GIAC CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Security Essentials: Network, Endpoint, and Cloud (Security 401)"
at http://www.giac.org/registration/gsec

Buffer Overrun Vulnerabilities in Microsoft Programs: Do You Really Need to Apply All of the Security Patches?

Submitted on September 21, 2003

by

Ed Rietscha

in partial fulfillment of the requirements for

GIAC Security Essentials Certification

Option 1, Version 1.4b

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

ABSTRACT

Microsoft ® has released 39 Security Bulletins so far this year. Over half of them address buffer overrun vulnerabilities. Each of the buffer overrun Security Bulletins recommends that administrators download a patch and install it onto all of their vulnerable computers. For sites with hundreds or thousands of affected systems, applying a security patch is a huge undertaking. Administrators may be tempted to ignore patches or wait until a convenient time to apply them.

This paper provides information about buffer overrun vulnerabilities in Microsoft programs and explores the risks associated with not applying security patches as Microsoft recommends for these vulnerabilities. It explains what buffer overruns are and tells why they exist. It describes how these vulnerabilities are discovered, and lists the Microsoft programs that have them. It then explains in detail how much damage a hacker can cause by overrunning a buffer and shows the methods that hackers use against Microsoft's overrun vulnerabilities. The final section recommends that administrators apply all security patches that affect their systems in the timeframe that Microsoft recommends, and it describes other methods they should use to protect their systems.

INTRODUCTION

On September 10, 2003, Microsoft released Security Bulletin MS03-039[1] to warn users that three more vulnerabilities had been found in Windows ® software. Two of these were buffer overrun vulnerabilities. If exploited, they could allow a hacker to cause such serious damage to Windows computers that Microsoft assigned the Bulletin a "Critical" security rating. The Bulletin recommended that system administrators immediately download a new security patch and install it on every computer running one of the affected operating systems. For companies with Microsoft server and client operating systems, this equated to patching almost all of their computers.

MS03-039 was released less than two months after a similar Security Bulletin on a nearly identical vulnerability.[2] The earlier Bulletin, released on July 16th and revised on August 25th, also recommended that administrators immediately download a security patch and install it on all vulnerable systems.

It seems that a new Microsoft buffer overrun vulnerability is discovered every couple of weeks. A review of the 39 Security Bulletins that Microsoft has released so far this year[3] reveals that this assessment is actually very accurate: over half of them – including the most recent four - address buffer overruns.

For sites with hundreds or thousands of affected computers, applying a security patch is a huge undertaking. IT departments that apply patches manually often have to choose among three poor options when they're notified that their systems have a vulnerability: dedicate extra resources to quickly secure the systems; run vulnerable systems until all patches can be applied; or disconnect systems from the network until

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

they can be patched. Even for companies with the best patch management software available today, updating a large number of systems is time consuming and usually requires manually applying patches to a percentage of the systems.

It took the large corporation that I work for weeks to patch all systems against the buffer overrun vulnerability that Microsoft announced in July. It'll take us about the same length of time to install Microsoft's latest patch. My corporation's Computer Emergency Response Team has directed that all sites update all of their computers in a short period of time or disconnect them from the corporate network – no exceptions. Other corporations, faced with installing a patch on systems that they just patched less than two months ago, may be wondering if it's worth the effort.

Are buffer overruns a big enough threat to justify all of the work required to install all of Microsoft's security patches? If a site has a firewall and applies the mitigating measures mentioned in the Security Bulletins, aren't these enough? Or, why can't companies just wait until two or three Security Bulletins have been released – it usually doesn't take more than a couple of months – and apply all patches at once?

This paper discusses the importance of installing security patches for buffer overrun vulnerabilities in Microsoft programs and provides answers to the following questions that personnel unfamiliar with overrun vulnerabilities may be asking:

1. What are buffer overruns?

2. Why do they exist?

3. How are they discovered?

4. Which current Microsoft programs have these vulnerabilities?

5. How much damage could a hacker cause by exploiting one of these overruns?

6. How can hackers exploit the overrun vulnerabilities that have been discovered this year?

7. What do system administrators need to do to protect their systems against these vulnerabilities?

Although this paper concentrates on overruns in Microsoft programs, these vulnerabilities exist in other programs, such as Oracle and Linux, also.[4, 5] Most of the background information provided applies to buffer overflow vulnerabilities in these other programs. I have limited my discussion of individual Microsoft overrun vulnerabilities to ones that have been discovered this year, because my research showed that Microsoft has already included patches for most of the overruns discovered last year in Service Packs for the affected software. Unfortunately, this is not true for most of this year's

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

vulnerabilities. The administrator or user must apply a separate patch for each – or accept the risk associated with not applying the patch.

WHAT ARE BUFFER OVERRUNS?

A buffer is a block of contiguous memory locations that a program has reserved to hold data. An overrun occurs if a program allows a user to enter more data into a buffer than it is designed to hold. For example, if a user enters 101 bytes of data into a buffer that's only 100 bytes in size, the extra byte of data is going to overrun the buffer.

In an article on its website, Microsoft describes three types of buffer overruns:

> 1. Static buffer overruns - A static buffer overrun occurs when a buffer, which has been declared on the stack, is written to with more data than it was allocated to hold. The less apparent versions of this error occur when unverified user input data is copied directly to a static variable, causing potential stack corruption.
>
> 2. Heap overruns - Heap overruns, like static buffer overruns, can lead to memory and stack corruption. Because heap overruns occur in heap memory rather than on the stack, some people consider them to be less able to cause serious problems; nevertheless, heap overruns require real programming care and are just as able to allow system risks as static buffer overruns.
>
> 3. Array indexing errors - Array indexing errors also are a source of memory overruns. Careful bounds checking and index management will help prevent this type of memory overrun. [6]

Most other websites refer to these vulnerabilities as buffer overflows, instead of overruns, and explain stack- versus heap-based overruns a bit differently than Microsoft. In addition to an area where the program's compiled code resides, a program may use two other storage areas in memory. In general terms, a stack is used for temporary storage of data during program execution; parameters, local variables (to include buffers) and other data is "pushed" onto the stack when functions are called and "popped" off when the functions terminate. A heap, on the other hand, holds data that is more static, such as global variables and static variables.[7]

Most Microsoft buffer overrun vulnerabilities are stack-based. The amount of damage that such an overrun can cause depends on two things: what kind of data the user enters into the buffer, and how the program is using the memory locations immediately after the buffer. If the user overruns a buffer with random data, the likely effect is to cause the program to crash.

Although having a program crash is a bad thing, overruns can cause much worse damage than this. In a more serious stack-based overrun, a hacker can overwrite memory locations that hold a value known as a function return address.[8] When a function is called during program execution, the system pushes the address where it

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

was called from (the function return address) onto the stack. Once the function exits, the system pops this address from the stack and returns the program to what it was doing before the function was called.[9] By writing data past the end of a buffer, a hacker may be able to overwrite the function return address with an address of his choice, thereby causing a program to execute any code in memory that he chooses.

Given an opportunity to modify both the data in a buffer and the value of a function's return address, a hacker could cause serious damage. He could fill the buffer with malicious code and then direct the program to start executing this code upon returning from a function. If a buffer is too small to hold all of the malicious code that the hacker wants to execute, he could load a small program into the buffer that downloads and executes a larger program from the Internet.

WHY DO THEY EXIST?

Most buffer overrun vulnerabilities are the result of poor practices by programmers working with arrays in C and C++. These languages have a number of built-in functions that allow more data to be entered into an array than it's designed to hold. Programmers should avoid using these functions, or they should write their code to check for invalid input.

Microsoft's website says the following about this issue: "Preventing buffer overruns is primarily about writing good code. Always validate all your inputs and fail gracefully when necessary."[6] Microsoft recommends that developers who want to avoid buffer overruns in their programs read two books published by Microsoft Press on developing "solid" and "secure" code.

It sounds easy: all it takes to avoid overrun vulnerabilities is to validate inputs or fail gracefully. Microsoft applied these security principles and many more during the development of its newest server operating system, Windows Server™ 2003. According to a BusinessWeek article written by Alex Salkever:

> For the 2003 version, Microsoft threw even more resources into making the code safe. It halted all programming work for 10 weeks during the winter of 2002 to teach its engineers more about how to build secure software. It used in-house hackers to try to tear apart Win 2003. And it built innovative automated software to check for common types of security flaws.[10]

Despite all of this, Windows Server 2003 was among the operating systems affected by the most recent overrun vulnerabilities. Haven't Microsoft's programmers read the Microsoft Press books on developing solid, secure code? Don't they know how to write "good" code?

Actually, Microsoft's programmers write excellent code. The problem is, there's just so much of it. Windows Server 2003 has over 50 million lines of source code! This is 10 million more lines than Windows XP, and 15 million more than Windows 2000. This

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

code is divided into thousands of files in thousands of directories. With programs this large, even the best programmers can overlook mistakes.

HOW ARE THEY DISCOVERED?

Microsoft overrun vulnerabilities are discovered in a number of ways. Many are reported to Microsoft by companies like Next Generation Security Software, Ltd., which specialize in computer system security. Others are identified and made public by groups such as Last Stage of Delirium Research Group or The Hackademy. For vulnerabilities discovered this way, Microsoft patches are generally available for download before hackers are able to launch large-scale attacks.

Some of the organizations that have discovered Microsoft overrun vulnerabilities did so by looking through the source code for the affected programs. Through its Enterprise Source Licensing Program (ESLP), Microsoft provides its enterprise customers access to Windows 2000, Windows XP, and Windows Server 2003 operating system source code at no cost. Microsoft's ESLP web page says that any organization that "meets the specified criteria—and signs the source licensing agreements" may access Windows source code.[11] In addition to contributing to improvements in future versions of Windows, ESLP provides Microsoft with a pool of trusted individuals to look for bugs in current versions.

A number of companies provide training on working with Windows source code. For example, Azius offers both introductory and advanced classes. An advertisement for its Windows Source Code Workshop says it includes a "guided tour" of the "thousands of directories and tens of millions of lines of code," and instruction on "building the operating system from the source … and debugging with source code visible in the debugger." [12]

Unfortunately, it's also possible to find overrun vulnerabilities in programs without access to the source code. The method that hackers use is generally referred to as disassembling code. Thousands of websites provide instruction and tools for doing this. The main tools required are a disassembler and a debugger. A good anti-hacking website describes these tools as follows:

> Disassemblers or decompilers can translate an application's code back into Assembler, no matter in which language the application was originally written. … Once an application is translated it's easy for the cracker (if he knows the particular language) to find sections of interest to him and to determine how they work. … The best decompilers even comment on the translated code, which makes the code that much easier to understand.
>
> Debuggers allow crackers to trace an application, instruction by instruction, and to stop it at any point and trace its important sections. [13]

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Most hacking web sites recommend IDA Pro disassember (http://www.datarescue.com/idabase/ida.htm) and SoftIce debugger (http://www.compuware.com/products/devpartner/softice.htm). One hacker, who calls himself Barnaby Jack, provides the following endorsement of IDA Pro: "without a doubt, THE tool for reversing code. … If I were to cover everything this tool can do I would be here all day, and I'd still be missing something. With the combined effort of IDA and [a good debugger], there are no barriers."[14]

WHICH CURRENT MICROSOFT PROGRAMS HAVE THESE VULNERABILITIES?

Buffer overrun vulnerabilities have been discovered in all currently supported Microsoft operating systems – Windows 98 through Windows Server 2003. In addition, buffer overruns have been found this year in the following Microsoft applications:

1. Internet Explorer versions 5.01 through 6.0 (Microsoft Security Bulletins MS03-015 and 020).

2. BizTalk Server 2002, an "Enterprise Integration product … used in intranet environments to transfer business documents between different back-end systems" (MS03-016).

3. DirectX versions 5.2 through 9.0a (MS03-030).

4. Data Access Components versions 2.5 through 2.7 (MS03-033).

5. Office 97, 2000 and XP; Word 98; FrontPage 2000 and 2002; Publisher 2000 and 2002; and Works Suite 2001, 2002 and 2003 (MS03-036).

6. Visual Basic for Applications SDK 5.0, 6.0, 6.2 and 6.3 (MS03-037).

7. Access 97, 2000 and 2002 (MS03-038).

HOW MUCH DAMAGE COULD A HACKER CAUSE BY EXPLOITING ONE OF THESE OVERRUNS?

How much damage could a hacker do if he exploits one of the Microsoft overrun vulnerabilities? Microsoft's most recent Security Bulletin provides an eye-opening answer to this question: "An attacker who successfully exploited either of the buffer overrun vulnerabilities could gain complete control over a remote computer. This would give the attacker the ability to take any action that they wanted on the system, including changing Web pages, reformatting the hard disk or adding new users to the local administrators group."[1] In addition, a hacker could view, add, delete or modify files on the system; install programs; or change security settings.

In addition to these latest vulnerabilities, seven other overrun vulnerabilities discovered this year could also allow hackers to take complete control of a system.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

These are described in Security Bulletins MS03-001, MS03-005, MS03-006, MS03-007, MS03-013, MS03-024 and MS03-026.

Nine of the other vulnerabilities could allow a hacker to take actions in the security context of a user on a system. These are described in Security Bulletins MS03-008, MS03-015, MS03-020, MS03-023, MS03-027, MS03-030, MS03-36, MS03-37 and MS03-38. The amount of damage that a hacker could do would depend on the privileges that the user account has on the system. If the account has administrator privileges, the hacker could control the system.

MS03-016, MS03-019 and MS03-022 address vulnerabilities in IIS. Exploiting these overruns could allow a hacker to run code in the security context of the IIS service. By default, the new version of IIS runs as a user account.

The MDAC vulnerability explained in MS03-033 is a special case: A hacker who exploited this flaw could gain "the same level of privileges over the system as the application which initiated the broadcast request. The actions an attacker could carry out on the system would be dependent on the permissions that the application using MDAC ran under."

HOW CAN HACKERS EXPLOIT THE OVERRUN VULNERABILITIES THAT HAVE BEEN DISCOVERED THIS YEAR?

It has been estimated that once an overrun vulnerability is found, it can be exploited about 90% of the time. So, how do hackers attack these vulnerabilities? Do they need physical access to a vulnerable computer to gain control over it? Can they exploit a vulnerability by tricking a user into opening an infected e-mail attachment, visiting a web page, or clicking on a link on a web page? Or, can a hacker remotely exploit a vulnerable computer without the assistance of any user on the computer?

A review of the twenty-one Security Bulletins that Microsoft has released this year on buffer overruns reveals that all of the methods mentioned above – and more – can be used to exploit these vulnerabilities. Hackers can exploit seven of the overruns from across the Internet without the aid of any users on a vulnerable system. Even an unskilled hacker can attack these vulnerabilities. All information required to launch an attack can easily be gathered or guessed. These vulnerabilities are described in the Security Bulletins listed below, and they can be exploited by sending the indicated types of malformed requests across the network:

1. MS03-001 – an RPC call that would employ the Locator service to resolve a logical name.

2. MS03-007 – a WebDAV requests to a web server running IIS 5.0.

3. MS03-016 – a request to the HTTP receiver.

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

4. MS03-019 – a request to the server performing logging.

5. MS03-022 – a request to the server running Windows Media Services.

6. MS03-026 – a TCP request to an RPC interface.

7. MS03-039 – a request to the RPCSS service.

An eighth vulnerability can also be exploited from the Internet, but according to Microsoft Security Bulletin MS03-024, a hacker would need to know a valid user name and password on a vulnerable server in order to attack it.

Seven of the other overrun vulnerabilities - described in MS03-006, MS03-008, MS03-015, MS03-020, MS03-023, MS03-030 and MS03-038 - can be exploited through the use of malicious web pages. These pages can be hosted on web sites or e-mailed to users as HTML mail. Depending on the version of Outlook or Outlook Express used, attacks can be launched as soon as a user opens the e-mail or views it in a preview pane.

The MDAC overrun described in MS03-033 can be exploited without any user assistance, but a hacker has to be on the same network as the vulnerable system in order to attack it. The hacker has to detect a specific type of request from a vulnerable system and respond to it with a malicious packet.

To exploit the Windows shell overrun described in MS03-027, a hacker has to create a Desktop.ini with a corrupt attribute and lure a user to the network or Internet share where this file resides. Similarly, a user would have to be tricked into opening a document to be attacked by the MS03-036 vulnerability, or into opening a document or sending an e-mail message for MS03-037.

To exploit either of the two remaining vulnerabilities, an attacker would have to have physical access to a vulnerable computer and be able to log on to it. Once logged on, the attacker could run a program that provides incorrect parameter information to the Windows Redirector component of Windows XP (MS03-005) or a program that sends incorrect debugger messages to the Windows kernel (MS03-013).

WHAT DO SYSTEM ADMINISTRATORS AND USERS NEED TO DO TO PROTECT THEIR SYSTEMS AGAINST THESE VULNERABILITIES?

Overrun vulnerabilities in Microsoft programs are a serious threat. Each of Microsoft's Security Bulletins includes a severity rating and a recommendation regarding how quickly the appropriate security patch should be applied. Administrators need to follow Microsoft's recommendations. They need to read all of the Security Bulletins and install the patches that apply to their systems within the recommended time frames.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Microsoft's Security Bulletins have links to download locations for the security patches. Most of the patches released in 2002 and a few of the ones released in 2003 have already been included in Service Packs (SPs) for the vulnerable programs; administrators do not need to install these patches if they have installed the SPs.

In addition to installing security patches and SPs, administrators should use the following measures to limit damage from buffer overrun exploits:

1. A properly configured firewall can help to protect against many external threats, including the ones described in MS03-001, MS03-024 and MS03-026.

2. Services and settings that are not needed should be turned off. Many of these are enabled by default in different versions of software. The following services and features should be disabled to prevent the attacks indicated:

   a. MS03-001 – Locator service.

   b. MS03-007 – WebDAV or IIS.

   c. MS03-019 and MS03-022 – Windows Media Services.

3. A hacker has to be logged onto a system in order to exploit some of the vulnerabilities. Organizations should practice physical security to prevent unauthorized users from accessing the systems.

4. Antivirus software should be kept up-to-date. Hackers use viruses or worms to attack many of the vulnerabilities. Corporations should use a different antivirus package on servers and workstations, and they should block executable attachments to e-mail messages. Users should be trained not to open attachments from unknown senders. They should only open attachments from known sources if they're expecting something or if they've verified the content of the attachment.

5. Administrators and users should backup data frequently to prevent loss or corruption.

6. WindowsUpdate should be used to automatically detect required patches and SPs for newer versions of Windows software.

7. Web servers should run Microsoft's UrlScan tool, which prevents some buffer overruns. PC Magazine recently ran an excellent article written by Larry Seltzer on this tool. It summarizes URLScan's function as follows:

> UrlScan is a DLL that silently monitors requests to your Microsoft IIS Web server, blocking and logging those that violate the following rules set by administrators:
>
> > Extremely long URLs. By default, UrlScan blocks URLs longer than 16K.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

> File extensions. UrlScan can block requests for certain types of files. For example, it blocks all EXE file requests by default.

> HTTP verbs. You can set UrlScan to allow or deny specific HTTP commands, such as *OPTIONS*.

> Directory traversal. UrlScan can block attempts to traverse the directory tree with "./" and "../".

> URL sequences. UrlScan can block certain characters with malicious potential, such as the use of ":" to access alternative file streams. [15]

URLScan is part of the IIS Lockdown Tool, which turns off unnecessary IIS features. The Lockdown Tool is available for free download at the following location: http://www.microsoft.com/downloads/release.asp?ReleaseID=43955.

8. Networks need to be scanned for "back doors," such as modems or wireless LAN access points, which bypass the firewall. "War dialer" software can be used to look for modems. Portable scanners are now available for around $30 at most computer stores to scan for wireless access points.

9. To limit the damage from exploits that run in the security context of a user, limit user account privileges to the minimum necessary to complete assigned tasks.

10. To limit the damage from attacks that run in the security context of an exploited application, ensure that applications run under user accounts unless they need administrator privileges.

11. Limit the hours that authorized users can log on, and track users' actions while on line.

12. To protect against malicious web pages, restrict users from visiting web sites that are not required to accomplish their jobs.

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

List of References

1. Microsoft, "Microsoft Security Bulletin MS03-039." 10 September 2003. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS03-039.asp (10 Sep. 2003).

2. Microsoft, "Microsoft Security Bulletin MS03-026." 10 September 2003. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS03-026.asp (10 Sep. 2003).

3. Microsoft, "Security Bulletins." 10 September 2003. URL: http://www.microsoft.com/security/security_bulletins (10 Sep. 2003).

4. U.S. Department of Energy, Computer Incident Advisory Capability, "Information Bulletin N-128: Oracle Buffer Overflow in E-Business Suite [Oracle Security Alert 56]." 25 July 2003. URL: http://www.ciac.org/ciac/bulletins/n-128.shtml (21 Sep. 2003).

5. "Sandeep Grover" (posted by the Rookery), "Buffer Overflow Attacks and Their Countermeasures." URL: http://www.home.linuxjournal.com/article.php?sid=6701 (21 Sep. 2003).

6. Microsoft, "Avoiding Buffer Overruns." URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/security/security/avoiding_buffer_overruns.asp (21 Sep. 2003).

7. University of Hawaii, "Stack vs Heap Allocation." URL: http://www-ee.eng.hawaii.edu/Courses/EE150/Book/chap14/subsection2.1.1.8.html (21 Sep. 2003).

8. Litchfield, David, "Buffer Overflow for Beginners." URL: http://www.wbglinks.net/pages/reads/bofs/beginner.html (21 Sep. 2003).

9. Eleventh Alliance, "Buffer Overflows: Simply Explained." URL: http://www.wbglinks.net/pages/reads/bofs/bof6.html (21 Sep. 2003).

10. Salkever, Alex, "For Windows, Less Fat Means Fewer Bugs." 29 April 2003. URL: http://www.businessweek.com/technology/content/apr2003/tc20030429_6540_tc047.htm (19 Sep. 2003).

11. Microsoft, "Enterprise Source Licensing Program." URL: http://www.microsoft.com/resources/sharedsource/Licensing/Enterprise.mspx (15 Sep. 2003).

12. Azius, "WSC110: Windows Source Code Workshop." URL: http://www.azius.com/site/index.cgi?page=msc110 (10 Sep. 2003).

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

13. Unknown, “Tips and Tricks.” URL: http://www.anticracking.sk/coding.html (12 Sep. 2003).

14. “dark spyrit” (AKA Barnaby Jack), “Win32 Buffer Overflows (Location, Exploitation and Prevention.” URL: http://www.wbglinks.net/pages/reads/bofs/w32bofs.html (27 Aug. 2003).

15. Seltzer, Larry. “Blocking Malicious URLs.” PC Magazine. Volume 22, Number 13 (August 5, 2003): 76.

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.