Global Information Assurance Certification Paper

Code Red: We're all responsible
Caroline Scerri
June 31, 2001

Introduction

Over the past week you may have heard of a worm known as Code Red which exploits a vulnerability in IIS servers that has been known about for approximately a month now. A patch was released not long after the vulnerability was published and work arounds were available almost immediately. Each machine that was infected by this worm became a point for the worms exponential rate of self-propagation. It's very easy to blame whoever wrote the worm but all those IIS admins out there, you must know by now that you are running servers that are not only prefered targets but are typically insecure. Blame for spoofing and spam is beginning to rest with ISP's, so too should this kind of attack proliferation fall on lazy admins.

Code Red reportedly infected over 250,000 machines on the 19th alone, leaving those running the English version of IIS with a nice little ditty where their web pages used to be, the others it used as spring-boards to continue its infestation of the digital world. As the pièce de résistance the worm then blossomed into a DDoS focused square on the "www.whitehouse.gov" site. And on the "seventh day" the worm rested and beheld all that it had done. It has been reported that Code Red is time based and that between the 1st and 19th of next month the lifecycle of the worm ticks over to its awaken and propagate stage.

If you still haven't applied the patches or removed .ida mapping on your IIS server by the beginning of next month, then shame on you...

Through the eyes of a worm

Thanks to the people at eEye who worked to disassemble the Code Red worm and reveal what they found to the security community, I bring you the life of Red.

A copy of the code disassembled by Ryan Permeh and Marc Maiffret can be found at http://www.eeye.com/html/advisories/codered.zip.

First I would like to explain the vulnerability that this worm exploits. On the 18 June Microsoft published a vulnerability concerning an unchecked buffer in the Indexing Server ISAPI extension. By default IIS installs the idq and other ISAPI extensions, .dll's which provide "extended functionality", to provide support for administrative scripts and Internet Data Queries. The idq.dll contains an unchecked buffer in an area of code that handles input URLs. This vulnerabilty concerned all IIS servers running on NT 4.0, Windows 2000 and Windows XP platforms unless script mappings to .ida had been removed. The idq.dll runs in the System context, which effectively means the attacker gets complete control of the server. When the commands in a request for a .ida file are processed, the credentials check of the requesting user that is in place for idq.dll, are not performed as it is done before any indexing functionality is requested. You can bet that Microsoft acknowledged this was a serious vulnerability. However long it took between being notified by the vulnerabilities discoverer and its publishing, they provided a short term solution and a patch.

The advisory is available from www.microsoft.com/technet.

The discovery of this vulnerability was made by Riley Hassell of eEye. As explained by Riley this "buffer overflows in a wide character transformation operation". Which translates to mean that when the ASCII characters are taken in by the input buffer, one at a time, for conversion into char/unicode format. The buffer length is unchecked allowing for a malicious user to overwrite the EIP.

The code Red worm, which has been written in assembler code, uses a HTTP GET request as follows which causes the program flow to divert back to a point on the stack that jumps into the worm code held in the body of this request.

```
GET /default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
N%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u685
8%ucbd3%u7801%u9090%u9090%u8190%u00c3%u0003%u8b00%u531b%u53f
f%u0078%u0000%u00=a  HTTP/1.0
```

As explained by Riley, from this point the worm sets up some initial stack variables. The worm then sets up its internal jump table, which is a stack-based table used to store function addresses, allowing the worm to generate funtion addresses at run time.

After getting the addresses of the functions GetProcAddress and LoadLibraryA, which the worm can use to find all the other functions that lie between the addresses of these two, the worm loads the following function:

From kernel32.dll:
GetSystemTime
CreateThread
CreateFileA
Sleep
GetSystemDefaultLangID
VirtualProtect

From infocomm.dll:
TcpSockSend

From WS2_32.dll:
socket
connect
send
recv
closesocket

Documented at www.eeye.com.

The worm then stores the base address of w3svc.dll. If the IIS server is running the English version it will use this knowledge in the technique it uses to modify the sites web pages.

The worm then generates upto 100 instances of itself. The worm does this by counting the number of worm threads already in action. If there are less than 100 threads the worm creates a new thread which uses the same code base. If the thread count equals 100 the worm creates a thread whose function is to modify the web page if certain conditions are true.

Each thread of the worm replica first checks for the existance of c:\notworm. If this file already exist on the server then the worm does not attempt to go on to its infectious stage. Otherwise it goes onto check the UTC-time of the infected system. If the date is greater than UTC 20, meaning if its after or including the 20th day of the month for the local system, the worm will go into its DDoS stage. This part of the worm lifecycle involves creating a socket and connecting to 198.137.240.91:80 (www.whitehouse.gov/www1.whitehouse.gov). If the connection is made then the worm generates 18000h single byte SEND( )'s to www.whitehouse.gov. After 18000h SEND( )'s the worm takes a rest for about four and a half hours. After this time it will resume its assault on the whitehouse site.

The people at www.whitehouse.gov resolved the page request www2.whitehouse.gov(198.137.240.92) instead, assigning it a TTL of 872. Something that worm writters will undoubtedly learn from, as if the DDoS had been done by URL this fix would not have worked.

If the date is less than 20 UTC the worm goes into infectiouse mode, attempting to send itself to random IP addresses on port 80. On a successful completion of SEND, the worm closes the socket and goes back to the UTC check stage, repeating the time check, sleep/hack web page or infect loop indefinately.

The worm uses "hooking" to modify the web pages of infected systems that it detects as running the English version of IIS.

Hooking is a technique which describes the modification of code in memory to point to code that the worm has provided. Code Red modifies a w3svc.dll function called TcpSockSend, which is what w3svc.dll uses to send information back to the client. It searches out and replaces the address for TcpSockSend in the import table to point to an address within the worm, FAKE_TCPSOCKSEND.

Code Red then sleeps for ten hours, during which time all Web requests will return a page made up of the following code:

```
< html >< head >< meta http-equiv="Content-Type" content="text/html; charset=English" >< title >HELLO!
< 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>
```

Which has when written without the additional spacing within the tags, gets displayed as:

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

Hacked By Chinese!

Note the cute spelling for body.

The success of this worm is wholly dependant on the IIS admins failure to apply a fix. There was some discussion on BUGTRAQ that the advisory from Microsoft did not make clear that if you weren't running the Indexing Service you still had to apply the patch. What was stated on the advisory is that unless you selected not to install the Indexing Service at IIS initial installation you were vulnerable to this exploit and should apply the fix. If you weren't sure about whether you'd done it, you should perhaps incline towards the side of caution. Don't assume you selected not to install this feature that is set to install by default. Something really lovely I heard an IIS admin say one day was that he didn't want to apply the patch because Microsoft patches sometimes make other things fall over and he didn't have a testbed to check it out on. In this case I would have thought a quick risk assessment would say atleast remove the mappings to .ida and .idq.

Because of the verility of the worms self propagation, like a virus its affect on the web was an exponential rate of infection.

CAIDA observed the activity of Code Red based on packets seen going to non-existant addresses. A graphical representation of the worms activity can be seen at the www.caida.org. Another very interesting animated representation of the spread of code red is also available at www.caida.org.

Son of Red

There were a number of postings on BUGTRAQ in reference to another version of Code Red which appeared to generate truly random IP addresses.

To date the idea of a next generation Code Red worm is hypothetical, however in theory it has been backed up by a number of security professionals.

The original Code Red, according to Stuart Staniford, has what appears to be a bug; its random number generation. Each instance of Code Red, as stated earlier is comprised of the same code and the random number generator is initialised with a fixed seed. Which means that all threads of the worm attempt to compromise exactly the same sequence of IP addresses.

There appeared in the logs of several people what would seem to indicate a varient of the worm which was attacking truly random hosts. The following table comes from Ken Eichmann of cas.org. This data was collected by Ken on the 19th June, it shows non-legitimate HTTP SYN scans targeting his class-B address space. As seen in BUGTRAQ.

# Hour	# Code Red Worm Scans	Addresses Scanning	Addresses Being Scanned
00	12699	2450	562
01	13059	2577	562
02	13272	2590	541
03	13056	2564	525
04	13283	2632	507
05	13229	2612	502
06	13554	2601	468
07	13517	2608	506
08	13746	2685	612
09	16819	3325	1724
10	36589	7838	8338
11	116083	26823	28462
12	295348	68085	51459
13	466542	103522	59699
14	520973	113451	60881
15	513513	115124	60814
16	513894	90931	60900
17	499642	111175	60469
18	480850	106215	59987

19	449712	97699	58908
20	26687	7319	8507
21	9197	2181	3046
22	7782	1814	2570
23	7056	1648	2343

This table shows the number of unique ip addresses being targeted holding steady for the first few hours, then a significant rise in addresses being attacked suggests a variation of the worms behaviour, which is not evident in the original Code Red worm as posted by eEye thus leading to the conclusion that a next generation Code Red may be out there.

Detection and Removal

How do you know if you are being attacked?

If you are receiving GET request on port 80 with the following sigature you have been attacked, note this does not mean you have been infected. Code Red send these GET requests to any machine that it achieves a socket connection with on port 80.

```
GET /default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
N%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u685
8%ucbd3%u7801%u9090%u9090%u8190%u00c3%u0003%u8b00%u531b%u53f
f%u0078%u0000%u00=a  HTTP/1.0
```

A copy of the ISAPI overflow attack signatures as seen from some popular IDSs can be viewed at www.whitehats.com.

How do you know if you are vulnerable?

There have been several ways of detecting whether your server is vulnerable to this worm posted on BUGTRAQ.

The Microsoft patch fixes the buffer overflow problem with the Indexing Service dll's, it does not remove script mapping to .idq and .ida. If you are unsure of whether or not you applied this patch back in June you should check that the file date of the idq.dll is 5/24/2001. This means your server is probably .patched.

Microsoft suggest preforming the following to check the patch has been properly installed.

For Windowns NT 4.0:

- Confirm the following registry key has been created:

  HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Hotfix\Q300972.

For Windows 2000:

- Confirm the following registry key has been created:

  HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Updates\Windows 2000\SP3\Q300972

- Verify the individual files by checking their date/time and version information which is provided in the following registry key:

  HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Updates\Windows 2000\SP3\Q300972\Filelist

Otherwise, if you want to see that you have successfully removed the script mapping to .ida you can test how your server responds to getting sent the malformed GET request, if mapping is disabled your server should respond with a 404 error code.

Or you can try the following as suggested by CIAC:

Enter the following URL into your web browser, replacing the www.yourservername.com with the name of your server.

http://www.yourwebservername.com/default.ida

If it returns 'The IDQ file default could not be found.' then you still have .ida enabled.

How do you remove this vulnerability?

If you are more concerned with getting this buffer overflow vulnerability out of the way, or have a testbed to make sure it won't wreck your IIS setup, the best solution is to apply the patch.

For IIS servers running on NT 4.0 apply the following patch:

http://www.microsoft.com/Downloads/Release.asp?ReleaseID=30833

For IIS servers running on Windows 2000 Professional, Server and Advanced Server apply this patch:

http://www.microsoft.com/Downloads/Release.asp?ReleaseID=30800

For IIS running on Windows 2000 Datacenter Server Microsoft advised that the patches were hardware-specific and that you should get them off your equipment manufacturer.

For IIS running on Windows XP beta Microsoft advised that the vulnerability would be eliminated in the next beta update as well as in the final version of XP.

To remove mappings to .ida and .dq to avoid the buffer overflow being exploited follow these steps.

- Open Internet Services Manager.
- Right-click the Web server | Properties | Master Properties | WWW Service | Edit | HomeDirectory | Configuration and remove these references:

Internet Data Query	.idq
Internet Data Administration	.ida

If everyone who runs an IIS server patches their servers there will not be another Code Red episode, otherwise it is bad news for the next site its authors wish to target.

Going out on a limb here, but rather than being political terrorism, this worm seems more to be a good wake-up message for people who run IIS and that don't respond to vulnerability notifications.

Conclusion: Lessons Learnt

The Code Red worm found its way around the globe with frightening ease, what has been termed CRV2, the next generation Code Red, will be a basket-full of pain for the world wide web, unless IIS administrators unite against it which is sadly a simple matter of applying a patch. Sad because its so simple but seemingly such a big ask.

The lowdest and repeated message that I hope people have heard from the existance of this type of worm is "Patch Your IIS!" and do it before the 31st of July.

References:

Microsoft Security Bulletin MS01-033 "Unchecked Buffer in Index Server ISAPI Extension Could Enable Web Server Compromise", V1.1 (June 18, 2001) URL:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-033.asp

Discovery: Riley Hassell, Exploit: Ryan Permeh "All versions of Microsoft Int ernet Information Services Remote buffer overflow", June 18, 2001 URL:
http://www.eeye.com/html/Research/Advisories/AD20010618.html

CIAC "L-117: The Code Red Worm", July 19,2001 URL:
http://www.ciac.org/ciac/bulletins/l-117.shtml

Ryan Permeh and Marc Maiffret "ida "Code Red" Worm", July 17, 2001 URL:
http://www.eeye.com/html/Research/Advisories/AL20010717.html

Stuart Staniford "Analysis of spread of the Code Red worm", URL:
http://www.silicondefense.com/cr/

David Moore "The Spread of the Code-Red Worm (CRv2)", July 26, 2001 URL:
http://www.caida.org/analysis/security/code-red/

BUGTRAQ

http://www.securityfocus.com/archive/1/198347 CodeRed: the next generation, Marc Maiffret

http://www.securityfocus.com/archive/1/198357 Re: CodeRed: the next generatio n, Stuart Staniford

http://www.securityfocus.com/archive/1/198340 Re(2): Re(2): 'Code Red' does n ot seem to be scanning for IIS, Ken Eichman