Министерство образования и науки Украины
Донецкий Национальный Технический Университет
Кафедра ПМИ

Автореферат

По теме: «Обнаружения уязвимостей и предотвращение атак на экономические системы»

Руководитель:
доцент, к.т.н. Губенко Н.Е.

Выполнил:
студент группы ЭКИ-99а
Гошко Станислав

Донецк 2003

На сегодняшний день операционные системы производства фирмы Microsoft завоёвывают всё большую и большую популярность. На основе решений на базе ОС: Windows NT, Windows 2000, Windows XP, Windows 2003, строятся многие экономические системы. Данные системы считаются относительно стабильными и зачастую используются как файловые и web сервера. Использование данных систем в качестве серверных систем ставит вопрос их безопасности на первое место. Даже использование данных систем как пользовательских не снимает требования к их безопасности. Глобализация и гипер рост сети Интернет так же накладывает свой отпечаток на развитие операционных систем и их служб. На данный момент заметно значительное улучшение дел в области компьютерной безопасности. Корпорация Microsoft уже приобрела несколько более мелких фирм занимающихся компьютерной безопасностью.

Но надеяться на одни лишь решения данной не безызвестной фирмы по крайней мере было бы глупо. Необходимо заниматься компьютерной безопасностью, как в рамках отдельно взятой фирмы, так и в рамках конечного пользователя.

1. Проблема атак на отказ в обслуживании в экономческих системах.

Данная проблема существует в мире компьютерной безопасности очень давно. Сильное же распространение в Windows системах она получила с 1997 года.

1.1 Что такое DoS атаки

Чтобы понять инциденты, описанные ниже, будет полезно отступить немного назад и рассмотреть более простую форму этой атаки – отказ в обслуживании. Отказ в обслуживании, или DoS - самая базовая категория атак в сфере компьютерной безопасности, которая может использоваться в нескольких вариантах. Этот термин может быть применен к любой ситуации, в которой атакующий пытается помешать использованию кем-либо какого-либо ресурса. Это может быть реализовано различными методами, физическими и виртуальными. Например, атакующий может перекрыть доступ к телефонной системе путем перерезания главного телефонного кабеля, идущего к зданию, непрерывных звонков по всем свободным телефонным линиям, или взломав мини-АТС. Во всех трех случаях, атакующий достигает цели, закрывая доступ пользователей к ресурсу, т. к. становится невозможно произвести ни входящие, ни исходящие звонки.

Концепции DoS легко применимы к миру сетей. Маршрутизаторы и серверы могут обрабатывать ограниченный объем трафика в любой момент времени, в зависимости от таких факторов, как характеристика оборудования, количество памяти и полоса пропускания. Если этот предел превышается, новый запрос будет отвергнут. В результате, будет проигнорирован легитимный трафик, и пользователи получат отказ в доступе. Таким образом, атакующий, который хочет нарушить работу определенного сервиса или устройства, может сделать это, просто забросав цель пакетами, которые поглотят все доступные ресурсы.

DoS не является обычным взломом, в котором целью атакующего является получение неавторизованного доступа, но может оказаться столь же зловредным. Цель DoS – нарушение работы и причинение неудобств. Успех измеряется в продолжительности хаоса. Примененные против ключевых целей, таких как корневые DNS сервера, эти атаки могут быть серьезной угрозой. Угроза DoS атак зачастую стоит на первом месте при обсуждении концепций информационной войны. Их легко осуществить, трудно остановить, и они очень эффективны.

Существуют две разновидности атак на отказ в обслуживании: это локальная и удалённая. Локальные атаки данного вида встречаются довольно редко и как пример можно рассмотреть такую атаку.

Создаётся папка, после этого создаётся ярлык на эту папку и этот ярлык помещается в эту же папку. Затем пытаемся войти в папку через созданный нами ярлык, это влечёт колоссальное пожирание мощностей компьютера вплоть до его повисания.

Теперь перейдём к удалённым атакам на отказ в обслуживании, так как существует достаточно большое количество, я думаю, нам стоит совершить экскурс в прошлое, чтобы разобраться с основами данной проблемы.

## 1.2 Атака WinNuke

Начнем, конечно же, с классического и широко известного WinNuke, появившегося 7 мая 1997 года. Автор метода поместил его описание и исходный текст программы в несколько news-конференций. Ввиду крайней простоты метода практически каждый мог вооружиться новейшим оружием и пойти крушить направо и налево.

Первой жертвой стал сервер www.microsoft.com. Он прекратил откликаться в пятницу вечером (9 мая) и только к обеду понедельника вновь обрел устойчивость.

Конечно же, наряду с жертвой номер раз в мае 97-го пали многие серверы, на которых красовалась гордая надпись "Windows NT Powered", а также серверы и без оной, но работающие под Windows NT. К чести Microsoft следует заметить, что заплатки появились довольно быстро.

Итак, что же такое WinNuke? Наряду с обычными данными стандарт предусматривает пересылку по TCP-соединению и срочных (Out Of Band, OOB) служебных данных. На уровне форматов пакетов TCP это выражается в ненулевом значении соответствующего поля (urgent pointer). Большинство компьютеров, работающих под Windows, используют сетевой протокол NetBIOS, нуждающийся в трех IP-портах: 137, 138, 139. Как выяснилось, если соединиться с Windows-машиной через 139-й порт и послать туда несколько байт OOB-данных, то NetBIOS, не зная, что делать с этими данными, попросту подвешивает или перезагружает машину. В Windows 95 это обычно выглядит как синий текстовый экран с сообщением об ошибке в драйвере TCP/IP, и работа с сетью становится невозможной до перезагрузки ОС. NT 4.0 без service pack'ов перезагружается. NT 4.0 лишь со вторым service pack'ом выпадает в синий экран, сообщая о необработанном исключении в коде ядра (эту картинку нередко называют blue screen of death).

Судя по информации в Сети, такой атаке подвержены и Windows NT 3.51, и Windows 3.11 for Workgroups, и WinFrame, в основе которого лежит Windows NT 3.51.

Тут к месту упомянуть довольно забавную историю. Как выяснилось вскоре после выпуска SP3 (service pack 3 для Windows NT 4.0), WinNuke, запущенный с компьютеров Macintosh, легко "пробивал" защиту service pack'a. Причиной послужило существование двух разных стандартов на IP-пакеты, содержащие OOB-данные, - стандарта от Berkley и стандарта, описанного в RFC. Они по-разному вычисляют Urgent Pointer, и результат вычислений отличается ровно на единицу. Третий service pack, защищающий от "своих" OOB-пакетов, оказался беззащитен против пакетов другого стандарта. Поэтому почти сразу после SP3 вышел дополнительный OOB-fix.

Следует заметить, что если для написания оригинального WinNuke достаточно самых тривиальных функций работы с TCP/IP (программа на PERL занимает семь строк), то, чтобы "пробить" SP3, потребуется работать с TCP на низком уровне либо запускать стандартный WinNuke с платформы, поддерживающей другую реализацию OOB. Само существование OOB-данных, безотносительно WinNuke, вызывает немало проблем именно из-за существования двух стандартов, или, вернее, отсутствия стандарта. Поэтому гарантировать корректную работу программы, использующей OOB, не может никто (правильнее сказать, легко можно гарантировать ее некорректную работу).

Умные люди рекомендуют вообще не использовать OOB-данные в своих программах, и многие так и поступают. Таким образом, OOB сегодня - это наличие потенциальных дырок, позволяющих злоумышленникам изобретать все новые и новые методы атак. Попросту "постреляв" OOB-данными по открытым TCP-портам, можно попытаться найти новую дырку в какой-либо программе. Некоторым, говорят, удается...

## 1.3 Атака SPing

Еще одна ошибка обнаружилась буквально через месяц после нашумевшего дебюта WinNuke. Объектом атаки на сей раз стал протокол ICMP, точнее, его реализация. Этот протокол издавна привлекал любителей сетевых диверсий. Поскольку ICMP представляет собой внутренний механизм поддержки работоспособности IP-сетей, то с точки зрения злоумышленника он является очевидным объектом атаки. Как пример можно упомянуть ping с большим размером пакета. Поскольку ICMP-пакеты имеют определенные привилегии в обработке, то ping большого размера может парализовать работу компьютера или даже целой сети, IP-каналы которых будут передавать только ICMP-пакеты.

Такой способ часто используется людьми, имеющими достаточно мощные каналы связи, против тех, кто адекватных каналов не имеет. Сей метод, основанный на грубой силе, очевидно, не требует большого ума (точнее сказать, не требует никакого ума и, видимо, поэтому так любим молодыми сотрудниками провайдерских организаций). Защиты от него в общем случае не существует, самый адекватный метод реакции - установив адрес злоумышленника, написать жалобы куда только можно.

Но мы опять отвлеклись от основной темы. Объектом нашего исследования является известная ошибка, называемая SPing (Jolt, SSPing, IceNuke, IcePing, IceNuke, Ping Of Death...). Множество названий вовсе не означает множества различных модификаций. Просто разные люди называют одну идею по-разному, и встретить программу можно под всеми вышеперечисленными именами. Как выяснилось, Windows-системы неадекватно реагируют на получение сильно фрагментированного ICMP-пакета (кусочками до одного килобайта) большого размера (чуть больше 64 килобайт). Реакция заключается в безоговорочном повисании, включая мышь и клавиатуру. В конце июня 1997 года жертвой SPing пал сервер Microsoft, после чего его закрыли каким-то хитрым брандмауэром (думается, информация о типе и настройках этого брандмауэра относится к важнейшим секретам Microsoft).

В отличие от WinNuke, жертвами SPing могут стать не только Windows-системы, но и Mac OS и некоторые версии Unix (заплатки для них, к счастью, уже имеются). Официальная заплатка от Microsoft для NT 4.0 с установленным SP3 (далее везде по тексту будет подразумеваться, что NT 4.0 имеет установленный SP3).

Заметим, что SPing намного серьезнее, нежели WinNuke, поскольку использует ICMP-пакеты, которые чаще всего не фильтруются брандмауэром, а если и фильтруются, то подобного рода защиту можно попытаться преодолеть, используя приемы спуфинга.

## 1.4 Атака Land

Следующий метод атаки, называемый land, замечателен в первую очередь огромным числом поражаемых систем. Кроме Windows NT, этой напасти подвержены и Mac OS, и множество вариантов Unix (от бесплатных до коммерческих), и такие экзотические системы, как QNX и BeOS, и даже многие аппаратные маршрутизаторы (в том числе от Cisco и 3Com). Практически для всех систем уже имеются исправления, хотя, конечно же, установили их далеко не все владельцы компьютеров.

Что же такое land? При инициации TCP-соединения посылается пакет с установленным флагом SYN. Нормальной реакцией на получение SYN-пакета является подготовка ресурсов для нового соединения, посылка SYN-ACK-пакета (пакета подтверждения) и ожидание реакции с другой стороны. Если в течение определенного времени ответа не последует, SYN-ACK-пакет передается повторно несколько раз, как правило, со все большей задержкой.

Очевидным методом атаки, использующим вышеописанный механизм, является классический SYN-Flood, заключающийся в следующем: на компьютер-жертву посылается множество SYN-пакетов с искаженными адресами отправителя. Компьютер-жертва тратит массу вычислительных ресурсов, пытаясь подтвердить соединения с абсолютно ничего не подозревающими или даже с несуществующими компьютерами. При достаточно большом количестве фальшивых запросов ресурсы компьютера-жертвы могут исчерпаться, и все другие процессы будут остановлены либо аварийно завершены (другой вариант: будут сброшены все имеющиеся соединения).

Но это очень старый метод, основанный на грубой силе. Механизм работы land хитрее. Посылается SYN-пакет с адресом отправителя, совпадающим с адресом получателя, жертвы. Пакет посылается на любой открытый порт. Для Windows-систем это почти всегда может быть порт 139 (наш старый знакомый по WinNuke). Для других систем это может быть любой известный порт (21, 80 и др.). Реакцией Windows-компьютера на land является абсолютное повисание.

Заплатка для Windows NT 4.0 доступна уже давно.

## 1.5 Атака Teardrop

Найденный в том же несчастливом 1997 году метод атаки Teardrop основан на ошибках в реализации TCP/IP-стека. Атаке подвержены Windows-системы, а также компьютеры с Linux. Заплатка от Microsoft для Windows NT лежит по адресу ftp://ftp.microsoft.com/bussys/winnt/winnt-public/fixes/usa/NT40/hotfixes-postSP3/simptcp-fix. Строго говоря, эта заплатка, появившаяся довольно быстро, предназначалась для отражения другой атаки (которая заключается в посылке большого количества UDP-пакетов с искаженным адресом отправителя на 19-й порт, что приводит к повышенному UDP-трафику).

Фурора, подобного тому, который вызвал WinNuke, на сей раз не было. Возможно, пользователи уже свыклись с тем, что новые методы DoS-атак появляются регулярно. Тем не менее, Teardrop замечателен тем, что стал первым из семейства подобных (об этом будет рассказано ниже). Он прекрасно иллюстрирует тот факт, что любые программы, даже насчитывающие несколько строк, содержат ошибки. (Если вы уверены, что программа безошибочна, значит, вы просто не заметили ошибки, которая после обнаружения будет казаться очевидной.)

Совершим небольшой экскурс в тайны реализации TCP/IP-стека. Передача данных в TCP/IP-сетях осуществляется не с помощью неких идеальных носителей информации, а по вполне реальным каналам (часто отвратительного качества), и в сам стандарте заложено, что передаваемый пакет может разбиваться на несколько меньших пакетов, а принимающая сторона возвращает ему первоначальный вид. Точнее, более высокоуровневые, чем IP, протоколы TCP и UDP могут передаваться фрагментировано на уровне IP. Каждый фрагмент характеризуется смещением от начала исходного пакета и своей длиной. Драйвер TCP/IP-стека собирает фрагменты на принимающей стороне до тех пор, пока не получит их все (или, во всяком случае, пока не решит, что принял все).

Безусловно, при передаче возможны различные ситуации, которые умная реализация TCP/IP должна распознавать. В частности, может быть, что несколько полученных фрагментов будут пересекаться. Нас интересует ситуация, когда новый фрагмент имеет смещение, лежащее внутри уже полученного фрагмента.

Что же делает TCP/IP-стек в этом случае? Во-первых, вычисляется размер пересечения: смещение старого фрагмента плюс длина старого фрагмента есть смещение нового фрагмента. А затем в буфер копируется только та часть нового фрагмента, которая "выступает за границу" старого фрагмента. Все просто и очевидно. Однако возможна ситуация, когда новый фрагмент не только начинается внутри старого, но и заканчивается в нем же.

По идее, такой фрагмент должен быть просто пропущен, но как раз этого программисты, писавшие Windows NT и Linux, и не предусмотрели. Что же происходит в этом случае? Пусть у нас есть уже полученный фрагмент A, со смещением A_offs и длиной A_len. Пришел новый фрагмент B, со смещением B_offs и длиной B_len. Причем A_offs < B_offs < B_offs + B_len < A_offs + A_len, то есть фрагмент B лежит внутри фрагмента A.

Проследим действия принимающей стороны по шагам. Начало нового фрагмента лежит внутри имеющегося. Пересекающаяся часть имеет смещение A_offs + A_len - B_offs. А тот кусочек, что нужно добавить в буфер, начинается с A_offs + A_len и имеет длину (B_offs + B_len) - (A_offs + A_len).

Длина-то меньше нуля или (если вспомнить о машинной зацикленной арифметике) является очень большим числом. И вот такого большого размера блок памяти будет копироваться, уничтожая при этом все встречающееся на пути (обычно под "горячую руку" попадает операционная система). Собственно, вышеописанное и есть Teardrop. Приведенное объяснение абсолютно верно для Linux (поскольку ее исходный код открыт), но, вероятно, аналогичный механизм делает потенциальной жертвой Teardrop-атаки и Windows NT.

Вслед за появлением метода Teardrop возникло несколько его модификаций, которые были способны пробивать Windows NT с установленной против обычного Teardrop заплаткой. Известность получили Bonk (Boink), NewTear, SynDrop. Все эти атаки закрываются еще одной заплаткой: ftp://ftp.microsoft.com/bussys/winnt/winnt-public/fixes/usa/nt40/hotfixes-postSP3/teardrop2-fix.

Наиболее оригинальным среди них является SynDrop. По сути, он представляет собой оригинальный Teardrop с тем отличием, что в посылаемых фрагментах устанавливается флаг SYN (снова этот излюбленный флаг).

Последней атакой на отказ в обслуживании является RPC DoS.

Уязвимость обнаружена в части RPC протокола, который используется для обмена сообщениями через TCP/IP. Удаленный атакующий может аварийно завершить работу всех RPC служб.

Как сообщается, уязвимость связанна с неправильной обработкой некорректных сообщений в RPC Endpoint Mapper процессе, который слушает на 135 порту. В результате удаленный пользователь может нарушить работу этой службы, что приведет к нарушению работы всех RPC служб на уязвимой системе.

Уязвимость обнаружена в Windows 2000 и Windows XP. Windows NT 4.0 уязвима к этой проблеме, но ее архитектура не позволяет выпустить патч, устраняющий эту уязвимость.

Microsoft оценила риск обнаруженной уязвимости как “ Important”

Как мы можем заметить атаки данного вида получили громадное распространение.

## 2. Проблема несанкционированного доступа в операционные системы семейства Microsoft Windows NT/2000/XP/2003

Данная проблема не специфична для конкретного семейства операционных систем, она существует во всех операционных системах. Но мы решили остановиться на данном семействе операционных систем по той причине, что операционные системы семейства Windows завоевали огромную популярность в последние годы не только у обычных пользователей, но и зачастую используются как серверные системы. Если операционные системы данного семейства используются, как серверные, то пренебрежение таким существенным вопросом  как безопасность чревато огромными потерями.

Давайте посмотрим на основные уязвимости систем Microsoft. Итак, что мы можем видеть, летом 2003 года наблюдался чрезвычайный всплеск хакерской активности, основными атаками были атаки на переполнение буфера. Самым распространённым червём последнего времени считается “Lovesan“, который размножается через переполнение буфера в RPC службе систем обсуждаемого нами семейства.

### 2.1 Сетевая атака червя Worm.Win32.Lovesan

Вирус-червь. Распространяется по глобальным сетям, используя для своего размножения уязвимость в службе DCOM RPC Microsoft Windows

Червь написан на языке C, с использованием компилятора LCC. Имеет размер 6KB, упакован UPX. Размножается в виде файла с именем "msblast.exe".

Содержит текстовые строки:

“I just want to say LOVE YOU SAN!!
billy gates why do you make this possible ? Stop making money and fix your software!!”

Признаками заражения компьютера являются:

- Наличие файла "msblast.exe" в системном (system32) каталоге Windows.

- Сообщение об ошибке (RPC service failing) приводящее к перезагрузке системы.

Размножение

При запуске червь регистрирует себя в ключе автозапуска:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
windows auto update="msblast.exe"

Червь сканирует IP-адреса, начинающиеся с "base address" и пытается соединиться с 20 IP-адресами для инфицирования уязвимых компьютеров.

После этого червь "спит" в течение 1,8 секунды, а затем снова сканирует 20 IP-адресов и повторяет этот процесс в бесконечном цикле. Например, если "base address" является 20.40.50.0, червь будет сканировать следующие адреса:

20.40.50.0
20.40.50.1
20.40.50.2
...
20.40.50.19
---------- пауза 1.8 секунды
20.40.50.20
...
20.40.50.39
---------- пауза 1.8 секунды
...
...
20.40.51.0
20.40.51.1
...
Червь выбирает один из двух методов сканирования IP-адресов:

В 3 случаях из 5 червь выбирает случайный "base address" (A.B.C.D), где D равен 0, а A, B, C случайно выбраны из диапазона 1-255. Таким образом "base address" находится в диапазоне [1-255].[1-255].[1-255].0.

В 2 случаях из 5 червь сканирует подсеть.

Он определяет адрес локального компьютера (A.B.C.D), устанавливает D в ноль и выбирает значение C. Если C - больше чем 20, то червь выбирает случайное число от 1 до 20. Если C меньше или равно 20, червь не изменяет его.

Например, если инфицированная машина имеет IP-адрес "207.46.134.191", то червь будет сканировать адреса с 207.46.[115-134].0. Если IP-адрес - "207.46.14.1", то червь будет сканировать адреса, начиная с 207.46.14.0. Используя уязвимость в Microsoft Windows 2000/XP, отправляет на 135 порт выбранного IP-адреса команды эксплоита DCOM RPC и запускает на удаленной машине командную оболочку "cmd.exe" на TCP порту 4444.

После этого червь, при помощи команды tftp get, через 69 порт загружает себя на удаленную машину в системный каталог Windows и запускает на исполнение.

После заражения инфицированная машина выводит сообщение об ошибке RPC service failing, после чего перезагружается.

С 16 августа 2003 года червь запускает процедуры DDoS атаки на сервер windowsupdate.com, пытаясь таким образом затруднить или прервать его работу.

До описанного выше червя была ужасная по своим масштабам эпидемия червя «Helkern».

## 2.2 Сетевая атака червя Worm.SQL.Helkern

Интернет-червь, заражающий сервера работающие под Microsoft SQL Server 2000. Распространяется от компьютера к компьютеру пересылая на очередной (заражаемый) компьютер, через порт 1434, свой код и запуская этот код на выполнение путём использования ошибки в программном обеспечении MS
SQL (см. ниже).

Червь имеет крайне небольшой размер - всего 376 байт.

Червь присутствует только в памяти зараженных компьютеров и не создаёт своих копий в дисковых файлах. Более того, при работе червя никакие файлы не создаются, и червь никак не проявляет себя (помимо сетевой активности зараженного компьютера).

При активизации на заражаемом компьютере червь получает адреса трёх функций Windows:

GetTickCount (KERNEL32.DLL)
socket, sendto (WS2_32.DLL)

Затем червь в бесконечном цикле посылает свой код (командой "sendto") на случайно выбранные адреса в сети (при этом использует случайные данные от команды "GetTickCount").

Поскольку SQL-сервера часто используются в качестве стандартной базы данных на Web-серверах, то данный червь может замедлить работу Интернета в глобальных масштабах, поскольку все зараженные сервера в бесконечном цикле посылают пакеты на случайно выбранные адреса в сети - и, следовательно, сильно увеличивают сетевой трафик.

В коде червя видны строки:

```
h.dllhel32hkernQhounthickChGet
Qh32.dhws2_f
etQhsockf
toQhsend
```

Реализация атаки

Для реализации атаки на сервера используется одна из ошибок в защите IIS типа:

Remote Buffer Overrun Vulnerability

Название конкретной применяемой атаки:

Unauthenticated Remote Compromise in MS SQL Server 2000

Данная ошибка была обнаружена в июле 2002 года и исправлена последующими патчами к MS SQL Server 2000.

Подробное описание уязвимости можно найти на сайте Microsoft: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS02-039.asp
(Microsoft Security Bulletin MS02-039)

Перед ним был Интернет червь “Code Red”, который распространялся через переполнение буфера в веб сервере IIS(Internet Information Services).

2.3 Сетевая атака червя IIS-Worm.CodeRed (aka Code Red, Bady)

Первый представитель данного семейства сетевых червей, "Code Red" (также известный под именем "Bady"), по данным ZDNet, уже заразил около 12 000 серверов по всему миру и провел крупномасштабную DDoS атаку на Web сервер Белого дома (www.whitehouse.gov), вызвав нарушение его нормальной работы.

"Code Red" заражает только компьютеры под управлением Windows 2000 (без установленных сервисных пакетов), с установленным Microsoft Internet Information Server (IIS) и включенной службой индексирования (Indexing Service). Вместе с тем, именно это программное обеспечение чаще всего используется на коммерческих Web, FTP и почтовых серверах, что и определило широкое распространение червя. Масштабы эпидемии могли бы быть еще более внушительными, если бы червь поражал и другие версии Windows, например Windows NT и Windows XP. Однако автор червя умышленно "нацелил" его только на системы Windows 2000.

Для проникновения на удаленные компьютеры червь использует обнаруженную в июне 2001 г. брешь в системе безопасности IIS, которая позволяет злоумышленникам запускать на удаленных серверах посторонний программный код. Для этого "Code Red" посылает на случайно выбранный удаленный сервер специальный запрос дающий компьютеру команду запустить основную программу червя, которая в свою очередь попытается таким же образом проникнуть на другие серверы. Одновременно в памяти компьютера может существовать сразу сотни активных процессов червя, что существенно замедляет работу сервера.

18 июня 2001 г. Microsoft выпустила заплатку, устраняющую данную брешь, однако подавляющее большинство пользователей еще не успело обновить свое программное обеспечение.

Важной особенностью "Code Red" является то, что в процессе работы он не использует никаких временных или постоянных файлов. Данный червь уникален: он существует либо в системной памяти зараженных компьютеров, либо в виде TCP/IP-пакета при пересылке на удаленные машины. Подобная "бестелесность" представляет серьезную проблему для защиты серверов, поскольку требует установки специальных антивирусных модулей на межсетевые экраны.

Помимо значительного замедления работы зараженных компьютеров, "Code Red" имеет другие побочные действия. Во-первых, червь перехватывает обращения посетителей к Web сайту, который управляется зараженным IIS-сервером, и вместо оригинального содержимого передает им следующую страницу:

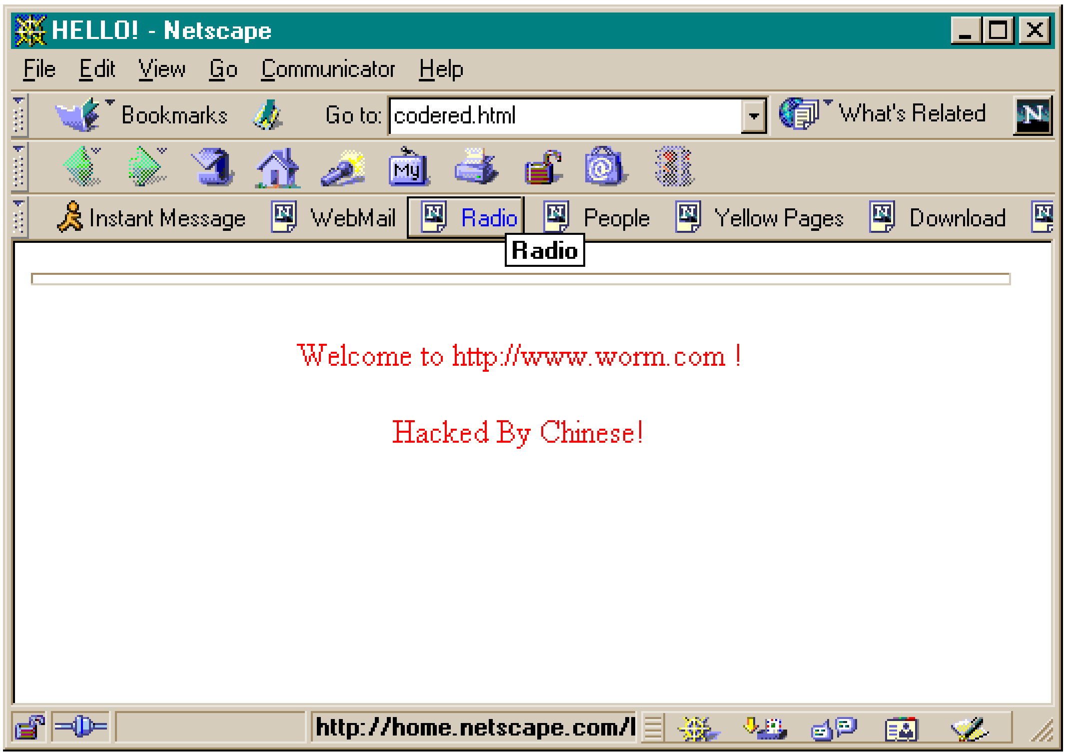


Рис. 2.3.1 Фальсифицированная web страница

После показа фальсифицированной стартовой страницы взломанного Web сайта в течение 10 часов, червь автоматически возвращает все на свои места и посетители видят оригинальную версию сайта. Важно отметить, что данный эффект проявляется только на системах, где по умолчанию используется язык "US English".

Во-вторых: между 20 и 27 числами каждого месяца включительно червь осуществляет DDoS (Distributed Denial of Service) атаку на сайт Белого дома США (www.whitehouse.gov). Для этого копии червя на всех зараженных компьютерах посылают многочисленные запросы на соединение, что вызывает зависание сервера, обслуживающего данный Web-сайт.

Как вы могли заметить, подавляющее большинство атак на Windows системы в основном вызвано атаками типа – переполнение буфера.

Мы рассматривали в основном примеры крупномасштабных вторжений, которые были реализованы Интернет червями, на самом деле количество различных уязвимостей в операционных системах семейства Windows NT/2000/XP/2003, крайне велико.

Почему же так получается, что и в системах семейства Windows, и в операционных системах семейства *nix, самой распространённой атакой является атака на переполнение буфера? Ответ здесь кроется глубоко внутри этих операционных систем. Данные операционные системы писались на языке программирования “C” с ассемблерными вставками. Так как язык “C” не проверяет корректность размеров буфера источника и буфера приёмника, то из-за этого и возникают такие ошибки. Чаще всего память для буфера приёмника в конкретных функциях выделяется в стеке. А как вам известно, в стеке хранится адрес возврата функции, и хакеры научились подменять как адрес возврата так и сам буфер таким образом, чтобы самим получать управление и выполнять произвольный код на данной системе.

Переполнение буфера (buffer overflows) - название самой распространенной уязвимости в области безопасности программного обеспечения. Первая атака с применением данной уязвимости использовалась в вирусе-черве Морриса в 1988 году. С тех пор их число увеличивается с каждым годом. В настоящее время можно говорить, что уязвимости, связанные с переполнение буфера являются доминирующими при удаленных атаках, где обычный пользователь сети получает частичный или полный контроль над атакуемым хостом. Анализ атак и обнаруженных уязвимостей последних лет показывает, что данная проблема является первостепенной. Так, например, 9 из 13 выпусков CERT (Computer Emergency Response Team site) в 1998 году и по крайней мере половина выпусков 1999 года связаны с переполнением буфера. Информационный обзор популярного списка рассылки Bugtraq показывает, что примерно 2/3 респондентов считает переполнение буфера основной причиной нарушения сетевой безопасности. Отметим, что переполнение буфера присуще также программному обеспечению ряда аппаратных средств. Примером может служить уязвимость принтера HP LaserJet 4500. Очевидно, что эффективное решение данной проблемы позволит исключить большую долю самых серьезных угроз компьютерной безопасности.

Основа атак с использованием этой уязвимости - принцип функционирования операционных систем, где программа получает привилегии и права запустившего ее пользователя или процесса. Таким образом, менее привилегированный пользователь или процесс, который взаимодействует с данной программой может использовать ее права в своих целях. Штатные средства программного обеспечения не позволяют выполнять такие действия. Однако “переполнение буфера” все же делает это возможным. Использование данной уязвимости подразумевает изменение хода выполнения привилегированной программы, например, запуск командной оболочки с правами администратора. Реализации атаки требует решения двух подзадач.

- Подготовка кода, который будет выполнятся в контексте привилегированной программы.

- Изменение последовательности выполнения программы с передачей управления подготовленному коду.

Рассмотрим пути решения подзадачи подготовки кода.

- Подготавливаемый код представляет собой исполняемый машинные инструкции соответствующего процессора и может передаваться в программу в качестве ее параметра или команды. При этом параметр или команда сохраняется программой в отведенном для этого буфере. Буфер может находится в любой области памяти: в стеке (локальные, автоматические переменные), в динамически распределяемой памяти, в области статических данных. Например, программе, запрашивающей ввод строки, под видом строки может быть передан нужный атакующему код, которому в дальнейшем будет передано управление.

- Нужный код не передается в программу, так как он уже присутствует в ней самой или в ее адресном пространстве и требуется лишь его параметризация. Например, подготовка параметра для функции запуска программы. В данном случае атакующему требуется изменить или сформировать нужный параметр (для нашего примера указатель на строку с названием программы), а не сам код. Параметр также может находится в любой области памяти.

- Если параметризированный код уже присутствует в адресном пространстве программы, то подготовки кода не требуется.

Далее рассмотрим способы передачи управления подготовленному коду. В основе этих способов лежит переполнение буфера, т. е. блока памяти, выделенного под переменную. Переполнение возникает при отсутствии проверки выхода за границы буфера. Таким образом, искажается содержимое других переменных состояния и параметров программы, которые входят в область переполнения буфера. Типы искажаемых объектов-переменных определяет способ передачи управления коду атакующего, и могут быть следующими.

### Искажение адреса возврата из функции

Так как вызову функции сопутствует занесение адреса возврата в стек, то при его подмене атакующим, управление передается по заданному им адресу. Здесь используется переполнение буфера локальных переменных функции, которые также создаются в стеке. Простым примером служит следующий фрагмент программы на Си.

```
int namelen (void) {

char name[21];

gets(name);

return strlen(name);

}
```

Из примера видно, что при вводе имени размером более 20 символов частью строки будет замещен адрес возврата из функции. Далее, при выполнении инструкции возврата из подпрограммы, управление будет передано по адресу, который образуют соответствующие позиции введенной строки и в обычной ситуации будет получено сообщение об ошибке операционной системы.

Такие атаки на переполнение буфера получили название “атаки срыва стека” (stack smashing attack).

### Искажение указателя функции

В данном случае атаке подвергаются переменные, содержащие указатели на функции. Эти переменные могут располагаться в любой области памяти, не только в стеке но и в области динамически и статически выделяемых данных. Атакующий организовывает переполнение буфера, которое искажает данные указатели, и далее при вызове функций по этим указателям управление передается подготовленному коду. Приведем пример уязвимости указателей в виде следующего фрагмента программы на Си.

```
void dummy(void) {

printf("Hello world!\n");

}

int main(int argc, char **argv) {
```

```
void (*dummyptr)();

char buffer[100];

dummyptr=dummy;

strcpy(buffer, argv[1]); // Уязвимость

(*dummyptr)();

}
```

Здесь переполнение буфера buffer приводит к подмене указателя dummyptr и последующему изменению хода выполнения программы.

## Искажение таблиц переходов

В результате компиляции часто создаются так называемые таблицы переходов, которые могут использоваться в целях оптимизации, динамического связывания кода, обработки исключений и т.п. Искажение таких таблиц вызванное переполнением буфера позволяет передать управление подготовленному коду.

## Искажение указателей данных

Данный способ не предусматривает явной передачи управления подготовленному коду, но предполагает, что передача происходит на основании оригинального алгоритма программы. Другими словами, подготовленный код запускается в ходе обычной, не искаженной, последовательности исполнения программы.

Рассмотрим следующий фрагмент программы на C.

```
foo(char * arg) {

char * p = arg; // уязвимый указатель

char a[40]; // переполняемый буфер

gets(a); // применение gets() реализует уязвимость
```

```
gets(p); // искажение кода

}
```

Здесь переполнение буфера а вызывает подмену указателя p и последующую запись строки по адресу искаженного указателя. Вводимая строка содержит код атакующего. Такая схема атаки часто используется для корректировки (patch) части кода программы или кода динамических и статических библиотек, располагающихся в памяти по фиксированным адресам. Например, корректировка-подмена системных функции выхода из программы или запуска процесса.

Другой пример атаки подобного рода - искажение указателя кадра стека локальных переменных (frame pointer overwrite attack). Эта атака основана на стандартных операциях пролога и эпилога подпрограмм, в результате чего подменяется указатель базы кадра локальных переменных.

На мой взгляд, комбинация всех методов подготовки кода и целей переполнения буфера (типа искажаемых структур) определяет виды всех возможных атак по переполнению буфера, что позволяет их классифицировать. Результат комбинации приведен в следующей таблице.

Таблица 2.3.1. Классификация атак по переполнению буфера

| Подготовка кода<br><br>Цель переполнения | Внедрение кода | Внедрение параметров | Не требуется |
|---|---|---|---|
| Искажение адреса возврата из функции | Атака “срыв стека” | Атака “срыв стека” с параметризацией | Атака “срыв стека” с передачей управления |
| Искажение указателей функций | Атака на указатели функций | Атака на указатели функций с параметризацией | Атака на указатели функций с передачей управления |
| Искажение таблиц переходов | Атака на таблицы переходов | Атака на таблицы переходов с параметризацией | Атака на таблицы переходов с передачей управления |
| Искажение указателей данных | Атака с искажением указателей данных | Атака с искажением указателей данных с параметризацией | Атака с искажением указателей данных с оригинальным кодом |

3. Возможности решения данных проблем

Вот некоторые варианты защиты от атак на отказ в обслуживании. Для небольших и средних компаний, имеющих локальные сети с каналом в Интернет, можно посоветовать использовать межсетевые экраны. Это, как правило, отдельно стоящий компьютер или маршутизатор, который пропускает через себя трафик из нескольких, строго определенных, сетей, причем, естественно, контролируемый трафик должен физически (!) проходить через этот экран. Сетевая ОС на этом экране (Firewall) осуществляет фильтрацию трафика с последующей его логической обработкой. Например, можно настроить Firewall таким образом, что из защищенной сети можно будет заходить в Интернет, тогда как "из внешнего мира" подключиться к какой-нибудь рабочей станции будет невозможно, несмотря на доступность этой сети из Интернета. Firewall будет "отсекать" попытки подсоединиться к рабочей станции (распространенный вид DDoS когда взломщик подключается к 139 порту Windows и вызывает зависание этой ОС) или, к примеру, можно лимитировать канал на тот или иной трафик, например, если сделать лимит на ICMP трафик не более 50 кбит/с, то забить тот или иной канал уже намного сложнее. Такие системы должны настраивать специалисты, т.к. существует много тонкостей в протоколах TCP/IP.

Конечным пользователям можно посоветовать системы защиты индивидуального характера, такого рода программы есть и для Windows. Но применять его советую лишь в случаях, когда нет возможности поставить отдельно Firewall (как правило, это пользователь, подключившийся по dial-up), поскольку в отличие от сетевых ОС, которые изначально ориентированы на работу с трафиком, это лишь своего рода "надстройка" над Windows, и, естественно, невозможно так же надежно контролировать трафик. Хотя это все же лучше чем "лезть" в Интернет с "открытой" Windows.

Выше были приведены рекомендации по защите от специалиста Интернет агентства «РуСтар».

Я же хочу дополнить его и поправить одной установки Firewall недостаточно, потому что если данный программный комплекс не будет грамотно настроен, то он сам может послужить источником потенциальной уязвимости. Данному утверждению существует множество подтверждений в рассылках по компьютерной безопасности.

Мной предлагаются два варианта защиты отдельных компьютеров:

- Защита средствами самой Windows
- Защита средствами независимых разработчиков

Оба варианта имеют свои как положительные, так и отрицательные стороны.

Теперь необходимо сосредоточиться на проблеме защиты от атак, имеющих целью получение несанкционированного доступа.

В основном экспертами по компьютерной безопасности в качестве защиты от вторжений рекомендуются следующие методики:

- Тонкая настройка операционной системы

- Установка всех системных обновлений

Мной предлагаются следующие дополнительные варианты решения данной проблемы:

- Анализ настроенной системы сканером безопасности

- Разработка программы для анализа локальных программ на атаки типа переполнение буфера

- Анализ локальных приложений и сервисов.

# ВЫВОДЫ

В ходе данной работы мы смогли увидеть и понять опасность удалённых и локальных атак на операционные системы семейства Microsoft Windows NT/2000/XP/2003. Так же мы выяснили, что готовых решений данных проблем не так уж и много. Существующие же решения крайне дорогостоящие и практически не доступны малым и даже некоторым средним фирмам.

Что позволяет нам предложить несколько новых и не дорогостоящих решений для анализа компьютерной безопасности операционных систем Microsoft Windows NT/2000/XP/2003.

Список используемых источников:

1. Максим Степин. DoS-атаки на WINDOWS

2. Андрей Колищак. Атаки на переполнение буфера

3. Стюарт Макклуре, Джоел Скембрэй, Джордж Куртц. Секреты хакеров. Проблемы и решения сетевой защиты. – М.: Лори, 2001.

4. Стюарт Макклуре, Джоел Скембрэй. Секреты хакеров. Безопасность Windows 2000 – готовые решения. - М.: Лори, 2002.

5. Fred B. Schneider, Steven M. Bellovin, Martha Branstad, J.Randall Catoe, StephenD. Crocker, Charlie Kaufman, Stephen T. Kent, John C. Knight, Steven McGeady, Ruth R. Nelson, Allan M. Schiffman, George A. Spix, and Doug Tygar. Trust in Cyberspace. National Acadamy Press, 1999. Committee on Information Systems Trustworthiness, National Research Council.

6. Steve Bellovin. Buffer Overflows and Remote Root Exploits. Personal Communications, октябрь 1999.

7. Crispin Cowan, Perry Wagle, Calton Pu, Steve Beattie, and Jonathan Walpole. Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade., Department of Computer Science and Engineering, Oregon Graduate Institute of Science & Technology, ноябрь 1999.

8. "Aleph One". Smashing The Stack For Fun And Profit. Phrack, 7(49), ноябрь 1996.
9. "Mudge". How to Write Buffer Overflows. http://l0pht.com/advisories/bufero.html, 1997.
10. Nathan P. Smith. Stack Smashing vulnerabilities in the UNIX Operating System. http://millcomm.com/nate/machines/security/stack-smashing/nate-buffer.ps, 1997.
11. Вадим Колонцов. Переполнение буфера. WDL magazine., http://www.tversu.ac.ru/wdl/articles/overflows.html, 1996.
12. klog. The Frame Pointer Overwrite. Phrack Magazine, выпуск 55, сентябрь 1999.
13. M.Bishop. How to Write a Setuid Program. :login;,12(1), январь/февраль http://olympus.cs.ucdavis.edu/bishop/scriv/index.html, 1986.
14. Anon. Linux Security Audit Project. http://lsap.org/.
15. AnupK Ghosh, Tom O’Connor, and Gary McGraw. An Automated Approach for Identifying Potential Vulnerabilities in Software. In Proceedings of theIEEE Symposium on Security and Privacy, Oakland, CA, май 1998.
16. "Solar Designer". Non-Executable User Stack. http://www.false.com/security/linux-stack/.
17. Casper Dik. "Non-Executable Stack for Solaris". Posting to comp.security.unix,

18. Compaq. ccc C Compiler for Linux. http://www.unix.digital.com/linux/compaq_c/ , 1999.
19. Richard Jones and Paul Kelly. Bounds Checking for C. http://www-ala.doc.ic.ac.uk/phjk/BoundsChecking.html, июль 1995.
20. Reed Hastings and Bob Joyce. "Purify: Fast Detection of Memory Leaks and Access Errors". Winter USENIX Conference, январь 1992.
21. Synthetix: Tools for Adapting Systems Software. http://www.cse.ogi.edu/DISC/projects/synthetix.
22. Crispin Cowan, Calton Pu, Dave Maier, Heather Hinton, Peat Bakke, Steve Beattie, Aaron Grier, Perry Wagle, and Qian Zhang. StackGuard: Automatic Adaptive Detection and Prevention of Buffer-Overflow Attacks. 7-я конференция USENIX Security Conference, San Antonio, TX, январь 1998.
23. Crispin Cowan, Steve Beattie, RyanFinnin Day, Calton Pu, Perry Wagle, and Erik Walthinsen. Protecting Systems from Stack Smashing Attacks with StackGuard. Linux Expo, Raleigh, NC, май 1999.
24. Tobias Haustein. Buffer overflow in HP JetDirect module, рассылка Bugtraq, 19 ноября 1999.
25. Alexander Snarskii. FreeBSD Stack Integrity Patch ftp://ftp.lucky.net/pub/unix/local/libc-letter, 1997.
26. Drew Dean, Edward W. Felten, and Dan S. Wallach. Java Security: From HotJava to Netscape and Beyond. IEEE Symposium on Security and Privacy, Oakland, CA, 1996, http://www.cs.princeton.edu/sip/pub/secure96.html.
27. "Vendicator". Stack Shield: A "stack smashing" technique protection tool for Linux. октябрь 1999, http://www.angelfire.com/sk/stackshield.