Global Information Assurance Certification Paper

Secure Windows Initiative Trial by Fire:
IIS 5.0 Printer ISAPI Buffer Overflow

Corey Pincock
GSEC 1.2e

Introduction

Microsoft's commitment

On April 10, 2001 at the 10th annual RSA Security Conference, David Thompson, vice president for the Windows product server group, announced that there has been a company wide effort and focus on improving the security of Microsoft's products, from the top of the organization down.

To illustrate this statement Thompson highlighted the Microsoft Windows 2000 operating system and its built in security features, an improved security response process, the company's Safe Internet consumer security and privacy Web site, and the SafeNet 2000 security and privacy summit which was hosted by Microsoft. Microsoft has even gone so far as to make the Windows source code available to a select group of universities for testing and validation.

Thompson credited these new efforts aimed at improving security to the Secure Windows Initiative. According to Microsoft, the Secure Windows Initiative (SWI) helps to expand the security knowledge of Microsoft's engineers and developers while encouraging them to constantly look for ways to improve the security of the company's products. At that same RSA conference in April, Scott Culp, Security Program Manager at Microsoft reiterated that Microsoft was strengthening its commitment to security and "Recognize now that every piece of software has vulnerabilities and bugs, and we have to deal with it." SWI is Microsoft's attempt to "deal with it."

Another significant component of Microsoft's renewed security effort is its Security Services Partner Program, which has now grown to 50 companies. The Security Services Program gives security providers a direct connection to the Microsoft Security Response Center. This direct connection means that providers can receive immediate notification of issues from the center or contact it to help identify risk issues and better assess the extent of risk for their clients. George Kurtz, the CEO of Foundstone, a computer-security services provider, says, "It's a convincing demonstration that Microsoft understands the importance of security to the growth of e-commerce and all other aspects of technology. It's also a demonstration of how important security is to Microsoft as a key driver of its continued growth. This sends a very important message to the rest of the industry, helping to raise awareness of security issues with customers of all sizes -- all of whom need comprehensive and up-to-date security solutions."

Despite these convincing moves, Microsoft's commitment to security has come under fire recently with the discovery of several new vulnerabilities. One of these was the discovery of the IIS 5.0 Printer ISAPI buffer overflow. The announcement of this critical vulnerability quickly grabbed the attention of security professionals. One is now left to wonder how effective is SWI and is Microsoft really serious about security. Is SWI another marketing effort or a true demonstration that Microsoft understands the importance of security?

The Exploit

The discovery

According to the advisory posted to Bugtraq by eEye Digital Security on May 1, 2001, while Riley Hassell, of eEye Digital Security, was updating the vulnerability assessment tool called Retina to check some of the new features of Windows 2000 for unknown vulnerabilities he made a startling discovery. Retina was able to discover a buffer overflow vulnerability by using some proprietary technology called CHAM (Common Hacking Attack Methods). Eeye claims that when you turn on the CHAM functionality with Retina it performs a basic vulnerability scan and then in phase two it uses the information it gathers to discover unknown vulnerabilities. By pre-selecting specific protocols in Retina's policies menu (FTP, POP3, SMTP, HTTP), Retina will attempt various hacker exploits. The attacks include overflows, format string attacks, path attacks, munged byte attacks, among others.

One of the features that were added by Hassell to be audited by CHAM was the printer ISAPI filter extension. ISAPI is an application-programming interface for the Internet Services in Windows 2000. ISAPI allows web developers to develop custom code that provides additional web services. This custom code can either be implemented in an ISAPI filter, if the new functionality provides a low-level service, or conversely an ISAPI extension, if the new functionality provides a high-level service. In this case, the targeted code was an ISAPI extension. The vulnerability assessment tool proceeded to send a large amount of data into the ISAPI extension in order to attempt a buffer overflow.

According to eEye, within a matter of minutes, a debugger kicked in on inetinfo.exe because of a "buffer overflow error." The overflow error indicated to Hassell that this particular ISAPI extension could be vulnerable to a buffer overflow attack. After the discovery, Ryan Permeh of eEye Digital Security was called in to try and exploit the vulnerability. Subsequently, Permeh created an example exploit to be used as a "proof-of-concept" which when run against an IIS 5 Web server would perform a system level task of the attacker's choosing. The exploit worked and as coded, created a text document on the remote server with instructions directing readers to a Web page on eEye.com with information on how to patch the system. At this point Microsoft was notified and provided with a working exploit. Shortly thereafter came the announcement that possibly every default installation of Windows 2000 is vulnerable to a buffer overflow attack giving a remote attacker system-wide access. An announcement like this gets the attention of the information security profession pretty quickly

Details

The affected ISAPI extension is one that implements the Internet Printing Protocol (IPP). IPP provides a way to request printing services and learn the status of print jobs across the Internet via HTTP. For example, a traveling sales person could use IPP to send a print job across the Internet to be printed on a printer at the corporate network. He also could find out whether the print request had completed without error. The capability to use IPP is enabled by default in Windows 2000.

The Windows 2000 Internet printing ISAPI extension contains msw3prt.dll that handles user requests. Due to an unchecked buffer in a section of code of msw3prt.dll that handles input parameters, a maliciously crafted HTTP .print request containing approx 420 bytes in the 'Host:' field could enable an attacker to overflow the buffer with the code of their choice.

Typically a web server would stop responding in a buffer overflow condition; however, once Windows 2000 detects an unresponsive web server it automatically performs a restart of the Internet services by the system. The restart then allows the malicious code to run at system level context with administrative rights and permissions. This attack would give an attacker full control of the exploited server.

For this attack to be possible, a Windows 2000 server needs to be available on port 80 (HTTP) or port 443 (HTTPS) and have the mapping for the Internet Printing ISAPI extension, which is the default.

The aftermath

On May 1, 2001 shortly after eEye provided Microsoft with a working proof of concept. Microsoft announced Microsoft Security Bulletin MS01-023 “Unchecked Buffer in ISAPI Extension Could Enable Compromise of IIS 5.0 Server,” acknowledging the vulnerability and providing a patch to be applied immediately. Meanwhile, Microsoft engaged their security partner channel and informed them of the vulnerability. The security partners then informed customers in key sectors of the critical security hole. With the notification, Microsoft's Security Services Partner Program effectively went into action.

As a result of this huge vulnerability that seriously affects nearly every default installation of Windows 2000, Microsoft decided to hold Service Pack 2 until it can integrate the patch with the update. "The update was in the can, and we delayed it because this fix has to go in," said Scott Culp the Security Program Manager at Microsoft.

A day later, CERT put out an official advisory (CERT Advisory CA-2001-10 Buffer Overflow Vulnerability in Microsoft IIS 5.0) advising administrators to address and patch the vulnerability immediately. Meanwhile the Common Vulnerabilities and Exposures (CVE) group assigned the identifier CAN-2001-0241.

After the announcement of the exploit and a posting of the proof of concept code by

eEye, additional code was written to exploit the vulnerability. Dark Spyrit posted to Bugtraq, code that he named jill.c. According to the posting, the code would give a remote attacker a "remote command shell, reverse telnet style". A day later, portings to Perl and Window's binaries began appearing, allowing a greater audience the ability to exploit the critical vulnerability. The sample exploit code could easily give a script-kiddie the ability to gain true administrative access to a default installation of Windows 2000 Server even behind a firewall.

What is a buffer overflow

History

The buffer overflow is common to all computer architectures and operating systems. It is perhaps the most misunderstood and difficult exploit, yet the most dangerous. Of the fifty-four CERT advisories published between 1999 and the first quarter of 2001, ten were related to buffer overflow issues. To summarize, 19% of published CERT advisories during the period dealt with buffer overflows.

Despite the large number of buffer overflows in general, buffer overflows are just beginning to be understood and exploited on the Win32 platform. During the period from 1999 to 2001, out of the ten CERT advisories, only one in 2001 and one in 1999 were directly exploitable on the Win32 platform (This does not include Microsoft applications or third party findings of buffer overflows).

Arguably, one of the first buffer overflow attacks that was seen as successful was Robert Morris's Internet Worm. In 1988, when the 23-year-old Cornell graduate student let his program (worm) loose, it succeeded in crippling 5 to 10 percent of the 60,000 hosts then connected to the Internet. One of the methods Morris used to gain access to a vulnerable system was a buffer overflow bug in the *fingerd* daemon. Once it gained access to a vulnerable system, Morris's program installed itself on the machine, and used several methods to attempt to spread itself to other machines. Supposedly, Morris did not intend the code to be so devastating but due to some programming errors it propagated quicker and with greater impact on the host. Fortunately, this use of a buffer overflow was extreme and not many other attempts at buffer overflow's followed.

However in 1995, Mudge of the L0pht wrote an article called, "How to Write Buffer Overflows." Shortly after this, in a 1996 publication of the online hacker magazine Phrack, appeared an article called "Smashing the Stack for Fun and Profit," by Aleph One. Taking off where Mudge left off, Aleph One explains in detail how to write a buffer overflow exploit against a Unix system. As a result of this diffusion of knowledge, in 1997 and 1998, buffer overflow exploits became routine against open source Unix systems.

A paper published by the Oregon Graduate Institute of Science & Technology[1] and

[1] http://www.cse.ogi.edu

funded in part by the Defense Advanced Research Projects Agency said that,[2] "Buffer overflows have been the most common form of security vulnerability for the past 10 years." Microsoft also has noted that between two-thirds and three-quarters of computer security problems are buffer overrun issues.[3]

Technical details

Buffer overflows are not always malicious and can creep into your daily routine as nothing more than an irritation. At a basic level a buffer overflow will cause your application to crash. However, the presence of a buffer overflow indicates that that the application may susceptible to an exploitation which could allow a remote attacker to run malicious code. Luckily, not all buffer errors can be forced to run malicious code at the end of the overflow.

Before we get into the technical details of a buffer overflow attack, here are a few relevant definitions.

Process
The executable code or program that calls functions.

Function
Is a programmatic procedure, which performs a unit of work and may or may not return a value.

Buffer
A temporary space reserved in memory that a function can use to store data while it does its work.

Address
A location in memory that is numerically defined.

Instruction pointer
Points to a memory address where a function is run from.

A computer executes processes to perform work. Functions are a component of processes and may call other functions. In order to perform, a function is allocated a region of memory called a stack to store variables and data while it is working. When a function is finished it must return control to the previous function, which is found at a return address on the stack by an instruction pointer. Generally, a function will attempt to make sure that it has enough room in its buffer to store variables. However, a poorly written function that doesn't validate the length or type of input into a buffer may corrupt the stack. A buffer overflow occurs when a function stores more data in the buffer than the space reserved for it. It's like overfilling a glass. The overflow of the buffer causes

[2] http://www.darpa.mil/
[3] http://www.microsoft.com/TechNet/security/bulletin/fq99-049.asp/

memory adjacent to the buffer to be overwritten, corrupting the values previously stored there.

A buffer overflow is exploited when the attacker is able to overwrite the saved instruction pointer that indicates the return address. The attack will be successful when the current function finishes and returns to an address placed in the saved instruction pointer by the attacker. After the function finishes and returns to the address in the saved instruction pointer the code of the attacker will be executed.

The basic reason that buffer overflows exist is due to poor programming practices. According to Rik Farrow in a November 1999 article named Blocking Buffer Overflow Attacks in Network Magazine, "C subroutine calls that copy data but do no bounds checking are the culprits (as well as the programmers who use these calls). The strcat(), strcpy(), sprintf(), vsprintf(), bcopy(), gets(), and scanf() calls can be exploited because these functions don't check to see if the buffer, allocated on the stack, will be large enough for the data copied into the buffer. It is up to the programmer to either use a version that makes the check (such as strncpy()) or to count the bytes of data before copying them onto the stack."

Buffer overflows in Windows Servers

To date there has been six acknowledgements by Microsoft of buffer overflow conditions that could lead to arbitrary execution of code remotely by an attacker on Windows 2000. They include:

- Microsoft Security Bulletin MS00-079, "HyperTerminal Buffer Overflow Vulnerability";
- Microsoft Security Bulletin MS00-094, "Phone Book Service Buffer Overflow Vulnerability";
- Microsoft Security Bulletin MS01-013, "Windows 2000 Event Viewer Contains Unchecked Buffer";
- Microsoft Security Bulletin MS00-085, "ActiveX Parameter Validation Vulnerability";
- Microsoft Security Bulletin MS01-025, "Index Server Search Function Contains Unchecked Buffer";
- Microsoft Security Bulletin MS01-023 "Unchecked Buffer in ISAPI Extension Could Enable Compromise of IIS 5.0 Server."

All of these acknowledgements have an associated patch.

Conclusion

A dangerous vulnerability

This exploit is dangerous because some of the usual preventative and detective controls do not help. A remote attacker can gain full control of the host even behind a firewall if ports 80 or 443 are open. In addition, when this exploit is executed it will kill the web

service, but because the web server will restart automatically, an administrator will be unaware that his web server has been compromised. This would allow the attacker to install a root kit and attempt to further exploit the victim's network. However, there are number of standard defenses the security professional and the industry as a whole should be aware of and use.

How to prevent this attack

Because this attack is a buffer overflow attack the usual buffer overflow preventative measures should apply. They include:

- Good programming practices
- Secured operating system
- Intrusion Detection
- Awareness

Good programming practices

Because buffer overflows begin with poor programming practices it is essential that vendors train their programmers to write secure code. One basic programming practice is to make sure that the C subroutine calls programmer's use to copy data does bounds checking. Another basic practice is ensuring that programmers only give a program the privileges it needs to do its job. This prevents exploited programs from accidentally giving system wide rights to attackers.

In addition, to aid in the auditing of code, recently Secure Software Solutions[4] released a product under version 2 of the GNU Public License called RATS. RATS is a security auditing tool for C and C++ code. According to Secure Software Solution, "RATS scans through code, finding potentially dangerous function calls. The goal of this tool is not to definitively find bugs. Instead, this tool aims to provide a reasonable starting point for performing manual security audits."

However, after the release of a product that contains exploitable buffer overflows there is little that the security professional can do but to focus on the other three defenses.

Secured operating system

Out of the box all operating systems have an existing level of exploitable vulnerabilities. As a result it is critical that system administrators harden their O/S. Each operating system has specific steps an administrator must take, but generally speaking an O/S should do it's intended function by only running the services, daemons, or applications to support that function. In addition, the administrator should apply all relevant patches and bug fixes. After hardening the O/S an administrator should compare it to an established security configuration baseline.

A common complaint against Microsoft is that the default configurations of their products are very insecure. (For example, out of the box Windows 2000 has a feature turned on

[4] http://www.securesw.com/

called IPP.) To assist the administrator in locking down the default configuration, Microsoft has produced a checklist[5] and a configuration template called Hisecweb.inf.[6] These tools are available to the administrator for base lining and deploying secure web servers. In addition, there is a download available that incorporates the configuration template Hisecweb.inf and makes a number of registry modifications to lock down the web server.[7] Many of steps recommended by Microsoft should be standard practice for deploying any web server. The steps include disabling and removing unneeded services and applications and making policy changes at various levels:

- Service settings
- IPSec settings
- SCE settings
- IIS settings

After making your changes it is now possible using a tool included with Windows 2000 called Security Configuration Manger to ensure that your system continues to meet your baseline security configuration.

To help the wary administrator there is also a tool called StackGuard which according to it's maker WireX, "Is a compiler that emits programs hardened against "stack smashing" attacks... Programs that have been compiled with StackGuard are largely immune to stack smashing attack. Protection requires no source code changes at all. When a vulnerable program is attacked, StackGuard detects the attack in progress, raises an intrusion alert, and halts the victim program."

Intrusion Detection

In general terms, network intrusion detection systems can be used to watch for known buffer overflow attacks and their associated network signature. However, there are a few specific things you can look for to prevent this particular attack. Number one, with the aid of your firewall and network based intrusion detection system (IDS) you can be alerted and cautious of anyone attempting to gather information regarding your information systems through port scans or version queries. Reconnaissance or information gathering is usually the precursor of any attack. After being watchful of information gathering activities, any GET requests of .printer with a buffer of greater than or equal to 420 bytes sent within the HTTP Host: Header, is a good indication someone is attempting to exploit the vulnerability.

Awareness: Subscribe to security mailing list

And finally, keep abreast of new developments in security by subscribing to a mailing list or security digest from an organization like SANS.org or SecurityFocus.com as well as the security mailing list for your vendor. No operating system is perfect and new vulnerabilities are disclosed weekly if not daily, by being aware of changes and new developments you can maintain a proactive approach to securing you information

[5] http://www.microsoft.com/technet/security/iis5chk.asp
[6] http://download.microsoft.com/download/win2000srv/SCM/1.0/NT5/EN-US/hisecweb.exe
[7] http://www.microsoft.com/Downloads/Release.asp?ReleaseID=19889

systems.

To keep administrator abreast of new patches and updates Microsoft has a tool called HFCheck.[8] This tool allows IIS5.0 administrators to ensure that their servers are up to date on all security patches. The tool can be configured to run continuously or periodically using either a local database or a remote database on the Microsoft web site to ensure that a server is up to date on all the current and relevant patches. When the tool finds a patch that hasn't been installed, it can display a dialogue box or write a warning to the event log. Using this tool, administrators can be assured that their systems are up to date with relevant patches and fixes.

An evaluation of Microsoft's SWI

Is Microsoft demonstrating their commitment to security when their flagship Windows 2000 server can be exploited with such a common and dangerous vulnerability? Has the Secure Windows Initiative been successful at encouraging a security mindset at Microsoft?

Although, part of the SWI initiative is internal training and awareness, it is still unclear whether this will encourage and allow Microsoft developers to write secure code. However, overall Microsoft has done a pretty good job at providing administrators with the tools they need to initially secure their operating system, maintain an appropriate level of security, and to make them aware of patches through tools like HFCheck.

Unfortunately, many of these problems could be prevented if Microsoft took a more secure approach to programming and a more closed box approach when shipping their server products. However, in the end it must be the administrators responsibility to make the box secure. If an administrator had followed Microsoft's recommendations and secured the system initially they would not have been vulnerable to this exploit.

[8] http://www.microsoft.com/Downloads/Release.asp?ReleaseID=24168

Bibliography

- Aleph One, “Smashing the Stack for Fun and Profit” Phrack Volume 7 Issue 49. 14 May 2001 http://www.securityfocus.com/data/library/P49-14.txt.
- DilDog, “The Tao of Windows Buffer Overflows” 15 May 2001. http://www.cultdeadcow.com/cDc_files/cDc-351/ .
- Farrow, Rik. “Blocking Buffer Overflow Attacks” Network Magazine. 1 November 1999. 15 May 2001. http://www.networkmagazine.com/article/NMG20000511S0015.
- Fisher, Dennis. “Microsoft makes 'clean break' on security policy” eWeek. 11 April 2001.
- “Microsoft Puts Spotlight on Security Leadership at RSA Conference 2001” 10 April 2001. Microsoft. 15 May 2001 http://www.microsoft.com/presspass/press/2001/Apr01/04-10ThompsonPR.asp.
- “Microsoft Security Bulletin MS01-023, Unchecked Buffer in ISAPI Extension Could Enable Compromise of IIS 5.0 Server” 1 May 2001. Microsoft. 5 May 2001 http://www.microsoft.com/technet/security/bulletin/MS01-023.asp.
- Permeh, Ryan. “IISHack 2000” 1 May 2001. eEye. 1 May 2001 http://www.eeye.com/html/research/Advisories/iishack2000.c.
- Russell, Ryan, et al. “Hack Proofing your Network” Rockland, MA: Syngress, 2000.
- “Security in Numbers: New Microsoft Partner Program Boosts Access to Industry’s Best, Say Executives from Foundstone, Guardent” 9 April 2001. Microsoft. 15 May 2001 http://www.microsoft.com/presspass/features/2001/apr01/04-09rsa.asp.
- Simon, Istavan. “A Comparative Analysis of Methods of Defense against Buffer Overflow Attacks” 31 January 2001. California State University. 15 May 2001 http://www.mcs.csuhayward.edu/~simon/security/boflo.html .
- Thomas, Evan. “Attack Class: Buffer Overflows” Hello World April 1999 Issue 2, Volume 1. 12 May 2001 http://www.cosc.brocku.ca/~cspress/HelloWorld/1999/04-apr/attack_class.html#6.