Case Study: Welchia Worm vs. Policy Makers Fighting Malware with Policy, not with Fire

Submitted by Benjamin Corll

GCIH Practical Assignment (v.3)

Date Submitted: 12/24/03
Date Resubmitted: 02/09/04

Table of Contents

Deleted: 2
Inserted: 2
Deleted: 3
Deleted: 2
Inserted: 2
Deleted: 5
Deleted: 2
Inserted: 2
Deleted: 7
Deleted: 2
Inserted: 2
Deleted: 9
Deleted: 2
Inserted: 2
Deleted: 15
Deleted: 2
Inserted: 2
Deleted: 17
Deleted: 2
Inserted: 2
Deleted: 18
Deleted: 2
Inserted: 2
Deleted: 22
Deleted: 2
Inserted: 2
Deleted: 22
Deleted: 2
Inserted: 2
Deleted: 23
Deleted: 2
Inserted: 2
Deleted: 33
Deleted: 2
Inserted: 2
Deleted: 35
Deleted: 2
Inserted: 2
Deleted: 37
Deleted: 2
Inserted: 2
Deleted: 38
Deleted: 2
Inserted: 2
Deleted: 41
Deleted: 2
Inserted: 2
Deleted: 43

Introduction

The year 2003 has been riddled with network-borne worms. These worms have cost corporations millions of dollars. This loss is both from lost revenue and from having to pay system administrators to clean and patch machines (incident response). I believe these corporations did not have to pay this type of price. If they had followed their own security policy and actually enforced that policy, they would have faired better during these types of incidents.

Two of the most destructive worms in 2003 were the Blaster and the Welchia (Nachi) worms. Both of these worms cost organizations millions of dollars in lost revenue and man-hours cleaning up during and after the outbreaks. However, these both were preventable if only the organizations had followed their own security policies.

Every company from the Fortune 500 companies to the small “mom and pop” ISP’s have written policies and procedures. These policies are used to direct which organization or person is to perform which act. It gives a logical frame on which to build the organization’s hierarchical structure.

Looking at our corporate approved policies, the system administrator is responsible for the installation of all patches that a vendor recommends. It goes on to state that the network administrators are responsible for port management, router access lists, and router changes. And the policies lay out what the individual end users are to be responsible for.

Usually it’s the system administrators whom are tasked with keeping the systems they maintain healthy and in proper working order. Then the network administrators are tasked with keeping the network up and running and as uncluttered as possible. Part of their combined job is the health of the company’s network infrastructure. For the system administrators, this includes the patching and security of the machines in their area of influence.

According to our policies and procedures, the end user is not tasked with maintaining their own machine. The only concern with the machine that the end user has to take into account is not disabling any of the security measures on the machine. This includes both the antivirus agent that runs on the machine and the live update mechanism for Microsoft Windows machines.

Understandably, system administrators get overworked and get bogged down with different tasks. They have to maintain servers, desktops, user accounts, network shares, and work with network administrators. It can seem trivial at times to ensure that certain patches get pushed to each and every machine. Especially when there are so many patches that are released by a vendor. Microsoft alone released 51 separate patches in 2003. Some of these patches superceded others, but they still released 51 of them. Keeping up with each patch and ensuring every single machine receives each pertinent update can be a full time position of its own.

The problem comes when the system administrator fails to add a critical patch that a vendor puts out and recommends immediate action. The corporate policy states that the system administrator will follow those recommendations. However, they are so consumed in other tasks that they fail to follow those guidelines. There are no repercussions on the system administrator. There are also no mechanisms in place that will verify if a system administrator is in compliance with said policy. There are also no rewards or incentives that would motivate an administrator to keep up to date on their patches.

Until corporations start holding the responsible parties accountable for the work that is outlined in the corporate sponsored and 'signed off' policies and procedures, they will never have a secure environment. Patches will continue not to be added to machines. And then viruses and worms will continue to run rampant through corporate networks.

This has to be approved and held accountable at least at the CIO or CTO level. It is their budget that is used for patch management. It is also their budget that is used to fight the onslaught of a viral infection. Patch mitigation starts at the top and must flow down the organization. If the executive staff cannot convey the critical need for patches during normal operations, then how can one expect the system administrators to have a sense of urgency when something critical does face the infrastructure?

If the organization continues to fail to hold individuals accountable on these policies, they will continue to feel the effects of these worms. Each worm has gotten just a little smarter and little more powerful. It's just a matter of time before a hybrid worm comes out that can effect multiple operating systems using multiple exploits and do an undeterminable amount of damage.

This paper is written to bring awareness to the industry that passing a corporate policy does not ensure the policy will be enforced. This will be done as I describe and outline an outbreak of a malicious worm onto a corporate network. The fact of the matter is that the worm outbreak would not have been as damaging if the policy that was signed off by the CTO had been enforced.

I also want to outline that properly patching a machine and keeping the security of each machine up to date is essential in the security of the network. Locking all the doors by installing a firewall and monitoring the network with IDS is not the only avenue to a security posture. An organization must ensure that their administrators are keeping the machines current. By patching a system, the administrator can very well deter an attacker as they look for the easiest targets.

The Malware / Exploit

The exploit that I am going to reference in this paper is the Welchia worm. I am going to show that this worm was able to cause significant amounts of damage simply because the organizations were not following their own policies.

I will list the information pertaining to the worm gleaning information from different sources. This will show what the worm is written to do, its characteristics, and how it spreads. I will also highlight on how the spreading of the worm would have been suppressed by proper policy management.

By the end of the paper, I will have shown that with this specific piece of malware, if system administrators had patched their machines, the worm would not have been able to spread.

Here is specific information on Welchia from Symantec (www.symantec.com):
Name:
W32.Welchia.Worm:

Also Known As:
W32/Welchia.worm10240 [AhnLab], W32/Nachi.worm [McAfee], WORM_MSBLAST.D [Trend], Lovsan.D [F-Secure], W32/Nachi-A [Sophos], Win32.Nachi.A [CA], Worm.Win32.Welchia [KAV]

Type: *Worm*
Infection Length: *10,240 bytes*

Systems Affected:
The worm is not picky; it is an equal opportunity infector. It will infect a machine that is running server just as well as it will infect a machine running workstation.

The breakdown of systems infected (from Microsoft):
Microsoft Windows Server 2003, Standard Edition
Microsoft Windows Server 2003, Enterprise Edition
Microsoft Windows Server 2003, Datacenter Edition
Microsoft Windows Server 2003, Web Edition
Microsoft Windows Server 2003, 64-Bit Datacenter Edition
Microsoft Windows Server 2003, 64-Bit Enterprise Edition
Microsoft Windows XP Professional
Microsoft Windows XP Home Edition
Microsoft Windows XP Media Center Edition
Microsoft Windows XP Tablet PC Edition
Microsoft Windows XP 64-Bit Edition Version 2002
Microsoft Windows XP 64-Bit Edition Version 2003
Microsoft Windows 2000 Professional

Microsoft Windows 2000 Server
Microsoft Windows 2000 Advanced Server
Microsoft Windows 2000 Datacenter Server
Microsoft Internet Information Server 5.0
Microsoft Windows NT Advanced Server
Microsoft Windows NT Server
Microsoft Windows NT Server 4.0 Terminal Server Edition
Microsoft Windows NT Server, Enterprise Edition
Microsoft Windows NT Workstation 4.0

Systems Not Affected:
Linux, Macintosh, OS/2, UNIX, Windows 3.x, Windows 95, Windows 98, Windows Me

Protocols/Applications:
Remote Procedure Call (RPC) over port 135
Distributed Component Object Model (DCOM)
World Wide Web Distributed Authoring and Versioning (WebDAV)

Here is a definition of the protocols and a little bit about what each of them do:

Taken from www.whatis.com*: DCOM (Distributed Component Object Model) is a set of Microsoft concepts and program interfaces in which client program objects can request services from server program objects on other computers in a network. DCOM is based on the Component Object Model (COM), which provides a set of interfaces allowing clients and servers to communicate within the same computer (that is running Windows 95 or a later version). One of the DCOM interfaces is with RPC.*
(Note: This is the interface that the Welchia worm uses.)

http://ftp.ics.uci.edu/pub/ietf/webdav/intro/webdav_intro.pdf

WebDAV is extending HTTP to provide a standard infrastructure for asynchronous collaborative authoring across the Internet. The WebDAV extensions support the use of HTTP for interoperable publishing of a variety of content, providing a common interface to many types of repositories and making the Web analogous to a large grain, network-accessible file system.

Known Variants:

Welchia is a variant of the MS Blaster worm. Blaster is a malicious worm that also targets Microsoft systems and exploits them using the same MS RPC DCOM vulnerability that Welchia uses. However, Welchia added the MS03-007 (WebDav exploit - had to one up the Blaster author).

Unlike Blaster, which had variants (teekids.exe and penis.exe), there are currently no known variants in existence for Welchia.

CVE References (both still candidate numbers):
CAN-2003-0109 - Buffer Overflow in NTDLL.DLL
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0109

CAN-2003-0352 - Buffer Overflow in Certain DCOM Interface for RPC
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0352

W32.Welchia.Worm is a worm that exploits multiple vulnerabilities:

- *The DCOM RPC vulnerability (described in Microsoft Security Bulletin MS03-026) using TCP port 135. The worm specifically targets Windows XP machines using this exploit.*

- *The WebDav vulnerability (described in Microsoft Security Bulletin MS03-007) using TCP port 80. The worm specifically targets machines running Microsoft IIS 5.0 using this exploit. As coded in this worm, this exploit will impact Windows 2000 systems and may impact Windows NT/XP systems.*

W32.Welchia.Worm does the following:

- *Attempts to download the DCOM RPC patch from Microsoft's Windows Update Web site, install it, and then reboot the computer.*
- *Checks for active machines to infect by sending an ICMP echo request, or PING, which will result in increased ICMP traffic.*
- *Attempts to remove W32.Blaster.Worm.*

Symantec Security Response has developed a removal tool to clean the infections of W32.Welchia.Worm.

Signatures of the attack:

The Welchia worm sends out massive amounts of ICMP Echo Requests. The machine not will be sending a few ICMP packets; it will be sending thousands of ICMP packets a day. These requests have a signature such as the one below.

From (http://www.esphion.com/ETB-82.htm): (the IP Addresses have been removed)

```
0x0000   4500 005c bfe9 0000 6b01 a0a7 ---- ----        E..\....k.......
0x0010   ---- ---- 0800 2c8b 50de 2541 aaaa aaaa        ......,.P.%A....
0x0020   aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa        ................
0x0030   aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa        ................
0x0040   aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa        ................
0x0050   aaaa aaaa aaaa aaaa aaaa aaaa                  ............
```

The packet length is always 92. The TTL which is used when the packet is created is 128. The packet payload is filled with 0xAA (decimal 170).

Another “signature” of the attack, is the worm will add two services along with registry entries:

Service Name: RpcTftpd
Display Name: Network Connections Sharing
File: %System%\wins\svchost.exe

Service Name: RpcPatch
Display Name: WINS Client
File: %System%\wins\dllhost.exe

Both of these services will be set to start automatically.

The registry entry that is added is:
HKEY_LOCAL_MACHINE>SYSTEM>CurrentControlSet>Services>

subkeys:
RpcPatch
RpcTftpd

Note: You may simply right click on these subkeys and delete them if you are manually removing the worm.

The worm is programmed that on January 1, 2004, when the program executes, it will remove itself. Since this worm was released into the wild in August of 2003, this was a very long time for the worm to be active before it would delete itself!

Here is a chart generated at: http://isc.sans.org/port_details.html?port=0
See diagram 1:

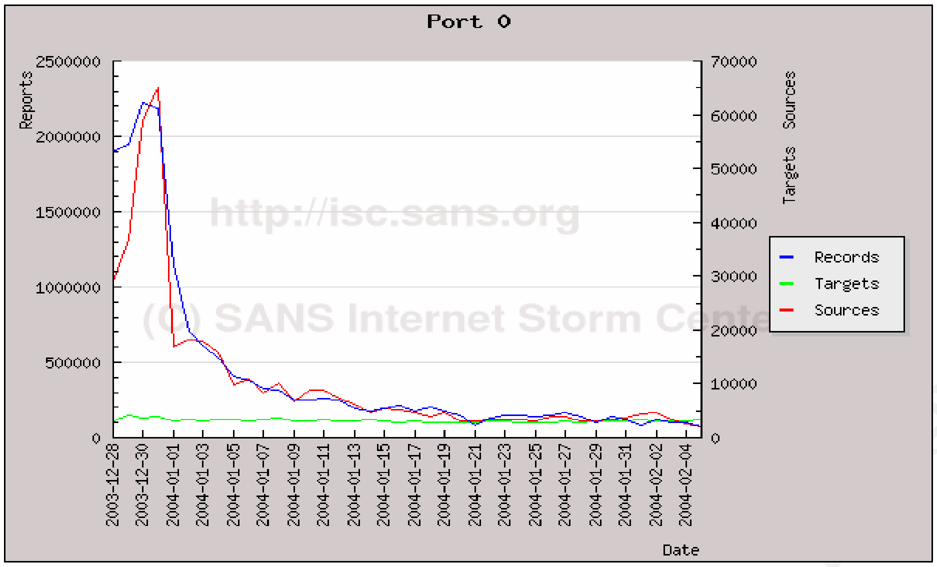


DIAGRAM 1

In my personal testing (in my lab environment), when I turned my clock forward to January 1, 2004 and rebooted the machine, my machine did NOT remove all entries of the Welchia worm. The service was no longer listening on port 707, but the entries in the registry were not removed. The machine did stop sending large amounts of ICMP traffic. It looked to be a dormant worm.

This was confirmed if you look at the diagram above. As the date rolled forward into 2004, there was a dramatic decrease in ICMP traffic.

Malware / Exploit Details

On July 16, 2003 Microsoft and LSD (Last Stage of Delirium Research Group) released information on a vulnerability dealing with the RPC DCOM. (Remote Procedure Call) (Distributed Component Object Model). The problem actually did not lie with the RPC service but with the DCOM service.

The description from LSD's webpage (http://lsd-pl.net/special.html):

This is a stack buffer overflow vulnerability that exists in an integral component of any modern Windows operating system, an RPC interface implementing Distributed Component Object Model services (DCOM). In a result of implementation error in a function responsible for instantiation of DCOM objects, remote attackers can obtain remote access to vulnerable systems.

The internal experiments conducted in the LSD labs confirmed that this vulnerability affects all default installations of Windows 2000 (sp 1-4), Windows

XP (sp 1) and Windows 2003 Server (regardless of the service packs installed). According to definitely more extensive tests performed by Microsoft (security bulletin), Windows NT 4.0 and Windows NT 4.0 Terminal Services Edition are also vulnerable to this threat.

The buffer overflow is in the Microsoft RPC implementation. Microsoft, in their MS03-026 bulletin, states "There is a vulnerability in the part of RPC that deals with message exchange over TCP/IP. The failure results because of incorrect handling of malformed messages. This particular vulnerability affects a Distributed Component Object Model (DCOM) interface with RPC, which listens on TCP/IP port 135. This interface handles DCOM object activation requests that are sent by client machines (such as Universal Naming Convention (UNC) paths) to the server."

Almost a month later on August 10, 2003, a worm that was soon to be named MSBlaster was released. This was a very fast moving / fast spreading worm. According the write-up on Symantec's webpage, this worm infected approximately 330,000 machines.

This worm was relatively easily mitigated by the fact that all a user had to do was patch their machine before the release of the worm. But the clean-up of the worm was anything but easy. After the worm's release, companies were quick to come out with counter measures to help in the identification, containment, and the remediation of the worm.

One of the "nasties" that the Blaster worm was coded to do was perform a Distributed Denial of Service (DDoS) against windowsupdate.com. In our internal DNS servers, we made an entry for windowsupdate.com and directed it to a specified log server. This allowed us to prevent our network from assisting in a DDoS against Microsoft, but it still allowed us to log which machines were attempting to send the traffic.

With clean-up on this worm going well and starting to look up, things got a lot worse on the 18th of August. This was due to a dramatic increase in ICMP traffic on the Internet. It was later noted that the increase in ICMP traffic was accredited to the newly arrived Welchia worm.

From a 10,000-foot view, the Welchia worm looked to be a good worm. It was 'designed' to aide system administrators clean a machine that was infected with the MSBlaster worm.

However, the problem with this 'good' worm is it would take over a machine by attempting two different exploits – the RPC DCOM (like Blaster) and the WebDAV exploit. And it would also install a backdoor! It would do this without the user's permission and most times, without the user's knowledge.

Another bad habit that the Welchia worm was written with is it would flood a local network with large amounts of ICMP traffic (ICMP echo requests). If several machines on a small subnet (i.e. a network with a 56K connection or a business with an ISDN line) were infected this could cause a degradation or a complete denial of service (DoS).

By design, the Welchia worm was supposed to infect a machine, download the MS03-026 patch from Microsoft, and then remove the MSBlaster infection.

The problem with Welchia is the worm was a malicious worm. My reason for stating this is the amount of damage that this worm did on organizations. It also left a backdoor open on the machine. And the increase in the bandwidth utilization on infected subnets does cost organizations revenue. In my opinion, it went from being a 'white worm' to a 'black worm' for these very reasons. Yes, it would patch a machine and plug a hole in the security of the system. However, this is no different than a rootkit or a bot that owns a machine and then patches the hole it exploited so that no other 'hacker' or 'cracker' could exploit the same vulnerability and own the box themselves.

I do not believe it is feasible for system administrators or anyone in the security field to fight fire with fire, or in this case, fight a 'black worm' with a 'white worm'. Fires, like Internet worms, can behave in different ways. With fires, if the air in a room becomes so hot, the entire room can burst into flames in one huge explosion. This is called a flashover. Or a fire can start off on its own destroying a certain part of a forest. If left on its own, it will continue to burn and devour more trees. However, it can 'meet' up with another fire and become one. The force of the two united will far exceed the power of each separately. Some internet worms have the ability to lay dormant for a period of time, while others move with the speed of a flashover. In the case of Blaster and Welchia we had two different fires working toward a common destructive goal. They were both released and began simmering on their own. If left unattended, each worm on its own would continue to burn through a network, further infecting more and more machines. Eventually their destructive nature would have met up. The Welchia infections would have attempted to overtake the Blaster infections. This would have caused a merge between the two fires and caused more traffic and chaos on the network. This was the case in our network. Welchia overtook the Blaster infections and became the major blaze of our environment.

Much like the virus infections on a network, there can be separate infections or one major infection. They can spread rapidly, or they can trickle slowly from machine to machine. They can infect one machine and stop there. Or they can have logic written in to allow them to gather information about the network but not activate and cause any damage until a certain time (or a trigger). Then they may begin an attack such as a distributed denial of service (DDoS). The only limitation to the destructive force of a worm is the imagination and the technical ingenuity of the coder who writes the malicious code.

I am focused on why I believe this worm was so destructive. Many corporations have policies that state that all machines must be currently up to date on their patches and have antivirus software installed and updated. However, most corporations neither enforce nor comply with this policy. They have the “M&M” security mindset. They have a hard shell on the outside. They see this as the greatest threat so they install (firewalls, ACL’s, IDS sensors in the DMZ, etc) and other devices. But then they do not take the inside threat seriously. They have but a soft inside. Once a user is inside the network, they are a trusted user and can do just about anything as an authorized and trusted user. So all the ‘bad guy’ has to do is crack the exterior defense and gain entry. This contributes to why worms are able to propagate and damage corporate networks on such a large scale.

The Welchia worm is a dual pronged attack. It will use either the WebDAV exploit or the RPC DCOM exploit to take over a machine. Once it takes over a machine it will attempt to further propagate itself.

In order to explain exactly what Welchia did when it was executed on a workstation, I will reference Symantec’s website again (www.symantec.com):

When W32.Welchia.Worm is executed, it performs the following actions:

1. *Copies itself to:*

   %System%\Wins\Dllhost.exe

   NOTE: *%System% is a variable. The worm locates the System folder and copies itself to that location. By default, this is C:\Winnt\System32 (Windows 2000) or C:\Windows\System32 (Windows XP).*

2. *Makes a copy of %System%\Dllcache\Tftpd.exe as %System%\Wins\svchost.exe.*

   NOTE: *Tftpd is a legitimate program, which is not malicious, and therefore Symantec antivirus products will not detect it.*

3. *Adds the subkeys:*

   RpcPatch

   and:

   RpcTftpd

   to the registry key:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services

4. *Creates the following services:*

   Service Name: *RpcTftpd*
   Service Display Name: *Network Connections Sharing*
   Service Binary: *%System%\wins\svchost.exe*

   This service will be set to start manually.

   Service Name: *RpcPatch*
   Service Display Name: *WINS Client*
   Service Binary: *%System%\wins\dllhost.exe*

   This service will be set to start automatically.

5. *Ends the process, Msblast, and deletes the %System%\msblast.exe file, which W32.Blaster.Worm drops.*

My note: starting to eliminate its arch-enemy – Blaster. This is why the worm is categorized as a white worm.

6. *Selects the victim IP address in two different ways: The worm uses either A.B.0.0 from the infected machine's IP of A.B.C.D and counts up, or it will construct a random IP address based on some hard-coded addresses.*

   After selecting the start address, the worm counts up through a range of Class B-sized networks; for example, if the worm starts at A.B.0.0, it will count up to at least A.B.255.255.

My note: This is how the worm will decide which IP address to scan. The local class C is going to be scanned first. This is why the local subnet is swamped with ICMP packets.

7. *Sends an ICMP echo request, or PING, to check whether the constructed IP address is an active machine on the network.*

My note: This is where the traffic begins. The local subnet may soon be over-burdened with traffic (this is a prime symptom that something is wrong on the network!!!)

8. *Once the worm identifies a machine as being active on the network, it will either send data to TCP port 135, which exploits the DCOM RPC vulnerability, or it will send data to TCP port 80 to exploit the WebDav vulnerability.*

My note: This is the dual prong attack.

9. *Creates a remote shell on the vulnerable host that will reconnect to the attacking computer on a random TCP port, between 666 and 765, to receive instructions.*

   ***Note:** In the vast majority of the cases, the port is 707, because of the way the worm threading model interacts with the implementation of the Windows C runtime .dll.*

   ---

10. *Launches the TFTP server on the attacking machine and instructs the victim machine to connect and download Dllhost.exe and Svchost.exe from the attacking machine. If the file, %System%\dllcache\tftpd.exe, exists the worm may not download svchost.exe.*

My note: This is where the exploit code is actually downloaded and infects the machine.

11. *Checks the computer's operating system version, Service Pack number, and System Locale. It also attempts to connect to Microsoft's Windows Update and download the appropriate DCOM RPC vulnerability patch (see note from Network Associates on the hard coded links).*

My note: This is where the worm will attempt to download and patch the machine with MS03-026. This is where the machine is trying to clean a machine

12. *Once the update has been downloaded and executed, the worm will restart the computer so that the patch is installed.*

My note: If the patch was not able to be downloaded, the machine will not reboot.

My note: If the machine is patched and rebooted, it will still continue to be infected with the Welchia worm. Basically it will continue to scan it's local subnet and random IP addresses that it can attempt to infect and patch.

13. *Checks the computer's system date. If the year is 2004, the worm will disable and remove itself.*

My note: As I stated earlier, when tested, the machine did stop sending packets, but the registry settings where not removed.

According to Network Associate's webpage, the worm has hard coded links to the following patches: (http://vil.nai.com/vil/content/v_100559.htm)

- http://download.microsoft.com/download/6/9/5/6957d785-fb7a-4ac9-b1e6-cb99b62f9f2a/Windows2000-KB823980-x86-KOR.exe
- http://download.microsoft.com/download/5/8/f/58fa7161-8db3-4af4-b576-0a56b0a9d8e6/Windows2000-KB823980-x86-CHT.exe
- http://download.microsoft.com/download/2/8/1/281c0df6-772b-42b0-9125-6858b759e977/Windows2000-KB823980-x86-CHS.exe
- http://download.microsoft.com/download/0/1/f/01fdd40f-efc5-433d-8ad2-b4b9d42049d5/Windows2000-KB823980-x86-ENU.exe
- http://download.microsoft.com/download/e/3/1/e31b9d29-f650-4078-8a76-3e81eb4554f6/WindowsXP-KB823980-x86-KOR.exe
- http://download.microsoft.com/download/2/3/6/236eaaa3-380b-4507-9ac2-6cec324b3ce8/WindowsXP-KB823980-x86-CHT.exe
- http://download.microsoft.com/download/a/a/5/aa56d061-3a38-44af-8d48-85e42de9d2c0/WindowsXP-KB823980-x86-CHS.exe
- http://download.microsoft.com/download/9/8/b/98bcfad8-afbc-458f-aaee-b7a52a983f01/WindowsXP-KB823980-x86-ENU.exe

Once the worm determines the operating system it will use the links above to download the appropriate patch.

These are direct links to the patch. They are all still active and available as of the 1st of February 2004.

The Platforms/Environments:

The network that I will reference is a network set up at an insurance company. There are seven members of the incident response team that were involved in the 'actual' incident. They each used multiple machines and multiple tools in order to effectively battle this worm. Some of the tools used were: nmap, nessus, FoundStone tool suite, etc.

The default workstation that is issued to each employee is the Windows XP professional (workstation). Each machine has different tools and scripts on them as each individual adds to their toolkit. There is a lot of sharing of ideas and tools, but each machine is unique.

The machines on the local subnet are:

(7) Windows XP Professional workstations

- Hardware
  - PIII 800 MHz
  - 512 MB RAM
  - Intel based

- Applications
    - ISS tool suite
    - Foundstone tool suite
    - Windows version of nmap (3.0)
    - VMWare
    - SSH client

(4) RedHat 9.0 workstations

- Hardware
    - PIII 600 MHz (laptops)
    - 512 MB RAM
    - Intel based
- Applications
    - *nix based nmap
    - Nessus server & client
    - *nix based nbtstat tool
    - Scanssh

(2) RedHat 7.2 workstations

- Hardware
    - PIII 800 MHZ
    - 512 MB RAM
    - Intel based
- Applications
    - *nix based nmap
    - Nessus server & client
    - *nix based nbtstat tool
    - Scanssh
    - VMWare

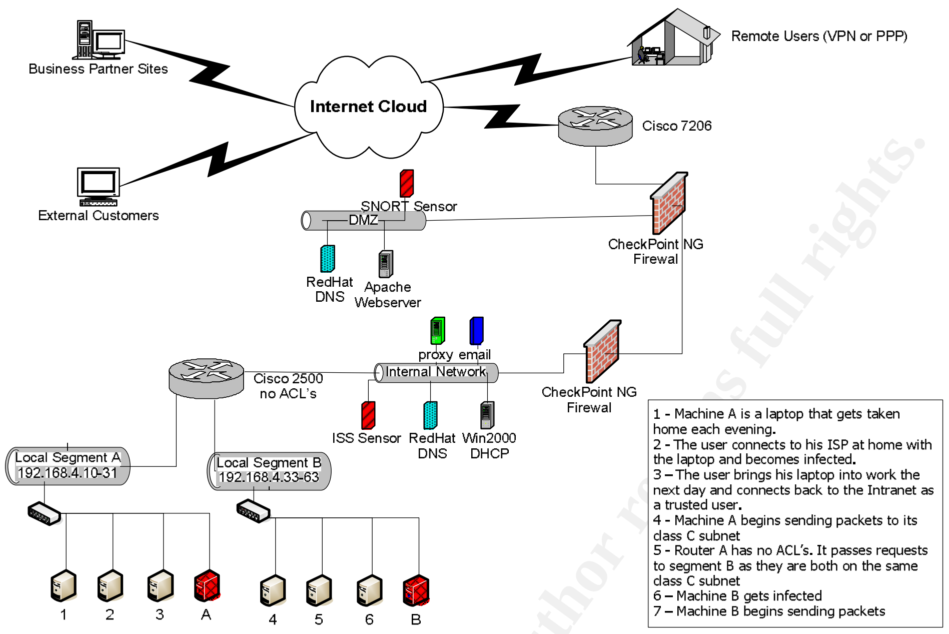
(1) Windows 2000 Advanced Server (Log Server)

- Hardware
    - P4 – 2GB Processor
    - 1 GB RAM
    - Intel based
- Applications
    - ISS tool suite
    - Foundstone tool suite
    - Windows version of nmap (3.0)
    - VMWare
    - SSH client

These machines are all connected to the internal network via a Cisco Catalyst switch. The only thing that separates the Incident Response Team (IRT) architecture from the 'standard' internal network setup is the IRT has a DSL connection back through Sprint. This allows the IRT to download tools from sites that the proxy has blocked.

Network Diagram

See diagram 2:



1 - Machine A is a laptop that gets taken home each evening.
2 - The user connects to his ISP at home with the laptop and becomes infected.
3 – The user brings his laptop into work the next day and connects back to the Intranet as a trusted user.
4 - Machine A begins sending packets to its class C subnet
5 - Router A has no ACL's. It passes requests to segment B as they are both on the same class C subnet
6 – Machine B gets infected
7 – Machine B begins sending packets

DIAGRAM 2

The network diagram above shows a network that is adequately protected from the Internet and from the Internet threats. All external facing machines are kept in a DMZ. There are multiple firewalls that are between users and the Internet. And there is actively monitored IDS sensors installed. This does create the illusion to upper management that the network is adequately secured. However, the fact of the matter is only the perimeter is secured. To truly have security in depth, you must also have security enforcement inside the network.

All internal users go through a proxy server versus going directly to the Internet. This helps for logging and for troubleshooting issues. This also allows the organization to restrict user's access to certain websites that may be harmful to the organization.

However, this does not help the organization when users are allowed to take home laptops. These laptops can be connected to the ISP via dial-up or high-speed connections. There are limited security precautions taken by ISP's to protect their users, as they don't want to hinder their user's access. If a laptop becomes infected at a user's home, they will carry that infection into their organization.

When the user does connect back to their infrastructure, they are treated as a trusted user. They can authenticate and are authorized to work on the network. However, they also are carriers of what ever infection they obtained on their home network. This will begin to use the laptop as a launch pad and start infected the rest of the network.

It is also possible for a laptop to be connected to both an ISP and the corporate infrastructure at the same time. Essentially the user is creating a bridge from the external network into an organization's infrastructure. This can be very dangerous for the organization as the laptop is a trusted user and has authorization to be on the network. Its traffic will most likely be trusted explicitly.

Stages of the Attack:

Welchia is a worm that will infect a machine and then attempt to spread. It does this fast and effectively. What makes it so effective is the multi-prong attack vector. With this being a worm and not an active exploit that is deliberately run by an end-user, the stages do vary from an exploit that must be manually ran by a user. This includes a virus that requires an action be ran (i.e. a user opening up an email attachment).

(1) Reconnaissance:

The reconnaissance stage is difficult to determine, as Welchia is a worm that was completely written and released prior to the exploitation attempt. The worm writer seems to have written in the code for the worm to scan on a local subnet (see "scanning" section for more information). Again, the reconnaissance most likely took place prior to the coding of the worm. The writer of the code may have had a specific target in mind or just wrote as 'generic' a worm as possible. Either way, the coder would have had to do their research prior to the release of the worm into the 'wild'.

The coding of this worm and the way it would chose the next host to scan is actually quite simple. This also helps explain why this worm (and the Blaster worm) was so effective. Rather than just chose a random IP address as the next target, both of these would scan the local subnet first. Logically speaking, if one machine is unpatched and vulnerable and on a network, chances are there is another machine that is also unpatched and vulnerable.

(2) Scanning:

Once a machine is infected, it will begin to scan the local subnet for vulnerable hosts. In order to do this, the machine begins by sending ICMP echo requests on the class C network. The hosts that respond will be the ‘chosen’ targets that the worm will attempt to spread to.

As I quoted from the Symantec webpage, the way the worm is coded to operate its scanning is:

1. *Selects the victim IP address in two different ways: The worm uses either A.B.0.0 from the infected machine's IP of A.B.C.D and counts up, or it will construct a random IP address based on some hard-coded addresses.*

   After selecting the start address, the worm counts up through a range of Class B-sized networks; for example, if the worm starts at A.B.0.0, it will count up to at least A.B.255.255.

2. *Sends an ICMP echo request, or PING, to check whether the constructed IP address is an active machine on the network.*

3. *Once the worm identifies a machine as being active on the network, it will either send data to TCP port 135, which exploits the DCOM RPC vulnerability, or it will send data to TCP port 80 to exploit the WebDav vulnerability.*

From TruSecure:

The TCP scanning behavior of the malcode will also generate a significant volume of ICMP Host-Unreachable messages, particularly if the host is located on an RFC1918 (privately addressed) or sparsely populated network. Welchia TCP scanning can be identified in that it sends 48 byte packets to port 139 of targets, having source ports which increment by one with each successive packet.

(3) Exploiting the system:

The Welchia worm will attempt to exploit a system using a couple of different methods. It will attempt to use the same RPC DCOM vulnerability that the Blaster worm used (MS03-026). Or, it will use a HTTP exploit that Microsoft highlighted in (MS03-007). Both of these exploits had patches that were readily available long before the exploit code was released.

By adding the ability to exploit the MS03-007 exploit, this pronged attack helped this worm to be a little more devious than the worm that it was actually written to eradicate.

Aside from the attack vector, what made this worm so harmful was its use of the ICMP echo packets. Although the single packet itself did no damage, an infected machine would send a single packet to every chosen address on the local subnet. This is in addition to the randomly chosen Internet addresses. This caused network latency and possibly could create a denial of service (DoS). If enough packets were sent to a router, that router would have to start storing (queuing) those packets while it determined the best way to route them. This could cause the tables on the router to fill and possibly cause a router to dump out and die. This too would cause a denial of service as that entire subnet containing the infection would now be an isolated network. There would be no entry or exit from that subnet until the router was recycled.

(4) Keeping access (Keeping control / ownership):

This worm overwrites several files and adds itself into the Registry of an infected machine. The first four steps that Symantec highlighted on are how the worm will keep access on the machine:

1. *The worm copies itself to:*

   %System%\Wins\Dllhost.exe

   ***NOTE:** %System% is a variable. The worm locates the System folder and copies itself to that location. By default, this is C:\Winnt\System32 (Windows 2000) or C:\Windows\System32 (Windows XP).*

2. *Makes a copy of %System%\Dllcache\Tftpd.exe as %System%\Wins\svchost.exe.*

   ***NOTE:** Tftpd is a legitimate program, which is not malicious, and therefore Symantec antivirus products will not detect it.*

3. *Adds the subkeys:*

   RpcPatch

   and:

   RpcTftpd

   to the registry key:

   HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services

4. *Creates the following services:*

   ***Service Name:** RpcTftpd*
   ***Service Display Name:** Network Connections Sharing*
   ***Service Binary:** %System%\wins\svchost.exe*

   This service will be set to start manually.

Service Name: *RpcPatch*
Service Display Name: *WINS Client*
Service Binary: *%System%\wins\dllhost.exe*

This service will be set to start automatically.

Another way that the worm kept ownership of the machine is it would open up a backdoor. This is commonly port (tcp) 707. However, it can be any range between the ports 666 – 765.

With the worm embedding itself into the system it is not restricted to running in virtual memory. The clean tool (or a manual removal) is required in order to clean the machine.

(5) Covering tracks

The worm does **not** do a very good job 'covering its tracks'. The reason that I state this is that fact that the machine is very noisy once it is taken over by the worm. It also opens a backdoor on the machine. These two facts lead me to believe that the worm does not do an adequate job covering its own tracks.

However, I do not believe the writer of the code intended to have this worm act in the background and not be seen. I believe the author quickly put this code together and released it prematurely. If this worm was to be a white worm, then the amount of destruction and financial loss that was realized by the worm lead me believe it was released prematurely. If I am mistaken, then this worm would have to be categorized in the "black worm" status and not in the "white worm" status even though it does close a vulnerability hole.

Of course, the writer of this worm could have been at odds with the writer of the MSBlaster worm, and wanted not to write a 'white worm' but a worm that would outdo the Blaster worm and remove the Blaster worm from machines it infects.

The backdoor that is opened on the machine is TFTP. When I walked through the steps of infection, the worm creates a service. The service that is created and set to run automatically is RpcTftpd. TFTP is trivial file transfer protocol. TFTP by default runs on tcp port 69. Again, as I stated earlier, this is not the port that is utilized by this worm. This is how the victim machine will actually download the malicious code from the 'attacking' machine.

We did have an ACL added to each of our routers that would explicitly block port 69. These were applied to both internal and external interfaces. This was put in place to attempt to help slow the spread of the worm.

The Incident Handling Process:

Defending against a malware attack is the same principle as the process I am going to walk through. I will focus on the steps we took to combat the Blaster / Welchia worms; however, these steps are broad enough to combat most malware problems. The difference between each malware incident will differ when it comes to port numbers to block and other specifics of the malware.

I am going to focus on how my organization handled the incident. I will try to keep this as general as possible so it will work with any organization and with any type of worm / malware exploit.

Preparation:

In order to defend against this attack, a system administrator must first be on the ball with patch management. The original patch to combat against the MS RPC DCOM vulnerability came out on July 16th. If all of our system administrators had installed this patch, which Microsoft labeled as a critical patch, then the bulk of vulnerable machines would not have been vulnerable to the worm.

If each machine in an organization had been patched according to the policies and procedures that were signed by the CEO, then this would not have become an incident. In all organizations, there will be a few straggler machines that I'm sure would have missed the patch. So it is very possible the worm would have affected the organization. However, if the majority of the machines had been patched then the organization as a whole would not have suffered anywhere near as much as they did. It was poor patch management and negligence on from both the management chain and the systems administrator that enabled this worm to do the extent of damage that was done.

Again, all the systems administrator had to do was download the critical patch from Microsoft:
http://www.microsoft.com/security/security_bulletins/ms03-026.asp
This is still a valid link as of February 1, 2004.

If the systems administrator had patched their machine for the vulnerability, that machine would no longer be vulnerable to the malware. I do understand that a lot of administrators are overworked and work in a reactive role. However, taking the proactive role would have been far less costly than taking the reactive role with patch management.

On September 10th (well after both Blaster and Welchia were released) Microsoft superceded their own patch (MS03-026) and issued MS03-039. This also was issued as a critical patch and once again should have been installed on all applicable machines. Here is a link to the updated bulletin:

http://www.microsoft.com/security/security_bulletins/ms03-039.asp
This is still a valid link as of February 1, 2004.

Our organization was made aware of the vulnerabilities as they were released by Microsoft. They chose not to release any publications mandating the update of the patches. Instead, they were confident that the policies in place that mandated a system administrator maintain their machines would be followed.

The organization had talked about purchasing applications that would either install patches for a system administrator or let them know when new patches were available. This was found to be a valid concern and a necessity, however, it was not found to be feasible due to the size of our infrastructure.

As we all ready knew, the system administrators were ill-prepared to install a patch on each of their machines. The system administrators leaned on their procedures that stated a patch must be tested by our internal engineering department before they would install it. This would ensure that the patch once installed would not break any of the operating systems or applications in use. This also bought them time to begin installations so they could meet any timelines that were outlined for them.

At this point, this is where the CTO / CIO (management) should have been contacted regarding risk mitigation. They need to be made aware of the possibility of an infection, the possible ramifications of the infection, and the cost of mitigating the risk. They are in their position to give direction to the organization on how the organization as a whole will mitigate a risk. In our case, we made the CIO aware and he chose not to take action as eh did not see an imminent threat from this vulnerability.

Unfortunately virus / worm coders were faster than our engineering department. The exploit code for Blaster was released. The impact from that was bad enough. Our incident response team was engaged to help eradicate the worm from our environment. But two weeks later Welchia was released. It used the same primary vulnerabilities (MS RPC DCOM). Since our system administrators were unable to patch all of their machines, the infection rate was in the thousands.

Our procedures for the Incident Response Team (IRT) were put to the test. They were found to be a little old and outdated. Contact lists were no longer valid. However, the core procedures on how to react and how to handle / escalate the incident were adequate. They provided our incident handlers information on how to proceed in each of the steps of incident response. More importantly, they provided the staff that augmented the team information on how the IRT worked and the steps they took when responding to an incident.

Identification:

Welchia was posted on the Symantec webpage on August 19th. By following our policies and procedures, we spent the first half of the morning reading the latest vulnerabilities on the Internet. This is how we were made aware of the existence of this Blaster variant. We noted the signatures and the information pertaining to the virus. We passed this on to our local management (IRT Manager) for their dissemination up to the upper management chain (CISO, CIO, CTO).

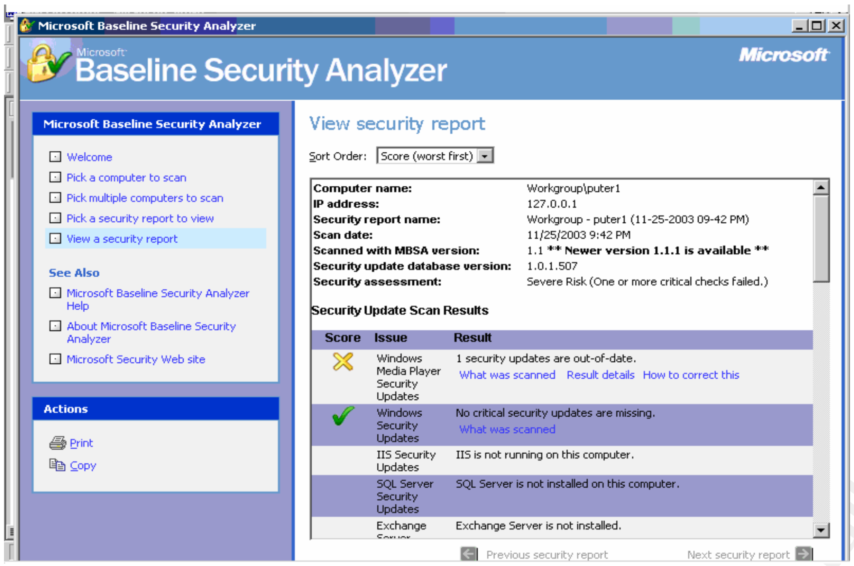
We did note that this was a variation of the MSBlaster worm. It used the same exploit (RPC DCOM). We were actively working on cleaning our network from the MSBlaster infection.

Within four hours of discovering the worm's existence, our perimeter started seeing an increase in ICMP traffic. We verified other teams were seeing the same type of traffic by using our connection with the Forum of Incident Response and Security Teams (www.first.org). We also went to the Internet Storm Center to verify the trends in specific port traffic (http://isc.sans.org).

This worm quickly went from a Level 1 threat up to a Level 3 threat (according to the threat rating assigned by Symantec). Once it was deemed a Level 3, they alerted their premier partners. We received a call from one of those partners about the updated virus definition pattern. We quickly downloaded it and pushed it to all of our workstations via the virus console.

Since we had all ready been working on mitigating the risk of the MSBlaster worm, we were aware of the patches for MS03-026. But to be safe, we utilized Microsoft's baseline security analyzer to verify where we stood on our patching levels. This tool will scan a machine (as long as you have administrative rights on the machine) and determine common security vulnerabilities. It will determine if you are missing patches, if patches are out of date (been superceded), or even if your password schema is too weak (i.e. Too short, allowing users to have non-expiring passwords, etc).

See diagram 3:


Microsoft Baseline Security Analyzer
Microsoft Baseline Security Analyzer
Microsoft
Microsoft Baseline Security Analyzer
Welcome
Pick a computer to scan
Pick multiple computers to scan
Pick a security report to view
View a security report
See Also
Microsoft Baseline Security Analyzer Help
About Microsoft Baseline Security Analyzer
Microsoft Security Web site
Actions
Print
Copy
Previous security report
Next security report

View security report

Sort Order: Score (worst first)

Computer name:	Workgroup\puter1
IP address:	127.0.0.1
Security report name:	Workgroup - puter1 (11-25-2003 09-42 PM)
Scan date:	11/25/2003 9:42 PM
Scanned with MBSA version:	1.1 **** Newer version 1.1.1 is available ****
Security update database version:	1.0.1.507
Security assessment:	Severe Risk (One or more critical checks failed.)

Security Update Scan Results

Score	Issue	Result
✗	Windows Media Player Security Updates	1 security updates are out-of-date. What was scanned Result details How to correct this
✓	Windows Security Updates	No critical security updates are missing. What was scanned
	IIS Security Updates	IIS is not running on this computer.
	SQL Server Security Updates	SQL Server is not installed on this computer.
	Exchange	Exchange Server is not installed.

DIAGRAM 3

Comment: you might want to consider adding figure numbers and a description of the figure below it

The baseline analyzer was run against each of the XP workstations. All seven of them came back clean. We of course did not scan the RedHat machines, as this was not a blended threat that would affect Linux as well as Microsoft products. Being all of our machines were patched, we turned our focus onto the rest of the infrastructure.

By 11pm that evening, our sensors started going off. Our duty agent started receiving automated pages from our desktop intrusion detection sensors. The Windows 2000 server that houses our logs was sending out large amounts of ICMP traffic. This machine was kept in a rack out of our area of influence. The team tasked with the patch management of that machine had yet to download the patch for the Microsoft DCOM/RPC vulnerability.

By the time the agent arrived on site the machine was already trying to make contact / infect other machines on the local network. Since this was a machine that was only connected to the internal network we knew it would send packets only to internal sites (minus the truly random packets). This also meant that the infection had to come from the internal network, not from the Internet.

Realizing where the infection was, the agent was able to gain access to the rack and then log on to the Windows 2000 machine. In order to contain the worm, he first updated the Symantec Antivirus software using Symantec's LiveUpdate button. This updated the machines definition pattern file. With the RealTime protection enabled, the tool was able to quarantine the worm and prevent it from further sending out any packets.

Then to eradicate the worm (which I go more in-depth in the eradication section) he ran the FixWelch.exe tool from Symantec while downloading the MS03-026 patch for Windows 2000 machines. The clean tool removed the virus and left the machine clean. However, it was still vulnerable. So the agent used the .msi package to install the MS03-026 on the server. Once the patching was complete and the machine rebooted, it was then completely clean and no longer vulnerable to the attacks from the Welchia worm. However, there was no way to tell when the machine became infected and how many other machines it had touched and spread the infection to.

With the infection running wild in the infrastructure, the incident handler utilized our escalation procedures to alert management and technical resources. This included our upstream provider. They were able to begin putting ACL's on the core and backbone routers. This stopped the worm from spreading past a backbone router and into another subnet.

Our technician initiated our emergency procedures and alerted the rest of the incident response team. By checking our sensors and logs we quickly realized we had rampant infections of both the MSBlaster infection and the Welchia worm in our infrastructure. We did debate for a short period just allowing the Welchia worm to run and let it clean the Blaster infection. However, the amount of traffic it was creating did not make that a feasible option. That added to the fact that each machine would still have to be 'touched' and cleaned for a worm make that a mute point. Management decided to instruct the IRT to locate all infected machines and eradicate the infections.

Now with the confirmed infection of one of our own machines, we were 100% certain that the worm was within our network. At this point we began scanning internal using several freeware tools. One tool we utilized was Angry IP Scanner 2.19. We utilized this tool as it was configurable to allow us to scan for machines that were open on tcp port 707.

When we saw the sheer number of infections, we knew that the IRT alone could not handle tracking each machine. Following our procedures we informed our management of the situation and they recruited our stand-by team to augment our group. These were members of other teams that we our liaisons. They were familiar with our team, with our processes, and with our team members. They were vital in assisting in the response aspect of this incident. All of this took place within the first 12 – 16 hours after the first responder arrived on site.

As we all engaged, one of the many difficulties that we did run into is that we could not find owners of all of the machines. We would verify the machine was infected, then we'd run a command from the command prompt:

```
 "nbtstat –A <IP>"
```

This command enabled us to see if we could tell who was logged into the machine. The hostname was helpful in some cases in trying to locate an owner. Other times, the user logged into the machine was able to help us locate the owner.

For machines that did not fall into those categories we had to get a little more inventive. We ended up writing a script. If a script was not available, we would manually run the command:

```
'netsend <IP> <crafted message>'
```

The crafted message would tell them their machine was infected and it would give them our phone number to call if they had any questions.

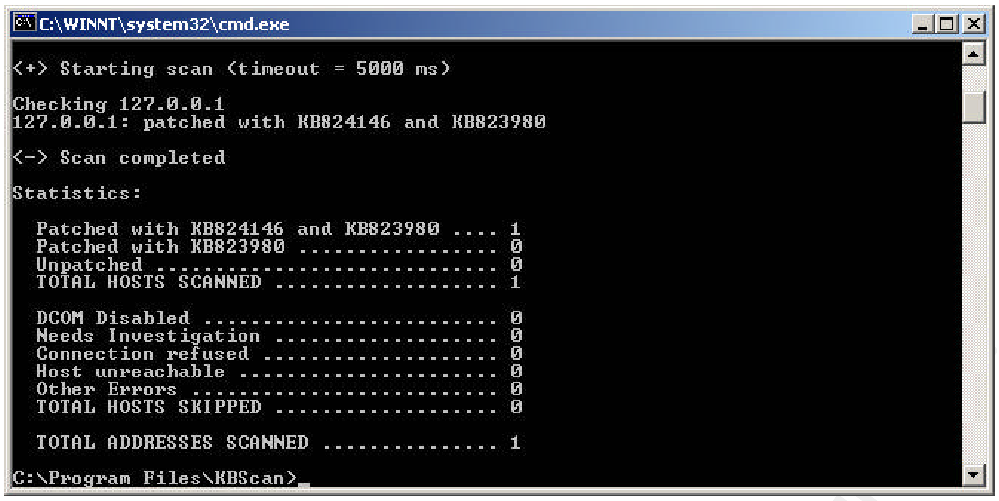
Simple Batch Script – netsend.bat

- SETLOCAL
- FOR /F "tokens=1,2" %%a in (input_file.txt) DO SET XTAR=%%a&CALL :WORK
- ECHO %XTAR%
- :WORK
- IF "%XTAR%"=="" GOTO END
- net send %XTAR% "Your machine is infected with the Welchia worm. Please call <#>. We will assist you in downloading and running the FixWelch file and updating your Symantec Antivirus software. We will also help you in locating and installing the appropriate patch for your system" >> output_file.txt
- SET XTAR=
- GOTO :EOF
- :END
- ENDLOCAL

By the time the user called in we had used the MS scanning tool to determine if they were patched or not. So we would be able to tell them if they needed to patch their machine, to clean their machine, and/or update the antivirus software.

To be proactive, the system administrator should have also scanned their local network to find out which machines were patched and which needed the patch. Several freeware tools were released that would enable a system administrator to do this:

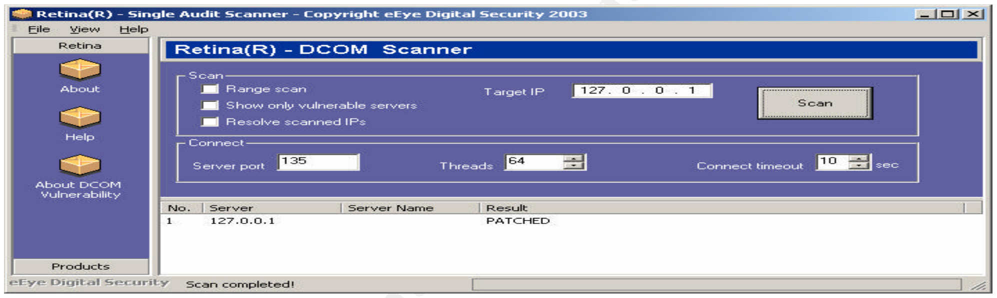
- Microsoft came out with KB824146Scanner (this was actually the updated tool for MS03-039). The original tool was for MS03-026.
    - o This is a command line scanner (see diagram 4):



```
<+> Starting scan (timeout = 5000 ms)

Checking 127.0.0.1
127.0.0.1: patched with KB824146 and KB823980

<-> Scan completed

Statistics:

  Patched with KB824146 and KB823980 .... 1
  Patched with KB823980 ................. 0
  Unpatched ............................. 0
  TOTAL HOSTS SCANNED ................... 1

  DCOM Disabled ......................... 0
  Needs Investigation ................... 0
  Connection refused .................... 0
  Host unreachable ...................... 0
  Other Errors .......................... 0
  TOTAL HOSTS SKIPPED ................... 0

  TOTAL ADDRESSES SCANNED ............... 1

C:\Program Files\KBScan>_
```

DIAGRAM 4

- eEye came out with their scanner (had to register to download this tool). This was a DCOM vulnerability scanner. I do not feel it was as reliable as the Microsoft scanner. I found it to be flakey on exactly what was and what was not vulnerable. I noticed this when scanning a subnet that still contained machines running Windows 9x. The tool would give an error message if the machine was running an application running DCOM.
    - This is a GUI scanner (see diagram 5)



Range scan
Show only vulnerable servers
Resolve scanned IPs
Target IP 127. 0 . 0 . 1
Scan
Server port 135
Threads 64
Connect timeout 10 sec

No.	Server	Server Name	Result
1	127.0.0.1		PATCHED

Scan completed!

DIAGRAM 5

If a custom scanner is not available, one way to determine if a machine may be infected with the Welchia worm is to check if the machine is listening on port 707. This is the most common port that Welchia listens on. However, the worm can listen to a range of ports 666-765.

This scan can be performed by either using a scanning tool (nmap) or simply telnet'ing to the machine on port 707:

```
C:\telnet 10.10.10.10 707
```

If the machine is listening then the result is:

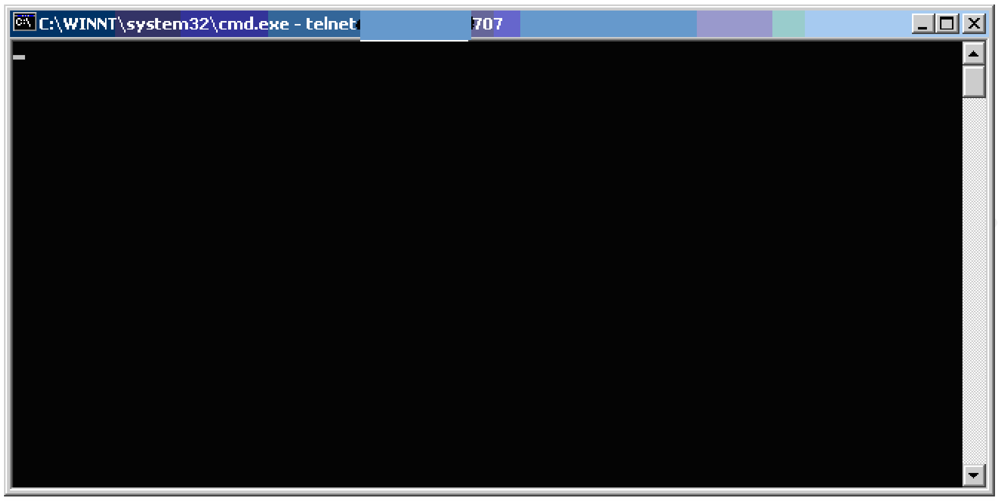
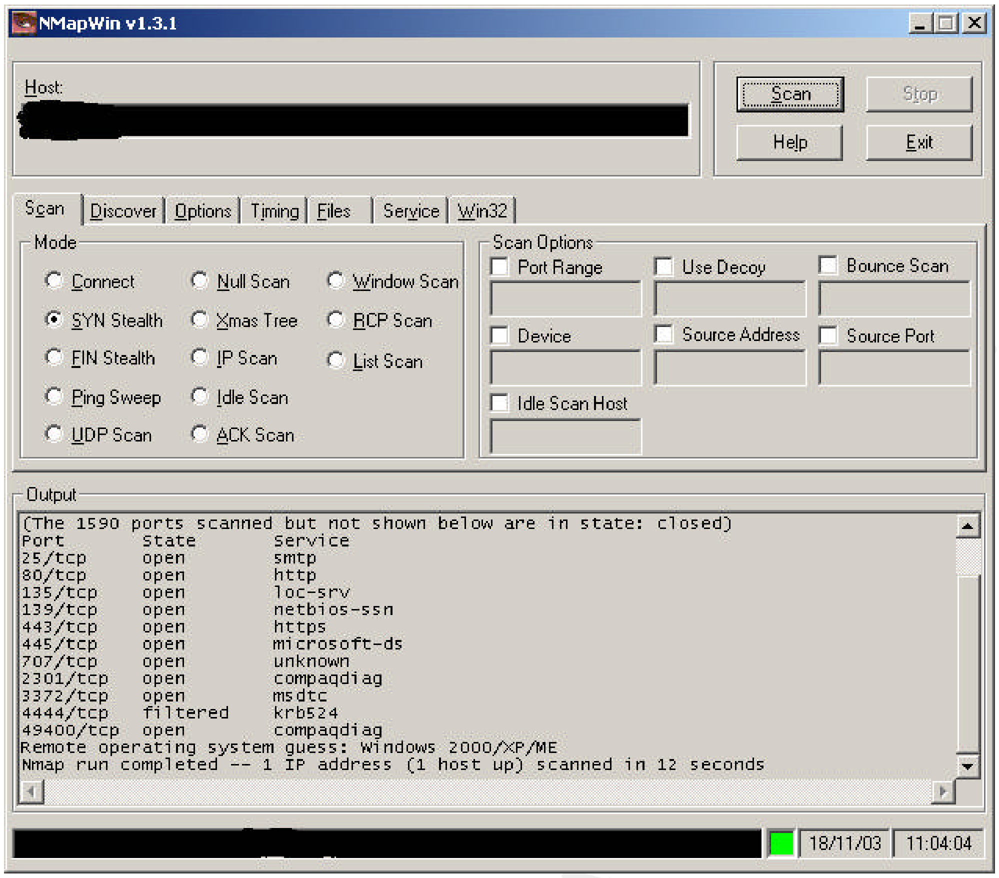


DIAGRAM 6

If the machine is not listening the result will be “connect failed”. This is a good thing. If the machine is not listening on the given ports then the worm is either not on the machine or else it is not active on the machine.

Here is an example of the Windows nmap version: (see diagram 7)



Host: [redacted]

Mode: [✓ SYN Stealth]

```
(The 1590 ports scanned but not shown below are in state: closed)
Port       State       Service
25/tcp     open        smtp
80/tcp     open        http
135/tcp    open        loc-srv
139/tcp    open        netbios-ssn
443/tcp    open        https
445/tcp    open        microsoft-ds
707/tcp    open        unknown
2301/tcp   open        compaqdiag
3372/tcp   open        msdtc
4444/tcp   filtered    krb524
49400/tcp  open        compaqdiag
Remote operating system guess: Windows 2000/XP/ME
Nmap run completed -- 1 IP address (1 host up) scanned in 12 seconds
```

[redacted]

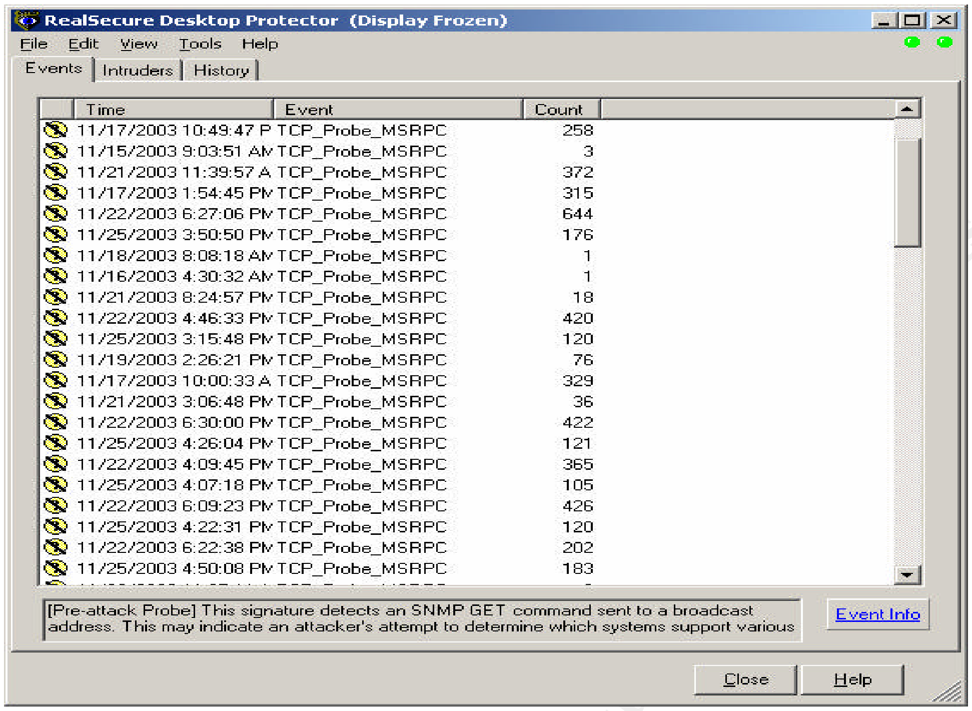
DIAGRAM 7

Note – TCP port 707 is open. The application does not assign a Service to this port, but the port is open / listening. This is a good indication that the machine is infected with the worm. At this point the system is identified and the system administrator needs to engage this machine and take steps to remediate this specific risk. Details of eradication will be in the eradication section.

Instead of actively scanning and trying to be proactive in the identification stage, a system administrator can take the reactive role and set up a host based intrusion detection system (HIDS) on a local machine. By installing an HIDS they can then monitor the logs for machines that are attempting to infect them.

This example can also be the first realization that something is wrong on the network. If the local HIDS sensors start picking up on anomalous traffic and alerting the logging server, it's probably a good idea to investigate. In the example below the sensor is noting the attack signature as TCP_Probe_MSRPC.

Below is an example of ISS' Real Secure Desktop Protector (this was formerly BlackICE Defender). Internet Security Systems – http://www.iss.net (see diagram 8):


RealSecure Desktop Protector (Display Frozen)
File Edit View Tools Help
Events Intruders History
Event Info
Close Help

Time	Event	Count
11/17/2003 10:49:47 P	TCP_Probe_MSRPC	258
11/15/2003 9:03:51 AM	TCP_Probe_MSRPC	3
11/21/2003 11:39:57 A	TCP_Probe_MSRPC	372
11/17/2003 1:54:45 PM	TCP_Probe_MSRPC	315
11/22/2003 6:27:06 PM	TCP_Probe_MSRPC	644
11/25/2003 3:50:50 PM	TCP_Probe_MSRPC	176
11/18/2003 8:08:18 AM	TCP_Probe_MSRPC	1
11/16/2003 4:30:32 AM	TCP_Probe_MSRPC	1
11/21/2003 8:24:57 PM	TCP_Probe_MSRPC	18
11/22/2003 4:46:33 PM	TCP_Probe_MSRPC	420
11/25/2003 3:15:48 PM	TCP_Probe_MSRPC	120
11/19/2003 2:26:21 PM	TCP_Probe_MSRPC	76
11/17/2003 10:00:33 A	TCP_Probe_MSRPC	329
11/21/2003 3:06:48 PM	TCP_Probe_MSRPC	36
11/22/2003 6:30:00 PM	TCP_Probe_MSRPC	422
11/25/2003 4:26:04 PM	TCP_Probe_MSRPC	121
11/22/2003 4:09:45 PM	TCP_Probe_MSRPC	365
11/25/2003 4:07:18 PM	TCP_Probe_MSRPC	105
11/22/2003 6:09:23 PM	TCP_Probe_MSRPC	426
11/25/2003 4:22:31 PM	TCP_Probe_MSRPC	120
11/22/2003 6:22:38 PM	TCP_Probe_MSRPC	202
11/25/2003 4:50:08 PM	TCP_Probe_MSRPC	183

[Pre-attack Probe] This signature detects an SNMP GET command sent to a broadcast address. This may indicate an attacker's attempt to determine which systems support various

DIAGRAM 8

The above example had the "intruder" column removed. The display can list both the intruder hostname and IP address if selected. This enables the administrator to find machines that are actively scanning their network attempting to infect other machines.

The same will pertain to an NIDS sensor (Network Based Intrusion Detection System). The alerts will come in from a sensor on the network versus coming from the actual machine that is having the exploit attempt to infect it. The sensor will most likely have a NIC (network interface card) listening in promiscuous mode. It will see all traffic on its network segment. By analyzing this traffic, it will let a system administrator know how many machines on each subnet may be infected by the type of traffic to and from these machines.

Once an infection has been determined, the system administrator needs to clean the machine. I would still recommend the system administrator begin by updating the antivirus software with the latest definition file and then scanning the local machine. This will quarantine any infections and will let the system administrator know if they are dealing with one infection or multiple infections. It also helps the administrator know which worm / virus they are dealing with. This is helpful as a system administrator could get extremely frustrated when they can not clean a machine that is infected with the SoBig virus when they keep running the FixBlast or FixWelch tool!

Also, by keeping the antivirus software on the local desktops current and up to date, the system administrator stays ahead of the game in combating future worms/virii. Up to date definition patterns on desktop machines (or servers), often contain the "real time scanning" for products such as Symantec and McAfee. When those run, they will quarantine any infection they detect. (This can be annoying if you have a 'security' CD in the cdrom tray – real-time scanning software may attempt to quarantine several of those tools). If the real-time scanner does quarantine the worm, the machine may still be infected, but it will not be able to infect other machines. Most importantly, the worm may be quarantined but it will still require the FixWelch.exe tool to be run on it (or a manual removal may be performed).

In order to ensure that all of our machines were current with their definition files, we were able to write a script that would check the registry entry on a machine. We looked for the registry setting for Symantec's definition file. We needed to make sure the definition file was dated at least August 18, 2003. The registry setting was under:

HKEY_LOCAL_MACHINE \ SOFTWARE \ Symantec \ SharedDefs \ DefWatch_10 (The value in the "Value Data" will contain a link to the definition file date and revision number)

The patching of a machine will also NOT clean the infection. This is the same as locking the front door once a burglar is all ready in the house. It does prevent other burglars from gaining entry but does not do anything to hinder the one that is already ransacking your valuables. That is why the clean tool (such as the one from Symantec) is required in order to remove the infection. This tool scans each file on the system and will attempt to remove the infection. The directory the tool is ran in will contain a log file that will let the system administrator know what action was taken by the tool.

The second step for the system administrator to do is download the fix tool (FixWelch.exe). This tool will scan the entire machine's file structure for infected files (it took a little over 10 minutes to run on my machine (Pentium III, 800MHz, 512MB RAM, 30 Gig HD with 70% unused).

Then, the system administrator will be working with a clean machine. They can then download the patch and install it on the machine. This will complete the process for cleaning and preventing further infection from the Blaster / Welchia / and future variants that use the DCOM vulnerability.

Containment:

Comment: much of what you said above in Identification is more applicable to containment. you might want to shift things around a bit

With the rapid spread of this worm, it was necessary for us to take steps to slow the worm down. We had to isolate the problem before we had any chance of trying to eradicate the infections. In order to do this, we set up access control lists (ACL's) on our routers that blocked ICMP from traversing them. A user could 'ping' all they wanted on their own subnet, but as soon as they tried to send ICMP traffic through the router, they were dropped. This served useful in preventing the worm from being able to infect other machines not on their subnet. The worm would only attempt to infect machines that they were successful in pinging.

We could have put up an ACL blocking tcp port 707 traffic, but we needed to be able to telnet to machines on port 707 to verify infections. So we opted not to add that level of 'protection' there (yes, took written management buy-off / risk mitigation to make this decision).

Working with our upstream provider was an essential key to our mitigation. We had them add ACL's inbound and outbound to our network. They did add both the ICMP and the TCP port 707 blocks. This aided us as they were able to log their ACL hits. They would copy those logs and then send us emails of machines that were either sending large amounts of ICMP or attempting to 'talk' on port 707.

Here is an example of the ACL showing large amounts of ICMP traffic:

```
#sh access-list 150
Extended IP access list 150
    deny   ip host xx.xx.xx.xx any (2713 matches)
    permit icmp any any

    permit ip any any (64 matches)
```

We also took the advice of Cisco:

Here are some of the recommendations that Cisco came out with to mitigate the Welchia (Nachi in their paper) worm:
(<http://www.cisco.com/warp/public/707/cisco-sn-20030820-nachi.shtml>)

```
    ! --- block ICMP
    ! ---
    ! --- you do not need to deny ICMP packets if you already filter
    ! --- them by using PBR
    ! ---
    ! --- blocking ICMP packets with access lists will cause ping utility and
    ! --- other similar diagnostic tools not to work
    access-list 115 deny icmp any any echo
    access-list 115 deny icmp any any echo-reply

    ! --- block vulnerable protocols
    ! --- Nachi related

    access-list 115 deny tcp any any eq 135
    access-list 115 deny udp any any eq 135

    ! --- block TFTP

    access-list 115 deny udp any any eq 69

    ! --- block other vulnerable MS protocols

    access-list 115 deny udp any any eq 137
    access-list 115 deny udp any any eq 138
    access-list 115 deny tcp any any eq 139
    access-list 115 deny udp any any eq 139
    access-list 115 deny tcp any any eq 445
    access-list 115 deny tcp any any eq 593

    ! --- Allow all other traffic -- insert
    ! --- other existing access list entries here

    access-list 115 permit ip any any
    interface <interface>
    ip access-group 115 in
    ip access-group 115 out
```

The worm will attempt to send packets to random IP addresses, some of which may not exist. When that occurs, the router will reply with an ICMP unreachable packet. In some cases, replying to a large number of requests with invalid IP addresses may result in degradation of the router's performance. To prevent that from occurring, use the following command:

```
Router(config)# interface <interface>
    Router(if-config)# no ip unreachables
```

Following the basic mitigation risks from Cisco, the network administrator should have configured access control lists on their routers - ingress and egress filters. Or they should have engaged the proper group to have those filters put in place.

In our case, we had been battling the MSBlaster incident. So we already had in place access control lists blocking TCP and UDP ports 69 (tftp), port 135, and port 4444. These were blocked inbound and outbound. *Note: If port 135 is blocked all together it may cause some problems with MS Outlook.*

We had an open bridge (teleconference) that all technical users could call into. They could get advice / instructions on this call. The amount of users that were turned away as the telecon was full told us that this was a good idea.
The amount of infections that we had in the organization left us with another problem: How to report these infections. We ended up setting up a webpage that listed each of the machines IP addresses and host names. Once a machine was verified to be infected, it was added to the page. Then when the machine was verified to be cleaned, it was removed.

The administration of the webpage was manual. This was a bad idea, even though the webpage was a good concept. Automation of the updates (adding and removing) would be helpful for future breakouts.

If we were unable to find an owner of a machine and could not get a response from a customer via the 'netsend <ip> <message>' command, we would go to the Cisco switch and turn off that machine's port. All we needed to do this was have the MAC address of the machine. This was gathering in our early assessment of the machine when we did an "nbtstat –A <ip>":

```
VMWare Network Adapter VMnet1:

Node IpAddress: [192.168.2.1] Scope Id: []

NetBIOS Remote Machine Name Table

Name            Type        Status

---------------------------------------------

MAC Address = 00-50-56-C0-00-01
```

Eradication:

With the ACL's in place and machines unable to infect beyond their own subnets, we are able to begin scanning local subnets for infections. The subnets with the larger amounts of infections were the priority. The subnets with smaller infections were sent emails to local administrators asking them to ensure their machines were patched (and we included the Microsoft KB scanning tool). We gave them the instructions on how to clean infections, and how to determine if a machine was infected.

We also contacted our Help Desk and ensured that they had in-depth knowledge of this worm. They also needed to have steps to help callers who called in. By alerting the Help Desk, it would prevent them from working with a user on the phone trying to determine the root cause of a problem on their machine that was part of this incident. Again, it just saved time by working with them and keeping them informed.

The eradication of each machine actually took less than a half hour. As stated above, the system administrator would start by either installing Symantec Antivirus software or simply updating the definition file. This would quarantine any infection and prevent any further spreading of the infection. Then the machine would be cleaned using the FixWelch.exe tool from Symantec. Once that was completed we used the appropriate patch from Microsoft to patch the machine with MS03-026 (of course we did update our policy with the release of MS03-039).

Our team did not do all of the patching / cleaning. We used the local administrators at the customer sites and we used our Help Desk to assist in the cleaning process. When it was appropriate, we used remote administrative rights to push a patch and reboot a machine. This was not the preferred method, but it was an effective one.

After the machines were cleaned by the local system administrator or by the help desk, they would either call us with an IP address or they would send us an email. This would enable us to verify their work.

Once the machine was verified to be clean, the ACL was removed. We were also able to re-enable the port on their local switch (if applicable). Essentially we were able to let the machine operate as usual. However, it was not uncommon for a machine to be cleaned and stay blocked. If the local administrator did not call us and have us verify the machine was cleaned, they stayed blocked.

We found that most system administrators did not make patching their machines a priority. This lack of action on their part allowed the worm to spread quickly and infect multiple machines on their network. The system administrators that had patched their machines had very little to do and were able to assist other administrators.

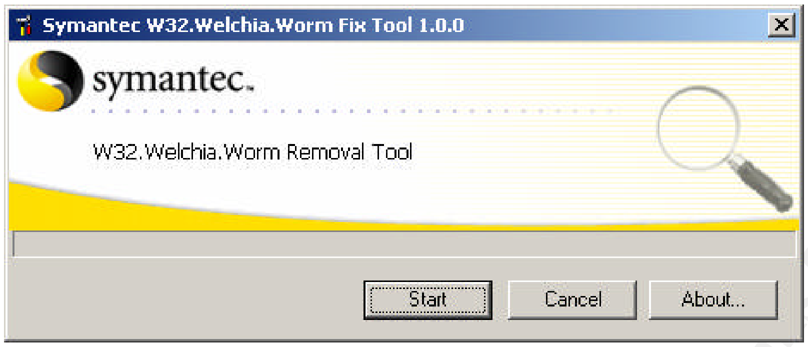
Due to the lack of patching from the system administrators, the clean-up / eradication phase of incident response took a lot longer than expected. We had allotted two weeks for the eradication of the worm. However, it took over four weeks until we finally got a complete scan of our infrastructure to show up with no new infections.

Recovery:

In order for us to verify that a system was in fact clean, rather than take a system administrator’s word for it, we did utilize the Microsoft KBscanning tool to ensure the system was patched. We would then use a scanner (FoundStone's 4.0 scanner) to ensure that the machine was not listening / open on port 707.

If this was true, then we were allowing the machine back on the network with unfettered access. This was a removal of any ACL's that may have been enacted either local or by our upstream provider.

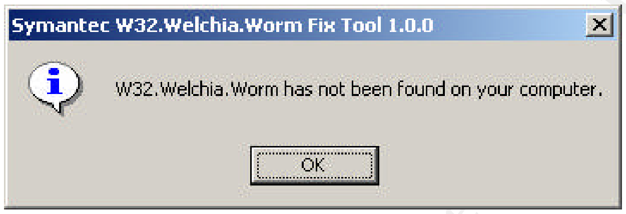
When possible, we would utilize the Symantec cleaner: (see diagram 9)


Symantec W32.Welchia.Worm Fix Tool 1.0.0

W32.Welchia.Worm Removal Tool

DIAGRAM 9

When we had a person on site to do the cleaning, we used this tool. This would verify that the registry entries, directory entries, and the entire infection was removed from the machine. (see diagram 10)


Symantec W32.Welchia.Worm Fix Tool 1.0.0

W32.Welchia.Worm has not been found on your computer.

DIAGRAM 10

When we had the system administrator on the phone and had them as a captive audience, we took advantage of that situation. We would scan their machines entire subnet to ensure that there were not any other infected machines on their network. If there were, we had the administrator responsible and could alert them to the other infections. This proved to be a good practice in the remediation of the worm.

We performed continuous scans of all networks for port (tcp) 707. We also monitored ICMP traffic. This was helpful for watching for possible future breakouts. This was done not only in the identification of the worm, but also during the recovery stage.

Comment: was there any type of formal turn-over of previously infected machines to the sysadmins? any paperwork or signatures?

Lessons Learned:

Following up with the team after the worm breakout, we put together several things that we found were lacking.

Through our fight with the MSBlaster and then the Welchia worm, we learned that the best way to fight worms was to have an in-depth and enforced security policy. Yes, we all ready had the policy, but enforcement of the policy was lacking. In order to get the policy enforced we had to get more than upper management's signature on the document. We needed to have their approval to remove machines from the network that were not compliant. We also needed a way to verify machines were compliant with our policies. This whole incident would have been avoided (or been on a much smaller scale) if each of the machines had the vendor recommended patch and had updated antivirus definition patterns.

We now have in writing, permission from our CEO to have any machines that threaten or could possibly harm our infrastructure off the network. This is a key document as this may interrupt business and cost the organization revenue. However, by not removing the machine from the network may also cost the organization revenue. This action came directly from our lessons learned meeting.

Another thing we learned the hard way was that the first thing an organization needed to do is inform their users! I believe this comes before anything else as the user can be either the weakest or the strongest link. If the users are aware of what is going on in the organization, they can assist, even if it means them doing nothing. If they do not know what's going on, they can cause undeterminable damage (and a lot of overtime hours for the IRT).

This is not to say that the IRT is not to start working before informing the users in the organization. Hopefully the manager of the team can draft a message on what is going on that can be feed to the customer sites informing them of the problem. Open and honest communications is essential the entire time of an incident. Let the users know what is going on and they will not panic. The same thing goes for the upper management.

Our current process and procedures were modified to add this as part of those procedures. A communication will be put out to the organization detailing what is happening, what users in general are to do, and who and how to contact the appropriate personnel.

When it comes to mitigating the risk, a system administrator needs to patch their machines. Keeping the machines up to date with vendor recommended patches stops the exploitation that most worms / virii execute to gain access of that machine. Most exploit code is written with publicly available exploits that have previously been remediated by vendors via a patch / security update. The obvious exception to this is the 0 day exploits. And the best defense against a zero-day exploit is the defense in depth. Simply stated, defense in depth is having multiple layers of security to secure a system or network to the highest degree possible.

Our network administrators needed to block unnecessary ports. In this case it was TCP traffic on port 135. This may not always be a feasible option (especially with Microsoft Outlook). However, you may be able to block inbound port 135 requests.

Since we are talking about blocking, you may also want to block inbound ICMP Echo requests. It's even better if your device can recognize the pattern that packets use.

Also monitoring of the network is essential to every organization. Look for anomalous traffic. It is better to be a quick responder and put out a small fire than to be facing a complete Denial of Service (DoS) and trying to determine where it started and how to stop it.

Another fault that was found lacking was communication. In an incident, the entire incident response team needs to be 'on the same page' during the entire process. This begins when the team either notices or is informed of the anomalous traffic, up until the incident is completely eradicated. A spokesperson for the team needs to be officially mandated to report to management and customers.

The IRT needs to have contacts in all other security departments and in customer sites. These contacts need to be kept up to date and checked at least quarterly. It's a bad situation calling a contact and finding that person has been gone for months (or years).

We have updated all of our contact lists, our email address lists, and our emergency contacts. We have updated our process and procedures to state that we will continue to maintain these lists by verifying them at least quarterly.

And asset management is another key to successful remediation. If you don't know what's on your network then you'd better find a way to get that information. This helps on every step of the Incident Handling process. Obtain a list of assets on the network, assign those assets to an owner, and then keep that list up to date. It only takes one machine to begin a new infection.

Remote users and laptops that get plugged into a user's home network (ISP) pose a huge threat to corporations. An administrator has little control over these machines. They can be infected at home and then when the user carries that laptop in and plugs it into the corporate network, he's sharing that infection with the rest of the infrastructure. Again, corporate policy should dictate that laptops should be kept up to date on their virus patterns and patch management. If a user knows the policy is enforced they are more likely to follow.

Conclusion:

This document was written for a couple of different reasons. (Ok, for one reason, mandatory to obtain my GCIH!). I tried to convey my opinion that the reason most organizations are thwarted with malware incidents is not because the code writers are getting smarter and writing better code. I believe the main reason lies with the lack of policy enforcement. An organization needs to hold individuals accountable for policy compliance. This also includes recognizing those who are compliant. This may be a merit bonus for those who do comply. Money is a great incentive for those who do not take security or corporate policy seriously.

I also wanted to highlight why I do not believe it is a viable option in the security world to fight fire with fire. In my opinion, two wrongs to not make a right. Creating a worm that will infect other's machines and use them essentially as robots to further infect more machines is NOT a viable option. No matter how noble the cause, it's still a bad practice.

Welchia is what some term a white worm, meaning it has 'good intentions'. This is the opposite of a black worm, or those with malicious intentions. The two main problems that I have with worms (white or black) are that they affect a user's machine without their permission (and often without their knowledge). And whether they download code to break or to fix, they alter the person's machine. This can in fact break the system, whether the worm was coded for good or for evil.

Again, I want to stress another major reason for my writing this document is that I honestly believe the reason that worms are being so effective even though they are using old exploit code is that the systems they are exploiting are behind on their patches. This is a corporate policy issue, not an end user issue. If an organization would be serious about following their own policies and procedures, they would reap the benefits from that decision.

I will keep my same mentality that the best way to combat worms and viruses is to keep your systems patched and to have the latest virus definition patterns on the machines. This is to accompany the ACL's on routers, firewalls with good rules sets, and a set of IDS systems that are being actively monitored. Defense in depth is the key to good security.

For an organization to set a policy but not really enforce it is the same as setting a speed limit on a highway. For the most part, each driver will go about ten miles over the speed limit. Some just totally disregard it completely.

Nobody really follows that speed limit until they personally see an officer (whether it's a state trooper or a local sheriff's deputy). Then each driver will slow down and follow the posted speed limit. However, as soon as that officer is out of sight, they will increase the speed and act as if they never saw the officer and don't know there is a set speed limit.

These same drivers will complain if they are pulled over and ask why they were singled out and they'll say that they were just going with the flow of traffic.

It's the same with a network. Unless the corporate officer specifically mandates an update for a patch, the system administrators do not make it a policy to update their machine. They would rather work on other issues than take the time to test and install a patch on each of their machines.

The system administrators like the speeding drivers will go with the flow. If there is no enforcement of the patch management from the corporate HQ, they will be reluctant to have a strict policy. They will let patches and updates wait until they pile up and then they'll install them in batches.

Again, I believe actively monitoring and enforcing the corporate security policy is one of the best ways to ensure that a network is kept virus free and operational. Accountability for following the corporate mandates is necessary if those corporations ever want to be at the level of security that they dictate is acceptable.

References:

RPC Overflow Exploit Code
http://www.securiteam.com/securitynews/5LP0B0AB5C.html

Symantec's data – Security Response on Welchia
http://securityresponse.symantec.com/avcenter/venc/data/w32.welchia.worm.html

W32.Welchia.Worm Removal Tool
http://securityresponse.symantec.com/avcenter/venc/data/w32.welchia.worm.removal.tool.html

Cisco Security Notice: Nachi Worm Mitigation Recommendations
http://www.cisco.com/warp/public/707/cisco-sn-20030820-nachi.shtml

NAI
http://vil.nai.com/vil/content/v_100559.htm

Windows RPC DCOM Long Filename Heap Overflow Exploit (MS03-039)
http://www.securiteam.com/exploits/5WP0B20B5C.html

MS03-026: Buffer Overrun in RPC May Allow Code Execution
http://support.microsoft.com/default.aspx?kbid=823980

eEye's write-up / analysis of Blaster
http://www.eeye.com/html/Research/Advisories/Blaster_Analysis.txt

CVE (Common Vulnerabilities / Exposures)
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0109
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0352

LSD's write-up – exploit code for RPC DCOM – proof of concept
http://lsd-pl.net/special.html

Code Review - Windows RPC Interface
http://marc.theaimsgroup.com/?l=bugtraq&m=105914789527294&w=2

PSS Security Response Team Alert - New Worm: Nachi, Blaster-D, Welchia
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/alerts/nachi.asp

Examining the Nachi Exploit – a write up on Nachi by Linda Bourbeau
http://www.giac.org/practical/GCIH/Linda_Bourbeau_GCIH.pdf

F-Secure Virus Descriptions
http://www.f-secure.com/v-descs/welchi.shtml

Trend Micro
http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_NACHI.A

SecuriTeam
http://www.securiteam.com/windowsntfocus/5SP0C20AKG.html

Welchia packet details:
http://www.esphion.com/ETB-82.htm