INCIDENT RESPONSE

A study of an actual Nimda Worm infection on an enterprise scale

by

Jeff Neithercutt

A paper submitted in partial fulfillment of the requirements for the certificate of

GCIH

From the SANS Organization

2002

TABLE OF CONTENTS

LIST OF FIGURES

Part I: The Exploit: Background of the exploit

NIMDA, ORIGIN OF THE SPECIES

The Nimda worm, also known as W32/Nimda@MM, I-Worm Nimda (AVP), w32.Nimda.A@mm (NAV), W32/Nimda.eml, Mimda, and Win32.Nimda.A@MM(Mcafee), was first reported "in the wild" on September 18, 2001. It navigates the networks of the world via four distinctly different methods, but is primarily focused on infecting computers using Microsoft's IIS Web Server software. It also infected many users of the Internet after they opened an infected attachment that came with an email, usually from an infected friend of theirs, which prompted them to trust the email and it's malware attachment. Fortunately, Nimda's focus was on spreading, rather than on data alteration or destruction, so other than slowing email and network servers to a crawl to deal with the infected traffic, and costing millions in man-hours to rebuild the infected machines, it had minimal effect to actual data integrity. Some networks experienced a service interruption, commonly known as a Denial of Service, as a result of the excess infected traffic. Nimda is Operating System specific, and does not affect Macintosh computers or machines running variations of the Unix operating system, other than to cause occasional unusual activity in certain print servers and router/switch devices with a specific type of software, primarily from HP and Cisco. There were reports from some customers that when Nimda entered their enterprise, some of their HP printers using the Jet Direct software from HP failed, and had to have their BIOS software reinstalled. On the Cisco switches and routers that were affected, various strange behaviors were reported, but nothing was experienced within our enterprise specifically.

Name:

The name Nimda could be derived from the word Admin spelled backwards, as kind of a snide way of rubbing in the unlimited access enjoyed by the worm once an infection occurred. But who knows what the twisted author of this particular malware had in mind when naming it. It is also speculated that it might have been a reference to the Admin.dll file where the Virus starts its run on an infected machine. Nimda was reported to the CVE, Common Vulnerability and Exposures index, and it was assigned the reference number CA-2001-26

Operating System:

Microsoft Windows operating systems that were vulnerable to the NetBIOS shares exploit and other methods of infection included Windows 95, 98, 98SE, ME, NT, and 2000. Additionally, the Microsoft Personal Web Server versions 1.0 and 3.0 were vulnerable as well.

Protocols/Services/Applications:

Any Microsoft IIS Server version 3.0, 4.09, or 5.0 that was not patched with the service packs released when the vulnerability the worm exploits was first announced.

TCP/IP (Transmission Control Protocol/Internet Protocol)
UDP (User Datagram Protocol)
HTTP (HyperText Transfer Protocol)
SMTP (Simple Mail Transfer Protocol)
MAPI (Message Application Program Interface)
TFTP (Trivial File Transfer Protocol)

NetBIOS (Network Basic Input/Output System)
MIME (Multipurpose Internet Mail Extensions)
IIS (Microsoft Internet Information Server)
IE (Microsoft Internet Explorer) Browser

- MAPI (Message Application Program Interface): A standardized set of C functions placed into a DLL (Dynamic Link Library). MAPI allows Windows application developers to take advantage of the Windows messaging subsystem, which is supported by default with Microsoft Mail or Microsoft Exchange. Any application in Windows can become "mail-enabled" by writing to the generic MAPI interface. MAPI standardizes the way messages are handled by mail-enabled applications, making it so that the application does not have to include vendor-specific code for every messaging system.
- MIME (Multipurpose Internet Mail Extensions) Extends the format of Internet mail to allow non-US-ASCII textual messages, not-textual messages, multipart message bodies, and non-US-ASCII information in message headers. (Sauder)

The worm uses holes in the HTTP, SMTP, TFTP, and TCP/IP processes to spread between computers via trusted shares. It also used the following methods of infection:

1. Microsoft IIS 4.0 / 5.0 vulnerable to directory traversal via extended Unicode in url (MS00-078) http://www.kb.cert.org/vuls/id/111677

2. Microsoft IE Mime Execute Code: Internet Explorer HTML emails with incorrect MIME headers could allow execution of code. http://www.iss.net/security_center/static/6306.php

3. Microsoft IIS/PWS Escaped Characters Decoding Command Execution Vulnerability: http://www.nyu.edu/its/security/virus/nimda-server.html

4. Microsoft Office DLL Execution Vulnerability: http://www.iss.net/security_center/static/5263.php

Brief Description:

The worm began its attack by probing each IP address within a subnet that was randomly selected. It was trying to detect and then exploit the above known weaknesses that were widely known to exist in unpatched Microsoft IIS Servers. If the IIS Web Server would read a Web page containing the infected JavaScript, it would automatically execute, causing the infected JavaScript to propagate to the remaining Web pages within the targeted IIS Server.

Once users with Microsoft Internet Explorer browsers, version 5.01 or earlier, visited an infected site on the exploited Server, the virus would be downloaded in their cache and would execute on their machine, causing the worm to be sent to other computers via one of the other exploit methods used by Nimda. Depending on the number of vulnerable users visiting an infected site, propagation could be rapid, and since there was no way to predict when a vulnerable user was going to surf that site, the only way to stop the spread was to disconnect the affected server pending a complete rebuild, or at least an exhaustive clean-up operation.

After the actual infection occurred, Nimda would spread to any other computers that had a trusted network share with the infected machine.

As a final act, the Nimda script would send a command to e-mail itself out to everyone listed in the infected system's windows address book. This was only successful with Microsoft mail products, however, and it is highly likely that the spread was greatly curtailed by this application limitation. The e-mail would contain an enticing subject line, and an attachment named readme.exe, which caused users who either opened, or previewed the email (which was a web page with the JavaScript built into it) to further infect and propagate the worm.

Variants:

There were five main variants of the Nimda worm. There are more than five total, but they all were centrally based on the original package, or one of the five main variants listed below:

1. W32/Nimda.b@MM. This is an Internet Worm packed with a PE packer, and the filenames Puta!!.scr and Puta!!.eml replace the original Nimda package files Readme.exe and Readme.eml.

2. W32/Nimda.d@MM. This is an Internet worm as well, but it uses different filenames than the original by replacing Readme.exe with Sample.exe, MMC.exe with CSRSS.exe, and Admin.dll with HTTPODBC.dll.

3. W32/Nimda.e@MM. This is an internet worm that is essentially the same as the ".d" variant listed above, with the exception of two dynamic link library files, Cool.dll and Httpodbc.dlll Other differentiations from the original include files named Sample.exe, Sample.eml, and riched20.dll.

4. W32/Nimda.f@MM. This is also an Internet worm, but it contains only minor differences from ".d" and ".e".

5. W32/Nimda.g@MM is also only slightly different than the original, and ".d" and ".e".

References:

1. CERT® Advisory CA-2001-26 Nimda Worm -- http://www.cert.org/advisories/CA-2001-26.html

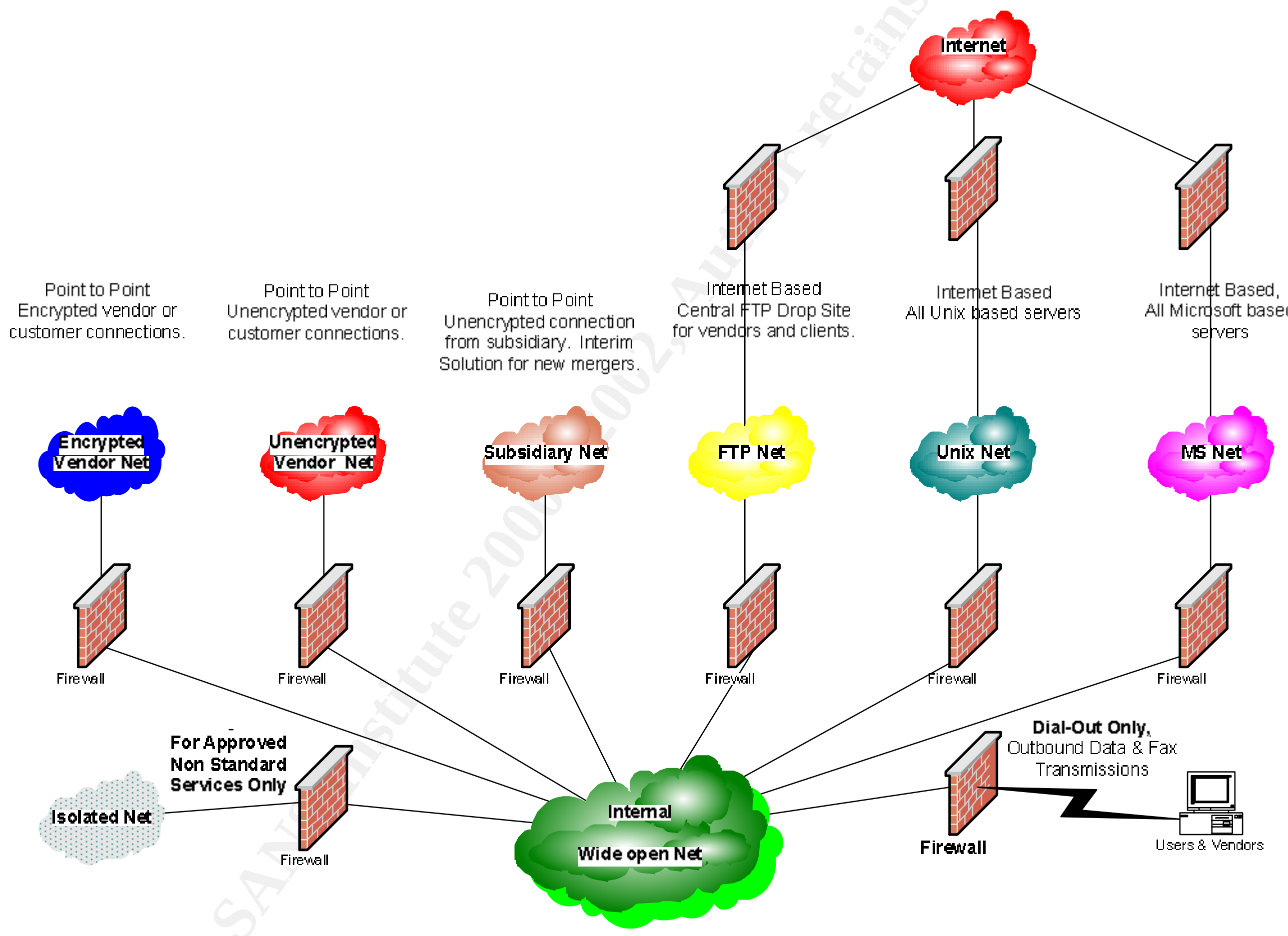
2. ADVISORY 01-022 "Mass Mailing Worm W32.Nimda.A@mm -- http://www.nipc.gov/warnings/advisories/2001/01-022.htm

3. FedCIRC Advisory FA-2001-26 Nimda Worm -- http://www2.fedcirc.gov/advisories/FA-2001-26.html

Part II – The Attack

Description and Diagram of the Network:

The network in our enterprise consists of roughly 4000 NT based servers, and another 4-5000 *nix based servers. We have approximately 120,000 end users. Our infrastructure contains four supernodes, operating in Datacenters, and we hold many OC3 pipelines to allow for connectivity and speed. The below Diagram illustrates how our Network is laid out at a very high level:



Internet
Point to Point Encrypted vendor or customer connections.
Point to Point Unencrypted vendor or customer connections.
Point to Point Unencrypted connection from subsidiary. Interim Solution for new mergers.
Internet Based Central FTP Drop Site for vendors and clients.
Internet Based All Unix based servers
Internet Based, All Microsoft based servers
Encrypted Vendor Net
Unencrypted Vendor Net
Subsidiary Net
FTP Net
Unix Net
MS Net
Firewall
For Approved Non Standard Services Only
Isolated Net
Internal Wide open Net
Dial-Out Only, Outbound Data & Fax Transmissions
Users & Vendors

The Wide Open Net is by far the largest segment of our network, with most user machines sitting inside it. The other nets have connections from the Wide Open through firewalls with strict ACL's on their way out to the Internet, and the Internet Based Firewalls contain our IDS system on the perimeter. We run our Host Based IDS system in the Wide Open net area, protecting servers in the other nets with a specific business need.

Protocol Description:

HTTP is the Hyper Text Transfer Protocol, and is used to exchange and parse files for graphic images, sound files, video clips, text files, and other multimedia files across the internet using TCP/IP for transport, usually via Port 80 on the computers involved in the transaction. Essentially, when typing text, or attaching an image to an existing document, HTTP allows tags to be placed within the text or surrounding the location of the image or sound file, that instructs a browser to change the format of the text, or get the image or sound file from the location, without further interaction from the individual who is "surfing" the page with the information on it. It is a stateless protocol, in that each individual command is executed without any knowledge or reliance upon any former commands which may have been run, and it is in this manner that the Nimda worm uses web servers to exploit the IIS vulnerabilities described above. HTTP can also be used locally, in that on an intranet, or even a stand-alone computer, a document can reference a location tag and the served document will create a hyperlink to that document, even though traversal of the Internet or intranet isn't required in order to receive the document.

Microsoft Internet Information Server (IIS) is Microsoft's premier web server product. It allows for the hosting of web pages in a manageable and coordinated manner. It provides access points for people from within an intranet, and from the external Internet, even separating the two with security rules to prevent traversal to the intranet from the Internet. It was a primarily a weakness in this protocol that allowed the Nimda worm to access internal documents and server resources on infected local servers from the internet.

Terminal Control Protocol/Internet Protocol (TCP/IP) is the set of communication protocols that direct connections between hosts on the Internet. This is the primary protocol used for the world wide web on the internet, and is not Operating System dependent, meaning it translates the information it serves so that a document served from an Apple OS X server can be translated and viewed by a workstation running Free BEOS, or any of a number of other operating systems, and they will see the document as it was intended, and originally created by the author. This protocol standard was the single most important step necessary for the explosive growth of the internet, as it allowed all of the many varied computer systems around the world to share a common way of connecting and sharing information in a readable/usable format. It has it's share of problems and limitations, however, not the least of which is that source addresses are not verified against a known table of static IP addresses, and the result is that many of the sources of Nimda infection were actually "spoofed" or stolen IP addresses, greatly complicating the task of tracking down and eradicating infected hosts in order to quickly stop the spread across the internet.

Simple Mail Transfer Protocol (SMTP) is used by mail servers from many different operating system platforms to send and receive email messages. It usually uses port 25 on the respective machines, and is called simple because there are not many pieces to it. It pretty much sends and receives mail, and that's about all it is capable of. SMTP is also used to facilitate communication between a mail client and the serving agent, and Nimda used this transport agent to send itself across the net to all the addresses found in the infected user's address book.

Network Basic Input/Output System (NETBIOS) adds special functions to the Dos Bios for Local Area Networks. It generally uses port 135-139 on Windows based computers, and specifically uses UDP port 137 for the NetBIOS name service, and UDP port 138 for the NetBIOS Datagram service. SMB takes advantage of this service, and since it uses UDP ports, it generally sends out packets for use, without verifying their receipt. This works great for just shipping Nimda straight across the net to trusted network shares without needing to verify it was received. For the portions of communication where verification was necessary, it utilized the TCP protocol and port 445 to what is known as the Direct Host. These ports and protocols are instrumental in the sharing of files, devices, and directories in Windows Operating System based computers.

Trivial File Transfer Protocol (TFTP) is a very basic file transfer protocol that provides no security features, and uses UDP, which means that the files are just sent out, with no concern about user names, passwords, or whether they were properly received. This was another way in which the Nimda worm propagated. According to RFC 1350[1], TFTP has been around for a while, having originally been designed by Noel Chiappa, and then redesigned by Noel, Bob Baldwin, and Dave Clark. They used TCP to base the acknowledgement and retransmission code on, and used PARC's EFTP abort message to inspire their error handling mechanism. TFTP runs on top of the UDP protocol, as well as many other protocols, but has been used specifically with UDP to solve the file transfer issues relating to UDP's one-way packet transfer nature. Essentially, UDP just throws it's packets from the source to the destination, without ever confirming receipt. This is not conducive to good file transfer and reception guarantees, so TFTP is used on top of UDP to facilitate this particular need. Using 8 bit bytes of data, it can not list directories or authenticate users, but it reads and writes files and email quite well, from source to destination. That is about all it can do, however. It was designed this way, in order to remain easy to put in place, and small in overall stature. Nimda made quick work of these features, using TFTP to ship it's payload from infected source to newly infected target.

[1] http://www.faqs.org/rfc/rfc1350.txt

Summary of the Exploit:

How Nimda Works:

Nimda started out as a series of emails initially detected on October 15, 2001, originating from an infected machine that appeared to be in Canada somewhere. There were hundreds of these original emails, and they were sent to addresses around the world. The emails had a source listed as mikko.hypponen@datafellows.com, which was originally the email address of F-Secure's Director of Anti-Virus Research. When F-Secure bought Datafellows, they changed the URL to F-secure.com. Mr. Hypponen was not actually affiliated with the email, and this is an example of one of the problems with the SMTP protocol, in that a well-respected individual could be impersonated as having done a heinous thing. This original email consisted of an attachment with the Nimda.a version of the Nimda worm, and did not enjoy great success at spreading. The next Variant however, struck on September 18, 2001 and quickly spread throughout the world rapidly, infecting an estimated 500,000 servers and home computers in less than a week.[2]

Ironically, Mr. Hypponen was actually involved in F-Secure's response and attempts to eradicate the worm for their customers, and in fact was a major contributor to the paper footnoted above and below this page.

Originally, the worm spread by redundantly scanning random subnets of IP addresses looking for vulnerable IIS Servers. The well-known vulnerability it was scanning these servers for was the IIS/PWS Extended Unicode Directory Traversal Vulnerability, and Microsoft had provided a patch to this particular vulnerability months prior to Nimda's birth. Due to a large number of unpatched, and therefore vulnerable servers, the worm began spreading at an astronomical rate. Once it found an IIS server vulnerable to the exploit, it copied the Admin.dll file to that server, infecting it, and causing it to begin scanning more subnets for more servers to infect. Nimda also scanned for the backdoor left by the previous mass infections by the Code Red II worm, and used any vulnerabilities it found to copy the admin.dll file to those servers as well. It used TFTP to upload the file if it found either of the two vulnerabilities on the server it was scanning.

Once the web server was infected, the worm moved through the system identifying all files ending in .html, .htm, and .asp extensions, and added a piece of infectious JavaScript to them. It then added a multi-part MIME-encoded version of the worm in the directories where any of the previous files were located. Once the readme.eml was placed in the proper directory, the infection was complete. One of the reasons this method of infection was so successful, was that when Microsoft issued the patches to protect servers from infection by the Code Red II worm, many system administrators found the instructions confusing, and oftentimes the patch simply did not properly execute. As a

[2] http://www.europe.f-secure.com/v-descs/nimda.shtml

result, there were a large number of vulnerable servers sitting exposed to the Internet when Code Red II struck, leaving the backdoor open for Nimda when it eventually arrived on the scene. We noticed in our Response process that many of the System Administrators we came in contact with protested loudly that they definitely had patched their servers, and there was no way they could be infected, but either the patch didn't properly execute, or for some other reason they were still vulnerable, and they were infected quite effectively. Microsoft also mentions in several later patches and upgrades to the IIS product that the System Administrator might need to re-run the previous updates and patches, as the newer ones unloaded some of the patches that protected them from this vulnerability. Not very forward thinking of them, and in fact, they contributed greatly to the spread of this worm by their very patching system.

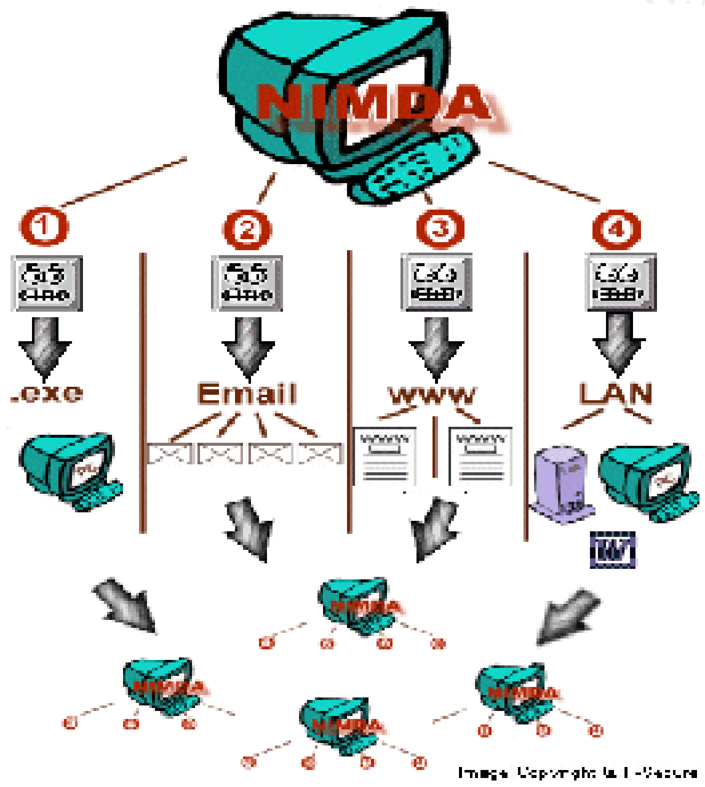
When using E-mail to move itself around, Nimda embedded itself as an attachment called Readme.exe, and used an enticing subject line to get the receiving party to execute the attachment. It also took advantage of an additional exploit in some versions of the Internet Explorer browser to execute automatically, even when the user had merely viewed the email, and hadn't actually executed the attachment. After infecting the receivers machine, the worm opened the Messaging API and MAPI sections of the infected system and extracted all of the listed email addresses, sending itself to each and every one, sometimes multiple times. This added an additional level of social engineering to the delivery mechanism, because users were much more likely to open an attachment that was coming from a source address they knew and probably felt they could trust. But it was Nimda calling, not their trusted friend. Even the patched versions of Microsoft Outlook and Outlook Express caused problems because the worm code attempt at executing would generate a window asking the viewer if they wanted to execute the attachment or not. Since most Windows users are used to getting spurious windows asking inane questions getting in their way, it is not uncommon for the user to just "hit enter to make it go away" without actually reading the content. In this manner, even patched machines became infected, and began to spread as well. Additionally, the Worm had its own built in SMTP engine, bypassing the outgoing filtering attempts put in place on the exchange servers in our enterprise in the initial stages of the infection in an attempt to stem the spread. The worm sets a cycle in place so that every ten to eleven days, all the email addresses are again accessed and bombarded with the loaded email. Amazingly, the worm even uses a counter!! It can be located in the registry key at HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MapMail, Cache. Every time the emails are sent out, the worm resets the counter, until the worm is eventually removed from the system. If it is never removed from the system, it continues to try and re-infect servers via all of the methods described above, including via NetBIOS Shares.

It traveled through trusted shares via the NetBIOS system in much the same way that a file being copied by the trusted share administrator would have. It essentially scanned the system for any open shares, then copied Riched20.dll in as many places, and on as many drives as it could, provided there were .dml or .doc or both types of files present. This way, anytime someone opens this type of file, which usually requires Riched20.dll to run, the worm would be executed. Many editing programs use Riched20.dll, including Microsoft Word, WordPad, Outlook, Project, and others, and is included in the installation of Microsoft Windows 95, 98, ME, NT, 2000, and XP.[3] It is used for extensive editing functions, and is called the Rich Edit Controls file. Once infected, the C: drive of the target system is shared out with everyone allowed access. The worm then disables the sharing security on the target by deleting all sub keys from the System\Current\ControlSet\Services\lanmanserver\Shares\Security Registry key.

Description and Diagram of the attack:

The following description of the Nimda exploit is based on information found on the F-Secure Nimda Virus Definitions web pages located at http://www.europe.f-secure.com/v-descs/nimda.shtml:

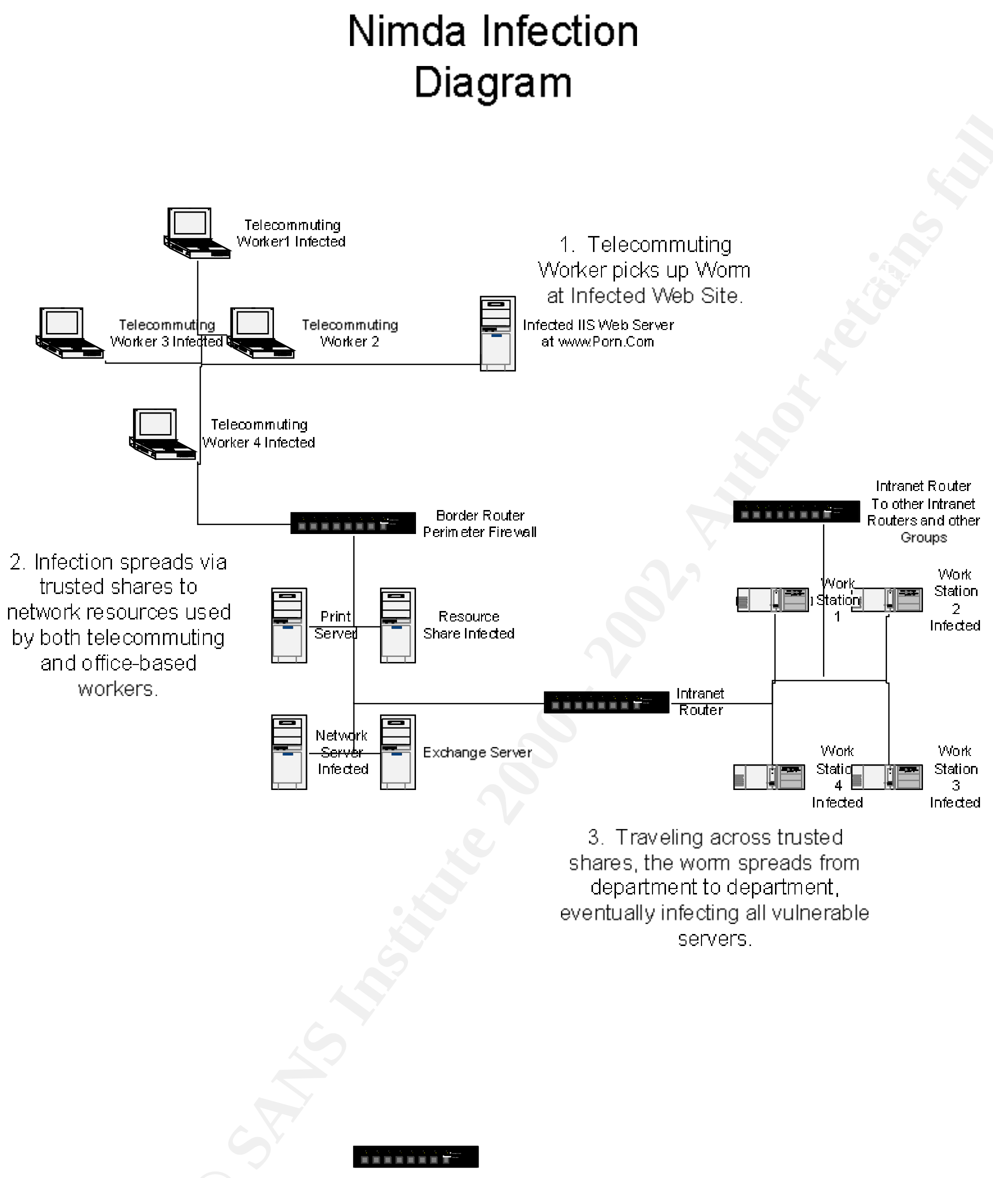
The following diagram was created by F-Secure to show the multi-partite spread pattern of the Worm:



---

[3] http://msdn.microsoft.com/library/default.asp?url=/library/en-us/winui/richedit_9d2r.asp

The following is a Diagram of the infection vector of NIMDA in our network:


Nimda Infection Diagram
Telecommuting Worker1 Infected
1. Telecommuting Worker picks up Worm at Infected Web Site.
Telecommuting Worker 3 Infected
Telecommuting Worker 2
Infected IIS Web Server at www.Porn.Com
Telecommuting Worker 4 Infected
Border Router Perimeter Firewall
Intranet Router To other Intranet Routers and other Groups
2. Infection spreads via trusted shares to network resources used by both telecommuting and office-based workers.
Print Server
Resource Share Infected
Work Station 1
Work Station 2 Infected
Intranet Router
Network Server Infected
Exchange Server
Work Station 4 Infected
Work Station 3 Infected
3. Traveling across trusted shares, the worm spreads from department to department, eventually infecting all vulnerable servers.

Depending on whether the infected box is a Server, or a workstation, which version of NT or Windows it is running, and depending on which infection path the worm has taken, several different modes of infection and transfer will occur. Different file names and program paths, as well as different command lines will be found based on the mode of infection and transfer as well. Starting on a server:

When the worm first arrives on a Server, it looks for the file Admin.dll, which, if it finds, it will create a mutex of, or," A mutual exclusion object that allows multiple threads to synchronize access to shared resources ".[4] This particular property of the worm became critical in our eventual detection methods in our enterprise, because when System Administrators called us for information concerning how to detect the presence of the worm, we would often tell them to add the Threads column to their Task Manager Processes window, as the infected file would be shown with hundreds of threads running in the processes window of an infected executable. Once it determined the Admin.dll file was present, it would create the 'fsdhqherwqi2001' file name and copy itself to the root\windows directory as the mmc.exe, which is usually the Microsoft Management Console, and runs with the rights of the user currently logged in. If the infected server is a web server, the worm will start to infect files on all of the removable and network drives, and all of the executable files with the extension EXE will be infected, with the exception of winzip32.exe, which is not a Microsoft file, and was not vulnerable to this infection technique, which in itself was unusual in that the worm embedded a copy of the exploit code inside itself, extracting it only after the infection was complete and the executable was activated.

After running the embedded file code, an attempt is made to delete the executable infected with the code in the first place. If the attempt at deletion is not successful, another file named wininit.ini is created that contains instructions that will delete the infected executable the next time the Operating System is started.

The worm also scans certain registry keys looking for additional files to infect. It specifically infected files listed in the drill down areas of the registry keys where Microsoft stores its application paths, listed at Software\microsoft\windows\currentversion\app path.

If it can locate them, the worm will scan the server users personal folders and infect any executable files it finds there. After that, it begins scanning the hard drives available to it for files with the extensions HTML, ASP, and HTM. Once it locates any of these, it creates a multi-partite message in the same directory, and names it readme.eml, attaching some JavaScript code to any of the files it located, which then point to the file it created. It's rather ingenious really, because then when someone visits a page with the JavaScript embedded in it, the script runs, activating the worm and infecting the system. Of course it runs itself in stealth mode, so the user has no idea what is happening, but if they are using Internet Explorer version 5.0 or 5.1, with all of the warnings and patches that have been published about it, then I guess there isn't much else that will stop them from getting infected eventually anyway. One of the ways we used to detect whether a server had been infected or not was to do a search on the hard drive for recently created EML and NWS files, particularly large numbers of them, as that was another "feature" of this worm. The

[4] http://www.dictionary.com/search?q=mutex

existence of a file called riched20.dll, with the hidden and system attributes set could also be found in any folder where DOC or EML files reside.

If the infected system is a workstation, it most likely got infected by clicking an attachment to an email that appeared to be coming from a trusted source, or after having visited an infected web site, though trusted shares were also used to transfer the file named readme.exe, or any other five character name, sometimes randomly generated, with an executable extension. You would find this/these files in a local temporary file with an extension of either MEP or TMP, which will make it difficult to differentiate on some workstations. I have seen many where there were literally thousands of suspect looking TMP files on board. There is a command line option associated with this particular method of infection labeled "dontrunold". This command option prevents the file from being run if it is older than the specified system date, which the worm checks on initial execution. Once activated, a DLL file is loaded with the worm inside, and of a specific file size. If it is less than 100 bytes, the worm will unload If the worm is started from README.EXE file (or a file that has more than 5 symbols in its name and EXE extension), it copies itself to temporary folder with a random name that has 'MEP*.TMP' name and runs itself there with '-dontrunold' command line option. When started, the worm loads itself as a DLL library, looks for a specific resource there and checks its size. If the resource size is less than 100, the worm unloads itself; otherwise it extracts its resource to a file and launches it. Checking the resource size is done to be able to detect if a worm runs from infected EXE files.

Then the worm gets current time and generates a random number. After performing a few arithmetic operations with this number the worm checks the result. If a result is bigger than worm's counter, the worm starts to search and delete README*.EXE files from temporary folder.

F-Secure says that at this point:

"After[5] that the worm prepares its MIME-encoded copy by extracting a pre-defined multi-partite MIME message from its body and appending its MIME-encoded copy to it. The file with a random name is created in a temporary folder. The worm then looks for EXPLORER process, opens it and assigns its process as remote thread of Explorer. On some platforms the worm fails to run as Explorer's thread. The worm gets API creates a mutex with 'fsdhqherwqi2001' name, startups Winsock services, gets an infected computer (host) info and sleeps for some time. When resumed, the worm checks what platform it is running. If it is running on NT-based system, it compacts its memory blocks to occupy less space in memory and copies itself as LOAD.EXE to Windows system directory. Then it

[5] http://www.europe.f-secure.com/v-descs/nimda.shtml

modifies SYSTEM.INI file by adding the following string after SHELL= variable in [Boot] section:

explorer.exe load.exe –dontrunold"

As you probably already know, loading it into the [Boot] section means that every time Windows starts, the command will run, and the worm will load. Since the worm has renamed itself to riched20.dll and placed itself in the system folder, it will execute and check all of the trusted shares on the computer, looking for and scanning any files it can access on the remote systems it can now access via those shares.

While scanning the remote systems it has accessed via the trusted shares, it looks for any directories with DOC and/or EML files in them, and copies itself into that directory in binary form. This greatly increases the likelihood that the worm will be activated when the infected web page is accessed.

If it finds executable files while running from a workstation, for some reason it doesn't try to infect them. This is odd behavior, compared to it's server based and web based nemesis, but is only true of the workstation version.

The worm can also spread via E-mail addresses that it finds in the infected machine's Temporary Internet Files folder. Depending on the email client being used, it may also be able to collect addresses from the victim in this manner, and then use it's own internal SMTP email engine to continue spreading itself to the victim's friends and acquaintances. This particular mode of operation is very successful because of the social engineering aspect it uses. An individual who is normally wary of unsolicited emails is much more likely to open one from a trusted source, like the recently infected friend who usually sends jokes.

When the worm finds an IIS server, it begins scanning for known backdoors, like the ones left behind by all of the Code Red variants after II. If it finds an IIS server that is vulnerable to this attack, it activates an automatic download on the host, which places the Admin.dll file on the root of either C or D and executes itself, infecting the server in this manner. We used a signature that scanned the root of system drives for this file as an effective manner of locating infected servers once we became aware of this method of infection.

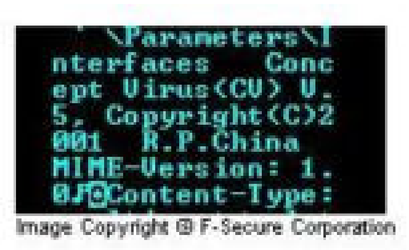
Once it has access, it changes Windows settings that regulate whether the user can see hidden files. It does this by accessing the registry keys at HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced and adjusting the 'Hidden', 'ShowSuperHidden', and 'HideFileExt' keys, making it impossible to see the worm's files in Windows Explorer any longer.

F-Secure also notes:

“After that the worm adds a 'guest' account to infected system account list, activates this account, adds it to 'Administrator' and 'Guests' groups and shares C:\ drive with full access privileges. The worm also deletes all sub keys from [SYSTEM\CurrentControlSet\Services\lanmanserver\Shares\Security] key to disable sharing security.

Additional information:

The worm has a copyright text string that is never displayed:



```
 \Parameters\I
nterfaces   Conc
ept Virus(CV) V.
5. Copyright(C)2
001  R.P.China
MIME-Version: 1.
0J@Content-Type:
```

Image Copyright © F-Secure Corporation

Concept Virus(CV) V.5, Copyright(C)2001 R.P.China”

Signature of the Attack:

The signature used to detect the Network Path was based on the scanning pattern seen with every packet of data sent out by an infected machine. If you check your web server logs, you will see the following:

```
GET /scripts/root.exe?/c+dir
GET/MSADC/root.exe?/c+dir
GET /c/winnt/system32/cmd.exe?/c+dir
GET/d/winnt/system32/cmd.exe?/c+dir
GET/scripts/..%5c../winnt/system32/cmd.exe?/c+dir
GET/_vti_bin/..%5c../..%5c../..%5c../winnt/system32/cmd.exe?/c+dir
```

You could see many variations of the last line, or any of the lines after the first four, as they are examples of attempts to connect to the open backdoor left behind by a previously Code Red infected box. The other lines are just exploits of the common Directory Traversal vulnerabilities, and have many possible variations on what I have listed, and many other possible examples exist.

Because the worm also modified several specific registry keys, we could have also created signatures based on registry modifications. However, we elected not to do that due to the sheer number of false positives we believed we would encounter, as almost anything you do on a Windows machine modifies the registry. One of the registry keys Nimda modifies is: HKLM\Software\Microsoft\Windows\CurrentVersion\Network\LanMan\[C$ -> Z$][6]

Additionally, we reactivated the signatures we had in place during the Code Red infections so that if Nimda triggered in the same way, we could capture it that way as well. You could also monitor router traffic for a sudden increase in Port 80 traffic, or set up ACL's at the firewall to trigger on a certain traffic level or pattern.

I am listing the Snort rules used by many people on the web to detect the Nimda worm, because I can not disclose the software or rules that we used here in our Enterprise:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg:"Nimda worm attempt";
uricontent:"readme.eml"; flags:A+;)
alert tcp $EXTERNAL_NET 80 -> $HOME_NET any (msg:"Nimda worm attempt";
content:"|2e6f70656e2822726561646d652e652e656d6c|"; flags:A+;)
alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"Nimda worm
attempt";
content:"|6e616d653d22726561646d652e65786522|"; flags:A+;)
alert tcp $HOME_NET any -> $EXTERNAL_NET 25 (msg:"Nimda worm attempt";
content:"|6e616d653d22726561646d652e65786522|"; flags:A+;)
alert tcp any any -> any 139,445 (msg:"SMB Nimda
RICHED20.DLL";content:"R|00|I|00|C|00|H|00|E|00|D|00|2|00|0"; flags:A+;)
```
[7]

It is also possible from a host-based perspective, to search for the Riched20 and Readme DLL files in directories where either Doc or EML files are normally found. On some systems, you would need to enable the view hidden file types in order to see these types of files. If the server you are tracking is infected, you would also find the Readme.eml file in any web folders where .HTM, .HTML, or .ASP files are located.

To find evidence of infection on a client running Windows, check the email attachments directory for the Readme.exe file. Windows Media player may suddenly and suspiciously start and stop on an infected box, and the file Load.exe will appear in the windows or winnt directories. Additionally, look for a line like, "shell=explorer.exe load.exe –dontrunold" in the system.ini file.

[6] http://www.symantec.com/avcenter/venc/data/w32.nimda.a@mm.html#technicaldetails

[7] http://lists.insecure.org/win2ksecadvice/2001/Sep/0081.html

Protecting from Nimda Infection:

You should always have the latest signature files from a reputable Anti-virus software package running on your workstations and clients. Additionally, you should have a carefully thought out Security Auditing program distributed throughout your enterprise. These two steps will assist you in determining where your weakest links are, but your real "ace in the hole" is to schedule regular vulnerability scans across your network. This must be done with permission, and with known good copies of a reputable vulnerability scanning software package. While software such as Nessus and Satan are free, and freely available, they may not contain the latest vulnerability information, and they do not provide a vendor to question with support problems, or configuration advice. Additionally, if a new vulnerability is disclosed on the Internet, you may have a delay in getting the new signature in place for your scans if a vendor is not backing your system with a service contract.

You must have a policy in place to deal with infected servers when they are discovered. Failure to do so will result in a chaotic, unorchestrated response, and could lead to a worsening of the infection, even while attempting to eradicate it. Infected servers SHOULD always be "nuked from high orbit", or have their system disks completely erased and reinstalled. "but I have years of patches and upgrades in there…" you say. Yes, but you have no way of knowing what changes were made to your system by the infection, and therefore, no sure way to verify that you were able to completely remove the infection. The only assurance you have of complete eradication, is to start fresh, from a known good copy of your operating system, and a known good copy of your data.

In the meantime, Microsoft has issued patches for several variants of this and many other worms, and they can be located at http://www.microsoft.com/technet/security/bulletin. This particular Directory Traversal exploit vulnerability patch information is available at http://www.microsoft.com/technet/security/bulletin/ms00-078.asp. You can also use the Microsoft HFNETCHK product to verify your system has the latest and proper updates employed.

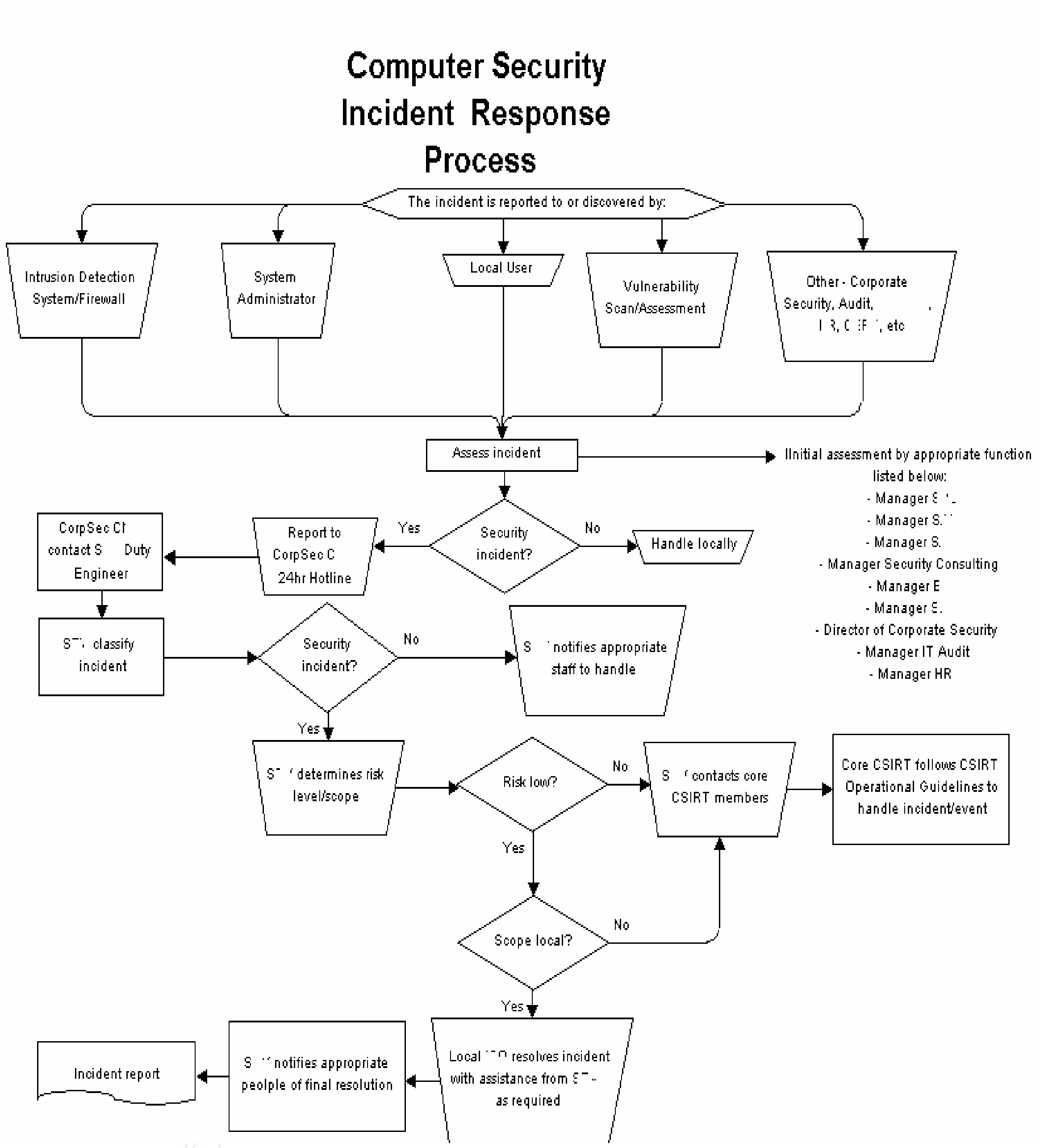
End users can pick up email patches for Outlook and Internet Explorer in the Microsoft Security Bulletin sections listed above as well. As of the date of this writing, however, simply installing the latest version of Internet Explorer, Outlook, Outlook Express, or Windows products will already be patched against these vulnerabilities. Removing unprotected file shares will go a long way toward protecting against this, and many other types of infections that are running amok on the Internet these days, as well as keeping your average "Script kiddie" from sharing out your drives for MP3's and the like. You should also load a personal firewall product like ZoneAlarm, Black Ice, or Tiny Personal Firewall, as they will either detect and report, or flat out prevent most file sharing violations or attempts to compromise your system. Of course, a difficult Administrator password is always the rule, not the exception, with any Windows product. And it goes without saying that this should include Alpha's and numerics, as well as special characters like the ASCII codes and function key strikes.

Web Server Administrators can find many tools available to help them here as well, but the main one is to remove the access to the backdoor left behind during Code Red II either with the tool provided by Microsoft, or by manually running the IIS Lockdown Tool in the Default mode issued by Microsoft. Again, the latest versions of IIS Web Server are no longer vulnerable to this particular exploit, but you should perform regular vulnerability assessments to verify this remains the case. Microsoft is now beginning to publish more and more security related documentation, and most of it is located at http://www.microsoft.com/technet/security

Part III – The Incident, and How We Handled It:

Preparation:

My company has a Computer Security Incident Response Team, but they have not been in existence for very long, and it consists of volunteers from the various Computer and Corporate Security Departments across the Company. Additionally, no training sessions involving a scenario, and all the members, has ever been conducted prior to the Nimda infestation, to the best of my knowledge, so most responses have historically been rather chaotic. That does not mean they do not get the job done. In fact, they do it very well. The existing Incident Response Process as it is supposed to function is outlined in the following Diagram: (It has been sanitized, and may appear choppy)


Computer Security Incident Response Process
The incident is reported to or discovered by:
Intrusion Detection System/Firewall
System Administrator
Local User
Vulnerability Scan/Assessment
Other - Corporate Security, Audit, HR, CERT, etc
Assess incident
Initial assessment by appropriate function listed below:
- Manager S
- Manager S
- Manager S
- Manager Security Consulting
- Manager E
- Manager S
- Director of Corporate Security
- Manager IT Audit
- Manager HR
Security incident?
Yes
No
Handle locally
Report to CorpSec C 24hr Hotline
CorpSec C contact S Duty Engineer
S classify incident
Security incident?
No
S notifies appropriate staff to handle
Yes
S determines risk level/scope
Risk low?
No
S contacts core CSIRT members
Core CSIRT follows CSIRT Operational Guidelines to handle incident/event
Yes
Scope local?
No
Yes
Local resolves incident with assistance from S as required
S notifies appropriate peolple of final resolution
Incident report

Our Enterprise also has a dedicated Antivirus group, and it functions well, providing regular "pushed" or passive installations of the latest signatures to the workstations on the network either daily, or every other day, but rarely does more than three days go by without an update being installed on our workstations. There is also an Antivirus Intranet Web page with a lot of very timely and well-organized information relating to current viruses and how to limit your exposure to them.

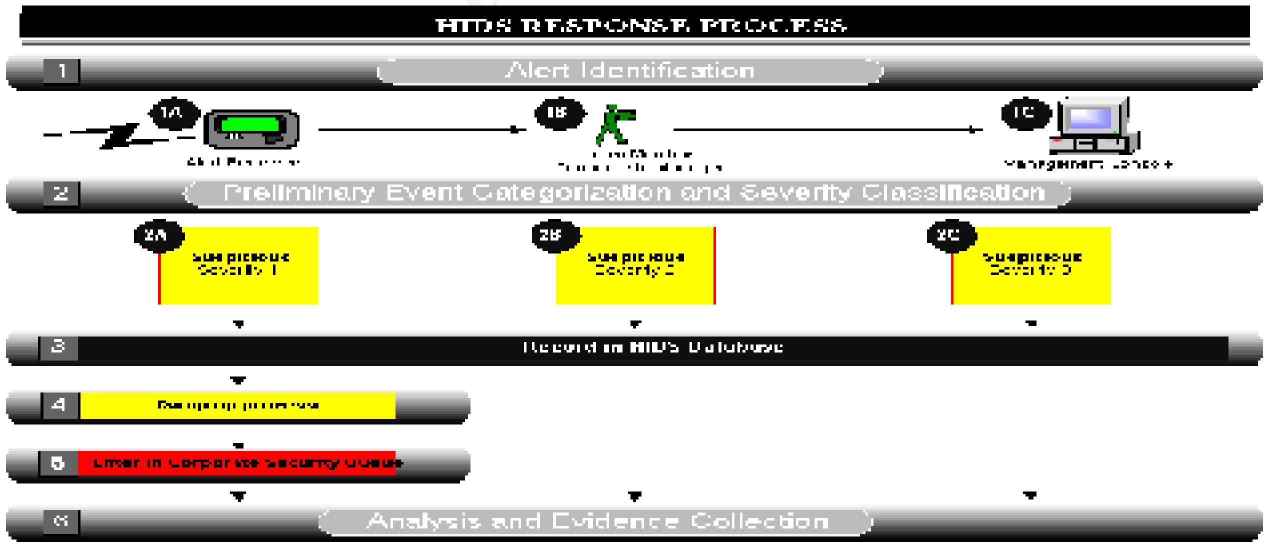
I belong to the Host-Based Intrusion Detection Group and we are responsible for monitoring real time alerts coming from the client servers we have installed throughout the Enterprise. However, at the time Nimda struck our Enterprise, we had only been installed on a very limited number of Servers. This greatly decreased our ability to detect and protect against infection in the majority of Servers that were vulnerable on our network.

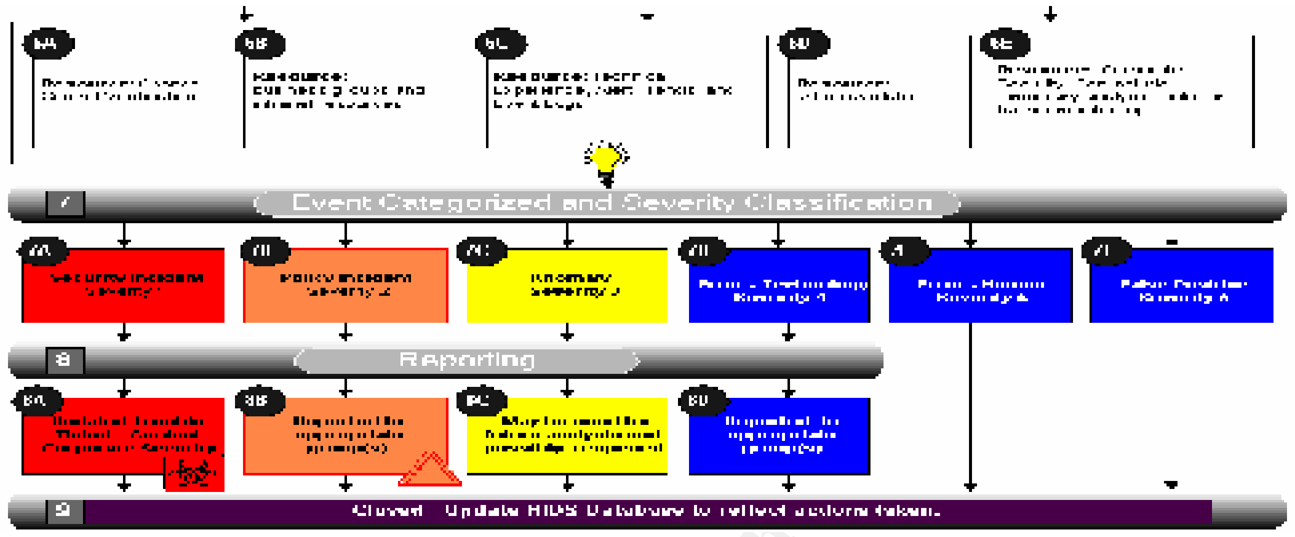
The HBIDS group has it's own incident response process, and it is detailed below, including a diagram. There is also a Perimeter Security Group responsible for maintaining and responding to alerts generated by the Firewalls on the Perimeter of our Enterprise, between the outside world and our Intranet. Additionally, there are several other groups monitoring Password usage, Data Base Security setup and configuration, Asset Protection and Management, Web Site Security assessment and management, Vulnerability Assessment and Management, Software Version Control, and probably many other Security groups which I either haven't found yet, or am just not aware of.

In an organization with 120,000 end users, this is not uncommon. I have been here for about a year and a half, and learn about many new people, things, policies, etc., every day!

Identification:

In this case, my group, the Host-Based Intrusion Detection Group, became aware of the infection via alerts being generated by our servers that didn't quite add up. Our HIDS Incident Response Process is detailed in the following Diagram, sanitized to protect the innocent:




Event Categorized and Severity Classification
Reporting
Closed Update HIDS Database to reflect actions taken.

My group was monitoring several of the public News Lists and had seen a description of the Worm on several of the lists, with the most comprehensive description coming from Eeye. They described some of the files which were dumped during the infection process, and we scrambled to produce an IDS signature that would trigger upon detection of these on one of our client systems. Once in place, we just waited for our first alerts to come in. It was about two hours before our first server triggered an alert. The signature was designed to alert if the Admin.dll filed appeared on any root drive on the server, and this particular server alerted that the Admin.dll file was located on the root of the D drive, which was set up as a Data Store, and was separate from the C or Root drive, which was housing the Operating System, which was, in this case, Windows NT.

We contacted the Server owner and asked them to verify the infection, but they stated that they had already run the patches and their machine was now clear. Since we do not have permission to run Vulnerability Scans across the network, (only the Vulnerability Assessment group can do that) we could not verify if they had successfully cleaned the machine. We advised them that the only recommended course of action with an infected box was a complete rebuild, but they stated that the machine was "too important" to take down for that long, and said they would take responsibility for it if the box was not properly patched by the Microsoft patch they had run.

Shortly after that, we got another alert, from the same machine, and when we re-contacted the server owners, they expressed frustration at having trusted Microsoft to come up with a reliable fix. We reminded them that Microsoft said the patch was to be run AFTER you rebuild the box and install from known good original media, and BEFORE plugging the machine into the network, possibly re-infecting it. They responded that they would do that now, but they were quite upset with us. "Don't kill the messenger….." comes to mind.

The infection had actually been detected in the network initially by the Virus Detection group, but I am not certain how quickly it was detected. I know the first Conference Call to discuss the Response was activated at 6 am on the morning of the 18th, so Infection must have been detected around 4, which means we were infected very early on during the initial spread of the virus.

Containment:

During the course of our investigation into the cause of these alerts, I contacted the Network Operations Center to have them assist in tracking the location of a server by IP address, so that I could contact the server owner and ask for assistance in troubleshooting the strange alert. Once connected to the NOC center, they told me they were very busy responding to the Nimda infection, and could we please assist them by tracking down and either cleaning, or disconnecting any machines in our subnet. They then gave me a conference call number to join, and I called in.

There were about 30 people on the call, most of them confused about why they had been contacted and asked to join, and the rest confused about what their roles were. The primary on the call was the Virus Support guy, who kept having to give a "quick rundown" to every upper level manager who joined the call in response to a page they received. After a few minutes of this, I was able to get a word in edgewise and asked where I could find information on the infected servers so that I could begin shutting down the ones in my subnet. I was given a list of IP addresses that had been set up by one of the CSIRT groups, which was scanning all the subnets and reporting infected IP addresses back to the NOC. I started with the IP's close to my own and began to try and track the boxes by running NBTSTAT –a to get a user name, and then comparing that user name to our Global Address book of user names in the email system. If that didn't work, I checked with an intranet website that listed a lot of information about servers from the work order generated when a server was first requisitioned, and would often be able to come up with the name and phone number of the server owner, or responsible group.

The rest of my tracking was just pure social engineering. Figuring out the user name and possible locations based on a series of pings and tracert reports, as well as the IP owner information and then wandering around the department trying to get a fix on who knew where the server was. I was able to find, disconnect, and assist with inoculation of about 15 boxes on each of the three days we were actively assisting with the infection.

The Virus Support guy was getting extremely exasperated, and did not seem to have any backup personnel, so he ended up having to jump from conference call to conference call in order to meet the different manager's demands for attention and information. Sometimes, he would be able to leave for a few minutes for a bathroom break or something, but the entire process was very broken. I have conducted some research, and have some fixes to the system that I will propose below. But basically, the first thing that needs to change is the reporting process. There is no formal reporting process known throughout the company.

When something breaks, everyone runs around asking the "old-timers" what the procedure is. There is no "911" or "emergency" link on the home page for the intranet, so you have to search for information based on what group you think might be responsible. This is exasperating, as there are so many groups, each with it's own set of responsibilities and rules, that it wastes valuable time to perform in this manner. A "911" button should be linked on the home page of the Intranet that links to a direct response agent who can direct the incident to the proper response entity.

So while the rest of us were busy tracking down and eradicating the boxes, only one group was authorized to perform scans across the network looking for infected machines that either hadn't begun to broadcast yet, or were vulnerable to only half of the attack, like the machine listed above, where the admin.dll was successfully dropped, but hadn't yet executed. They continued passing out lists of infected servers, and people kept assigning themselves to clean boxes on their subnets. It was an interesting process, and I likened it to a treasure hunt at times because there were many situations where I arrived at a reported infected machine, only to have been beaten there by another team doing the same function, that I was not aware was around and operating.

Finally, on the conference call, permission was given to the Network Operations Services Center to begin turning off router ports where infected machines were connected. It was decided this was the only way to effect containment while the boxes were located. Once this process was in place, things began to quiet down considerably. Unfortunately, it was an "out of sight, out of mind" solution, with no long-term solution for finding the infected machines and eradicating the infection ever made. The few groups that were actually tracking down infected machines resorted to sending NetBIOS messages to the screen of the infected machines telling the operators to turn it off and call them for information on how to rebuild them safely.

By the third day, the infection had been contained, and was no longer spreading. The conference calls were cancelled and the individual teams were handling the remaining one-off's left over that hadn't yet been located and eradicated. Most of us just went back to our regular positions, and no debrief or formal statement collection was ever completed. I would be interested in seeing if an actual report was ever completed and placed on file with Corporate Security in order to maintain a record of the events, and the order in which they occurred. I would be really interested in seeing if all of the people on the conference calls were even listed and recorded, much less documented.

Some time later, a memo went out system wide stating that access to outside Html based email systems like Hotmail and Aol was no longer allowed, because it allowed infected attachments to email to bypass the firewalls, so I assumed that was the cause of our particular outbreak, but since no official report was ever posted, no follow-up completed, and in fact, no actual documentation ever completed to tie the whole incident together, I really don't know where the infection began, and how it spread so efficiently. I can tell you that it hit many servers and workstations within our network, but didn't do much to our ability to operate, and other than taking up valuable man-hours to respond, wasn't much more than a nuisance to our Corporation. I heard that many others weren't so lucky, though.

As far as I know, no attempt was made to image a box and determine how the actual infection occurred. This has been the theme with most of the Worm exploits that have come out as the general consensus is that there is no point in attempting to forensically image and process a box if no attempt will be made to prosecute the offender. In this, as in most email Virus/Worm cases, no perpetrator was/is ever located, so any evidence collection would be done purely for general principle. Still, I would like to have seen an effort made to actually track the source of the infection in our network, but I also don't know whether that was ever actually done or not. It may have been, and in fact, that may have been why the email memo was generated limiting the use of the HTML external email systems.

Eradication:

All of the systems that were identified as being infected with the Worm were either completely rebuilt, or patched, with the exception of about a dozen, which were never located at all. These servers continue to send the infection mechanisms out constantly, but they have been blocked at their subnet router, to prevent their traffic from distributing across the network. Occasionally, as we saw about three weeks ago, a vulnerable server will be brought up on the subnet where one of these infected servers still resides. In the case three weeks ago, several boxes were rebuilt with an old NT 4.0 SP3 image, and were connected to the network so that the SA could download and install the latest patches. In theory, an acceptable practice, but in reality, the boxes were locked into a subnet with a Nimda infected server, which caused all of them to become immediately infected. Since our HBIDS software is now a required part of every new build, it was installed on these systems, and as soon as the infection struck, we were alerted, and were able to very quickly contact the SA responsible for the boxes, ensuring they were removed from the network and rebuilt from a newer, patched, build image, keeping them from being reintroduced to the network until they were no longer vulnerable.

Recovery:

Basically, there was no organized recovery effort. If the SA in charge of an infected box rebuilt it and reconnected it to the net and it was still vulnerable, it would be reinfected within minutes. If it was not vulnerable, they got to keep using it. If it was vulnerable and on an uninfected subnet, the

Vulnerability scans would detect it and summarily discharge it from the network, with no effort made to contact the SA and explain the reason. So when the SA found no more network connectivity, they would call the NOC center hopping mad, and be told they were vulnerable and no longer allowed to connect to the net, which did nothing to diminish their hopping madness.

Luckily, the Antivirus guy posted an "approved" Microsoft fix to the Antivirus website, and most SA's eventually ended up getting the correct patch file, and even applying it in most instances, but there were some servers that were never found and cleaned of the infection, and they still reside on the Network today, so I don't feel comfortable stating that a Recovery was ever completed. As far as residual testing goes, there are regular Vulnerability scans performed, and a listing of the servers found vulnerable is made available via an Intranet site, but I am not sure if any other follow-up is done, besides just kicking the servers off the network.

Lessons Learned:

It was obvious to me that the system was broken in a number of different ways. The Corporate Computer Incident Response process should go as detailed in the diagram listed previously.

If it did actually proceed according to the above diagram, I would not know it, as our group is not listed in this process, and therefore would not have been involved in the formal response. But judging from the Conference Call, it was not going as smoothly as the diagram recommends.

You could not go to a single location and enter the IP, then get the name and physical address of the server, despite the fact that all Server IP numbers are statically assigned and tracked by a group in the company. That group would not share the information, and so if a request needed processing, a form was required, management approval, etc…etc….meanwhile the infected box was re-infecting the vulnerable boxes it had trusted shares to, and scanning for other vulnerable servers on the intranet.

I found out later, that some of the boxes were simply never located, and they were just blocked off at the router in their subnet pending further activity by the groups that were tracking the infected boxes down, most of which were volunteers, and needed to return to their own work as soon as possible. In fact, we located one server that is still infected, but was never located, just the other day, when a new box with a vulnerable image on it was brought up in the subnet the infected box was locked into, and it immediately became infected. The group responsible for detecting the infected servers has no responsibility to pass the infected server information on to any other group, so if they are unable to locate it after a cursory search, they order it to be locked inside it's subnet, and move on to the next one. In my opinion, this is improper. Some group needs to be assigned the process of eliminating these infected boxes at all times. So that any time an infected server shows up on the net, it is tracked down and killed.

Usually, as in this case, it hasn't happened before, so they don't know, but they give you an obscure department reference to try in case they might know what to do. My first proposal is a system wide memo that dictates a reporting process both by phone and email, or hopefully the link on the Intranet home page, so that a central point of contact can be established.

Second, once that process is in place, all of the information necessary to respond to the alert needs to be immediately available to the responders. It should not take a lengthy reporting process that requires manager approval, in triplicate, with signatures, to find out the name of an infected server's "no longer working here" owner, just to start the whole process over again. A group needs to be made responsible for access to all of the data, if it is deemed too sensitive to share openly. Then they could dictate limited access to a specialized group of employees with a business need to know. Additionally, no group should have the right to refuse the information to those employees without an immediate managerial interference being available. Third, every system attached to the intranet should be assigned an IP address, even the workstations, or else a way of registering the physical location of dynamically assigned IP addresses should be established. The information for every computer system should be maintained in a central location, with regular updates to the contact information, with failure to comply with the update requests resulting in a penalty or disconnection of the system, or some other drastic escalation remedy. If this particular system upgrade had been in place on the day Nimda hit our system, it would have been dead in hours, instead of days.

Fourth, a Security Incident reporting process needs to be built into the training of all employees, new and otherwise, so that everyone knows what to do in the event of a cyber emergency, and any other emergency for that manner. If we are going to have regular fire drills, we should have regular cyber drills as well.

Fifth, compliance with the Computer Security Hardening Standards for systems needs to be made mandatory, with training made available to all groups with more than one system attached to the Intranet, and no system should be connected to the intranet without having first been given a security vulnerability assessment score or report. This would prevent any administrator from deciding that they have performed "due diligence" because they did their best, and then just plugging in a server that is swiss cheese to meet a deadline. They should have a source where they can find the information they need to properly harden their servers, and test them for vulnerabilities prior to connection to the intranet. In our company, that source is actually available, but there is a significant charge for the services to verify the vulnerabilities have been eradicated.

Sixth, the CSIRT should be made up of uniquely qualified individuals who regularly, as in not less than once per month, get together and train as a team on incident response scenarios, so that bugs in the system can be worked out, and skill levels can be maintained, so that responses go smoothly. It makes no sense to have a "team" of individuals. Each team needs to work together, regularly, to establish the type of integrated response that really effective teams everywhere have accomplished. Think of a Police SWAT unit. Each member may be trained in some specific specialty, but as a group, they all operate in a coordinated unit. They spend many hours each month practicing their entry techniques, and their tracking techniques. A CSIRT should do the same, so that everyone on the team knows who is doing what, and who can best handle each part of a situation. Even if it's just that everyone knows who is taking the detailed notes on this incident for later correlation into an acceptable forensic document.

Seventh, an emergency contact list for EVERY employee, no matter what their function, should be created, and updated MONTHLY, until they leave the company, so that if anyone needs to contact anyone, in an emergency, for any reason, they can do so. This information would obviously be sensitive, and should be stored in a safe location, with controlled access, but available to any designated member of the CSIRT team.

Advanced level information needs to be available as well. I just did a search on the intranet of my company for the word Nimda, and I came up with only THREE hits. None of which had anything to do with how to stop, contain, or eradicate the worm, much less where to go for information. Visiting the virus support web page is like journeying into a one-man war. This poor guy is doing a wonderful job of keeping up to date with all the latest stuff, but he can't do it alone, and the intranet doesn't even link him in a search for Nimda, Code Red, Code Red II etc…..Again, a central clearinghouse would solve this. Ironically, when searching the Virus Support web site for a mention of CSIRT, there is none, and the only CERT link takes you to www.cert.org, which is not my company CSIRT.

I did a search on the intranet and found an Incident Response Process Guide, but it contained only information concerning how to handle Physical Security incidents. There was a tab listed labeled "Computer Security", so I clicked on it, fully expecting to find information about what to do in a Cyber Emergency, of if you are infected with a virus, and instead, I found a physical security document with no references to Information Security at all. It did tell you how to lock your Laptop to a secure location in your hotel room while traveling, which was an interesting document in itself. Did you know you could "Request" a wheelchair accessible room because the support bars bolted to the wall in the bathtub make an excellent place to lock your Laptop lock cable around? I never would have considered it myself. I think I would rather carry the thing with me all the time than rest it on the toilet, locked to a wheelchair user's support structure.

While the information I found is necessary and helpful, it does not even begin to address the issues surrounding a virus infection or Nimda attack. I visited our Virus Support page for information and found a very comprehensive page, with a lot of vital information, good links to updates for anti-virus programs, how to clean machines already infected, searchable archives of information. But no links to a Cyber Emergency Response team, or any other information about the CSIRT at all. If I wasn't actually certain it existed, I would not be able to find anything about it on our Intranet. When I did a search for CSIRT on our Homepage, I found four links. One was to a mention of the Corporate Incident Response Chart listed above, and the other three were to outside agencies listed in our internal documents at other locations around the Intranet.

For my eighth recommendation, I would like to submit that each and every member of the CSIRT be given an identity card that describes their special duties, so that if a physical premises response is necessary, they won't have only a business card and a driver's license to show onsite security that they belong there. This may be one of the most important pieces in the Incident Response puzzle, because if the responders can't access the infected machines, it may not be possible to remove it from the net, thereby causing additional infections during the time it takes to either lock it down at the subnet switch (which still leaves the machines on the same subnet vulnerable to increased infection potential), or find the SA responsible for the box and make them fix it.

There were many ways information could have been disseminated during this crisis. None of which were utilized to their fullest capacity. One of the things I saw almost immediately was the Mass Email, but it contained only generic information and directed users with specific problems to return to a central page for additional information. Since everyone who didn't understand the problem all went to the internal page for info., the server promptly crashed, so that was not a good way to do the information exchange. No attempt was made to place door posters, remote access gateway banners, or send out a mass voice mail directing people what to do. They might have even placed a note on the home page of the Intranet, but they didn't.

I got the impression that the CSIRT was not activated for this response, and they should have been. But I am also not familiar with what the CSIRT activation process is, so I'm not certain that it wasn't.

There were no instructions given out on "how to determine if a system is infected" or "how to clean a box prior to reconnecting it to the net". In fact, on many of the conference calls, I repeatedly heard the Anti-virus guy stating that no verified cleaning mechanism had been established, and that all server owners were to "nuke from high orbit" and rebuild the boxes from scratch, restoring from a known good backup. Many system administrators refused to do this, and began implementing fixes located on Microsoft's website, and at other locations. No attempt was made to verify that all machines were properly cleaned, or that no backdoor's were left behind by the infection.

No attempt was made to verify that any of the fixes put into place met the security standards in place for our company. In fact, no one group seemed to even really be in charge of the whole process. It was a concerted effort by the many to solve the problem. But the definition of solving the problem seemed to be to remove all evidence of infection, not to verify cleaning and sanitation had occurred, and that the network didn't have boxes on it with back doors built into them.

1. A CSIRT needs to be officially sanctioned, trained, and equipped to respond to any situation at any time. They need to have free roam of the physical locations where computers and system equipment may be stored, and they need access to every type of information relating to the owner of every computer system in the company.

2. Regular Preparation and "fire drill" type scenarios need to be conducted throughout the enterprise to assist employees in the proper actions to take, and avenues to explore in the event of a system wide incident like Nimda.

3. A single group needs to be placed in charge of collecting and maintaining the appropriate patches and fixes to vulnerabilities inherent to the software programs used throughout the system, and they need to provide controlled access to these pieces so that verification of proper implementation can be established.

4. System policies need to be established, published, taught, and enforced at all levels of system access, from dumb terminals to mainframe servers. One group should maintain all of the records of who uses what, and how, and make certain that no system is connected to the net without proper testing and accreditation of some type.

5. Regular vulnerability scanning needs to continue (it is already in place and active) but needs to be followed up with a report to an enforcement group who is responsible for tracking down machines that are vulnerable or infected, until they are found and eradicated completely, or removed from the net.

6. A central location of information needs to be established, and responsibility for regular updates to that information needs to be assigned.

7. A central monitoring station needs to be established as a single point of contact for all Security emergencies so that only one source of information is established, and the flow of information surrounding an event can be controlled in content and dissemination.

8. An emergency notification system needs to be made available to all members of the corporation, whether as an email, phone, or panic button on the Home page of the Intranet, that leads immediately to the Central Monitoring Station for immediate action.

References:

Advisories and References:

4. CERT® Advisory CA-2001-26 Nimda Worm -- http://www.cert.org/advisories/CA-2001-26.html

5. ADVISORY 01-022 "Mass Mailing Worm W32.Nimda.A@mm -- http://www.nipc.gov/warnings/advisories/2001/01-022.htm

6. FedCIRC Advisory FA-2001-26 Nimda Worm -- http://www2.fedcirc.gov/advisories/FA-2001-26.html

Code Rainbow Loose In The Wild - Security Experts -- http://www.newsbytes.com/cgi-bin/udt/im.display.printable?client.id=newsbytes&story.id=170225