# Global Information Assurance Certification Paper

# Code Red II Analysis
## Version 1.5c GCIH Practical Option 2

**Heather Bard**

## Executive Summary

The Code Red worm and initial variants started off as a relatively harmless worm that just propagated a worm that performed a flood search on the 198.137.240.91 address. Code Red II and it's variants became more malicious by inserting backdoors using the relative shell path vulnerability to replace the explorer.exe program which causes compromised systems to require reformats instead of just cleaning the files that were originally affected to ensure that systems are clean. Code Red II variant d goes one step further by using %u encoding to bypass IDSs so that it can continue to attack after some of the initial updates to perimeter Network Intrusion Detection Systems (NIDS) and other IDS systems.

Code Red II d is discussed in detail here as it is a worm that exploits three well-known Microsoft vulnerabilities instead of just one and is more malicious and destructive than Code Red I. With this variant, as well as with earlier ones, they have all used vulnerabilities that had just recently been published on well known security focus web sites. This trend is a positive trend in that it appears that the "good guys" are starting to get enough of a network that they are starting to catch deficiencies in software and networking components either prior to or at the same time as the "bad guys" instead of having to analyze and respond to attacks.

The next step for the e-community is to be able to effectively and efficiently respond to and protect against these vulnerabilities prior to hackers being able to use these deficiencies. In this vein, most of the research that I have done on the Code Red variants, and other exploits, discusses exploits in very technical detail. This information is great for those that are advanced systems administrators. However, discussions with many individuals who have home computers, and even small home networks, shows that they usually don't understand the technical jargon and are somewhat intimidated by the discussions about exploits and the fixes for them. Most of these individuals have approached me wondering what they can do to not be so vulnerable, yet to make sure that they do not "mess up" their systems. I am hoping that my discussion below will give these individuals the information that they need in order to become more educated, as well as fix their systems, and at the same time making reference to much more detailed, technical papers that are available for those that want to roll-up their sleeves and jump into the meat of Code Red.

### Malicious Code and it's Exploit Details

Name of the Code: Code Red IId uses the following exploits
%u encoding IDS bypass vulnerability
Remote Buffer Overflow vulnerability (.ida)
Relative Shell path vulnerability

Variants:
CRv1, CRv2a and CRv2b, Code Red II

Operating System:
Windows NT 4.0 or Windows 2000, Cisco running IIS
Cisco Secure Intrusion Detection System (NetRanger Sensor)
Cisco Catalyst 6000 IDS Module
ISS RS Network Sensor 5.x, 6.x before XPU 3.2
ISS RS Server Sensor 5.5, 6.x before 6.0.1
Dragon Server 4.x
Snort before 1.8.1
Others may be affected

Protocols/Services:
MS IIS 4.0 or 5.0 webserver
Intrusion Detection System's (IDS) attack signatures

Brief Description: MS IIS implementation of non-standard %u encoding technique allows bypass of most IDSs as most commercial and freeware ID Systems need to break down (decode) http packets in order to analyze attack strings. This exploit allows the CodeRed worm (and potentially other worms) to propagte via encoding a scan in the first 19 days of any month. Then provide a means to pass an attack via encoding, such as replacing explorer.exe and opening backdoors (as CodeRed does). The Code Red II variant d worm initially uses %u encoding to bypass IDSs, then uses .ida buffer overflow injection to gain system level access, next it uses the relative shell path vulnerability to inject the Trojan explorer to gain a backdoor, and can cause ARP flooding DOS attacks during propagation.

**Service and Protocol Description**

There are three services that are affected by Code Red II vd. These are MS IIS that has been exploited in all of the Code Red variants. MS Explorer which was not affected in Code Red I variants, but was replaced in Code Red II variants as a Trojan. Finally, the latest service that was affected by Code Red II vd is the bypassing the IDSs. All will be discussed briefly here in order for the reader to understand these services and be able to make a more knowledgeable decision about the need for these and how to protect themselves.

Internet Information Server (IIS) is an integrated webserver application that Microsoft delivers with MS applications. Users may not be aware that their systems are even running this application as it is integrated into MS products and run in the background in many applications. IIS is an information transfer service that enables remote clients to get information from servers. When servers and clients are establishing connections they have to pass administrative information to each other in order to ensure information flows correctly. For those unfamiliar with this computer interchange, this connection information can be seen in the items that follow the http string when connecting to a web site, and the reason that a website that you may have typed in such as http://www.microsoft.com can end up being a very long string with % and ? in the string.

IIS uses a non-standard encoding technique, the language that computer applications use to talk to each other, of putting a %u in its URL to deliver payload between client and server. (eEye Research) MS IIS Web Server must gain information from another ms service, MS Indexing Services, through a filter, ISAPI (Indexing Service Application Protocol Interface), which has an .ida extension in MS files. IIS also uses Internet Data Query (.idq) drivers to gain information on web services. Neither of the above indexing functionalities needs to be running in order for a worm to gain access, as long as the mappings are present.

Intrusion Detection Systems sit on your network and or your hosts and observe and analyze your network traffic and traffic into and out of your host. In order for signature-based IDSs to function properly they must have the ability to decode various forms of HTTP encoded requests. In other words, they must be able to read the packets flowing past them and be able to understand the administrative data in them. Signature-based IDSs (the most common and oldest found on the market, such as Intrusion Detection Systems ISS) have to be preprogrammed with strings of characters to look for that signify an attack and thus are programmed with "standard" strings (very similar to virus detection software).

Where the MS IIS and the IDS services are intertwined in the Code Red worm is that MS IIS implemented a %u encoding in order to represent true Unicode/wide character strings where standard HTTP requests are encoded with UTF (%xx%xx) or hex (%xx) and thus this is what most IDS systems are programmed for.

Explorer is an integrated file search and server application that Microsoft delivers with MS applications. When a user uses this program, the program takes the request and then functions in looser restriction set then the average user operates at in order to find all the information that the user wants and has access to.

**Description of variants**

CRv1, appeared July 11, 2001, is the initial virus that looks for systems running IIS that have not patched the unchecked buffer vulnerability in .ida or removed the ISAPI script mappings. It propagates in a fairly pre-deterministic method trying 600 IP subnets for Chinese language servers and 300 paths for all others; it defaces some websites with "Hacked by Chinese", then floods, and sleeps. Uses .ida remote buffer overflow vulnerability.

CRv2a has no web defacement and has random target selection. It uses a fixed, static seed for its random number generator that causes it to output the same sequence of numbers each time the generator is invoked, although the numbers themselves are random in that they have no predictable relationship to each other. Uses .ida remote buffer overflow vulnerability.

CRv2b has no web defacement and has optimum per thread random target selection. It uses a random seed that uses an unpredictable starting point, so it generates a random sequence of random numbers, rather than a predictable series of random numbers**.** Uses .ida remote buffer overflow vulnerability.

Code Red II version c (W32/CodeRed.c.worm) This is actually a new worm. It does not deface web pages or contain a DDoS payload. It uses the atom "CodeRedII" for self-recognition and thus does not reinfect already infected CodeRedII systems. It can however re-infect those already infected with Code Red or its variant. It also opens back doors into systems by placing a Trojan on the system. Only runs on 2000 servers, but will crash NT servers. Also, disables cisco routers running web server application. Uses .ida remote buffer overflow vulnerability and the Relative Shell path vulnerability.

Code Red II version d (W32/CodeRed.d.worm). Uses %u encoding to pass the same worm as seen in Code Red II while bypassing unpatched IDS servers that do not look for this particular encoding. Uses .ida remote buffer overflow vulnerability, the Relative Shell path vulnerability, and the %u encoding vulnerability.

**How the exploit works**

The exploit occurs in three phases - infection, propagation, and trojanization. This discussion will start with a server becoming infected and complete the sequence. I will discuss the use of the buffer overflow vulnerability found in all versions of CodeRed; however, in order to discuss the more general description of the entire worm and it's many intricacies I will refer the reader to a very detailed description of the Remote buffer overflow vulnerability completed by eEye Digital Security at http://eeye.com/html/Research/Advisories/AD20010618.html. The following process comes mainly from the eEye Digital Security Research Advisory AL20010804, with modifications to clarify the process.

Infection
==========

The first thing the worm does when it gets to a host is setup a jump table so that it can get to all of its needed functions. The worm then proceeds to get its local IP address. This is later used to deal with subnet masks (propagation) and to make sure that the worm does not reinfect the local system. Next, the worm gets the local System Language to see if the local system is running Chinese (Taiwanese) or Chinese (PRC). At this point the worm checks if it has executed before, and if so, then the worm will proceed to the propagation phase. (See the propagation section). Next, the worm will check to see if a CodeRedII atom has been placed (GlobalFindAtomA). This functionality allows the worm to make sure not to re-infect the local machine. If it sees that the atom exists then it sleeps forever. The worm will add a CodeRedII atom. This is to allow the worm the functionality to check to see if a system has already been infected with the worm.

The worm now sets its number of threads to 300 for non-Chinese systems. If the system is Chinese then it sets it to 600. At this point the worm spawns a thread and begins the infection phase again. The worm will spawn threads according to the number set. Each new thread will be a propagation thread. This is where the worm calls the trojan functionality. You can find an analysis of the trojan mechanism down below in the Trojan System section. The worm then sleeps for 1 day if the local system is not Chinese, 2 days if it is. The worm then forces the system to reboot Windows so that the Trojaned explorer will become functional and remove the worm from memory.

Propagation
==========

The second phase of the worm is used to spread the worm further through propagation. The worm sets up the local IP_STORAGE variable. This is used for worm propagation functionality and to make sure not to re-infect the local system. The worm then sleeps for 64 milliseconds. The worm then gets local system time. The worm checks to see if the year is less than 2002 and if the month is less than 10. If the date is beyond either of those, then the worm reboots the local system. That basically limits the worm to 10/02 for its spreading (In a perfect world in which everyone's system is on the correct date/time). The worm then sets up SockAddr_In, by performing a GET_IP routine discussed here. The worm generates the IP address for the next host to connect to by doing the following…

```
call   GET_OCTET      ; load 4th octet (this is in reverse order due to byte ordering)
mov    bh, al
call   GET_OCTET      ; get 3rd octet
mov    bl, al
shl    ebx, 10h        ; shift bx to the top of ebx
call   GET_OCTET      ; get 2nd octet
mov    bh, al
call   GET_OCTET      ; 1st
mov    bl, al
call   GEN_OCTET      ; get first octet
and    eax, 7          ; and it by 7
call   CHECK_ADDR_MASK ; ecx has eip
```

For each octet, the worm generates a psuedo random byte between 1 and 254, then gets a random octet between 1 and 254 and mask it by 7. Finally, it uses this last byte to generate a 1st octet.

The most pertinent bit is CHECK_ADDR_MASK

this specifies the following:

```
dd 0FFFFFFFh        ; 0 - addr masks
dd 0FFFFFF00h       ; 1
```

```
dd 0FFFFFF00h        ; 2
dd 0FFFFFF00h        ; 3
dd 0FFFFFF00h        ; 4
dd 0FFFF0000h        ; 5
dd 0FFFF0000h        ; 6
dd 0FFFF0000h        ; 7
```

This mask is applied to the local systems IP address, and matched to the generated IP Address. This makes a new ip with 0,1 or 2 bytes of data with the local ip.

For instance, the worm will 1/8th of the time generate a random IP not within any ranges of the local IP Address, 0.0.0.0. Half of the time, it will stay within the same class A range of the local IP Address, 255.0.0.0, and 3/8th of the time, it will stay within the same class B range of the local IP Address 255.255.0.0.

Also note that if the IP the worm generates is 127.x.x.x, 224.x.x.x, or the same as the local systems IP address then the worm will skip that IP address and generate a new IP address to try to infect.

It fills up the masked space with the hosts computer's IP information instead of the attackers. On an infected server the exploit scans for vulnerable systems, using the IIS %u encoding IDS bypass vulnerability to mask the injection vector of the .ida vulnerability, on port 80/tcp or a non-blocking socket then it will create the socket and will connect non-blocking (max. 10 sec) to the ip address (no more SYN+ACK faking and slowing down). The worm does this by setting up the socket for transmission. This performs a Socket(), stores the handle, then makes it a non-blocking socket (this is important for speed dealing with connect() calls). The worm then connects to the remote host – using %u encoding, if it returns a connect right away the worm sets the socket to Blocking. This is so selection isn't required after the connection is made. If a connect is not immediately sent it performs a select to get the handle. If no handle is returned, then the socket is closed and the worm tries to propagate again. If a handle is returned then the worm uses the %u encoding to send a copy of the worm. The worm then does a receive which is not actually used anywhere. It then closes the socket and propagates again.

Trojan System
==========

The final action of the worm is performed in order to place a backdoor on the infected system. This portion of the worm is designed to dump root.exe (root.exe is cmd.exe) into msadc and scripts, and create a trojan on the local drive. First the worm, gets the System directory, this gets the native system directory (ie, c:\winnt\system32). It then appends cmd to the system directory string (c:\winnt\system32\cmd.exe). Next it sets the drive modifier to c: and copies cmd.exe to /scripts/root.exe (Actual path: Drivemodifier:\inetpub\scripts\root.exe). Then it copies cmd.exe to /msadc/root.exe (Actual Path: DriveModifier:\progra~1\common~1\system\MSADC\root.exe). Next it

initializes the area for explorer.exe and creates the Drive /explorer.exe (drive is c, then d). The worm now writes out explorer.exe. There is an embedded binary within the worm that will be written out to explorer.exe. It has the property that if an embedded byte is 0xFC, it gets replaced by 20h 0x00 bytes instead of the regular byte. For more on what the trojan explorer.exe binary does then goto the Explorer.exe Trojan section. Also the way NT works is that when a user logs into the local system it has to load explorer.exe (desktop, task bar etc...) however NT looks for explorer.exe first in the main drive path c:\ which means the trojan explorer.exe is going to be loaded the next time a user logs in therefore keeping the system trojaned over and over and over. Finally, the worm closes explorer.exe and changes the drive modifier to D, then the worm goes back to the original code. After it is done then it goes back to the infection process.

Explorer.exe Trojan
==========

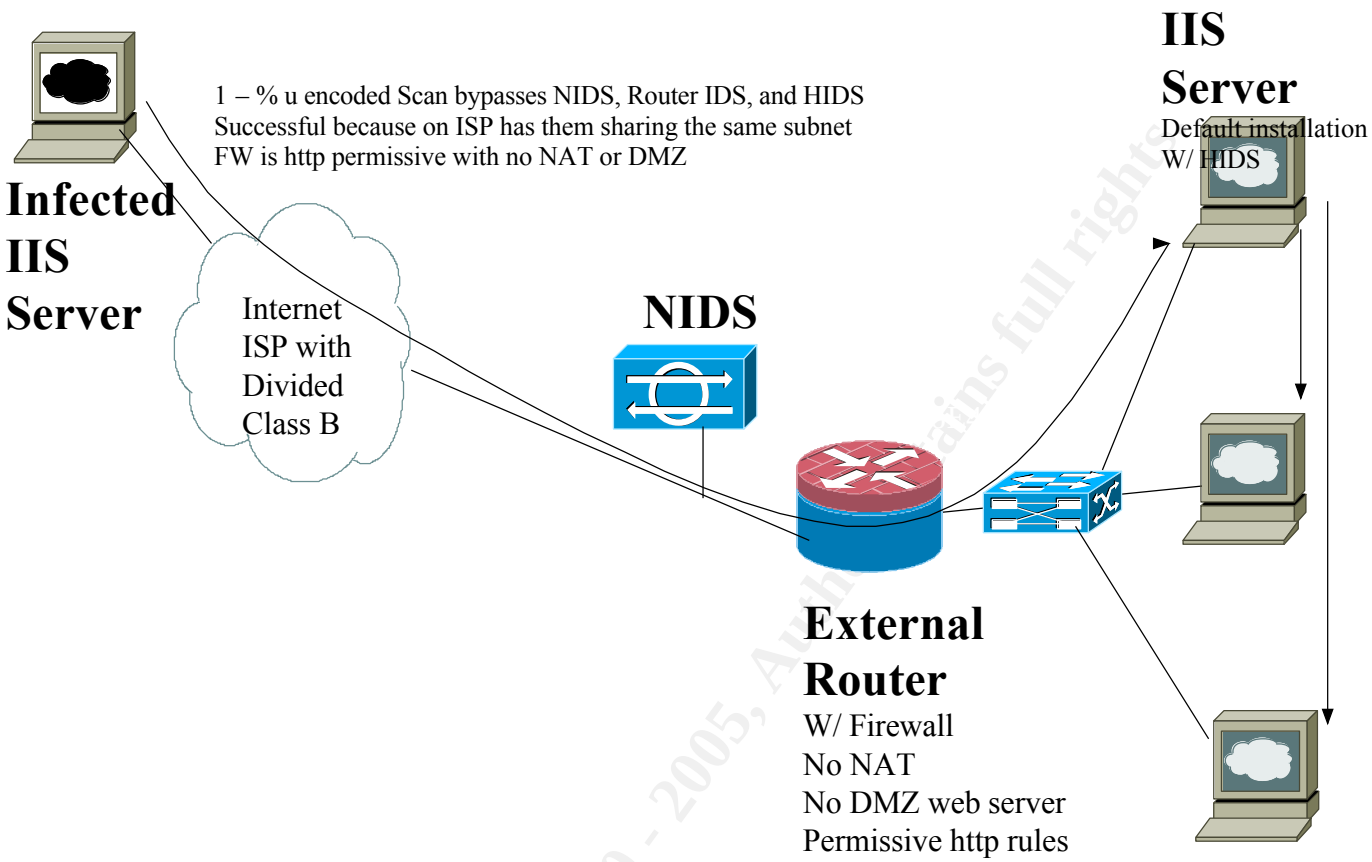
The following is a short description of the explorer.exe Trojan. The Trojan gets the local systems windows directory. It then executes explorer.exe from within the local systems windows directory. The worm now goes into the following loop:

set SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SFCDisable to 0FFFFFF9Dh, which basically disables system file protection.
set SYSTEM\CurrentControlSet\Services\W3SVC\Parameters\Virtual Roots\Scripts to ,,217
set SYSTEM\CurrentControlSet\Services\W3SVC\Parameters\Virtual Roots\msadc to ,,217
Set SYSTEM\CurrentControlSet\Services\W3SVC\Parameters\Virtual Roots\c to c:\,,217
Set SYSTEM\CurrentControlSet\Services\W3SVC\Parameters\Virtual Roots\d to d:\,,217
sleep for 10 minutes

Basically the above code creates a virtual web path (/c and /d) which maps /c to c:\ and /d to d:\. The writer of this worm has put in this functionality to allow for a backdoor to be placed on the system so even if you remove the root.exe (cmd.exe prompt) from your /scripts folder an attacker can still use the /c and /d virtual roots to compromise your system. As long as the trojan explorer.exe is running then an attacker will be able to remotely access your server.

These changes allow read/write access to the paths associated with these values and allow a remote attacker to carry out shell function on the local system by sending commands to it via URL. The "backdoor" will then cycle. With the completion of the cycle servers that have been rebooted will give full access to both c:\ and d:\ and to the script directories. As cmd.exe was copied to these directories by the worm before, the system can easily be compromised.

**Diagram**




This diagram is used to give a visual portrayal of the network with infected hosts and the IDS and vulnerable IIS Servers.

**How to use the Exploit**

In order to use the Code Red worm a hacker simply needs to get a copy of the executable code and place it on a system that will then be allowed to propagate through the Internet.

Manually exploiting the %u encoding vulnerability simply requires a "hacker" to insert the command that they wish to have by passed into the http string that they are sending to the IIS server that they wish to exploit.

Ravila White does a nice job describing how to use the other features of the Code Red v1 and v2 worms in her paper posted on the http://www.sans.org/giatc/cert website entitled "Revisiting the Code Red Worm"

**Signature of the Attack**

There are multiple ways in which this exploit and the Code Red Worm attacks can be spotted. Of course the best technique is to take a suspected host off of the Internet in

order to do some indepth analysis without potentially infecting other servers. First the user can look at a suspected host to see if any of the following files can be found.

C:\ inetpub\scripts\root.exe
C:\progra~1\common~1\system\MSADC\root.exe
D:\inetpub\scripts\root.exe
D:\progra~1\common~1\system\MSADC\root.exe
The file explorer.exe exists in either the c:\ or d:\ directory

Next the user or administrator of the IIS server can look in the IIS log files to find the following strings (From Cert Incident notice IN-2001-09).

/default.ida?XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX%u9090%u6858%ucbd3%u7801
%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090%u819
0%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a

The presence of this string in a log file does not necessarily indicate that a compromise has occured, it only implies that a "Code Red II" worm attempted to infect the machine. This differs from Code Red I in that I uses Ns instead of Xs as its filter character (Aris predictor, Code Red II Incident Analysis Alert).

The administrator can look at their IDS logs to see if they have any instances of this .ida signature, as well as searching to see if they have any cmd.exe stings or root.exe strings in the http requests.

The attacks would basically look like:

http://IpAddress/c/inetpub/scripts/root.exe?/c+dir (if root.exe was still there) or:

http://IpAddress/c/winnt/system32/cmd.exe?/c+dir Where dir could be any command an attacker would want to execute.

If the systems are still on the network, or an administrator wants to scan his network without interrupting service for all of her users. The user can place a sniffer on their network to see if a host is running an active instance of the "Code Red II" worm. The sniffer will show the host scanning random IP addresses on port 80/TCP looking for other hosts to infect. The IP addresses scanned by the "Code Red II" worm are determined in a probabilistic manner thus one can see the pattern in the scanning:

- There is a **one in two** chance that a given thread will scan random IP addresses with the same first byte as the infected host.
- There is a **three in eight** chance that a given thread will scan random IP addresses with the same first two bytes as the infected host.
- There is a **one in eight** chance that a given thread will scan random IP addresses.

**How to protect against the Code Red II worm**

To protect against the Code Red II worm in general and specifically against the specific vulnerability requires multiple steps by the user and the network administrator in order to ensure defense in depth in the event of a particular security failure.

For a user/system administer for end systems:

1) "Harden" all systems. Anytime you go to harden a system you must be fully aware of all the services that your system(s) are using. (see http://www.microsoft.com Technet Home – IT Solutions – Security – Bulletins - MS 00-52. MS 01-34. and MS 01-44)
   a. Remove IIS from your system if it is not needed
   b. Remove all unused ISAPI extension mappings
   c. Apply the patches available from Microsoft specifically best to apply the SP6a Security Rollup Package (SRP).
   d. Use the IIS Lockdown Tool – be aware that when using this it can disable other services that you may need open.
   e. Turn off interactive logins if not absolutely necessary
2) Update your virus definitions, as there is now a known signature for this.
3) Use windows file protection features (wfpf) and the system file checker (sfc.exe) tools.
4) Use strong authentication mechanisms that prevent unauthorized users from making system level changes.
5) For end users that are using an Internet Service Provider (ISP) for service, such as AOL, call/e-mail your ISP and ask them if they are updating their servers and are following the advice below.
6) For those users with cable modem type connection, it is best to keep your computer turned off or better yet, disconnected, when you are not using it. The less the amount of exposure, the less likely you are to be infected.

From a network perspective:

1) Disable web servers on routers
2) Ensure your IDS servers have the updated signatures for the %u and variants.
3) Setup a DMZ or screened network Web Server which is the only server that all external users can touch – therefore only providing one target of opportunity that the worm can find and that you MUST harden.

4) Use NAT or proxy firewalls so that the worm can not penetrate your network.
5) Employ a traffic analysis IDS, instead of and/or along with a signature IDS.

Can stop Code Red and other worms at the Internet gateway in a variety of ways. One example of this can be found at http://www.mcafeeb2b.com/products/webshield-eapp500/codered.asp.

**How User fix**

Apply the patches to the system and reboot the system. However, this is no longer sufficient with CodeRed II the best way to ensure a clean system is to reformat and reinstall all software – hardening the system as you do the initial installs and setup.

The user can use a tool that Microsoft has developed to eliminate the obvious affects of Code Red II. The tool can be obtained at http://www.microsoft.com/Downloads/Release.asp?ReleaseID=31878 or See the Anti-CodeRedII file attached to check systems.


AntiCodeRed2.vbs

**How Software vendor fix**

In most general terms the vendor can prevent worms and other malicious code by having all code that is written passed through quality control to ensure that all proper checksum s and other "security" procedures are implemented into software.

Specifically, Microsoft needs to fix the buffer overflow issue – for which they have provided a patch.

All companies should use standards in their protocols, such as hex encoding instead of non-standard techniques. Standards, similar to open code software, are reviewed by a large group of professional developers so that there is a better potential to ensure that security becomes incorporated into the standards. Any company that uses proprietary techniques/protocols should be responsible for ensuring that security services have the information necessary to protect the end users.

**Source code/Pseudo Code**

A very good location to find the source code for Code Red II is found at: http://www.eeye.com/html/advisories/coderedII.zip

On page 26 of her paper on http://www.sans.org/giatc/cert website entitled "Revisiting the

Code Red Worm", Ravalia White provides an excellent description of the pseudo code available for Code Red II.

**Additional Information**

The attached zip file contains Corecode's disassembly of the CodeRed II virus, the Code Red virus, the ida_root vulnerability, and the explorer Trojan.


coderedII.zip

## Conclusion

Code Red is a worm that should have been a wakeup to not only all system administrators, but to individual users, that systems administration and the continual updates of patches should not only be a requirement in any security policy but also are a necessity. However, the word either did not get out or the users/administrators failed to heed the warnings of the known vulnerabilities and their potential to do serious damage to systems and the network in general.

Code Red II was the "I told you so" in that it went a step further and actually installed a backdoor into all unpatched systems. This worm was also a wake up call to Intrusion Detection System providers that they need to be more flexible in their signature analysis and more forward looking into possible variants to known vulnerabilities.

**References**


http://www.incidents.org/react/code_red.php
http://www.incidents.org/react/code_redII.php
http://www.eeye.com/html/Research/Advisories/AL20010804.html
http://www.eeye.com/html/Research/Advisories/AD20010705.html
ida_root analysis by corecode.htm
McAfee Avert **W32/CodeRed.c.worm**
http://www.cert.org/incident_notes/IN-2001-09.html
http://aris.securityfocus.com/alerts/codered2/
Cisco Security Advisory & Cisco Secure IDS Signature Obfuscation Vulnerability
Security Wire Digest Volume # 68 Sept 6 2001
http://www.eeye.com/html/Research/Advisories/AD20010618.html
http://www.securityfocus.com/bid/2880 - buffer overflow vulnerability
http://www.caida.org/analysis/security/code-red
http://www.microsoft.com/support/kb/articles/Q222/1/93.asp