN и виртуальные сети, формируемые в рамках протокола *MPLS*. Последние помогают улучшить ситуацию лишь незначительно. Но пометка определенных потоков с помощью DSCP, меток *MPLS* или IPv6, не давая реальной защиты, заметно усложняют работу хакера (что само по себе уже неплохо).



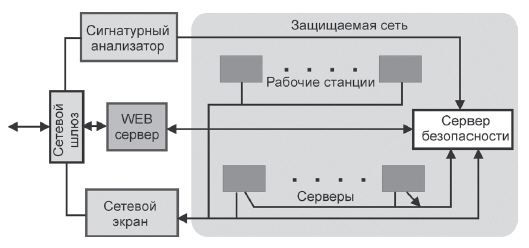
**Рис. 13.5.**Пирамида Maslow

На следующем уровне размещается *архитектура безопасности* сети и составляющих систем ( *Firewall*, системы управления доступом на прикладном уровне). Конечно, *Firewall* закрывает многие уязвимости, создаваемые дурным администрированием, но не все.

Далее следует уровень безопасности специфических программ организации, так как именно они становятся чаще всего мишенями атаки. Например, FreeBSD имеет **МАС** -механизм (*Mandatory* Access Control), который препятствует приложению вести себя некорректным образом. Аналогичные возможности имеет **SELinux** (Security Enhanced Linux). Но конфигурирование этих систем весьма сложно. На этом же уровне работают списки доступа **ACL** (Access Control List).

На вершине пирамиды находятся системы *IDS* / *IPS*. Сочетание всех этих средств обеспечит 85% безопасности, но оставшиеся 15% закрыть крайне сложно.

На [рис. 13.6](https://intuit.ru/studies/courses/1124/201/lecture/5223?page=4#image.13.6) отображена схема системы безопасности локальной сети LAN (выделена серым фоном). Связь LAN с внешней сетью осуществляется через сетевой экран ( *Firewall* ) и сетевой шлюз, с которого зеркалируются входящий и исходящий потоки данных на сигнатурный анализатор (например, *IDS* -SNORT). WEB-сервер доступен со стороны Интернет непосредственно. Доступ из Интернет в LAN через него не возможен. Сервер безопасности воспринимает данные от WEB-сервера, сигнатурного анализатора и сетевого экрана. При необходимости часть данных может передаваться им на WEB-сервер. При такой конфигурации сервер безопасности может выполнять функции *IPS*.



**Рис. 13.6.**Система безопасности локальной сети

Администратор может поменять стандартные значения номеров портов для обычных видов сервиса (SSH, FTP, WWW и т.д.). Это не делает сервисы безопасными, но заметно осложняет работу хакера. Он может просканировать порты и найти нужное значение, но это, вероятно, привлечет внимание администратора. Аналогично, можно поменять имена некоторых системных утилит, например, cmd.exe, telnet.exe, tftp.exe и т.д.. При этом номера портов нужно будет задавать явно, что осложнит и работу обычных пользователей. Такие меры можно рассматривать в качестве дополнительных.

Одним из средств противодействия атакам является введение производителями микропроцессоров флага NX (не исполнять), который позволяет разделить память на области, где вариация содержимого возможна, а где — нет. Такая технология препятствует модификации хакерами демонов и фрагментов ОС и позволяет заблокировать любые атаки, сопряженные с переполнением буферов. Данная технология внедрена компанией AMD в своем 64-битовом процессоре, а Intel — в процессоре Itanium (2001 год). Для широкого внедрения этой техники нужно переписать и перекомпилировать существующие ОС. В настоящее время эта техника внедрена в Microsoft Windows Server 2003 (Service Pack 1), Microsoft Windows XP (Service Pack 2), SUSE Linux 9.2 и Red Hat Enterprise Linux 3 с обновлением 3. Некоторые разработчики ОС внедрили эмуляции NX для процессоров, где этот флаг аппаратно не поддерживается. Развитие технологий NX может по оценкам экспертов (*Network Security*, V2005, Issue 2, февраль 2005, стр. 12-14) к 2009 году закрыть угрозу атак, сопряженных с переполнением буферов. Эта же методика может блокировать и распространение Интернет-червей.

Одной из наиболее частых мишеней атак являются базы данных, которые являются основой большинства информационных систем. Разработана схема, при которой исходная копия базы хранится на базовой ЭВМ, не связанной с Интернет. Копии этой базы передаются на компьютеры, которые обслуживают внешние сетевые запросы. При этом данные снабжаются цифровой подписью базовой ЭВМ. Вместе с откликом на запрос пользователь получает не только запрошенные данные, но и подтверждение того, что они получены из исходной базы. Как отклик, так и подтверждение снабжаются цифровой подписью, что позволяет клиенту проверить неискаженность полученных данных.

Пользователь сети должен ответить себе на следующие вопросы:

* Почему могут атаковать его сервер или рабочую станцию?
* Какие угрозы и при каких условиях могут ему угрожать?
* Насколько надежно и от каких угроз защищена сеть?

Ответив на эти вопросы, он сможет определить, какие средства защиты следует использовать. Некоторые пользователи, зная, что работают за *Firewall* или что почтовый сервер снабжен антивирусной защитой, считают, что им ничего не грозит. При этом никогда не следует исходить из предположения, что если у вас нет никакой привлекательной для воров информации, то ваша машина в безопасности. Она может быть нужна хакерам для атак других ЭВМ, для рассылки *SPAM* и т.д. Ваша ЭВМ может быть привлекательна для хакера просто потому, что плохо защищена. Конечно, сетевая безопасность является областью ответственности администраторов, но пользователи должны понимать, от чего они защищены, а от чего — нет.

Стирайте или еще лучше физически уничтожайте неиспользуемые диски, CD и другие носители.

При работе с беспроводными сетями следует применять VPN с шифрованием. При работе с беспроводными сетями используется протокол **IS-41-C** (США), который модельно сходен со стандартом GSM. Беспроводное оборудование должно отключаться сразу после завершения использования.

Администраторы должны выдавать новый пароль в случае утраты, только позвонив предварительно клиенту по телефону.

В случае работы с беспроводными сетями при выявлении подозрительного объекта желательно его локализовать. Это может быть сделано с помощью узконаправленной антенны c аттенюатором входного сигнала. Помочь этому может программа *GSP*, поставляемая вместе с Kismet. Система аудита беспроводной сети должна непрерывно контролировать сотни устройств внутри и вблизи здания, где работает мониторируемая беспроводная сеть.

Существуют специальные средства выявления уязвимостей сети. Следует учитывать, что полный доклад о таких уязвимостях может иметь размер телефонной книги. В такой ситуации трудно решить, с чего начать, если число уязвимостей больше 10000. Одним из возможных подходов является использование этих данных совместно с результатами работы *IDS*. Это сделано, например, в **ESM** (*Enterprise Security* Management) или в **SIM** (Security Information Management).

Выявление все новых уязвимостей вынуждает более внимательно относиться к системам обновлений (patches). В случае обеспечения безопасности отдельной ЭВМ применима системы Microsoft SMS (System Management Server) и SUS (Software Update Service). Существуют и общедоступные средства, например, [http://www.patchmanagement.org](http://www.patchmanagement.org/). Следует учитывать, что процесс копирования и обновления ОС или приложений сам по себе является уязвимым и проводить его следует со всеми возможными предосторожностями.

Разработчики и коммерческие поставщики программных продуктов часто выдают программы, структура которых не удовлетворяет самым элементарным требованиям безопасности. Из-за спешки иногда в новые версии программ переносятся уязвимости, выявленные в предыдущих вариантах.

*Хорошие результаты с точки зрения безопасности могут быть получены, если запретить пользователям устанавливать программное обеспечение по своему усмотрению*.

В последнее время разработана новая технология, которая обеспечивает приемлемый уровень безопасности. Это **VE** (*Virtual Environment*) (см. *Network Security*, V2004, Issue 11, ноябрь 2004, стр. 18-19). Сходная технология реализуется в Jail-серверах системы LINUX. Целью этой технологии является предотвращение возможного ущерба от вредоносной программы. Здесь каждая программа выполняется на отдельном виртуальном компьютере. Любые обмены между виртуальными машинами запрещаются. В таких условиях *вирусы*, например, не могут ничего повредить, кроме самих себя: о существовании других программ они просто не могут знать. Для обычного же пользователя все остается неизменным. Реализация VE не эмулирует ЦПУ или другие ресурсы и функции. Вместо этого осуществляются операции с системными объектами и маршрутизацией входных/выходных вызовов. Таким образом, исключается какое-либо замедление работы машины.

VE определяет схему взаимодействий и распределение прав доступа. VE инкапсулирует программу или группу программ, предоставляя весь спектр услуг (память, коммуникации и пр.). При этом дублируются все необходимые данные. VE не могут быть полностью изолированы друг от друга и, тем более, от ОС. Типичное взаимодействие между VE сопряжено с использование общих данных. Любые операции VE связанные с ресурсами ЭВМ находятся под жестким контролем.

Для файловой системы и Registry главным препятствием является дублирование ресурсов. C одной стороны, программа должна иметь возможность изменять существующие значения, например, ключей Registry или конфигурационных файлов ОС. С другой стороны, если программа является *вирусом*, любые изменения должны блокироваться. Если, например, программа запрашивает изменение ключа конфигурационного реестра (Registry), генерируется новая копия этого ключа и она становится частью соответствующего VE (метод Copy-On\_Write). Если программа решает уничтожить ключ, она ликвидирует локальную копию. Для всех программ данного VE ключ перестает существовать. Но копия ключа ОС остается неизменной (метод Leave-On-Delete).

Для обеспечения полной безопасности VE-технология должна размещаться ниже ядра, а не работать параллельно ему. Если переместить ядро в сектор 1 или 2, VE-устройству будет предоставлен полный контроль над ЭВМ. Существует несколько вариантов взаимодействия VE и ОС.

Так как самой массовой ОС для серверов в настоящее время являются различные разновидности LINUX, разработчикам новых версий этой ОС следует задуматься о встроенных средствах безопасности. Одним из таких подходов может стать минимизация ядра ОС. Только для ядра небольшого размера можно гарантировать его безопасность, тексты с миллионом строк привилегированного объектного кода неконтролируемы. В такое ядро должно быть включено минимальное число функций (управление памятью, *критическими ресурсами* и доступом). ОС не должна напрямую контактировать с внешними устройствами. Остальные функции должны иметь модульный характер и включаться в оболочку.

Работа Syslog допускает передачу данных посредством протокола UDP (существует версия syslog-ng, где транспортным протоколом может быть TCP). Если атакер имеет доступ к каналу между отправителем и получателем, он сможет отслеживать обмен и уничтожать или фальсифицировать данные, говорящие о его присутствии. Хакер может также посылать уведомления получателю (по известному порту), добиваясь переполнения диска и блокировки работы системы *журналирования*. Журнальные файлы, помимо достоверности и полноты, должны обладать также юридической корректностью, чтобы их данные можно было использовать при судебных спорах. Это предполагает применение электронных подписей, чего по умолчанию пока нет ни в одной из систем syslog. Но существуют версии, где делается попытка решить все эти проблемы: модульный syslog, *SDSC* Syslog, Syslog Ng и Kiwi.

Если атакер обычно собирает данные о будущем объекте атаки, то так же следует действовать и потенциальным жертвам. Необходимо формировать динамически обновляемые базы *сигнатур атак* (уже существуют) и атакеров. Возможным инструментом сбора таких данных может стать *Honeypot* (*HoneyNet*, Honeytoken, смотри [http://project.honeynet.org](http://project.honeynet.org/)). Эти программные средства позволяют выявить последовательность действий хакера. Это особенно важно для выявления сигнатур неизвестных атак. Если вам известна скомпрометированная ЭВМ, можно послать туда параметры доступа и адрес вашего *honeypot*, чтобы спровоцировать хакера. *Honeypot* интересен тем, что там нет полезных ресурсов и обычные пользователи не будут пытаться туда войти.

**Сетевая информация, которую полезно собирать**

Следует контролировать число и состав различных *сервисов*, работающих на ЭВМ (например, telnet, ssh, *rlogin*, http, https, ldap, vns, *irc*, rpc, smtp и т.д.). Анализ этого списка и его вариаций со временем позволяет составить представление о возможной уязвимости системы. Определенную пользу может принести выявление *ОС*, установленных на ЭВМ сети (LINUX, Windows, FreeBSD и т.д.). Администратору проще иметь дело с одной ОС, но тотальная унификация имеет и недостатки. Один и тот же *вирус* или червь может парализовать единовременно всю сеть целиком. Понятно, что система сбора любых сетевых данных не должна поглощать заметной части ресурсов.

Рассмотрим признаки, которые могут свидетельствовать об успешном вторжении в ЭВМ (вариант ОС LINUX) (см. *Network Security*, V2004, Issue 8, "LINUX *intrusion* discovery: when security fails", Anton Chuvakin, август 2004, стр. 10-12).

* Чрезмерная перегрузка ресурсов (ЦПУ, оперативной и дисковой памяти).
* Частые сбои в работе ОС или приложений (самопроизвольная перезагрузка системы). Это может быть результатом, например, переполнения буферов или попытки поменять конфигурацию системы.
* Появление необычных объектов (файлов, каталогов, акаунтов, процессов). К этому классу проявлений можно отнести запуск или остановку каких-то демонов, например, syslogd.
* Необычная сетевая активность (новые соединения или резкое замедление сетевых операций).

Часто администратор лишь чувствует, что что-то происходит не так. Этого должно быть достаточно, чтобы начать исследования. Ниже приводится перечень команд, которые позволяют выявить и конкретизовать названные выше аномалии.

1. Для выявления файлов, существенно модифицированных в последнее время, что может свидетельствовать, например, о запуске программы типа *sniffer*, следует выполнить команду (наберитесь терпения, поиск будет происходить, начиная с корневого каталога):

find / -size +1000k -mtime +7 -print

Чтобы обнаружить наличие активных программ типа *sniffer*, можно выдать команду:

ip link | grep PROMISC или /sbin/ifconfig

1. Наличие файлов Nobody может указывать на то, что атакер создал, использовал и затем удалил акаунт нового пользователя. Выявить такие объекты можно с помощью команды:

find / -nouser -print

1. Для обнаружения файлов SUID root в необычных местах (например, вне каталогов /sbin и /bin), что иногда указывает на попытку сформировать люк для последующего входа в систему, следует выполнить команду:

find / -uid 0 -perm -4000 -print

1. Для выявления файлов с необычными именами (например, ".", " ", "..." и т.д.), что указывает на работу начинающего хакера, следует исполнить команду:

find / -name "..." -print

1. Для поиска подозрительных акаунтов (новые акаунты с *системными привилегиями*) можно выполнить команду:

grep :0: /etc/passwd

Попытки хакера установить свое программное обеспечение часто приводит к повреждению системных или прикладных программ. В системе RedHat имеется встроенная система **RPM** (RedHat Package Manager), которая позволяет выявить такие повреждения. Например, команда

rpm -qa | # rpm -Va | sort

поможет решить эту проблему. Для этой же цели можно использовать программу **chkrootkit** ([http://www.chkrootkit.org](http://www.chkrootkit.org/)).

Просмотрев список активных процессов с помощью команды ps-*aux*, следует обратить внимание на демоны, проход к которым начинается с символа "точка", а также на любые необычные имена процессов. Чтобы узнать подробности, следует заглянуть непосредственно в каталог /proc. Например, выдав команду cat/proc/20999, где 20999 — pid процесса. Для выявления прослушивающих сервисов можно выдать команду netstat-*nap*.

Аналогичные средства диагностики существуют и для ОС Windows (смотри, например, <http://www.cert.org/tech_tips/WIDC.html>). Следует помнить, что файловые системы FAT и FAT32 (Windows) не имеют средств защиты, поэтому при построении систем повышенной безопасности следует отдать предпочтение системе NTFS.

1. Одной из наиболее серьезных угроз является *rootkit*. Это программа, во многом подобная троянскому коню. Она может скрывать свое присутствие, перехватывать пароли, инсталлировать секретные входы в систему ("черный ход") и атаковать другие машины. Некоторые виды *rootkit* не дают шансов себя обнаружить. Чтобы гарантировать успешное детектирование *rootkit*, соответствующие программы нужно запускать в изолированной ("чистой") среде. Ниже даны ссылки на некоторые источники, откуда можно взять программы детектирования toolkit.
   * Rkdetect – <http://www.security.nnov.ru/soft/>
   * *Rootkit* *Revealer* – <http://www.sysinternals.com/Utilities/RootkitRevealer.html>
   * BartPE бутабельная ОС, загружаемая с CD и запускающая двоичные коды Win32 – <http://www.nu2.ne/pebuilder/>
2. Полезно регулярно проверять перечень акаунтов пользователей на машине. Это можно сделать с помощью команды wmic useraccount.
3. Важно контролировать права пользователей. Для windows XP нужную информацию можно найти по адресу: <http://www.microsoft.com/resources/documentation/Windows/XP/all/reskit/enus/prnd_urs_mhnn.asp>.
4. Для контроля автоматического запуска неавторизованных приложений можно воспользоваться командой: wmic /output:C:\services.htm service get /format:hform (для Windows-XP). Полезно также регулярно просматривать содержимое файлов Autoexec. bat, config.sys, system.ini и win.ini. Некоторые программы для "черного хода" загружаются в качестве услуг на фазе загрузки системы. Просмотрев список загруженных услуг, нужно удалить все ненужные.
5. Целесообразно на регулярной основе мониторировать модификации двоичных системных файлов. Контроля размера файла недостаточно, так как некоторые троянские кони сохраняют размер файла. Наилучший способ – это вычисление контрольных сумм с привлечением алгоритмов MD5 или SHA-1. Смотри <http://support.microsoft.com/?kbid=310747> или <http://support.microsoft.com/?kbid=222193>.
6. Для контроля конфигурации сети на предмет поиска неавторизованных включений следует воспользоваться услугами <http://support.microsoft.com/?kbid=837243>. При этом полезно заглянуть в каталог %systemroot%\system32\drivers\etc\hosts. Следует искать необычные номера портов, просушивающие соединения. Это можно сделать с помощью команды: netstat –an. XP позволяет проверить принадлежность процессов определенному порту с помощью команды netstat –ao. Преобразование ID процессов в их имена можно осуществить посредством команды: wmic process where ProcessId="x" get caption.
7. Выявить неавторизованные квоты можно, воспользовавшись командой net share.
8. Некоторые хакеры, чтобы скрыть свои действия, не запускают программы типа *троянский конь*, "черный ход" и т.д. немедленно, а программируют их запуск в будущем. По этой причине надо мониторировать, какие программы ждут запуска. Это можно сделать с помощью команд: at или schtasks.
9. Контроль наличия неавторизованных процессов можно осуществить с помощью команд pulist.exe или tlist.exe. Первая из команд позволяет также выявить, кто запустил тот или иной процесс. Команда tlist.exe с флагом –t отображает список процессов, породивших *дочерние процессы*. Дополнительную информацию можно получить с помощью команды msinfo32.msc.
10. Сигналом опасности может быть появление необычных *скрытых файлов*. Выявить их можно с помощью Explorer. Для этого достаточно сначала выбрать "Tools, Folder Options, View" и затем "Show *hidden files* and folders". Годится для этой цели и команда dir/ah.
11. Чтобы выявить измененные параметры доступа к файлам и ключам реестра, можно воспользоваться программами xcalc.exe или showacls.exe.
12. Для контроля за изменением политики машины или пользователя можно воспользоваться командой gpresult/v.
13. Аудит детектирования вторжений можно осуществить, выполнив из командной строки команду run gpedit.msc. Можно также воспользоваться *dsa*.msc или GPMC.msc. (Смотри также <http://www.microsoft.com/windowsserver2003/gpmc/gpmcwp.mspx>.)

Список номеров портов для известных троянских коней можно найти в <http://www.simovits.com/nyheter9902.html>

**Список некоторых известных видов атак**

Ниже перечислены некоторые разновидности *сигнатур атак* (более подробный перечень можно найти по адресу <http://www.iss.net/security_center/advice/Intrusions/default.htm> и <http://book.itep.ru/6/intrusion.htm>).

**Land attack**. Атакер пытается замедлить работу вашей машины, послав пакет с идентичными адресами получателя и отправителя. Для стека протоколов Интернет такая ситуация не нормальна. ЭВМ пытается выйти из бесконечной петли обращений к самой себе. Имеются пэтчи для большинства операционных систем.

**Unknown IP protocol**. Традиционными транспортными протоколами являются UDP, TCP и ICMP, которые работают поверх протокола IP. Код протокола определяется полем тип протокола заголовка IP-дейтограммы. Существует большое число протоколов, которые идентифицируются с помощью номеров портов, например, HTTP, использующий в качестве транспорта TCP. Появление незнакомого протокола должно всегда настораживать, так как может нарушить нормальную работу программ.

**Ping of death**. Предпринимается попытка послать пакет, длина которого больше теоретического предела 65536 байтов. IP-уровень фрагментирует такие пакеты, а получатель осуществляет сборку, которая иногда приводит к разрушению ОС. Системы до 1997 года были уязвимы для такой атаки. Операция проделывалось, например, с помощью команды ping -l 65550.

**Nestea attack**. Опасное перекрытие IP-фрагментов, сформированное программой nestea. Ваша операционная система может стать нестабильной или разрушиться. Имеются пэтчи для большинства операционных систем. Адрес отправителя, вероятнее всего, не является истинным. Это означает, что отправитель использует фальшивый IP-адрес и прикидывается кем-то еще. К сожалению, не существует простых способов определить, кто в действительности посылает кадры с искаженным адресом отправителя.

**Empty fragment**. Когда пакеты слишком длины, они могут быть фрагментированы. Система контроля фиксирует фрагмент с нулевой длиной. Например, IP-заголовок имеет 20-байт, а данных вообще нет. Это может указывать, что:

* атакер пытается обойти систему контроля вторжения;
* некоторое сетевое оборудование (маршрутизаторы/переключатели) работают некорректно, генерируя такие фрагменты;
* имеется ошибка в стеке TCP/IP машины, посылающей пакет;
* атакер пытается предпринять *DoS-атаку* против вашей системы.

Ядра Linux для версий между 2.1.89 и 2.2.3 были уязвимы для атак DoS с привлечением этой методики. Каждый такой фрагмент вызывает некоторую *потерю памяти*. Повторно посылая такие фрагменты, можно вызвать кризис из-за отсутствия свободной памяти.

**Suspicious Router advertisement**. Подозрительное анонсирование новых маршрутов. Атакер возможно пытается перенаправить трафик на себя. Для этого может использоваться, в том числе, и ICMP-протокол. Соответствующие опции должны всегда вызывать подозрение.

**Corrupt IP options**. Опции IP на практике почти не применяются. По этой причине в их обработке имеется достаточно много ошибок. Ими часто пытаются воспользоваться хакеры. Появление в пакете IP-опций должно вызывать подозрения.

**Echo reply without request**. Это может быть результатом взаимодействия хакера с засланным троянским конем. Может быть это и результатом дублирования вашего IP-адреса. В любом случае такие пакеты достойны внимания.

**UDP port loopback**. Пакет UDP путешествует между двумя эхо-портами. Такие пакеты могут делать это бесконечное число раз, используя всю имеющуюся полосу сети и производительность ЦПУ. Имеется несколько стандартных услуг такого рода, которые могут работать в системе. Среди них:

echo (порт 7)

дневная квота (порт 17)

chargen (порт 19)

Атакер может создать проблемы, фальсифицируя адрес отправителя и вынуждая две *машины бесконечно* обмениваться откликами друг с другом. Таким образом, может быть парализована вся сеть (ведь хакер может послать много таких пакетов). Например, хакер может имитировать посылку пакета от ЭВМ-A (порт *chargen*) к ЭВМ-B (порт echo). Эхо-служба ЭВМ-B пошлет отклик ЭВМ-A и т.д.

**Snork attack**. Регистрируются UDP-дейтограммы с портом назначения 135 (Microsoft *Location Service*), и отправитель с портом 7 (Echo), 19 (*Chargen*) или 135. Это попытка замкнуть две службы (если они разрешены/активированы) и заставить их бесконечно обмениваться пакетами друг с другом. Существует пэтч для блокировки таких атак (смотри сайт Microsoft).

**TCP FIN scan**. Разновидность TCP-сканирования. Однако хакер здесь пытается осуществить так называемое "FIN-сканирование". Он пытается закрыть несуществующее соединение сервера. Это ошибка, но системы иногда выдают разные результаты в зависимости от того, является ли данная услуга доступной. В результате атакер может получить доступ к системе.

**DNS cache poison**. Атакер послал запрос DNS-серверу, который содержит также и отклик. Это может быть попыткой компрометировать DNS-сервер. Это может быть также результатом работы ISP, который перенаправляет своих клиентов на прокси-сервер. Но, вероятнее всего, это вторжение в систему. Для того, чтобы улучшить рабочие параметры, DNS-серверы пытаются "кэшировать" имена локально. Серверы просматривают секции откликов всех пакетов, приходящих в систему. Они запоминают эти отклики на короткое время для случая, когда кто-то еще нуждается в этой информации. Очевидной проблемой является вариант, когда такой пакет содержит ложную информацию. В частности, кто-то может послать запрос в DNS, содержащий, кроме того, дополнительную информацию в секции отклика. Старые серверы воспринимают эту информацию, кэшируют ее, и выдают в ответ на запрос, если таковой поступит. (Новые DNS-серверы лишены этой уязвимости, но существует еще достаточно много старых серверов). Если ваш DNS-сервер обновлен, атака такого рода невозможна.

**DNS BIND version request**. Сервер BIND DNS содержит в своей базе данных запись CHAOS/TXT с именем "VERSION.BIND". Если кому-то нужна эта запись, присылается версия программы BIND. Такой запрос сам по себе не является атакой, но может быть разведывательным рейдом. Однако если возвращается версия типа "4.9.6-REL" или "8.2.1", это указывает, что у вас установлена версия с хорошо известной дыркой, связанной с переполнением буфера.

**SMB malformed**. Существует ошибка в старой версии SMB (система Microsoft для совместного использования файлов и принтеров в сети). Эта ошибка может быть использована при авторизации, путем посылки специально сформированных пакетов. При реализации этого трюка машина крэшится. Данная атака может быть предпринята успешно для систем Windows NT 4.0 SP4 и Windows 95 (все версии). Заметим, что имеются пэтчи для всех систем. Для того, чтобы дырка работала, "File and *Print Sharing*" должно быть разрешено.

**HTTP URL directory traversal/climbing**. Ситуация выглядит так, как будто атакер пытается прочесть посторонние файлы вашей системы. Обычная ошибка web-браузера заключается в том, что хакер может специфицировать URL, который выглядит как /../../../foo/bar.txt. Эта атака может удаться, так как программист не осуществляет двойной проверки URL, чтобы убедиться, корректен ли файл web-сайта. Сигнатурой такой атаки может быть наличие в URL последовательности ../... Иногда такого рода атака может быть имитирована некорректными связями, размещенными на странице. Это говорит о некорректной конфигурации. Во-первых, проверьте параметры URL, чтобы выяснить, к какому файлу намерен получить доступ атакер. Затем проверьте, получил ли атакер доступ к файлу. Если это действительно критичный файл и атакер был успешен, необходимо предпринять срочные действия. Например, если атакер получил доступ к файлу паролей, необходимо заменить все пароли. Следует также проверить, является ли версия сервера новейшей и использованы ли все существующие пэтчи безопасности. Большинство таких атак предпринимается против "встроенных" web-серверов (т.e. web-серверов, добавленных в качестве части другого программного продукта), а не против реальных web-серверов типа Apache и IIS.

**Данные HTTP CGI содержат ../../../..**. Данные, переданные в URL, имеют подозрительный проход, содержащий ../../../..; Этот проход может быть использован для доступа к привилегированным файлам. Атакер пытается добраться по дереву каталогов до нужных ему файлов. Некоторые приложения Web используют проходы, содержащие ../../../.. . Вам следует рассмотреть URL и аргумент GET с целью проверки их корректности. Если проход в аргументе GET указывает на попытку доступа к привилегированным данным, возможно, ваш сервер скомпрометирован.

**HTTP asp with \ appended**. Происходит попытка доступа к файлу asp с завершающим символом \. В некоторых ситуациях вместо исполнения программы asp будет возвращен исходный asp-файл. Это раскроет атакеру критическую информацию о сервере. Исходный текст программы сервера часто содержит пароли, скрытые имена файлов или ошибки, которые в такой ситуации относительно легко найти. Хакер может затем использовать эту скрытую (hidden) информацию для вскрытия сервера.

**Telnet Bad IFS**. В UNIX переменная "IFS" специфицирует символ, разделяющий команды. Если значение этой переменной изменено, то система детектирования вторжения не будет способна корректно интерпретировать команды. Более того, кто-либо может изменить эту переменную для того, чтобы модифицировать работу некоторых скриптов ядра. В частности, плохой IFS станет разграничителем, используемым при разборе любых вводов, таких, как имена файлов или DNS-имена. В этом случае нужно установить IFS соответствующим символу '/' или '.'. Вообще, если вы хотите жить немного спокойнее, лучше заблокировать применение telnet, предложив пользователям перейти на SSH.

**Telnet RESOLV\_HOST\_CONF**. Переменная окружения RESOLV\_HOST\_CONF посылается с привлечением поля опций Telnet. Это поле не должно никогда изменяться таким способом. Это может означать попытку получения доступа к критическим файлам типа паролей и т.д.

**SMTP DEBUG command**. Кто-то сканирует вашу систему с целью выявления ее уязвимости. В 1988 году червь Morris уложил Интернет. Одним из путей распространения червя являлась программа sendmail. Sendmail поддерживала нестандартную команду "DEBUG", которая позволяет любому получить контроль над сервером. Червь Morris автоматизировал этот процесс для распространения через системы sendmail. Сейчас маловероятно, что вы найдете такую старую почтовую систему. Следовательно, такая атака будет означать, что кто-то использует универсальный сканер уязвимости. Иногда система может выйти из синхронизма (сбой в ISN), что вызывает большие потери пакетов. В таких условиях по ошибке может быть запущена команда DEBUG. Например, если вы работаете с версией сервера для системы 486, который обрабатывает большое количество e-mail, такая десинхронизация может произойти. Уязвимой системой является Sendmail/5.5.8.

**SMTP MIME null charset**. Обнаружено незнакомое почтовое сообщение, вероятно сконструированное, чтобы разрушить почтовый сервер. В заголовки "Reply-To:" сообщения встраиваются исполняемые Shell и PERL коды. Когда система откликается, эти коды будут исполнены. Например, старые версии list-сервера MAJORDOMO исполняют любой PERL-скрипт, который вставлен в это поле.

**FTP pipe in filename**. Совершается попытка исполнить программу на FTP-сервере. Допускающая это уязвимость имеется во многих FTP-демонах до 1997 года, таких, что, если перед именем файла присутствует символ PIPE (|), сервер пытается исполнить программу имя которой следует за ним. Так как FTP-сервер обычно работает с привилегией root, может произойти его компрометация. Проверьте, что у вас работает новейшая версия сервера.

**FTP Site Exec DotDot**. Попытка неавторизованного доступа. Некоторые версии wu-*ftpd* позволяют использовать команду site exec для удаленных машин. Путем предоставления прохода с определенными параметрами удаленный пользователь может исполнить произвольные команды на FTP-сервере. Эта атака позволяет атакующему выполнять команды на атакуемой машине. Это может привести к получению им доступа root-уровня. Такой дефект имеют системы wu-*ftpd* версий 2.4.1 и ранее. Чтобы защитить себя от таких атак, просмотрите команды, которые атакер использовал. Если они представляют угрозу для атакуемой ЭВМ, можете считать машину скомпрометированной. Обновите программное обеспечение FTP-сервера или замените его вообще.

**NMAP ping**. Для проверки вашей системы на уязвимость используется программа NMAP. NMAP — очень популярная программа, применяемая хакерами для сканирования Internet. Она работает под Unix, имеет много конфигурационных опций и использует несколько трюков, чтобы избежать детектирования системами, отслеживающими вторжение. NMAP не позволяет хакеру вторгнуться в систему, но допускает получение им полезной информации о конфигурации системы и доступных услугах. Программа часто используется как прелюдия более серьезной атаки.

**SNMP WINS deletion**. Совершена попытка стереть записи WINS через интерфейс SNMP сервера. Это может быть попытка реализовать Denialof-Service (DoS) или просканировать систему безопасности. Клиенты Windows контактируют с системой WINS, чтобы найти серверы. Многие системы уязвимы для атак при помощи SNMP-команд, посланных серверу для того, чтобы стереть записи. Это не взламывает сервер, но после стирания записи в базе данных клиенты и серверы смогут более найти друг друга. Однако этот отказ обслуживания (DoS) может предварять другие атаки. Это может быть частью широко диапазонного сканирования с целью поиска слабых точек в сети. Эта атака может быть против систем, где нет работающих SNMP или WINS.

**SNMP SET sysContact**. Совершена попытка удаленно установить поле sysContact через SNMP. Это, вероятно, сканирование уязвимых мест вашей системы. Группа SNMP "system" используется всеми MIB. Она часто сканируется хакерами при предварительной разведке. Одним из способов сканирования является выполнение команд SET для поля "sysContact" для того, чтобы просмотреть, реагирует ли какая-либо система. Такие SET часто используют для взлома хорошо известные пароли/communities. Для противодействия рекомендуется запретить исполнение команд SET удаленно.

**CGI bash**. Может быть совершена попытка исполнить скрипт bash, который в случае успеха позволит хакеру получить доступ к серверу. Это программа shell, которая может выполнять произвольные операции на сервере. Если она была случайно помещена в удаленно доступный каталог и если вы обнаружили, что какие-то параметры системы стали доступны хакеру, считайте свою систему скомпрометированной. Вы должны немедленно удалить эту программу из данного каталога, проверить систему на наличие троянских коней и проконтролировать, не изменялась ли ее конфигурация. Для этого у вас должна иметься копия всех конфигурационных файлов на отдельном носителе.

**CGI perl.exe**. Может быть произведена попытка исполнить perl.exe, что позволит хакеру неавторизованный доступ к серверу. Это программа shell, которая может выполнить произвольные операции на сервере. Если эта программа случайно помещена в удаленно доступный каталог и если выявлена модификация некоторого системного параметра, можете считать свою систему скомпрометированной. Вам следует немедленно удалить программу из ее каталога и поместить в более подходящее место, проверить вашу систему на наличие троянских коней, а также проконтролировать конфигурацию на предмет возможных модификаций.

**Правила обеспечения безопасности**

В 1997 году одна из подгрупп IETF выпустила справочник Site Security *Handbook* (RFC-2196). В нем содержатся рекомендации системным администраторам по разнообразным вопросам защиты сети, правилам использования и методике работы (см. также <http://book.itep.ru/6/rfc2196.htm> и <http://book.itep.ru/6/rfc2827.htm>). Этот документ предлагает включить в правила следующие разделы.

* **Рекомендации по закупке оборудования и программного обеспечения**. Участие системных администраторов в выборе оборудования и программного обеспечения может принести большую пользу, так как часто они знают о его недостатках и ограничениях то, чего не афишируют продавцы и производители.
* **Политика секретности**. Устанавливает степень контроля над почтой и действиями пользователей, а также политику размещения пользовательских файлов.
* **Политика доступа**. Определяет, кто может иметь доступ в систему, что можно делать в рамках этих прав доступа, какое программное обеспечение можно установить и пр. Данный документ должен включать те же меры предосторожности относительно авторизации доступа и степени контроля, что и политика секретности.
* **Политика учетных записей**. Содержит описание прав и обязанностей пользователей и системных администраторов.
* **Политика аутентификации**. Устанавливает правила использования паролей и порядка лишения доступа, а также методы аутентификации пользователей.
* **Политика доступа**. Определяет, в какое время система должна быть доступна, содержит расписание обслуживающих мероприятий, перечень действий при появлении проблем, а также инструкции по документированию проблем и оповещению о них администраторов и ориентировочное время их устранения.
* **Политика управления**. Устанавливает правила общения с внешним миром и порядок доступа для приглашенных из других организаций специалистов.

**Правила для пользователей**

В правилах для пользователей необходимо регламентировать следующие вопросы.

* Использование учетных записей совместно с друзьями и родственниками.
* Выполнение программ дешифрования паролей для расшифровки локального файла **passwd**, например, с помощью программы **crack**.
* Выполнение программ дешифрования паролей для расшифровки файлов **passwd** других систем.
* Нарушение нормального процесса обслуживания.
* Проникновение в чужие учетные записи.
* Неправильное использование электронной почты.
* Просмотр файлов других пользователей (есть ли возможность чтения и записи, одобряется ли?)
* Публикации в UseNet (запрещены? с оговорками? разрешены?)
* Импорт программ из Интернет (запрещен? разрешен? разрешен с оговорками?)
* Использование системных ресурсов (принтеров, дисков, модемов, процессора).
* Копирование лицензионного программного обеспечения.
* Выдача разрешений на копирование лицензионного программного обеспечения другим лицам.
* Копирование защищенных авторскими правами материалов (музыки, фильмов и пр.).
* Всевозможная незаконная деятельность: мошенничество, *клевета* и др.
* Вовлечение в деятельность, которая является запрещенной (например, порнография).
* Сканирование локальной сети, перехват пакетов в сетевом сегменте подключения.
* Подключение из дома к своему служебному компьютеру через модем.

*Полезно объяснить пользователям, что их поведение в сети должно быть этичным*!

Примером соглашения для доступа к компьютерам может служить документ такого рода для факультета информатики университета Мельбурна. (Смотри также [http://www.admin.com](http://www.admin.com/).)

*Следует, впрочем, иметь в виду, что, прочитав и подписав правила работы в сети, пользователь, если даже он понял, что он подписал, об этом немедленно забывает*.

*Попытайтесь убедить его не давать свой пароль даже очень большому начальнику*.

**Безопасность WEB-серверов**

WEB-сервер — достаточно сложная и потому уязвимая для атак программа. Причем угрозы могут исходить из самых неожиданных мест. Так в конце июня 1997 года было обнаружено, что Windows-95 (и NT) "повисает" (полный перечень причин повисания этой системы может занять целый том) при приходе на ее вход ICMP-пакета с длиной, которая не соответствует значению, указанному в его поле заголовка Длина.

WEB-сервер, так же, как любая сеть, должен иметь свою, желательно изложенную на бумаге политику безопасности. Это должен быть достаточно простой документ типа приведенного ниже, а доступ к WEB-серверу должен имеет пять уровней.

1. **Общедоступный** с возможностью только чтения всех URL за исключением тех, что помещены в каталогах /private.
2. **Доступ сотрудников фирмы или организации**, которой принадлежит сервер. Здесь также допустимо только чтение, но доступны и секции каталога /private.
3. **Разработчики WEB-сервера**. Имеют возможность модифицировать содержимое сервера, инсталлировать *CGI-скрипты*, прерывать работу сервера.
4. **Администраторы узла (сервера)**. Имеют те же привилегии, что и разработчики, но могут также реконфигурировать сервер и определять категорию доступа.
5. **Системные администраторы**. Имеют идентичные привилегии с администраторами сервера.

Для получения доступа на уровне 3-5 необходимо письменное разрешение директора организации или его заместителя по информационным системам. Доступ уровня 2 автоматически получают все сотрудники организации или фирмы при авторизации. Администраторы могут аннулировать авторизацию по решению заместителя директора по информационным системам, а при чрезвычайных обстоятельствах — самостоятельно, но с последующим уведомлением руководства. Работа с локальной консоли WEB-сервера разрешается только администраторам. *Удаленная работа администраторам запрещена, они должны работать только с локального терминала*. *CGI-скрипты* устанавливаются на сервер после их проверки и одобрения как минимум двумя членами группы администраторов. Скрипты, исходные тексты которых недоступны, устанавливаются только по решению заместителя директора по информационным системам.

Информация из каталогов /private, которая считается конфиденциальной, доступна только с терминала самой ЭВМ.

При работе с WEB-сервером не разрешается доступ к базам данных или файлам, если для этого не имеется соответствующего индивидуального договора .

Описание политики безопасности должно включать указание периода формирования резервных копий содержимого сервера, описания допустимых сетевых услуг и время профилактических остановок. Включается сюда перечень видов обязательного мониторинга сервера и просмотра дневника посещений.

Приведенный текст описания политики безопасности может варьироваться в широких пределах, он зависит от используемой ОС и набора сетевых утилит.

Наиболее безопасной сетевой средой считается Macintosh OS. Это связано с тем, что она не включает в себя *интерпретатора команд*, не поддерживает скрипты и не предоставляет каких-либо дополнительных сетевых услуг, — неавторизованный просмотр WEB-страниц на *Макинтоше* практически не возможен.

Системы Windows NT и UNIX обладают сопоставимыми и достаточно высокими уровнями безопасности. Большое число сообщений о дефектах безопасности UNIX свидетельствует о его массовом использовании.

Теперь рассмотрим, что нужно сделать, чтобы обеспечить максимально возможную безопасность WEB-cервера.

* Выбрать наиболее безопасную ОС и сконфигурировать ее с учетом требования безопасности. Использовать все известные корректирующие программы, выпущенные разработчиком ОС.
* Организовать мониторирование любой *подозрительной активности* на сервере (активность в ночное время, многократные попытки авторизации и т.д.).
* Контролировать доступ к конфиденциальным документам. Доступ к таким документам должен быть разрешен только ограниченному числу пользователей. Доступ к таким частям сервера должен быть организован с использованием протокола SSL.
* Тщательно разрабатывать и проверять используемые *CGI-скрипты* и аплеты.
* Установить жесткие требования к доступу для выполнения различных операций, особенно для модификации содержимого и конфигурации сервера.
* Защитить локальную сеть от WEB-сервера. Исключить возможность проникновения к жизненно важным ресурсам сети через WEB-сервер, например, с помощью *Firewall*.
* Отслеживать вновь обнаруженные слабости используемой ОС и программного обеспечения сервера. Делайте это чаще, если вам не безразлична безопасность вашего сервера, не надейтесь, что все хакеры ленивее вас.

При работе с ОС Windows NT следует отключить доступ TCP/IP от услуг NETBIOS. Это может быть сделано с помощью *Firewall*, блокировкой доступа к портам 137 и 138 для UDP и TCP. Можно решить эту проблему отключения NETBIOS от TCP/IP драйвера переконфигурировав Windows NT.

Если WEB-сервер нуждается в контроле доступа, то в настоящее время (в HTTP/1.1) имеется две возможности. Первая (basic) предполагает традиционный ввод и передачу по сети имени клиента и пароля. Эта схема проста, но допускает перехват параметров доступа (а между клиентом и сервером может быть достаточно много промежуточных узлов). Вторая схема (digest) для пользователя выглядит аналогично, но вводимое имя и пароль не передаются по сети непосредственно. На их базе формируется дайджест MD5, который пересылается по сети и используется для идентификации клиента.

### Выводы

1. Абсолютно надежных методов защиты сетевых объектов не существует, не следует ждать их появления в ближайшее время. По этой причине конфиденциальные материалы следует хранить в местах, недоступных сетевым образом.
2. Система защиты должна быть адекватна возможному ущербу, связанному с возможным вторжением, и не создавать чрезмерных неудобств пользователям сети.
3. Так как вероятность повреждения сетевых средств не равна нулю (это сопряжено не только с возможностью вторжения, но и, например, с повреждением системного диска, пожаром и т.д.) нужно организовать службу резервного копирования критичных файлов и хранения их в безопасном месте.
4. Даже при наличии изощренной системы защиты (имеется в виду сигнатурный контроль) всегда имеется временное окно уязвимости с момента изобретения атаки до появления средства противодействия. По этой причине системы регистрации аномалий позволяют уменьшить вероятность успеха неизвестных пока атак.
5. Никогда не следует загружать программы, происхождение которых надежно не известно. Если по какой-либо причине это сделать нужно, следует их сначала проверить на наличие *вирусов*, *spyware*, троянских коней и пр. Предпочтение нужно отдавать коммерческим программным продуктам, в крайнем случае, серверам общедоступного программного обеспечения с надежной репутацией.

### RFC и ссылки по проблематике сетевой безопасности

1. RFC-1108 U.S. Department of *Defense* Security Options for the Internet Protocol, S. Kent, 1991
2. RFC-1244 Site Security *Handbook*, P.Holbrook, J.Reynolds, 1991.
3. RFC-1281 *Guidelines* for the Secure Operations of the Internet, R.Pethia, S.Crocket and B.Frazer, 1991.
4. RFC-1507 DASS - Distributed Authentication *Security Service*, C. Kaufman, 1993
5. RFC-1508 Generic *Security Service* Application Program Interface, J. Linn, 1993
6. RFC-1510 The Kerberos Network *Authentication Service* (V5), J. Kohl, B. Numan, 1993
7. RFC-1535 A *Security Problem* and Proposed Correction With Widely Deployed DNS Software, E. Gavron, 1993
8. RFC-1675 Security Concerns for *Ipng*, S. Bellovin, 1994
9. RFC-1750 Randomness Recommendations for Security, D. Eastlake, S. Crocker, J. Schiller, 1994
10. RFC-1824 The *Exponential* *Security System* TESS: An Identity-Based *Cryptographic Protocol* for Authenticated Key-Exchange, (E.I.S.S.-Report 1995/4), H. Danisch, 1995
11. RFC-1825 *Security Architecture* for the Internet Protocol, R. Atkinson, 1995
12. RFC-1827 IP *Encapsulating Security Payload* (ESP), R. Atkinson, 1995
13. RFC-1847 Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted, J. Galvin, S. Crocker, N. Freed, 199
14. RFC-1858 Security Considerations for IP *Fragment* Filtering, G. Ziemba, D. Reed, P. Traina, 1995
15. RFC-1579 Firewall-Friendly FTP, S. Bellovin, AT&T *Bell Laboratories*, February 1994
16. RFC-2643 Cabletron's SecureFast VLAN Operational Model. D. Ruffen, T. Len, J. Yanacek. August 1999.
17. RFC-2660 The *Secure HyperText Transfer Protocol*. E. Rescorla, A. Schiffman. August 1999.
18. RFC-3007 Secure Domain Name System (DNS) Dynamic Update. B. Wellington. November 2000.
19. RFC-3207 SMTP Service Extension for Secure SMTP over Transport Layer Security. P. Hoffman. February 2002.
20. RFC-3335 MIME-based Secure Peer-to-Peer Business *Data Interchange* over the Internet. T. Harding, R. Drummond, C. Shih. September 2002.
21. RFC-3711 The Secure Real-time *Transport Protocol* (SRTP). M. Baugher, D.McGrew, M. Naslund, E. Carrara, K. Norrman. March 2004.
22. RFC-3767 Securely Available Credentials Protocol. S. Farrell, Ed.. June 2004.
23. RFC-3850 Secure/*Multipurpose* Internet Mail Extensions (S/MIME) Version 3.1 Certificate Handling. B. Ramsdell, Ed.. July 2004.
24. RFC-3851 Secure/*Multipurpose* Internet Mail Extensions (S/MIME) Version 3.1 Message Specification. B. Ramsdell, Ed.. July 2004.
25. RFC-3853 S/MIME Advanced Encryption Standard (AES) Requirement for the Session Initiation Protocol (SIP). J. Peterson. July 2004.
26. RFC-3854 Securing X.400 Content with Secure/*Multipurpose* Internet Mail Extensions (S/MIME). P. Hoffman, C. Bonatti, A. Eggen. July 2004.
27. RFC-3855 Transporting Secure/*Multipurpose* Internet Mail Extensions (S/MIME) Objects in X.400. P. Hoffman, C. Bonatti. July 2004
28. RFC-3971 SEcure Neighbor Discovery (SEND). J. Arkko, Ed., J. Kempf, B.Zill, P. Nikander. March 2005.
29. RFC-4033 DNS Security Introduction and Requirements. R. Arends, R. Austein, M. Larson, D. Massey, S. Rose. March 2005.
30. RFC-4034 Resource Records for the DNS Security Extensions. R. Arends, R.Austein, M. Larson, D. Massey, S. Rose. March 2005.
31. RFC-4035 Protocol *Modifications* for the DNS Security Extensions. R. Arends, R. Austein, M. Larson, D. Massey, S. Rose. March 2005
32. RFC-4130 MIME-Based Secure Peer-to-Peer Business *Data Interchange* Using HTTP, Applicability Statement 2 (AS2). D. Moberg, R. Drummond. July 2005
33. RFC-4162 Addition of *SEED* Cipher Suites to Transport Layer Security (TLS). H.J. Lee, J.H. Yoon, J.I. Lee. August 2005.
34. RFC-4178 The Simple and Protected Generic *Security Service* Application Program Interface (*GSS-API*) *Negotiation* *Mechanism*. L. Zhu, P. Leach, K. Jaganathan, W. Ingersoll. October 2005.
35. RFC-4217 Securing FTP with TLS. P. Ford-Hutchinson. October 2005
36. RFC-4250 The Secure Shell (SSH) Protocol *Assigned Numbers*. S. Lehtinen, C. Lonvick, Ed.. January 2006.
37. RFC-4251 The Secure Shell (SSH) Protocol Architecture. T. Ylonen, C. Lonvick, Ed.. January 2006
38. RFC-4252 The Secure Shell (SSH) *Authentication Protocol*. T. Ylonen, C. Lonvick, Ed.. January 2006
39. RFC-4253 The Secure Shell (SSH) *Transport Layer Protocol*. T. Ylonen, C. Lonvick, Ed.. January 2006
40. RFC-4254 The Secure Shell (SSH) Connection Protocol. T. Ylonen, C. Lonvick, Ed.. January 2006
41. RFC-4255 Using DNS to Securely Publish Secure Shell (SSH) Key *Fingerprints*. J. Schlyter, W. Griffin. January 2006
42. RFC-4256 Generic Message Exchange Authentication for the Secure Shell Protocol (SSH). F. Cusack, M. Forssen. January 2006
43. RFC-4261 Common Open Policy Service (*COPS*) Over Transport Layer Security (TLS). J. Walker, A. Kulkarni, Ed.. December 2005
44. RFC-4262 X.509 Certificate Extension for Secure/*Multipurpose* Internet Mail Extensions (S/MIME) Capabilities. S. Santesson. December 2005
45. RFC-4279 Pre-Shared Key Ciphersuites for Transport Layer Security (TLS). P. Eronen, Ed., H. Tschofenig, Ed.. December 2005
46. RFC-4301 *Security Architecture* for the Internet Protocol. S. Kent, K. *Seo*. December 2005
47. RFC-4303 IP *Encapsulating Security Payload* (ESP). S. Kent. December 2005
48. RFC-4304 Extended *Sequence Number* (*ESN*) Addendum to IPsec Domain of Interpretation (*DOI*) for *Internet Security* Association and Key Management Protocol (*ISAKMP*). S. Kent. December 2005
49. RFC-4305 *Cryptographic Algorithm* *Implementation Requirements* for *Encapsulating Security Payload* (ESP) and Authentication Header (AH). D. Eastlake 3rd. December 2005
50. RFC-4309 Using Advanced Encryption Standard (AES) *CCM* Mode with IPsec *Encapsulating Security Payload* (ESP). R. Housley. December 2005
51. RFC-4310 Domain Name System (DNS) Security Extensions Mapping for the Extensible *Provisioning* Protocol (*EPP*). S. Hollenbeck. December 2005
52. RFC-4344 The Secure Shell (SSH) Transport Layer Encryption Modes. M. Bellare, T. Kohno, C. Namprempre. January 2006

Наиболее полезными серверами *по* проблемам безопасности являются в данный момент серверы: <http://www.linuxberg.com/>, <http://freshmeat.net/>, <http://www.securityfocus.com/infocus/1216>, <http://user-mode-linux.sourceforge.net/>

|  |  |  |
| --- | --- | --- |
| **Название программы** | **Назначение** | **Ссылка** |
| snort | Система обнаружения попытки вторжения | <http://www.clark.net/~roesh/security.html> |
| <http://www.snort.org/docs/idspaper/> |
| <http://www.snort.org/docs/snort-rh7-mysql-ACID-1-5.pdf> |
| SAINT | *Security Administrator*'s Integrated Network Tool – Инструмент *администратора безопасности* | <http://www.wwdsi.com/saint/> |
| PMFirewall | Конфигурирование ipchains в *Firewall* и маскирования IP | <http://www.pointman.org/> |
| OpenSSH | Защищенная оболочка Secure Shell с открытым кодом | <http://www.openssh.com/> |
| Sentiel | Быстрое сканирование системных файлов | <http://zurk.netpedia.net/zfile.html> |
| Secure Shell (ssh) | Средство для защищенных подключений с использованием шифрования и взаимной авторизации | <ftp://ftp.cs.hut.fi/pub/ssh/> |
| <http://www.ssh.fi/> |
| Ripper | Взломщик паролей | <http://www.openwall.com/john/> |
| Honeyd | Ловушка для хакеров | <http://www.citi.umich.edu/u/provos/honeyd/> |

Информацию об атаках и различных средствах противодействия читатель может найти *по* адресу:

<http://www.cert.org/reports/dsit_workshop-final.html>

Существует много средств для детектирования *DDoS* (*Distributed* denial-of-service). Каждое из этих средств имеет свои слабости, и ни одно не является универсальным решением. Некоторые из них можно найти:

* <http://www.fbi.gov/nipc/trinoo.htm>
* <http://staff.washington.edu/dittrich/misc/stacheldraht.analysis>
* <http://www.iss.net/cgi-bin/dbt-display.exe/db_data/press_rel/release/122899199.plt>
* <http://www.sans.org/y2k/stacheldraht.htm>

*Архив* средств для детектирования *DDoS* атак можно найти *по* адресу

<http://packetstorm.securify.com/distributed/>

CERT/CC публикует справочные материалы и другие документы *по* данной проблематике, например:

* <http://www.cert.org/advisories/CA-2000-01.html>
* <http://www.cert.org/advisories/CA-99-17-denial-of-service-tools.html>
* <http://www.cert.org/tech_tips>