Nimda - A Step Into Complexity

Matthew Rothschild

Matthew Rothschild
Assignment Administrivia. 2.0; GSEC Practical v1.2f
11/27/01

Nimda – A Step Into Complexity

I have had only one direct experience with a virus. It was on my home PC in 1999, when I had just graduated from college. Based on the date it happened (April 26) and the payload, I assume that it was the Chernobyl virus that forced me to reformat my hard drive. For the last three years, I have been exposed to viruses in only two capacities: As an employee that has to put up having anti-virus software on his PC at work and as an IT Auditor who asks clients what their anti-virus policy is. The latter is usually in relation to a financial audit to determine if I need to make a suggestion to management to make an improvement.

Spreading of Nimda

Prior to reading about the most recent virus, Nimda, I thought that asking about the anti-virus software was enough. Now after doing a little more research specifically on Nimda, there are some additional questions that I, and IT Administrators need to be asking. To name a few:

> Do you allow executable files to enter your network through e-mail?
> Do you allow Internet users to access external mail sites?
> How up-to-date are the patches on your Internet applications and servers?

The best way to keep a virus out of your organization is to prevent the code from getting to your network. Nimda tries to do this four ways.

The first is what is now a common source of viruses: e-mail. 'Nimda is a complex virus with a mass mailing worm component which spreads itself in attachments named README.EXE.' This leads to my first two questions:

(1) Is the user community restricted to using e-mail for business purposes only? If so, most users don't need to receive .exe files. This allows an administrator to scan incoming e-mails and prevent possibly malicious attachments from ever getting into the organization.
(2) How does e-mail penetrate the organization's network? The only way to scan all incoming e-mail is to first understand what your users are doing. If they access public mail sites on the Internet, such as Yahoo! and Hotmail, this can be a danger to your organization. Those e-mails are not being scanned, which means an attachment can be downloaded onto a user's PC without being scanned.

The second is by looking for a back door into your IIS Web Server through several known vulnerabilities. Two more questions are:

(3) How current are your patches and service packs on your web server? Any administrator who is not current could have a vulnerable network.
(4) Do you have appropriate incident handling procedures? Previous viruses, such as Code Red, will leave backdoors into your system. One of which was used by Nimda to propagate itself. An administrator should have procedures in place to ensure that their system is safely restored after being compromised. Creating a plan on the fly is prone to missing steps.

The third way Nimda can get into your system is by a user browsing an infected web site with a vulnerable web browser.
(5) How current are your applications? New versions of applications, especially Microsoft's, usually contain security updates.

The fourth way Nimda can get into your system is through network shares. Actually, this almost does not fit under this heading. In order to take advantage of network shares, Nimda must already be inside your internal network.
(6) How quickly are you alerted to and how quickly do you react to virus warnings? Even though every major anti-virus vendor had published solutions to fight against the virus, many networks were affected for the entire week.

Initial Loading of Nimda

After finding out about the different routines that the virus runs through, the first thing that came to my mind was that this virus is quite complicated. Especially since the following is only the initial loading.

The first thing the virus does is perform a number of steps to install itself.

The first step is for the virus (ADMIN.DLL) to make sure that it isn't interfering with itself. A mutex named "fsdhqherwqi2001" is searched for and if found, means that the virus is already running on that machine. I will address this situation later. For now, lets say that the mutex was not found. The virus then proceeds with the next step after creating the mutex named above.

The next step is to actually overwrite or create the "mmc.exe." file in the Windows directory (and possibly in the Windows startup file). At the same time, it uses stealth. The executable is marked as system and hidden. This, in combination with a quick edit of the registry to hide file extensions in Microsoft Explorer allows the virus to exist without obvious detection.

Now it runs the newly created executable with a "-qusery9bnow" string. Keep in mind that the mutex exists, which causes a different path in the program to be taken.

The new path directs the worm to hide its tracks by deleting any "readme*.exe" files from the temporary folder. This occurs successfully about 20% of the time. Additional deletions of temporary files will occur later in the payload section.

At this point, the original file stops executing. However a few more initial loading steps are still left. The worm checks some global variables to find out information about the Windows directory, the system directory, the temp directory, and whether or not it is running NT. It also grabs the IP address of the local host, which it will use later for its network functions.

The second to last initial step is to create the e-mail that will be used later on for the "IFRAME" trick.

> "…the worm prepares its MIME-encoded copy by extracting a pre-defined multi-partite MIME message from its body and appending its MIME-encoded copy to it. The file with a random name is created in a temporary folder"

The last initial step is to attempt to hide itself as a thread to the explorer.exe process. This prevents the worm from being shown in the Task Manager. On Windows 9X, it may run as a service in the background.

Payload and Damage

After setting itself up in some critical directories and attempting some stealth routines, Nimda is ready to run the more damaging part of its program.

It uses three different stealth mechanisms in combination to infect executables.

(1) It acts as a prepending virus. That is, it takes the original or victim file from its original location and places it the end of the virus instructions. This new infected file is then used to overwrite the original file. The original program can now be called by a user and not even know that the file is infected. However, the virus is still able to run as well. Curiously, Nimda does not infect WINZIP.EXE files.
(2) The old icon of the original file is kept in working order. Even after the original file is deleted, the original icon still works. This is because Nimda takes the resource data (including the icon) of the original file and copies it to the new file. Traditionally, a prepending virus can be discovered in Explorer by finding an icon that is no longer working.

(3) The temporary files used to create the new file are marked for deletion to hide the activity of the virus. If the platform is not NT based, a "move" call is used to mark the file for deletion upon reboot. If the platform is NT based, a "wininit.ini" file is created with a line to delete the temporary files. This will cause a problem, which is discussed next

As each executable is overwritten, a new "wininit.ini" file is created. This is a problem because only the last two temporary files will be deleted on reboot. This can cause hundreds of temporary files to be retained and result in drives becoming full (just a side effect).

The next key steps Nimda takes is to copy itself to the LOAD.EXE file and to the RICHED29.DLL file (may or may not already exist). It then edits the SYSTEM.INI file to allow LOAD.EXE to run at startup. The RICHED29.DLL file is also copied to every directory where .doc or .eml files are located. These steps greatly increase the chances of the virus running.

Nimda then continues to distribute itself in a new and unique way. It infects .htm, .html, or .asp files with names containing 'default', 'index', 'main', or 'readme'. Nimda first looks for these files on local and shared remote drives, then mails the MIME-encoded copy prepared as described above and attaches a javascript program that will open the mailed file. This also causes anyone who opens these files to be come infected.

The next surprise that Nimda has is to create a couple of security holes. The virus installs an open file share on C:\. This allows anyone on the network full access to this directory. It also moves the Guest account into the Administrator group. All it takes is an outside hacker to attempt a login as Guest with a blank password to (further) compromise your system.

The mass mailing routine has several parts. It first uses the Internet Explorer browser cache and the default MAPI mailbox to search for and generate a list of possible e-mail addresses. The subject of the message can be pulled from the MAPI mailbox or is left blank. The file attached in the e-mail messages is README.EXE, which may be run automatically in some cases. (Thus it is still considered a worm) An SMTP server within the worm is used to send out the e-mails.

As the mailing is occurring, Nimda is running threads, which operate an attack using several known vulnerabilities. The attacks are run against randomly generated IP addresses. An example of an attempt is described below:

> "W32.Nimda.A@mm attempts to infect unpatched Microsoft IIS web servers. On Microsoft IIS 4.0 and 5.0, it is possible to construct a URL that would cause IIS to navigate to any desired folder on the logical drive that contains the web folder structure, and access files in it." [2]

Other vulnerabilities that Nimda looks to exploit include a couple of backdoors left behind from Code Red II. If the selected IP address is not vulnerable to any of these attacks, it returns and generates another random IP address. If it is successful, the virus begins running on the new host as described above.

What needs to be done?

The specific steps needed to stop Nimda depend on what Anti-virus software and Operating System is being run. Rather than go into that detail, these are a summary of thoughts on what needs to done on a high level to stop the ease of which viruses attack organizations.

Security and System administrators, with the support of top-level management, need to have the correct answers to the questions discussed above with how Nimda spreads. Organizations need to start with the creating policies to enable the administrators to perform their duties effectively. To also increase the administrators' effectiveness, the procedures discussed above should developed and followed.

I should also note that a private company has the capacity to do only so much to protect its software. As viruses and software gets more complex, software companies need to improve their standards. Publishing a patch or upgrade after a security vulnerability has been exploited does not do a lot of good for their customer. Having thousands of different patches that must be tested, sometimes debugged, and applied to multiple servers can make an administrator's day a little more than frustrating. On what can be done to improve Internet security, Richard Penthia, Director of CERT states:

> "[There is] a strange kind of complacency. The marketplace isn't demanding higher-quality products. I don't know where to impart the blame. Users have to recognize that, until they demand a higher-quality software [without bugs or flaws], they won't get it. What might have been acceptable five years ago as a commercial product isn't acceptable today. Eventually, the marketplace is going to recognize that, especially after the latest round of viruses. A huge amount of money was spent on cleanup. We need to virus-proof our systems. We know how to design systems so they can resist viruses. When importing software from the outside into a machine, you can constrain its execution. We just haven't designed systems using that technique. We have to change the way we write application software. And we need to really pay attention to the software flaws that turn into vulnerabilities out there. All the software vendors need to step up and do a better job. " [5]

In other words, a cheaper and more effective way is to eliminate the vulnerabilities.

References

[1] http://www.antivirus.com/vinfo/virusencyclo/default5.asp?VName=PE_NIMDA.A&VSect=T

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

[3] http://vil.mcafee.com/dispVirus.asp?virus_k=99209&

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

[5] http://biz.yahoo.com/bizwk/011106/ljy3yaexzzgokvob8hw_cg_1.html

[6] http://www.curenimda.com/desc01.htm

[7] http://dailynews.yahoo.com/h/nf/20011113/tc/14733_1.htm