SANS GCIH Practical Assignment
Version 2

Advanced Incident Handling and Hacker Exploits
In support of The Cyber Defense Initiative

Code Red Alert

November 5, 2001

Submitted By:

Kenneth M. Smith
Patriot SANS, Boston 2001

TABLE OF CONTENTS

Topping the charts, Port 80 **1**

Primary Function 1
Port 80 under attack 1
Communication Protocols 2
Internet Protocol (IP) *2*
Transmission Control Protocol (TCP) *2*
Hypertext Transfer Protocol (HTTP) *3*
Web Server Platforms 4
Vulnerabilities by Web Server *5*
Web Server Protection 7
Perimeter Protection *7*
Application Protection *7*

The Buffer **8**

Normal Processing 10
Buffer Overflow, stack smashed. 11

The Exploit **13**

Setting the Stage, the vulnerability 13
The Danger is Confirmed 13
An Exploit is born, Code Red Alert 14
Initial Analysis *14*
The Infection Process *15*
The Damage *21*
Eradication *21*
The Impact *21*
Code Red Version 2 22
Analysis *22*
The Infection Process *22*
The Damage *22*
Eradication *23*
The Impact *23*
Code Red II 25
Analysis *25*
1. Infection 25
2. Propagation 25
3. Plant the Trojan 26
The Infection Process *26*
The Damage *31*
Eradication *31*

The Impact 32

Lessons Learned **33**

Why Code Red was so successful. 33

IIS 33

False Sense of Security 33

Lack of Policies 33

Lack of Resources 34

What is being done? 34

IIS Security Training 34

Third-party products 34

Fighting Back 34

And finally… 35

Anti-Worm Checklist 35

References **39**

Additional Reading **40**

Topping the charts, Port 80

Primary Function

Port 80 is defined by IARN as reserved for HTTP traffic on the TCP protocol. Due to the explosive growth of Internet usage, spurred by the advent of html/http and the Web, Port 80 has become the most heavily used port on the Internet.

Web server applications, or daemons, listen for incoming connections and requests on this port. To communicate back with the browser, or user, ephemeral high ports (1025 – 65534) are used.

Port 80 under attack

According to the Sans Internet Storm Center and the Consensus Intrusion Database (CID), Port 80 is also the most actively probed and attacked port. The CID Graph, shown in Figure 1, depicts this trend and demonstrates the sheer volume of blocked probes on this particular Port. For example, in a period of one day, the total number of packets to Port 80 that were blocked approached 48,000.

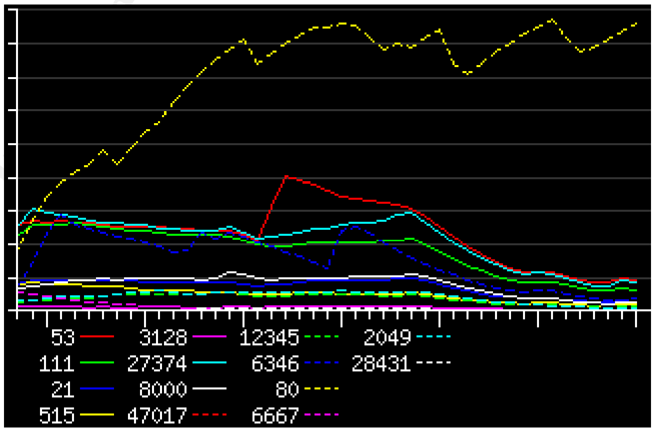


Figure 1: CID Graph, Top Probed Ports

Chart date 5-Oct-2001. The lines on the graph measure the number of ***packets to the top 10 probed ports*** *over time (as measured at Storm Center Partner locations). Values on the Y axis (left side) is the number of denied packets. Each line represents 5,000 probes Values on the X axis (bottom) is time measured in days. Each tick represents 1 day (6 weeks and 2 days total).*

A current CID graph can be obtained at www.incidents.org.

Communication Protocols

For Web communications to occur, there are a number of protocols involved. This section describes these protocols.

Internet Protocol (IP)

IP is a connectionless Network Layer (layer 3) protocol that is responsible for delivering datagrams to their destination host. A datagram is an individual message that contains a collection of data.

When a datagram reaches the Network Layer, from a higher layer, it adds it's own header to facilitate routing through gateway systems. This header contains a number of fields to facilitate packet delivery across internet network. The most important being the source and destination addresses, a protocol number, and a checksum. The protocol number tells the host on the receiving end what protocol the datagram should be delivered to. IP does employ a checksum, but only to confirm that the header remained intact during transit. There is no error detection or verification of the contents of the datagram that IP is tranporting.

Transmission Control Protocol (TCP)

TCP is a connection oriented Transport Layer (layer 4) protocol that manages the reliable delivery of data between hosts. Streams of data are sent to TCP from a higher layer for processing. At this layer, TCP breaks up the message into datagrams and adds it's own header to facilitate guaranteed delivery. Fields in this header include a sequence number, port number, and a checksum for the entire datagram, among others. TCP then sends the datagram to IP with a destination address for delivery.

The following sample network trace shows the initial TCP communication between a user (browser) and a web server (wwwserver).

```
Source      Destination    Protocol Info

browser      wwwserver     TCP      1866 > 80 [SYN]
wwwserver    browser       TCP      80 > 1866 [SYN, ACK]
```

```
browser      wwwserver     TCP      1866 > 80 [ACK]
```

Hypertext Transfer Protocol (HTTP)

HTTP is a connectionless Application Layer (layer 7) protocol. This protocol allows a client and a server to have a dialog for the purpose of exchanging information.

The most common use of HTTP is to facilitate document requests from clients to web servers. A client will request data, typically a document and images, using HTTP. The requested data is typically delivered to the client using the HTTP protocol. The actual data transferred is typically in the format of Hypertext Markup Language (HTML).

Once the TCP 'handshake' takes place, the client sents requests to the server using HTTP. Typically, you will see GET requests for HTML pages and images, like we see here. The actual requests would look something like you see below. In this example, the browser is requesting the HTML document /web1/home.htm.

```
GET /web1/home.htm HTTP/1.1
Accept: */*
Referer: http://www.wwwserver.org/web1/home.htm
Accept-Language: en-us
Accept-Encoding: gzip, deflate
If-Modified-Since: Tue, 21 Aug 2001 20:42:56 GMT
If-None-Match: "3bde1-33d5-3b82c7d0"
User-Agent: Mozilla/4.0 (compatible; MSIE 5.01; Windows NT 5.0)
Host: www.wwwserver.org
Connection: Keep-Alive
```

The web server will then respond by sending the requested data back to the browser in HTTP. The format of the body of the returned document is in HTML, which is then translated by the web browser.

```
HTTP/1.1 200 OK
Date: Thu, 04 Oct 2001 21:57:46 GMT
Server: Apache
Last-Modified: Tue, 21 Aug 2001 20:49:42 GMT
ETag: "97755-49e2-3b82c966"
Accept-Ranges: bytes
Content-Length: 18914
Keep-Alive: timeout=15, max=89
Connection: Keep-Alive
Content-Type: text/html
```

```
<html>
<head>
<title>I am an un-patched server, please attack me!</title>
</head>

<body background="/images/bg0.gif" link="#000080">

<table border="0" cellpadding="3" cellspacing="0">
  <tr>
    <td width="400"><img src="/images/images-r-us.gif"
width="384" height="100"
    alt="images-r-us.gif (4861 bytes)"></td>
```

On the network, this dialog would look something like this:

Source	**Destination**	**Protocol**	**Info**
browser	wwwserver	HTTP	GET / HTTP/1.1
wwwserver	browser	HTTP	HTTP/1.1 304 Not Modified
browser	wwwserver	HTTP	GET /web1/home.htm HTTP/1.1
wwwserver	browser	HTTP	HTTP/1.1 304 Not Modified
browser	wwwserver	HTTP	GET /images/logo1.gif HTTP/1.1
wwwserver	browser	HTTP	HTTP/1.1 304 Not Modified
browser	wwwserver	HTTP	GET /images/logo2.gif HTTP/1.1
wwwserver	browser	HTTP	HTTP/1.1 304 Not Modified
browser	wwwserver	HTTP	GET /images/dot1.GIF HTTP/1.1

Web Server Platforms

It is estimated that there are more than 35 million web servers on-line today. To try and get a better understanding of the overall impact that a worm would have on a specific web server platform, I consulted the Netcraft Server Survey. What they do is probe as many web servers as they can over a period of time. They look at the response header in the html, which will usually display the web server type, IIS, Netscape-iPlanet, Apache, etc. This data is compiled into graphs and tables for reference.

According to the latest figures available at Netcraft, the top three Web server applications are Apache, Microsoft IIS, and Netscape/iPlanet.

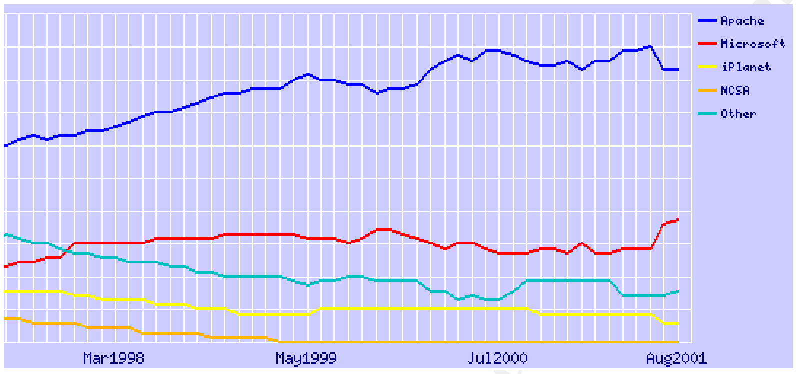


Figure 2: Netcraft Survey, Top Web Servers.

As you can see in the Netcraft graph, Apache is the market leader with almost 61 percent. Microsoft IIS is second, with just over 28 percent. Netscape/iPlanet is third with just over two percent of the market.

Vulnerabilities by Web Server

To get an idea of how many vulnerabilities do exist within each of the web server applications, I referred to the Common Vulnerabilities and Exposures (CVE) dictionary at MITRE, www.cve.mitre.org. This dictionary aims to create a standardized name to represent each vulnerability or exposure. The result is a common data format that allows different security databases and tools to speak the same language.

To gather my information, I ran queries against the CVE using the standard name of the web server application. For example, for Internet Information Server, I used IIS.

Web Server	***CVE Entries****
Apache	26
Microsoft IIS	84

Netscape/iPlanet	24

Table 1: Web Server Vulnerabilities according to CVE.

* includes CVE candidates.

As Figure 1 demonstrates, Microsoft Internet Information Server has the largest documented number of vulnerabilities, of the top three web servers, in the CVE dictionary. Lets take a closer look at some of the most common vulnerabilities in IIS.

Now if you were someone looking to wreak havoc on as many systems as possible, what web platform would you chase after? Based on the numbers, wouldn't you write the exploit to target Apache systems to get the most coverage? You would think that this would be the case, but that's not the way things have turned out.

Is IIS just that insecure overall? Are hackers specifically picking on Microsoft by targeting IIS? Does the close integration between IIS and the operating system actually create these opportunities?

Microsoft has done well for themselves by making everything simple, and by including many built-in features in their products. This is very true of IIS. It used to be that you needed to know something about http, perl, html and cgi, to get a site up and running.

Anyone can have a web server up in a few minutes. Included people who probably haven't really thought about the risks and ramifications in doing so. A few click of the mouse, and new customers will come knocking at your door (so they said).

As it turns out, there were so many sites out there that have never been patched that it's almost embarrassing. A great way to find them is to look at the contents of the web site itself. It the static content stale? 8 months old? Over a year old? If the content is not being kept up, then it's likely that the patches are not being applied either. Code Red helped to confirm this in a big way.

IIS Vulnerabilities

The overall number of vulnerable IIS web servers on-line today is surprisingly high. The number of servers running Microsoft IIS is estimated to be approaching 5.9 million. The following graph depicts the 10 most common vulnerabilities found on Microsoft IIS Servers, according to tests performed by Netcraft.

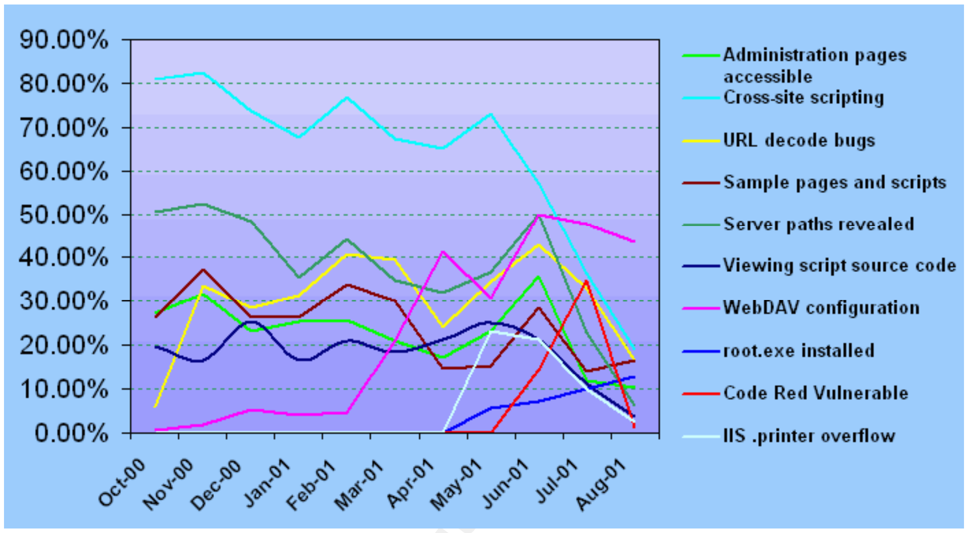


Figure 3: Top 10 IIS Vulnerabilities, as reported by Netcraft.

Web Server Protection

Perimeter Protection

Firewalls are the primary method of protecting networks and hosts from Internet-based attacks. A firewall-gateway allows the restriction of inbound and outbound traffic by using a static rule-base. This rule-base defines the services and hosts that are allowed or disallowed.

In the case of a publicly accessible Web Server environment, this Firewall rule-base is configured to allow HTTP Port 80 traffic inbound to the Web Servers. In addition, the Firewall will allow the Web Server to communicate back to the outside world using ephemeral high ports, either via state tables or a specific rule.

Application Protection

For Web Server environments, this is typically where the traffic control ends. As soon as Internet users are allowed into the network to access the web servers over port 80, the Firewall can do very little to prevent malicious activity. The Firewall has done its job. From here, it's up to the Web Server itself to manage the inbound requests, both valid and invalid.

The Buffer

The primary point of risk within Web servers is where they accept input (or requests) from the client. Additional functionality is provided to web users via Common Gateway Interface (CGI) applications. Processing requires the use of buffers.

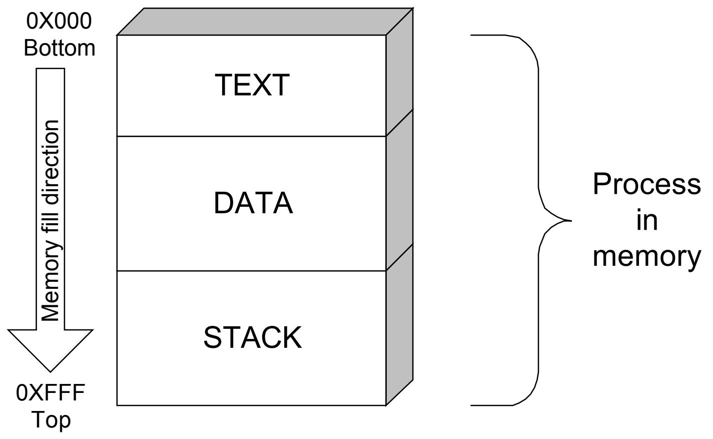


Figure 4: Process in memory.

A Buffer is a contiguous reserved space that stores a collection of data items. Each set of Processes in memory is organized into three areas.

- Text: Fixed size area of memory that stores code and read-only text data.
- Data: Used to store static variables. The size of this area can be changed via a system call.
- Stack: Temporarily stores operands and elements for the running process. The size of the stack can be changed. The address location of the top of the stack is constantly monitored and tracked by the Stack Pointer.

When a running process calls a subroutine, arguments or operands required by the subroutine are copied to the top of the stack. This process is called 'pushing' to the stack, along with the address that processing must return to when the subroutine is completed. A subroutine may also push elements or results of it's processing to the stack.

Elements of the stack are then read and removed or 'popped' from the top of the stack. While all of this is going on, the Stack Pointer or Register keeps track of exactly where the top of the stack is by storing it's memory location. The process of pushing and popping the stack works in a Last In First Out (LIFO) model.

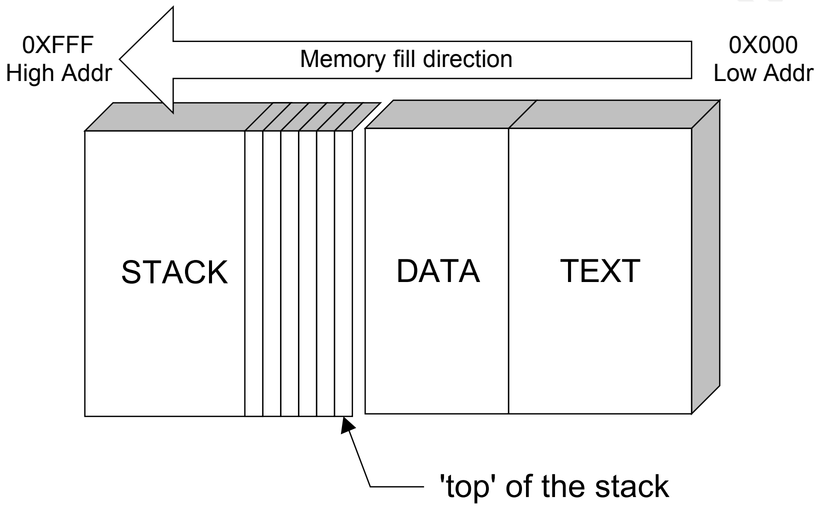


Figure 5: The Stack.

The 'top' of the stack is actually a lower memory address, while the bottom of the stack is a higher memory address. To simplify the concept, Figure 5 shows the buffer in a horizontal format. First, lets look at how this process normally occurs.

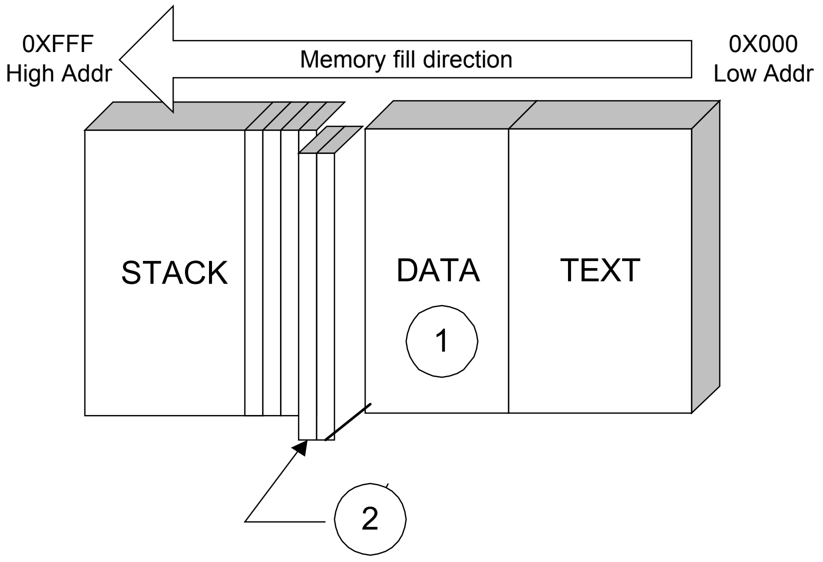


Figure 6: Normal Process Execution Begins.

Normal Processing

1. Processing begins.
2. The current process pushes data (parameters, variables, etc.) to the stack that will be needed by the subroutine.

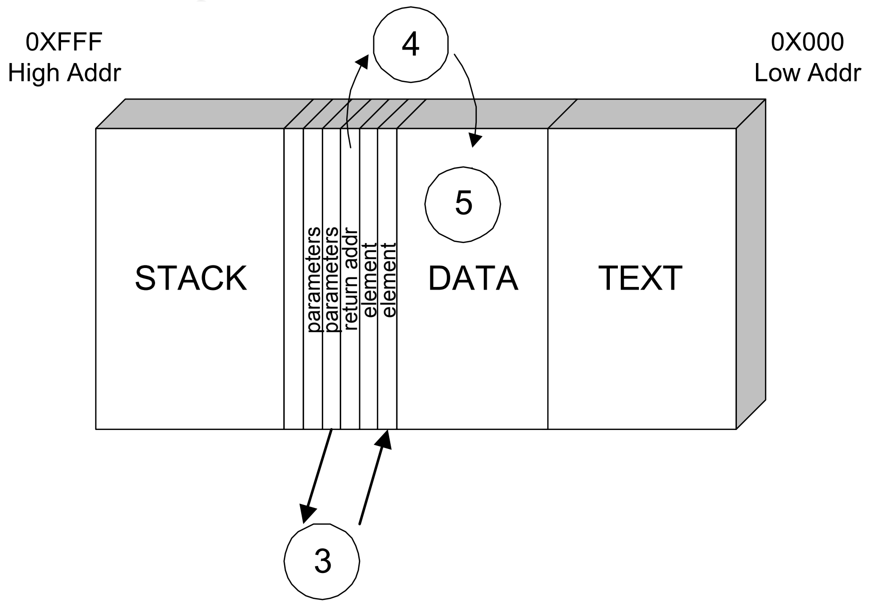


Figure 7: Normal Process Execution Ends.

3. Subroutine runs, reading stack frames as needed and returning or pushing elements (results) back to the stack.
4. The subroutine returns control to the calling process, based on the contents of the return address stack element.
5. Processing continues.

Now that we know what the stack does under normal circumstances, lets go through the same buffer being overflowed by malicious code, such as Code Red.

Buffer Overflow, stack smashed.

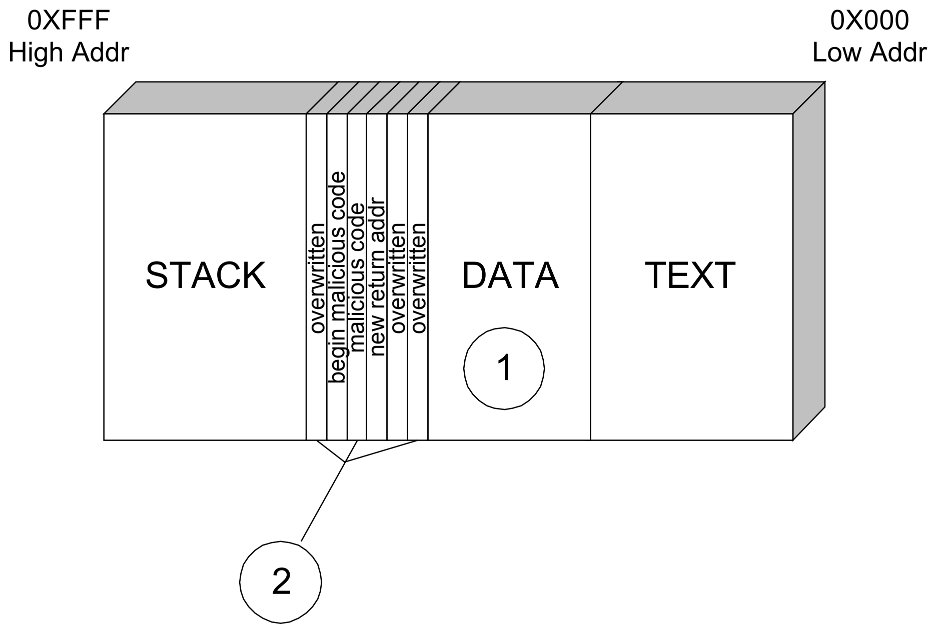


Figure 8: Buffer is Overflowed, Stack Smashed.

1. Processing begins.
2. The current process pushes data (parameters, variables, etc.) to the stack that will be needed by the subroutine. The length of some of this data is beyond the expected limits, an unexpected amount of stack frames is pushed to the stack as a result.

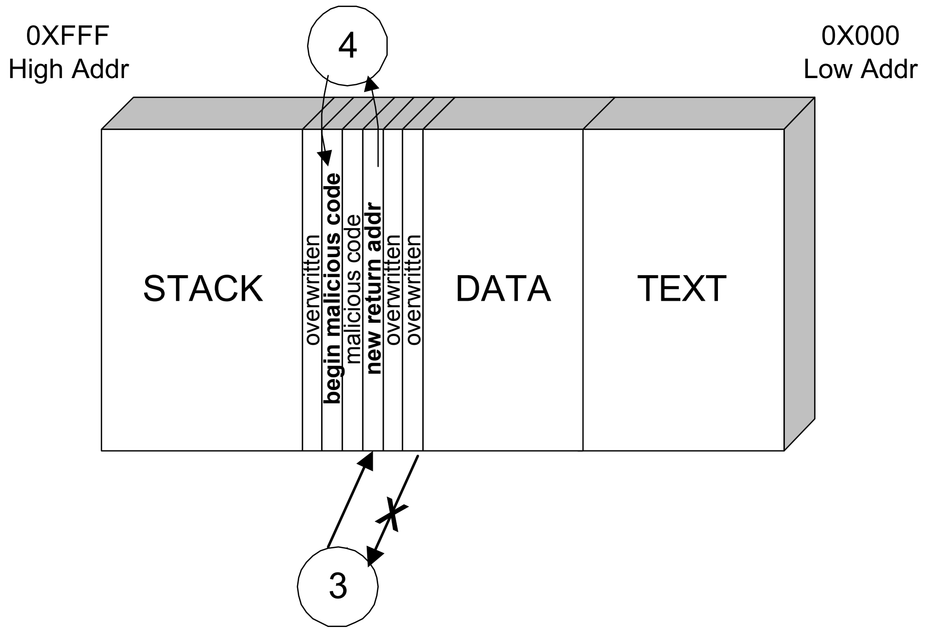


Figure 9: Control is Returned to Malicious Code.

3. Subroutine runs, attempting to read stack frames but finds that the stack has been overwritten or smashed. The subroutine fails and attempts to return control to the calling process, based on the contents of the return address stack element. But the buffer overflow has successfully overwritten this pointer with the address of malicious code that it has been placed in the stack.

4. Control is returned to the malicious code. The shell code is executed at the current privledge level.

5. The system has been compromized.

The Exploit

Setting the Stage, the vulnerability

On June 18th, 2001, eEye Digital Security discovered a new vulnerability in Microsoft's IIS Web Server software. The vulnerability lies within the code that allows a Web server to interact with Microsoft Indexing Service functionality. The vulnerable Indexing Service ISAPI filter is installed by default on all versions of IIS. The problem lies in the fact that the .ida (Indexing Service) ISAPI filter does not perform proper "bounds checking" on user inputted buffers and therefore is susceptible to buffer overflow attacks.

That same day, Microsoft released the security bulletin MS01-033 "Unchecked Buffer in Index Server ISAPI Extension Could Enable Web Server Compromise". This bulletin identified the vulnerability and the patch required to resolve it. eEye also released an Advisory, "All versions of Microsoft Internet Information Services Remote buffer overflow (SYSTEM Level Access) (AD20010618)." The Common Vulnerabilities and Exposures (CVE) Candidate identifier of CAN-2001-0500 was then assigned.

The Danger is Confirmed

To demonstrate the seriousness of this vulnerability, a number of 'proof of concept' exploits were written. The most notable is the "IIS5.0 .idq overrun remote exploit" by HSJ. This exploit code demonstrated that the vulnerability was indeed exploitable. The actual code for this exploit can be obtained from http://www.securiteam.com/exploits/5HP0N2A4KQ.html

An Exploit is born, Code Red Alert

On July 12th, almost four weeks later, reports of a new worm began to surface. This worm exploited the "ISAPI Extension Buffer Overflow" vulnerability on Windows NT and Windows 2000 systems running Microsoft IIS.

Unfortunately, many systems had not yet been patched and were still vulnerable. An obvious indication that a system had been compromised was that the default web page was defaced with the phrase "Hacked By Chinese".

Initial Analysis

On July 13th, the worm was disassembled and analyzed by Ryan Permeh and Marc Maiffret of eEye Digital Security. They found that this worm would infect vulnerable IIS servers running Windows 2000. Systems running Windows NT would just crash, instead. The infection consists of a series of memory resident processes that would do the following:

1. If the current system date fell between the 1st and the 19th of the month, a random list of IP addresses was generated.

2. This list of IP addresses was then probed in an attempt to infect those systems and further propagate.

3. The top level web page on many infected hosts was defaced with the phrase "Hacked By Chinese".

The worm would stop attempting to infect other systems on the 20th of the month. At this time, the second attack phase began.

4. Between the 20th and the 28th of the month, the worm launches

a Denial-of-Service attack against the Whitehouse web site, www.whitehouse.gov.

The eEye Digital Security Engineers had been up all night analyzing the worm, apparently drinking Code Red Mountain Dew, the highly caffeinated soft drink. This, plus the "Hacked By Chinese" web defacement phrase, led them to officially name the worm Code Red. The full eEye report, including a deep analysis, can be obtained from http://www.eeye.com/html/Research/Advisories/AL20010717.html

A sniffer trace of incoming Code Red probes would look something like this:

```
GET
/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNN%u9090%u6858%ucbd3%u7801%u9090
%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090%u8190%u00c
3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a    HTTP/1.0
```

For those with an Intrusion Detection Systems, this would also be the signature used to detect the Code Red worm.

The Infection Process

This example will run through a Code Red infection scenario. We will assume the following:

- The Date: July 15, 2001.
- Victim #1: A Windows 2000 Server running IIS 5 as www.victim1.com, using an IP Address of 10.1.1.101. This host has already been infected by the Code Red Worm.
- Victim #2: A Windows 2000 Server running IIS5 as www.victim2.com, using an IP Address of 10.1.1.102.

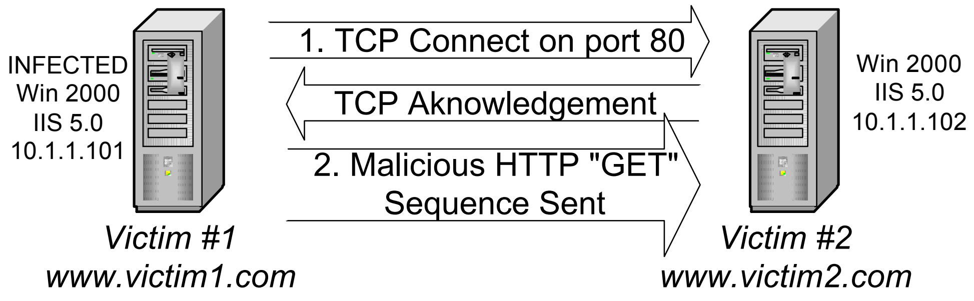


Figure 10: Code Red Infection Begins.

Step 1: Victim #1, already infected with Code Red, probes for more victims by sending an HTTP request to a list of target addresses (psuedo-randomly generated) on TCP port 80. One of these target addresses is 10.1.1.102, that of Victim #2. Remember that HTTP uses TCP and IP, so the TCP handshake must first take place.

```
Source      Destination    Protocol Info

victim1      victim2       TCP      1866 > 80 [SYN]
victim2      victim1       TCP      80 > 1866 [SYN, ACK]
victim1      victim2       TCP      1866 > 80 [ACK]
```

Step 2: If a TCP connection to port 80 cannot be obtained, the worm skips the address and moves on to the next victim. This would happen, for example, if Victim #2 were not a web server or was not listening on port 80.

Upon successful connection, the HTTP request would be sent:

```
GET
/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNN%u9090%u6858%ucbd3%u7801%u9090
%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090%u8190%u00c
3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a    HTTP/1.0
```

This request acts as the injection vector to facilitate the payload delivery.

Step 3: Victim #2 also happens to be running Microsoft IIS and listening on port 80. The host receives the request and IIS begins to process it.

- At this stage, if this system were running the current IIS patches, then only an error would result. The exploit would be unsuccessful.
- If this system were running another web server application, for example Apache, then only an error would result. The exploit would be unsuccessful.

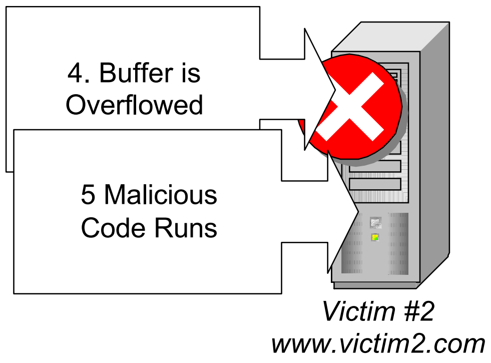


Figure 11: Buffer Overflowed

Step 4: Buffer is overflowed. See the Buffer Overflow section of this document for a detailed explanation of this process.

- If this system were running Microsoft Windows NT, then the web server would halt. The remainder of the exploit would be unsuccessful. Since Victim #2 is running Windows 2000, we will proceed to the next step.

Step 5: The malicious payload is delivered and executed at the current privledge level. Victim #2 is now compromised. This malicious code begins by performing the following:

a. Makes calls to functions to gather information needed for successful execution. Initial calls are to:

GetProcAddress

LoadLibraryA

b. GetProcAddress and LoadLibraryA are used to locate the memory locations of the following dll's:

kernel32.dll

infocomm.dll

ws2_32.dll

w3svc.dll

c. Once these dll's are located in memory, the following functions are called during code processing:

GetSystemTime
CreateThread
CreateFileA
Sleep
GetSystemDefaultLangID
VirtualProtect
TcpSockSend
Socket
Connect
Send
Recv
CloseSocket

d. The worm forces Victim #2 to send an HTTP GET back to Victim #1.

e. Generates a random list of IP addresses.

f. Victim #2 now begins to probe the systems in this list on TCP port 80 using the same HTTP request that was responsible for it's own infection:

```
Source        Destination     Protocol Info

Victim2        victim3        TCP      1866 > 80 [SYN]
Victim3        victim2        TCP      80 > 1866 [SYN, ACK]
Victim2        victim3        TCP      1866 > 80 [ACK]
```

```
GET
/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
```

```
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN%u9090%u6858%ucbd3%u780
1%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090
%u8190%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a
HTTP/1.0
```

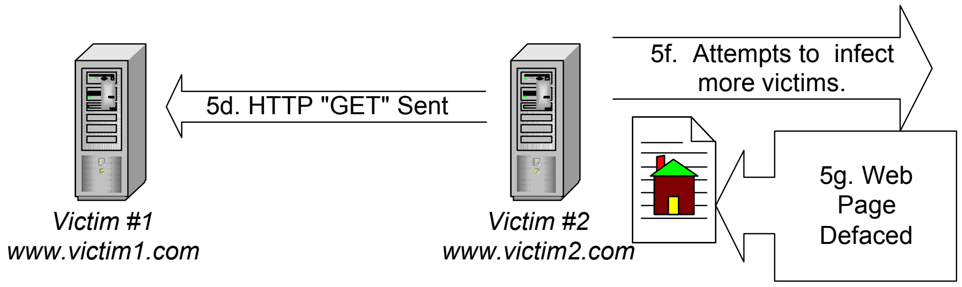


Figure 12: System Compromised

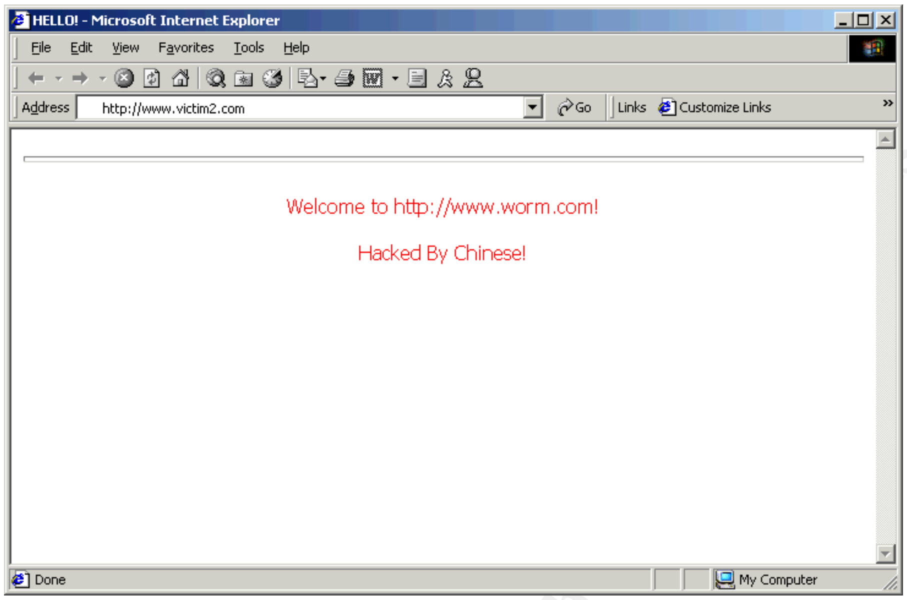
g. The worm code attempts to replace the top level web page on Victim #2 with phrase "Welcome to http://www.worm.com! Hacked By Chinese!". This is done by modifying the contents of memory, not by writing to a file. The HTML looks like this:

```
<html><head><meta http-equiv="Content-Type" content="text/html;
charset=english"><title>HELLO!</title></head><bady><hr size=5>

<font color="red"><p align="center">Welcome to
http://www.worm.com!<br><br>Hacked By Chinese!</font>

</hr></bady></html>
```

The top level web page for Victim #2 now looks like the example shown in Figure 10. Victim #2 is now completely infected and will continue to probe for new victims.


HELLO! - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://www.victim2.com
Go
Links
Customize Links
Done
My Computer

Welcome to http://www.worm.com!

Hacked By Chinese!

Figure 13: Web Page of Infected Server.

Step 6: If the system is still infected on July 20th, the system would begin performing a Denial of Service (DOS) attack against www.whitehouse.gov.

a. Connects to 198.137.240.91 on TCP port 80.

b. If this connection is successful, a loop process begins and will continue to send single byte packets to 198.137.240.91.

c. After the loop completes it's attack, this DOS process will sleep for approximately 4 hours and 30 minutes.

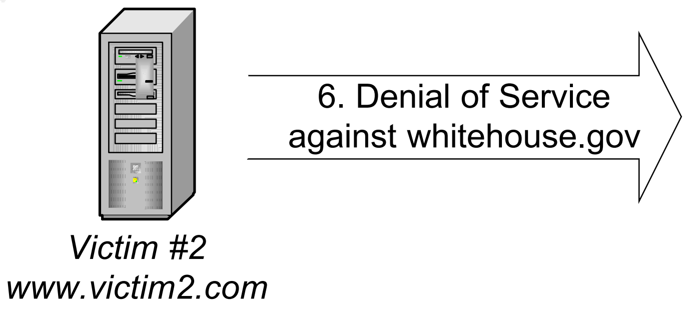


Figure 14: Victim #2 Launches DOS Attack.

The Damage

At the time, the only known damage inflicted by Code Red on the infected host was the "Hacked By Chinese" web page defacement and, of course, service disruption.

Eradication

Since the Code Red worm was memory resident, simply rebooting the infected system would disinfect it. Many quickly learned, however, that after being rebooted, systems were quickly infected again. For systems to be safe from re-infection, the Microsoft patch provided in MS01-033 was required.

The standard policy of many organizations is to rebuild a system that has been compromised in any way. If unsure about the state of a system, the security 'best practice' is to reformat the drives, reinstall and patch the operating system, and restore data from a known good pre-infection backup.

The Impact

The initial Code Red worm, often now referred to as Code Red Version 1, had a very limited global impact on systems. This was primarily due to the way that the worm generated target IP addresses. Since a "static" seed was used to generate the target IP addresses, the 'randomly generated' lists of addresses actually turned out to be similar from many infected hosts. This resulted in many systems being probed continuously, whether already infected or not. In addition, many of the systems probed were either non-vulnerable IIS systems or were not running IIS at all. The result was a slowly propagating worm that tied up infected servers resources and LANs while attempting to spread.

Code Red Version 2

On July 19th, seven days after Code Red was discovered, a new variant began to propagate quickly across the Internet. This new worm seemed identical in all aspect to Code Red, exploiting the same "ISAPI Extension Buffer Overflow".

Analysis

Further review by the team at eEye Digital Security (Ryan and Marc, at it again with the Code Red Mountain Dew I assume) found one small difference in this Code Red variant. This time, the seed used to generate the list of target IP addresses was completely random. This meant that each system that was infected would likely target completely different systems. This worm was quickly dubbed Code Red Version 2.

This seemingly trivial change to the worm resulted in a much higher rate of propagation and infection.

The Infection Process

The infection process of Code Red Version 2 was identicle to the Code Red Version 1 infection process.

The Damage

The damage inflicted by Code Red Version 2 to the infected host is the same as that of the original Code Red, the "Hacked By Chinese" web page defacement. This time, however, the service disruption was substantial.

Eradication

Again, since the Code Red Version 2 worm was also memory resident, simply rebooting the infected system would disinfect it. But the simple change in the way that IP addresses were generated now allowed this worm to spread very quickly. This worm traveled so fast that hosts were even infected between the time of reboot, and the application of the patch. To completely cleanup an infected host, it was usually necessary to disconnect it from the network long enough to reboot and install the patch.

The same patch used to close the "ISAPI Extension Buffer Overflow" vulnerability in IIS (MS01-033) exploited by Code Red also applies to Code Red Version 2.

As mentioned earlier, it is a security 'best practice' to rebuild compromised systems. This new version of the worm, however, infected so many systems that organizations quickly found it impractical to rebuild all infected systems and get back online in a reasonable amount of time. Also, all indications were that this was only a memory resident process and that it contained no additional malicious code. Based on these factors, this 'best practice' was largely ignored.

On July 30th, Microsoft released an article titled, "Protecting yourself from the Code Red Worm.

The Impact

Unlike Code Red Version 1, this new version had a large impact on the global Internet infrastructure. According to CAIDA, The Cooperative Association for Internet Data Analysis, more than 359,000 machines were infected with Code Red Version 2 in just 14 hours. This worm generated such a large amount of network traffic that slowdowns or outages were felt clear across the globe.

As if that weren't enough, many web-enabled appliances and network devices were also impacted. This included routers, switches, firewalls, and even printers. Cisco, for example, released a Security Advisory titled, "Code Red Customer Impact". Although most appliances and devices were not actually vulnerable, and could not be infected, some experienced erratic behavior, system hangs, or reboots, when they were probed. If this didn't affect the device, then the amount of traffic alone was enough to bring some devices to their knees.

To give you an idea of how fast this worm spread, Jeff Brown from University of California San Diego, created a series of animations. These animations, based on analysis by David Moore of CAIDA at San Diego Supercomputer Center, demonstrate the geographic spread of the worm. The animations are available at http://www.caida.org/analysis/security/code-red/coderedv2_analysis.xml#animations.

Computer Economics, of Carlsbad California, reported that the financial impact of the Code Red (Version 1 and 2) worms was estimated to be $2.62 billion. Their statistics on this and other worms can be found by visiting: http://www.computereconomics.com/cei/press/pr92101.html.

Code Red II

On approximately August 4, 2001, another worm began to work its way across the Internet. This worm was very different from Code Red Version 1 and 2. The only similarity was that it used the same IIS "ISAPI Extension Buffer Overflow" vulnerability to exploit the target system.

Analysis

While analyzing the worm, eEye Digital Security Engineers (Ryan Permeh and Marc Maiffret) found the string "CodeRedII" and named it as such. Upon closer review, it was confirmed that this indeed was a totally separate worm.

This worm is much more complicated than Version 1 or 2. I have summarized the processes that the worm attempts below.

This worm has three phases: Infection, Propagation, and the Trojan installation.

1. Infection

a. Checks to see if the system has already been infected with this worm.

b. Chinese language systems will spawn 600 threads, English language systems will spawn 300.

c. Installs a Trojan.

d. Sleeps for 2 days on Chinese language systems, 1 day for all others.

e. Windows is rebooted.

2. Propagation

a. Checks to make sure that the current date is before Oct 1 or before the year 2002.

b. Generates target IP address list.

c. Probes target hosts.

d. Attempts to infect more victims.

3. Plant the Trojan

This process installs the back-door as Explorer.exe.

a. Copies cmd.exe to /scripts/root.exe.

b. Copies cmd.exe to /msadc/root.exe

c. Writes out Explorer.exe

The Trojan functionality provides a virtual web path to the root of C:\ and D:\. This allows an attacker to run any command on the system as long as the Trojan Explorer.exe is running.

To send commands to the back-door, an attacker would send something like this:

```
http://address/c/inetpub/scripts/root.exe?/c+dir
```

OR

```
http://address/c/winnt/system32/cmd.exe?/c+dir
```

Where 'dir' would be the command that they would like to run.

The Infection Process

This example will run through a Code Red II infection scenario. We will assume the following:

- The Date: August 5, 2001.
- Victim #1: A Windows 2000 Server running IIS 5 as www.victim1.com, using an IP Address of 10.1.1.101. This host has already been infected by the Code Red II Worm.
- Victim #2: A Windows 2000 Server running IIS5 as www.victim2.com, using an IP Address of 10.1.1.102.

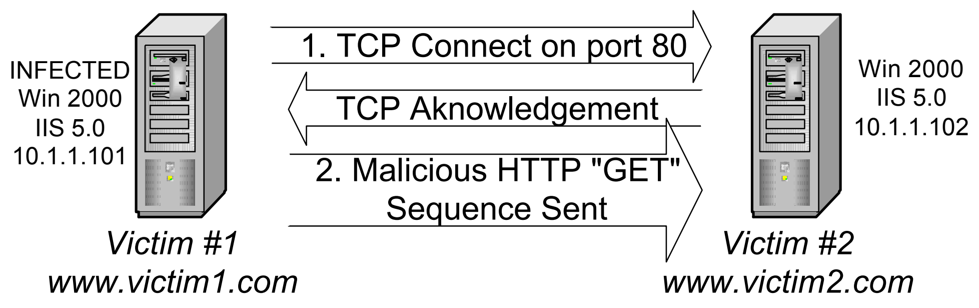


Figure 15: Code Red II Infection Begins

Step 1: Victim #1, already infected with Code Red II, probes for more victims by sending an HTTP request to a list of target addresses (randomly generated) on TCP port 80. One of these target addresses is 10.1.1.102, that of Victim #2. The TCP handshake will occur first.

```
Source      Destination    Protocol Info

victim1      victim2       TCP      1866 > 80 [SYN]
victim2      victim1       TCP      80 > 1866 [SYN, ACK]
victim1      victim2       TCP      1866 > 80 [ACK]
```

Step 2: If a TCP connection to port 80 cannot be obtained, the worm skips the address and moves on to the next victim. This would happen, for example, if Victim #2 were not a web server or was not listening on port 80.

Upon successful connection, the HTTP request would be sent:

```
GET
/default.ida?XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXX%u9090%u6858%ucbd3%u7801%u9090
%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090%u8190%u00c
3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a  HTTP/1.0" 404 205
```

This request acts as the injection vector to facilitate the payload delivery. Note that this HTTP request is slightly different than that of Code Red Versions 1 and 2. The padding characters are now "X" instead of "N".

Step 3: Victim #2 also happens to be running Microsoft IIS and listening on port 80. The host receives the request and IIS begins to process it.

- At this stage, if this system were running the current IIS patches, then only an error would result. The exploit would be unsuccessful.

- If this system were running another web server application, for example Apache, then only an error would result. The exploit would be unsuccessful.

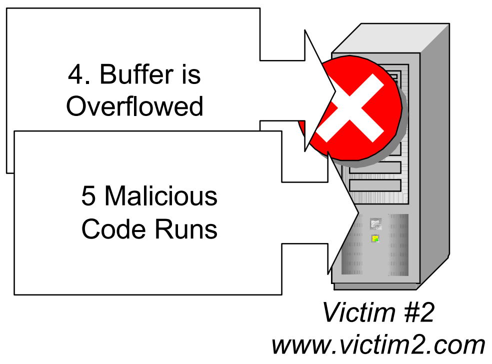


Figure 16: Buffer Overflowed

Step 4: Buffer is overflowed. See the Buffer Overflow section of this document for a detailed explanation of this process.

- If this system were running Microsoft Windows NT, then the web server process would halt. The remainder of the exploit would be unsuccessful. Since Victim #2 is running Windows 2000, we will proceed to the next step.

Step 5: The malicious payload is delivered and executed at the current privledge level. Victim #2 is now compromised. This malicious code performs the following:

a. Checks to see if the system has already been infected with this worm. If so, it will not continue.

b. If the system is running the local system language of Chinese, 600 threads are allocated for propagation. Systems running any other language will allocate 300 threads.

c. The Trojan 'back door' is installed.

1. Copies c:\winnt\system32\cmd.exe to /scripts/root.exe.
2. Copies c:\winnt\system32\cmd.exe to /msadc/root.exe

3. Writes out Trojan Explorer.exe

d. The worm sleeps for 2 days for Chinese systems, 1 day for all others.

e. Windows is rebooted.

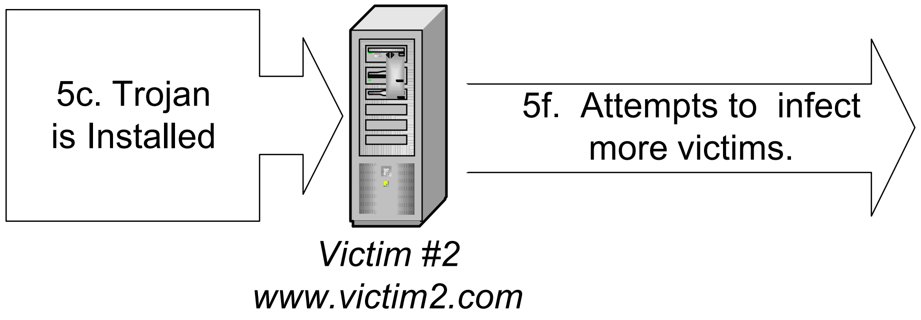


Figure 17: System Compromised

f. Propagation begins.

1. Checks the local system date and time. If the month is 10 or greater OR the year is 2002, then the system reboots. Otherwise, it continues to the next step. This would prevent to worm from propagating on or after October 1.

2. Generates target IP address list.

The target IP address is calculated as follows:

Addresses for target systems are randomly generated. A mask, based on the address of the local host, is then applied to each of these addresses. This masking process results in a good probability of finding vulnerable hosts. The reason is that it's more likely that systems sitting on the same network, as the already infected system, are also running IIS and have the same vulnerability.

The following table describes the IP address calculation probabilities, assuming the infected host's address was 10.1.1.102.

Address	***Probability***	***Example***
Same /8 network or class A	50%	10.xxx.xxx.xxx
Same /16 network or class B	37.5%	10.1.xxx.xxx
Random Address	12.5%	xxx.xxx.xxx.xxx

Table 2: Code Red II Target Address.

Due to the way the target addresses were calculated for Code Red II, a network that contained an infected host would experience a higher level of network slowdown and disruption of service than had been experienced with Code Red 1 and 2. This was due, in part, to the large amount of ARP traffic generated as the infected host attempted to locate hosts on the local subnet.

3. Probes target hosts by attempting to make a connection on TCP port 80. If a TCP connection to port 80 cannot be obtained, the worm skips the address and moves on to the next victim.

```
Source       Destination     Protocol Info

Victim2       victim3        TCP      1866 > 80 [SYN]
Victim3       victim2        TCP      80 > 1866 [SYN, ACK]
Victim2       victim3        TCP      1866 > 80 [ACK]
```

4. Victim #2 now sends the same HTTP request that was responsible for it's own infection:

```
GET
/default.ida?XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
X%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u68
58%ucbd3%u7801%u9090%u9090%u8190%u00c3%u0003%u8b00%u531b%u5
3ff%u0078%u0000%u00=a  HTTP/1.0"  404  205
```

At this point, System #2 has been completely infected with the Code Red II Worm, a Trojan Backdoor has been installed, and this host will continue to probe for new victims.

The Damage

The damage inflicted by Code Red II is quite different than Version 1 and 2. There was no web page defacement, and no Denial of Service against the Whitehouse web site. Instead, a Trojan named Explorer.exe is installed on the system, giving an attacker root level access to the entire system via a web browser. This back-door will continue to allow access even if the infected system is rebooted.

The Code Red II worm leaves the following on an infected server:

1. A file named Explorer.exe will be in the root of the C: and D: drives.

2. A file named Root.exe will be in the \inetpub\scripts\ folder.

3. A file named Root.exe will be in the \Program Files\Common Files\System\msadc folder.

Eradication

To completely clean a system, it must first be disinfected. To do this, Microsoft created a utility to "Clean the Obvious Effects of the Code Red II Worm". This tool can be found at: http://www.microsoft.com/technet/security/tools/redfix.asp.

This tool performs the following:

1. Removes the malicious files.

2. Removes the IIS mappings for /Scripts and /MSADC.

3. Reboots the system.

It's important to note that running this tool does not guarantee that a system has been 'cleaned'. Even Microsoft cautions that this tool only removes the “Obvious Effects” of Code Red II and does not eliminate the damage (or additional Trojans) that may have been installed to an infected system.

In addition, Microsoft includes a disclaimer that seems to agree with the ‘best practice’ of rebuilding internet-facing systems after a compromise. For assistance in dealing with a root compromise, CERT has a great document on this subject available at: http://www.cert.org/tech_tips/root_compromise.html.

To prevent further infection, the same patch used to close the “ISAPI Extension Buffer Overflow” vulnerability in IIS (MS01-033) exploited by Code Red 1 and 2 also applies to Code Red II.

On August 15, 2001, Microsoft released security bulletin MS01-044, which addresses a number of recent IIS vulnerabilities, including Code Red and it’s variants, and provides a cumulative patch. This new security bulletin supercedes MS01-033 and can be found at the following URL: http://www.microsoft.com/technet/security/bulletin/MS01-044.asp.

The Impact

The overall impact of this new worm to the global community was much less than that of Version 2. Although the algorithm used to generate target IP addresses was much more targeted, and much more intelligent that Version 1, many administrators had already applied the necessary patches in response to the earlier worms.

Code Red II made headlines on August 9th when Microsoft confirmed that a few of the MSN Hotmail servers were infected with the Code Red II worm. The full Computerworld article, by Joris Evers of IDG, can be found at: http://www.computerworld.com/storyba/0,4125,NAV47_STO62917,00.html

Lessons Learned

Why Code Red was so successful.

IIS

This goes without saying, I guess. If it weren't for so many IIS servers, then the worm would not have had such an impact. As I mentioned in the 'IIS Vulnerabilities' section, there are a lot of IIS servers that were just simply 'clicked' into action and that's the way they still operate today.

Many are of the opinion that IIS is just too insecure. As I am writing this, another highly publicized worm, called Nimda, is affecting Microsoft systems around the globe. Citing the many recent security vulnerabilities and worms that have exploited those vulnerabilities, Garner Research has gone so far as to advise companies to consider alternatives to IIS. The complete article can be found at http://www.gartner.com/DisplayDocument?doc_cd=101034

False Sense of Security

When I talk to companies about security audits, the most common response is, "We have a Firewall, we're all set." This says quite a bit about how that company sees security. When I hear this response, I know that this company would be a perfect candidate for an assessment.

Lack of Policies

A set of minimum requirement for all systems that contain or protect corporate data is a must. Many companies do have some best-practices, or recommended configuration guidelines. For the department that is trying eagerly to get a new web server online, these guidelines can easily be interpreted as optional.

Standard host configuration rules, for example, need to be documented and disseminated. Staff should be made aware that these rules are a "must", not optional.

Lack of Resources

Most network and host administrators are up to their ears with work. In addition, they are being asked to take on more and more tasks due to corporate reorganizing and layoffs. Many of the security-related tasks fall under the 'needs attention' category. Most are busy taking care of 'break-fix' items, however.

What is being done?

IIS Security Training

In response to these latest worms, The SANS Institute has created a 'Securing Microsoft's IIS Web Server' course. The intent is to educate people on the proper (and secure) installation and configuration of IIS servers. The course has been selling out quickly. For more information, visit http://www.sans.org/IIS/sec_IIS.htm.

Third-party products

The recent vulnerabilities in IIS have helped companies promote their add-on security products.

eEye Digital Security, the company responsible for much of the Code Red analysis, is the maker of 'SecureIIS Application Firewall'. This product claims to stop known and unknown attacks by using CHAM (Common Hacking Attack Methods) technology. You could call it 'instant bounds checking'. http://www.eeye.com/html/Products/SecureIIS/index.html

Similarly, Entercept Security Technologies offers 'Entercept Web Server Edition' specifically for web servers. Along with HTTP protection, they use a technique they call Shielding that also monitors at the operating system level. http://www.entercept.com/products/

Fighting Back

A few cool new tools have been developed in direct response to Code

Read and other worms in general. One of the most notable is an application that "tar-pits" attackers using functions of the TCP/IP communication protocol.

Tom Liston, with proof-of-concept assistance from Mihnea Stoenescu, developed a tool that uses a special method of 'hanging on to' incoming TCP exploit attempts, called bottlenecking or teergrubbing. This application reduces the effectiveness of the worm by tying up threads that would have otherwise tried to attack other systems. The current version of the application is called LaBrea (get it...tar pit?) and can be found at http://www.hackbusters.net.

And finally…

- Many IIS Servers have finally been patched!
- Microsoft released a cumulative patch to make life easier for everyone with IIS.
- Microsoft is working with anti-virus companies on a strategy make patch dissemination easier.

Anti-Worm Checklist

Nothing can guarantee complete protection from worms, viruses and other security threats. But there are things that can be done to greatly reduce the level of exposure. I have developed this checklist of 12 items that can help reduce exposure to the most common Internet threats.

Know the SANS/FBI Top 20 List – Using the combined knowledge of dozens of leading security experts, this list was developed to identify the top twenty vulnerabilities that account for the majority of system compromises. The latest version of the list, "The Twenty Most Critical Internet Security Vulnerabilities" can be found at http://66.129.1.101/top20.htm.

Stay Informed – Monitor security-related web sites and subscribe to mailing lists so you are aware of the latest threats. You need to know about it before you hear it on the television or radio news. I recommend the following:

SANS Newsletters: There are two weekly Newsletters and one monthly to choose from. The Security Alert Consensus (CAS), for example, lets you customize the content to your environment. You can choose to only receive news that pertains to the operating systems that you use. To subscribe to the SANS Newsletters, visit http://server2.sans.org/sansnews.

SecurityFocus Mailing lists: There are currently 26 lists to choose from. At a minimum, I would recommend the 'bugtraq' mailing list. To subscribe, to the SecurtyFocus lists, visit http://www.securityfocus.com/cgi-bin/forums.pl.

Incidents.org: This site provides daily updates and alerts on the latest threats, acting as an Internet Storm Center. Here you can access the latest CID Graph, probing statistics, and access the DShield Database. DShield is a system that acts as a central logging and analysis repository for IDS and Firewall logs. For more information, visit http://www.incidents.org/.

Protect your e-mail systems – Use an e-mail content filtering solution to protect your mail systems from malicious activity. It's preferable that this be a separate system located in a DMZ network. This solution should, at a minimum, allow filtering based on text strings within messages as well as the ability to monitor and restrict attachments. This system should integrate with an anti-virus product so that all attachments are virus checked.

Triple Check your Firewall configuration – Your firewall is a critical component of your security strategy. It's configuration and rule-base should be closely guarded. On a regular basis, a 'reality check' should be performed by multiple security administrators or a security consultant. Verify that your rule-base includes a 'Stealth Rule'. This is a rule that prevents anyone from talking with your firewall directly. This is usually placed as early in the rule-base as possible. For example, a Stealth Rule for Checkpoint Firewall-1 would look like this:

SOURCE	DEST	SERVICES	ACTION	TRACK	TIME	INSTALL	COMMENTS

Any	Firewall	Any	Drop	Long	Any	Firewall	Stealth Rule

<u>Test for vulnerabilities</u> – Perform frequent vulnerability assessments or penetration tests to identify vulnerabilities that may exist on your systems. Tests should be performed with multiple assessment tools from the Internet as well as from inside the network. As part of this test, run a password-cracking tool to test the strength of user passwords.

<u>Educate your users</u> – Employees expect that the company will keep their personal information secure. It's your company's duty to exercise diligence to make sure that social security numbers, health information, and other personnel information is kept private. You should expect the same level of security from your employees when it comes to corporate data. Regular Security Awareness training provides the opportunity to make users aware of best practices, corporate policies, and to review some dos and don'ts. In addition, it serves as a forum to allow your employees to ask questions.

<u>Keep the Operating System and applications current</u> – Keep all OS and application revision levels up-to-date. Not only does this usually provide a more stable system, but it also makes it easier to deal with hot-fixes and security patches later. Have you ever tried to quickly install a security patch, only to find that you needed to install an Operating System Service Pack first?

<u>Monitor your network</u> – Review router, firewall, and host logs on a regular basis. If this task takes too much of your time, then consider an integrated log management and reporting tool. If you have a central SYSLOG logging server, be sure that it's kept up-to-date and secure. An Intrusion Detection System will let you see and correlate activities that will go unnoticed when using simple logging only. If you have an IDS system, be sure to keep signatures current. If there is no IDS in place, you should immediately start looking into something that will meet your requirements.

<u>Put Internet accessible hosts on a DMZ</u> – Any host that can be directly accessed from the Internet should be in a Demilitarized Zone network, not your internal LAN. For example, web, mail, and ftp servers should be on a separate network segment that is connected to the firewall only. Firewall rules are then created to allow access to these systems from the Internet, and to allow these servers to communicate with systems on your LAN.

Apply those Security Patches! – Lately this seems to be a daunting task, keeping up with vendor provided security patches and hot-fixes. Whatever it takes, it must be done. There are some tools available to help manage this process for large numbers of servers. Some OS vendors are in discussion with anti-virus companies to map out a strategy for simplifying patch dissemination and application. For now, it's primarily a manual process. You should be keeping current with security-specific patches for all hosts, firewalls, routers, and appliances on your network. Monitor security and vendor web sites and mailing lists for the latest news on security patches.

Anti-Virus Everywhere – All systems, from notebooks and desktops to mission-critical servers need to be protected from viruses. If you have Anti-Virus software running everywhere, make sure it is configured to properly protect your systems. It should be against corporate security policy for users to disable Anti-Virus protection, unless authorized to do so. Virus signatures should be updated often. In light of the recent barrage of worms (Code Red, Code Red 2, Nimda, etc.) it's a good idea to check for signature updates on a daily basis.

Get working on those Security Policies – If you have written policies that relate to corporate information systems, they probably need to be updated. Many things have changed in a short period of time. If your company has no written policies, first create an Information Security Roadmap that outlines the current risks and how they can be addressed. From this, security policies can be developed for the topics that are pertinent to the environment.

References

The SANS Institute, Figure 1 “CID Graph”, Consensus Intrusion Database Project, Oct 2001
URL: www.incidents.org

Netcraft, Figure 2, Figure 5, and statistics, “Netcraft Server Survey”,
URL: http://www.netcraft.com/survey , Oct 2001.

Aleph One, “Smashing the Stack for Fun and Profit”, Phrack49,
URL:http://destroy.net/machines/security/P49-14-Aleph-One

Microsoft Security Bulletin MS01-033, “Unchecked Buffer in Index Server ISAPI Extension Could Enable Web Server Compromise”, June 2001
URL: http://www.microsoft.com/TechNet/security/bulletin/ms01-033.asp , Sep 2001.

Riley Hassell, Ryan Permeh, eEye Digital Security, “All versions of Microsoft Internet Information Services Remote buffer overflow”, June 18, 2001, used with permission,
URL: http://www.eeye.com/html/Research/Advisories/AD20010618.html, Sep 2001.

Mitre Corporation, “CVE Common Vulnerabilities and Exposures (CVE)”, Version 20010918
URL: www.cve.mitre.org , Sep-Oct 2001.

Ryan Permeh and Marc Maiffret, eEye Digital Security, “ida Code Red worm”, July 17, 2001, used with permission, URL: http://www.eeye.com/html/Research/Advisories/AL20010717.html , Sep-Oct 2001.

Computer Economics, “Computer Economics Cyberquake Index”, Sep 26, 2001
http://www.computereconomics.com/cei/press/pr92101.html , Sep 2001.

Thomas Dolan, “MS IIS Vulnerability – Indexing Services Buffer Overflow”, SANS GCIH Practical, July 2001. URL: http://www.sans.org/y2k/practical/Tom_Dolan_GCIH.zip , Sep 2001.

David Moore – CAIDA, The Cooperative Association for Internet Data Analysis at San Diego Supercomputer Center, “CAIDA Analysis of Code Red”,
URL: http://www.caida.org/analysis/security/code-red/ , Sep 2001

Jeff Brown - University of California San Diego, David Moore - CAIDA at San Diego Supercomputer Center, “Animation of the geographic spread of Code Red”
URL: http://www.caida.org/analysis/security/code-red/coderedv2_analysis.xml#animations , Sep 2001.

Ryan Permeh and Marc Maiffret, eEye Digital Security, “Analysis of Code Red II”, used with permission,
URL: http://www.eeye.com/html/advisories/coderedII.zip , Oct 2001.

Microsoft Security Bulletin MS01-044, “Cumulative Patch for IIS”, Aug 15, 2001
http://www.microsoft.com/technet/security/bulletin/MS01-044.asp , Oct 2001.

Joris Evers, IDG News Service, Computerworld, “Having Failed to Patch Servers, Microsoft Hit by Code Red”, Aug 9, 2001. URL:
http://www.computerworld.com/storyba/0,4125,NAV47_STO62917,00.html

John Pescatore, Garner Research, “Nimda Worm Shows You Can’t Always Patch Fast Enough”, Sep 19, 2001. URL: http://www.gartner.com/DisplayDocument?doc_cd=101034 , Oct 2001.

Additional Reading

Microsoft Security Alert, "A Very Real and Present Threat to the Internet", Protecting yourself from the Code Red Worm, URL: http://www.microsoft.com/technet/security/topics/codealrt.asp , Oct 2001.

Crispin Cowan, Perry Wagle, Calton Pu, Steve Beattie, and Jonathan Walpole Department of Computer Science and Engineering. Oregon Graduate Institute of Science & Technology.
"Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade".
URL: http://www.cse.ogi.edu/DISC/projects/immunix/discex00.pdf, 1999.

Dildog, "The TAO of Windows Buffer Overflow",
URL:http://www.cultdeadcow.com/cDc_files/cDc-351/ , Sep 2001.

HSJ, "IIS5.0 .idq overrun remote exploit", June 28, 2001
URL: http://www.securiteam.com/exploits/5HP0N2A4KQ.html , Sep 2001.

Cisco, "Code Red Customer Impact", Cisco Security Advisory, July 20, 2001.
http://www.cisco.com/warp/public/707/cisco-code-red-worm-pub.shtml , Sep 2001.

CERT Coordination Center, Carnegie Mellon University, "Steps for Recovering from a UNIX or NT System Compromise", April 17 2000, URL: http://www.cert.org/tech_tips/root_compromise.html

SANS Institute, SANS/FBI Top 20 List, "The Twenty Most Critical Internet Security Vulnerabilities",
Oct 2, 2001. URL: http://66.129.1.101/top20.htm