Global Information Assurance Certification Paper

An Examination of the W32/BugBear Worm

By Gary Delaney GCIA CISSP

GCIH Practical Assignment Version 2.1

Introduction

The most common type of virus seen in the wild at presence is the worm. A worm seeks to propagate itself through mediums such as email and computer networks. Earlier worms sought simply to propagate with no specific malicious intent. A denial of service could occur due to the level of activity that a worm created on a network, but this was not its main objective.

Since the first worms appeared they have been developing in sophistication. Their objectives are no longer solely to spread and propagate. They often have a payload that has additional functionality. This is what is most concerning. A cleanup may remove a worm from your network, but what damage has it done while it was present. It may have emailed passwords, it may have altered system files, and it may have placed a backdoor in the system. It is for these reasons that a cleanup involves far more than a simple removal of the malicious code.

It is essential that networks protect themselves from this type of infection. By its very nature a single incident can quickly turn into hundreds or thousands of incidents. The compromise to system security can take a great deal of time to assess and to eliminate. The BugBear worm is an excellent example of a worm that has precisely this impact. It will compromise system security though propagation but will also attempt to steal passwords, install a backdoor and run a key logger.

This paper is a review of the W32/BugBear worm, which was first seen on September 30th 2002. It also outlines an incident where the worm made its way into a network. This worm has many different functions and this paper outlines the compromises that the worm was able to effect within the network and also a description of the defences that were in place and how they stood up to the event.

This paper is based on Option 1 – Exploit in Action as outlined in GCIH Practical Assignment version 2.1

The Exploit

The worm uses an old exploit discovered in March 2001.

CVE-2001-0154

BUGTRAQ:20010330 Incorrect MIME Header Can Cause IE to Execute E-mail Attachment

Microsoft Security Bulletin (MS01-020)
'Incorrect MIME header can cause IE to execute e-mail attachment.'

Operating Systems Affected:

This exploit affects systems running IE 5.5 SP1 or earlier on Windows platforms. Windows 9x, ME, NT4, 2000, XP. Except IE 5.01 SP2 on x86. Also any software that uses IE to render html may be vulnerable.

Protocols/Services/Applications
Specifically IE 5.5 and earlier is affected. Other elements of the system are affected by virtue of the fact that they use IE to render html, for example Outlook. This is then exploited to run a piece of malicious code. Any software using a vulnerable version of IE to render html.

Brief Description
The exploit takes advantage of the fact that unpatched versions of IE may run certain MIME types automatically without the intervention of the user. An attacker reclassifies the code (in this case BugBear) as one of these MIME types and the code is executed on the victims system when the email is rendered in Outlook.

Variants
No known variants exist at this time.

The worm is known by a number of aliases including:

W32.Bugbear@mm (Symantec)
W32/Bugbear-A (Sophos)
W32/Bugbear.A@mm (F-Secure)
W32/Bugbear.worm
W32/Tanat
W32/Tanat-mm
Win32Bugbear (CA)
Worm/Tanatos (CentralCommand)
WORM_NATOSTA.A (Trend)

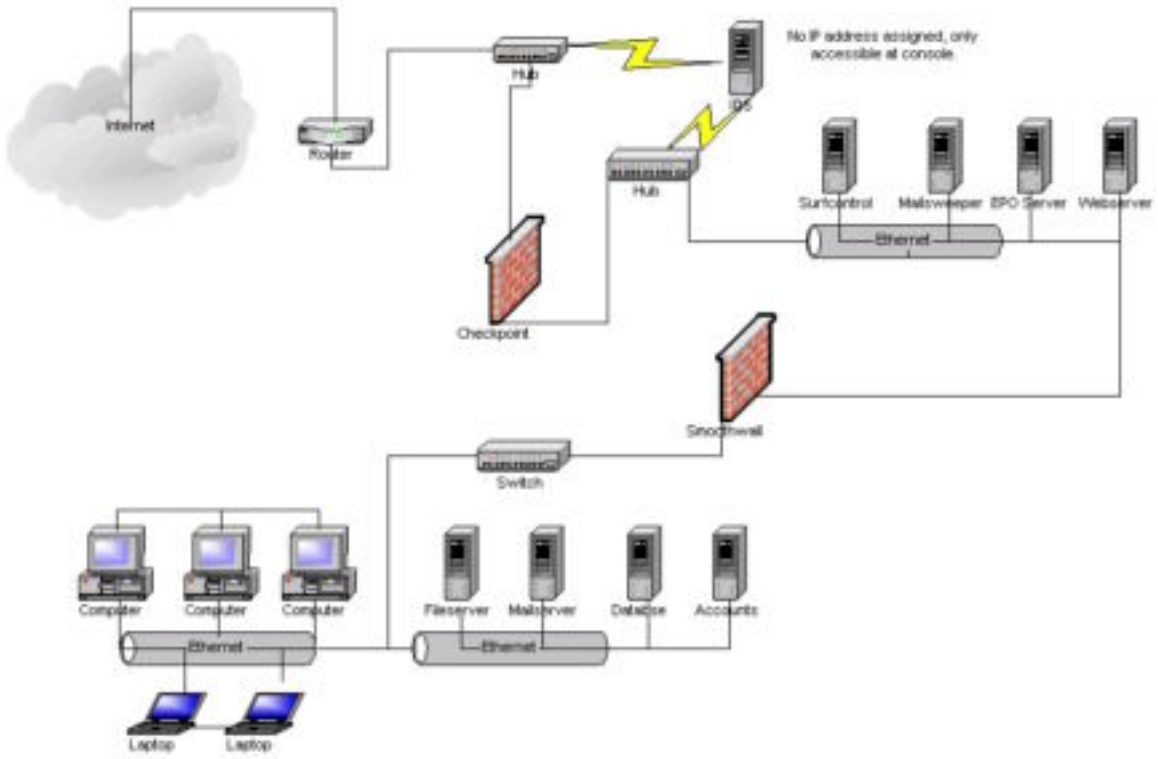
References:
www.cert.org/advisories/CA-2001-06.html
www.kb.cert.org/vuls/id/980499
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-020.asp
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0154

References for BugBear:

http://vil.nai.com/vil/content/v_99728.htm
http://www.sophos.com/virusinfo/analyses/w32bugbeara.html

The Attack

Network Diagram:



Network Description.

As can be seen from the above diagram, the network is split between an internal LAN and a DMZ type network that sits between two firewalls. None of the nodes on the inside of the network is addressable from the Internet either directly or through a NAT scheme. The nodes in the DMZ permit limited access to the Internet depending on their role and this is facilitated through the use of Network Address Translation (NAT).

Components

- IDS

An intrusion detection system has been deployed and monitors the DMZ and the segment between the external router and the DMZ. This system is based on FreeBSD 4.4 and Snort 1.9 with Snortsnarf for reporting.

- Checkpoint Firewall

The Checkpoint FW controls access to the DMZ and ultimately to the LAN. There is a NAT scheme in place for the purposes of facilitating email and Internet access.

Email Rules:

NAT Rule					
Original Packet			Translated Packet		
SRC	DEST	Service	SRC	DEST	Service
X.X.1.55	ANY	ANY	Y.Y.161.163	Original	Original

Security Policy				
SRC	DEST	Service	Action	Time
X.X.1.55	ANY	DNS, SMTP	Allow	Any
SRC	DEST	Service	Action	Time
Any	Any	Any	Drop	Any

Theses rules facilitate access to the mailsweeper machine from the Internet over ports 25 and 53 only (mailsweeper server runs DNS also). The subsequent cleanup rule – Any Any Drop prevents access over any other services. The NAT rule translates the internal private and non-routable address to the public IP address associated with the mx record of the organization thus facilitating the receipt of email.

Regular Internet access is provided by hiding all internal IP addresses behind the external interface of the Checkpoint firewall. This is regulated by the presence of a SurfControl machine, which enforces a security policy by content scanning all access to the Internet.

As can be seen in the attached diagram the internal LAN segment is isolated from the DMZ by use of a Smoothwall Firewall. This firewall runs on Linux 2.2.22 kernel. This is a fully functional firewall that can run on relatively low specification hardware. The policy of this firewall is relatively simple.

SRC	DEST	Service	Action	Time
LAN	ANY	ANY	Allow	Any
Mailsweeper	MailServer	25	Allow	Any
ANY	LAN	ANY	DROP	ANY

This rulebase facilitates access to the DMZ and ultimately the Internet for nodes on the LAN. External email is delivered to the mail server from the mailsweeper by use of port forwarding rather than NAT. The mailsweeper makes a connection directly to the smoothwall firewall on port 25 and it is configured to forward all traffic on that port to the mail server. Only the mailsweeper machine is permitted to do this.

External Service Access

Current rules:

Proto	Source	Destination port	Enabled	Mark
TCP	ALL	25	✓	☐

Port Forwarding

Current rules:

Proto	Source port	Destination IP	Destination port	Enabled	Mark
TCP	25	X.X.1.200	25	✓	☐

Anti Virus is deployed using McAfee EPO. Workstations run McAfee VirusScan 4.51 SP 1 with that latest DAT updated weekly. Servers run Netshield 4.5. All are managed through EPO, which indicates what version a machine is at, whether they have missed an update or whether they have been uncontactable.

Protocol Description.

The worm initially exploits the MIME header vulnerability within IE as described previously. Once an infection has occurred it seeks to exploit other elements of the operating system.

When vulnerable versions of IE encounter MIME encoded data they will execute it without reference to the user. There is a table in IE that specifies what applications are associated with what MIME type. When a particular MIME type is encountered the system executes it either stand alone or with its associated application.

The worm exploits the fact that Outlook and Outlook Express use IE to render html in email. Once the message is displayed on the target machine the worm is executed by exploiting the vulnerability. The worm then sets about propagating and performing the other functions that are built into it.

The delivery protocol is either SMTP or POP3 when a mail arrives or an alternative method of propagation is NetBIOS when the virus propagates through a share.

How the Exploit Works.

The Bugbear worm is able to exploit the vulnerability by altering the MIME header to match one of the types that will auto execute. An example of an auto executing MIME type would be audio/x-midi or application/x-msdownload.

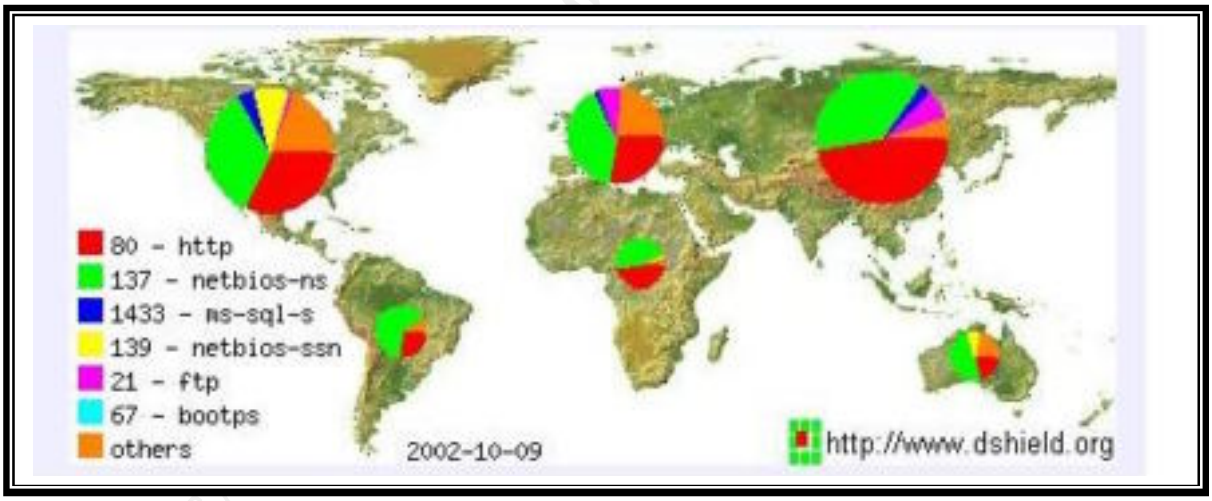
A typical example of how an infection occurs is:

An infected system acquires the victims mail address from a windows address book and sends a copy of itself to that victim. The victim is running one of the vulnerable versions of IE and also uses Outlook as an email client. The message is received and displayed in Outlook. Anti-Virus software on the system does not have a signature for this worm and fails to recognize it. Upon display the message renders using IE and the code is executed. IE doesn't force the code to execute using the associated application. If it did then when the MIME type was audio/x-midi it would fail to execute. The code is executed as an executable file even though it is classified as a midi file.

The worm contains its own SMTP engine and can email itself to addresses found on the system. It can and does forge source addresses so that another user may think that you have sent them the worm when in fact you have not. This can obviously be confirmed or rejected by examining the mail header for the source IP address.

The worm has quite a bit of functionality.

Firstly it will search for network shares in order to propagate. This has been responsible for users reporting printers spitting reams of paper with unintelligible characters. The worm treats a shared printer the same way as a shared folder and copies itself to it. There has also been a marked increase in the number of port 137 connection attempts on incidents.org and this has been explained by the presence of BugBear in the wild.



Port 137 Scans.

published: 2002-10-01

```
---                              update                                  ---
We now believe that these port 137 scans are due to the 'Bugbear'
mass mailing virus and the 'Scrup' worm.
```

Taken form www.incidents.org on October 10th 2002, approx 10 days after first detection.

The worm installs a backdoor Trojan onto the system, which will then listen on TCP port 36794. A listing of active connections will reveal this. This Trojan allows a user connecting remotely to perform a number of functions including:

- Listing and terminating processes.
- Retrieving files, which can also include the password file.
- Performing various file operations including copy, delete, write, read and execute.
- Retrieve system information including username, OS version, processor type, drive and memory information
- The user can use this access to open port 80 and allow access to the backdoor web server, which is part of the worm. If a remote user connects to the backdoor then you will see the following files created in the windows temporary folder:
  ~PHGGUM.tmp
  ~EAYLNLF.tmp

The worm drops a key logger onto the system, which will store keystrokes up to 64k and then attempt to email them to one of about 20 email addresses.

mshaw@hispostbox.com
mannchris@gala.net
gili_zbl@yahoo.com
c.willoughby@myrealbox.com
brdlhow@ml1.net
sc4579@excite.com
jwwatson@excite.com
stevechurchis@excite.com
langobaden@excite.com
jacopo58@excite.com
sctanner@myrealbox.com
erisillen@canada.com
sergio52@mac.com
rvre2736@fairesuivre.com
zr376q@yahoo.com
t435556@email.it
sdsdfsf@callme.as
boxhill@teach.com
stickly@login.pe.kr
vique@aggies.org
sm2001@mail.gerant.com
rwilson@singmail.com

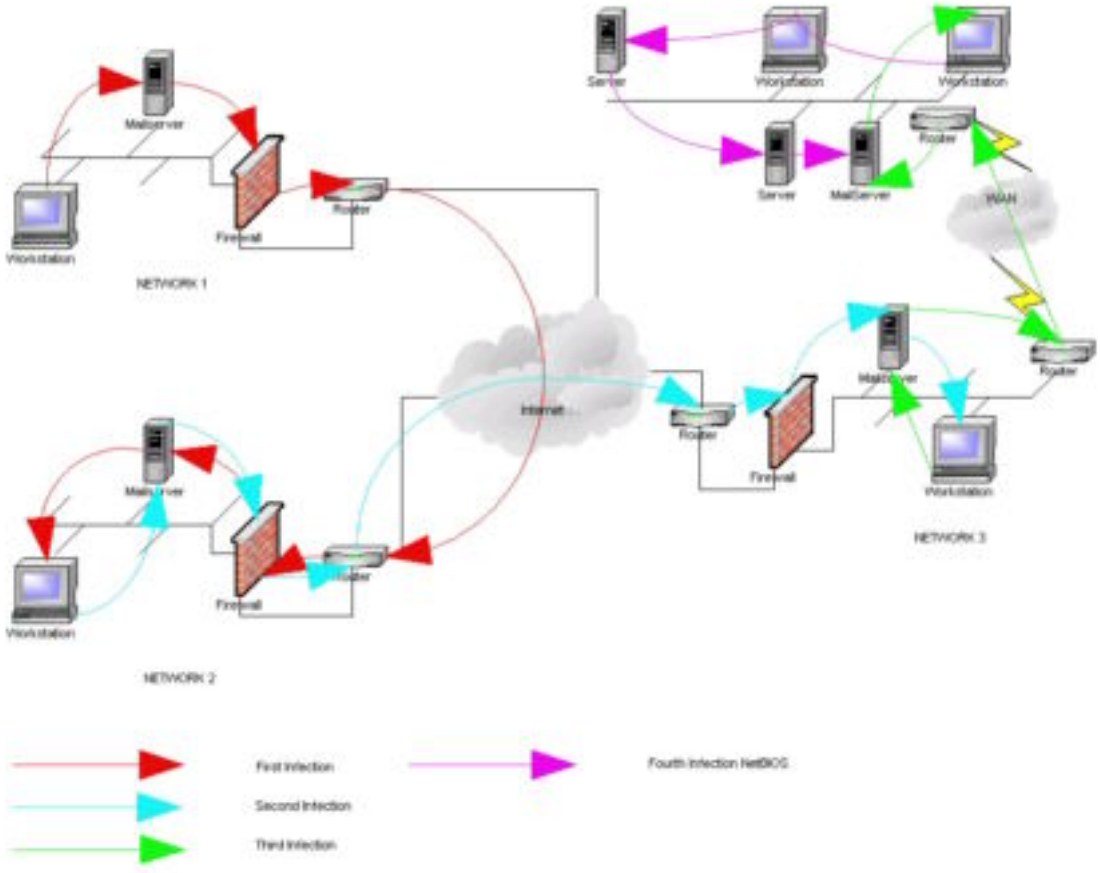
The worm attempts to terminate the processes of a large number of antivirus and host based firewall/IDS systems including the following:

ZONEALARM.EXE, WFINDV32.EXE, WEBSCANX.EXE, VSSTAT.EXE, VSHWIN32.EXE, VSECOMR.EXE, VSCAN40.EXE, VETTRAY.EXE, VET95.EXE, TDS2-NT.EXE, TDS2-98.EXE, TCA.EXE, TBSCAN.EXE, SWEEP95.EXE, SPHINX.EXE, SMC.EXE, SERV95.EXE, SCRSCAN.EXE, SCANPM.EXE, SCAN95.EXE, SCAN32.EXE, SAFEWEB.EXE, RESCUE.EXE, RAV7WIN.EXE, RAV7.EXE, PERSFW.EXE, PCFWALLICON.EXE, PCCWIN98.EXE, PAVW.EXE, PAVSCHED.EXE, PAVCL.EXE, PADMIN.EXE, OUTPOST.EXE, NVC95.EXE, NUPGRADE.EXE, NORMIST.EXE, NMAIN.EXE, NISUM.EXE, NAVWNT.EXE, NAVW32.EXE, NAVNT.EXE, NAVLU32.EXE,

NAVAPW32.EXE, N32SCANW.EXE, MPFTRAY.EXE, MOOLIVE.EXE, LUALL.EXE, LOOKOUT.EXE, LOCKDOWN2000.EXE, JEDI.EXE, IOMON98.EXE, IFACE.EXE, ICSUPPNT.EXE, ICSUPP95.EXE, ICMON.EXE, ICLOADNT.EXE, ICLOAD95.EXE, IBMAVSP.EXE, IBMASN.EXE, IAMSERV.EXE, IAMAPP.EXE, FRW.EXE, FPROT.EXE, FP-WIN.EXE, FINDVIRU.EXE, F-STOPW.EXE, F-PROT95.EXE, F-PROT.EXE, F-AGNT95.EXE, ESPWATCH.EXE, ESAFE.EXE, ECENGINE.EXE, DVP95_0.EXE, DVP95.EXE, CLEANER3.EXE, CLEANER.EXE, CLAW95CF.EXE, CLAW95.EXE, CFINET32.EXE, CFINET.EXE, CFIAUDIT.EXE, CFIADMIN.EXE, BLACKICE.EXE, BLACKD.EXE, AVWUPD32.EXE, AVWIN95.EXE, AVSCHED32.EXE, AVPUPD.EXE, AVPTC32.EXE, AVPM.EXE, AVPDOS32.EXE, AVPCC.EXE, AVP32.EXE, AVP.EXE, AVNT.EXE, AVKSERV.EXE, AVGCTRL.EXE, AVE32.EXE, AVCONSOL.EXE, AUTODOWN.EXE, APVXDWIN.EXE, ANTI-TROJAN.EXE, ACKWIN32.EXE, _AVPM.EXE, _AVPCC.EXE, _AVP32.EXE

Description and Diagram of Attack

This being a worm it is found "In the wild" a may be encountered at any time. A malicious user could of course purposely send it to you by email or allow it to propagate across network shares. Illustrated below are the various scenarios for infection.



The above diagram is a simplistic illustration of how the worm would propagate in the wild. The initial infection is at a workstation in Network 1. This could have been received

either by email or an infection from another share on the same network. The worm executes and uses the address book of the user on the workstation to propagate. It may forge email addresses in the from field so as to appear to have come from another IP address. Here we are illustrating an infection from network 1 to network 2. This was a simple email from the user in network 1 to a user in network 2.

The user in network 2 suffers the same fate and a copy of the worm is sent to network 3 among others. When the worm gets to network 3 it propagates initially through email across the WAN connection. A user on that network receives an infected email from the mail server and becomes infected themselves. Their system has access to network shares across this particular network and thus infects other systems on the network by this route. Below is a truncated example of the worm hopping from one machine to another via network shares.

NetBIOS worm propagation:

```
14:01:26.904508 IP 10.10.1.154.1028 > 10.10.1.123.139: S
2021364595:2021364595(0) win 16384 <mss 1460,nop,nop,sackOK> (DF)
                  4500 0030 0036 4000 8006 e369 0a0a 019a
                  0a0a 017b 0404 008b 787b 9373 0000 0000
                  7002 4000 1b79 0000 0204 05b4 0101 0402
14:01:26.904508 IP 10.10.1.123.139 > 10.10.1.154.1028: S
87054764:87054764(0) ack 2021364596 win 17520 <mss 1460,nop,nop,sackOK>
(DF)
                  4500 0030 bf02 4000 8006 249d 0a0a 017b
                  0a0a 019a 008b 0404 0530 59ac 787b 9374
                  7012 4470 b81b 0000 0204 05b4 0101 0402
14:01:26.904508 IP 10.10.1.154.1028 > 10.10.1.123.139: . ack 1 win
17520 (DF)
                  4500 0028 0037 4000 8006 e370 0a0a 019a
                  0a0a 017b 0404 008b 787b 9374 0530 59ad
                  5010 4470 e4df 0000 2020 2020 2020
.     .     .     .     .     .     .     .     .     .     .     .
.     .     .     .     .     .     .     .     .     .     .     .
.     .     .     .     .     .     .     .     .     .     .     .
.     .     .     .     .     .     .     .     .     .     .     .
.     .     .     .     .     .     .     .     .     .     .     .

14:02:34.924508 IP 10.10.1.123.139 > 10.10.1.154.1028: P 764:807(43)
ack 1034 win 16487 (DF)
                  4500 0053 bf0b 4000 8006 2471 0a0a 017b
                  0a0a 019a 008b 0404 0530 5ca8 787b 977d
                  5018 4067 a566 0000 0000 0027 ff53 4d42
                  7400 0000 0098 07c8 0000 0000 0000 0000
                  0000 0000 0000 fffe 0008 7000 02ff 0027
                  0000 00
14:02:34.924508 IP 10.10.1.154.1028 > 10.10.1.123.139: F 1034:1034(0)
ack 807 win 16714 (DF)
                  4500 0028 0043 4000 8006 e364 0a0a 019a
                  0a0a 017b 0404 008b 787b 977d 0530 5cd3
                  5011 414a e0d5 0000 2020 2020 2020
```

```
14:02:34.924508 IP 10.10.1.123.139 > 10.10.1.154.1028: F 807:807(0) ack
1035 win 16487 (DF)
                  4500 0028 bf0c 4000 8006 249b 0a0a 017b
                  0a0a 019a 008b 0404 0530 5cd3 787b 977e
                  5011 4067 e1b7 0000 2020 2020 2020
```

The worm is continually scanning for network shares:

```
13:52:58.614508 IP 10.10.1.154.137 > 10.255.255.255.137: udp 68
                  4500 0060 000c 0000 8011 23df 0a0a 019a
                  0aff ffff 0089 0089 004c eb63 8004 2810
                  0001 0000 0000 0001 2046 4845 5046 4345
                  4c45 4846 4345 5046 4646 4143 4143 4143
                  4143 4143 4143 4141 4100 0020 0001 c00c
                  0020 0001 0004 93e0 0006 8000 0a0a 019a
```

When it finds one it will copy itself to:

\\share name\%startup%\randomfilename.exe

The new copy of the worm will then execute on this system when it reboots.

The email that is usually the initial source of the infection arrives with from a faked source mail address with no message body and a random subject from the following list:

$150 FREE Bonus!
25 merchants and rising
Announcement
bad news
CALL FOR INFORMATION!
click on this!
Confirmation of Recipes…
Correction of errors
Daily Email Reminder
empty account
fantastic
free shipping!
Get 8 FREE issues - no risk!
Get a FREE gift!
Greets!
hello!
history screen
hmm..
I need help about script!!!
Interesting...
Introduction
its easy
Just a reminder
Lost & Found
Market Update Report
Membership Confirmation
My eBay ads
New bonus in your cash account
New Contests
new reading
Payment notices
Please Help...
Report
SCAM alert!!!
Sponsors needed

Stats
Today Only
Tools For Your Online Business
update
various
Warning!
Your Gift
Your News Alert

Once infected the SMTP engine that is built in to the system is used to send more copies of the worm to the first 170 email addresses that it finds on the infected system. It gets these from address books, cached email messages or mail boxes. It also checks the registry to ensure that it does not send itself to the currently infected user.

The worm contains two SMTP engines and they differ in the way that they encode mail. One sends the worm encoded as application/x-msdownload while the other has a type of audio/x-midi. This second type makes the body of the message html code that will auto execute when loaded in Outlook or Outlook Express, thus infecting the system.

This process continues on until the worm is removed, AV products are brought up to date so as to detect its presence and the MIME header vulnerability is patched to prevent exploitation by new worms / viruses.

Signatures

There are a number of signatures for this worm within the system. Firstly an infected system should be listening on port 36794

System Pre Bugbear:

Active Connections

Proto	Local Address	Foreign Address	State
TCP	test:epmap	test:0	LISTENING
TCP	test:microsoft-ds	test:0	LISTENING
TCP	test:1025	test:0	LISTENING
TCP	test:1027	test:0	LISTENING
TCP	test:netbios-ssn	test:0	LISTENING
UDP	test:epmap	*:*	
UDP	test:microsoft-ds	*:*	
UDP	test:1026	*:*	
UDP	test:netbios-ns	*:*	
UDP	test:netbios-dgm	*:*	
UDP	test:isakmp	*:*	

System Post Bugbear

Active Connections

Proto	Local Address	Foreign Address	State
TCP	test:epmap	test:0	LISTENING
TCP	test:microsoft-ds	test:0	LISTENING
TCP	test:1025	test:0	LISTENING
TCP	test:1027	test:0	LISTENING
TCP	test:36794	test:0	LISTENING
TCP	test:netbios-ssn	test:0	LISTENING
UDP	test:epmap	*:*	
UDP	test:microsoft-ds	*:*	
UDP	test:1026	*:*	
UDP	test:netbios-ns	*:*	
UDP	test:netbios-dgm	*:*	
UDP	test:isakmp	*:*	

The worm will alter the registry to ensure that the program starts on each reboot.

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\
CurrentVersion\RunOnce
<random string> = %System%\<random filename>.EXE

In addition to this the worm copies itself into the windows startup folder with a random 3-character name, e.g. yao.exe

A system administrator may observe spurious print jobs coming from shared printers on the network, this is due the worm copying itself to the printer as a share. The binary code is being copied directly to the printer and is thus producing the unintelligible printer output.

The presence of a new .exe file on the system. This file will be ****.exe where **** is random characters.

The worm also drops 3 *.dll and 2 *.dat files onto the system. One of these dll's is a key logger.

How to Protect Against the Attack

1 As stated the attack is based on a vulnerability that is over 18 months old (March 2001). The primary way to prevent infection by this worm is to patch IE so as not to be susceptible to the MIME header vulnerability.

http://www.microsoft.com/windows/ie/downloads/critical/q290108/default.asp
alternatively upgrade to IE 6.

2 Anti Virus software with signature files dated after the worms discovery will protect against the worm. This is an effective way to protect against infection by BugBear now, but it will not necessarily protect you against other worms or viruses that exploit the same vulnerability. This is why step 1 is the primary way to secure your systems.

http://www.mcafeeb2b.com/naicommon/download/dats/find.asp
http://www.trendmicro.com/download/pattern.asp
http://securityresponse.symantec.com/avcenter/download.html

Microsoft's patch ensures that html emails cannot automatically execute code and therefore the email will not automatically infect the system.

The Incident Response Process

Preparation

The organization has a security policy and also an incident response procedure. The security policy addresses many of the common issues such as appropriate use of email and the Internet. The policy is enforced with a number of technologies including Mailsweeper to protect mail from malicious content and viruses and SurfControl to limit access to the Internet. Antivirus is deployed throughout the organization and is managed through McAfee EPO. Backups are made nightly on a bi-weekly rotation with an additional monthly backup. All tapes are stored off site by the system administrator.

In addition to the above procedures the system administrator is required to take some proactive steps to ensure the security of the network. Keeping informed of the current issues, vulnerabilities and alerts is an essential aspect of this procedure. The system admin is subscribed to the Microsoft security notification service: http://register.microsoft.com/regsys/pic.asp
Additionally he is subscribed to the CERT notification mailing list and also checks the McAfee and Trend Micro websites daily for information regarding new threats to the network in the form of viruses and worms.

The administrator is required to patch systems, applications and software as new vulnerabilities come to light and maintain all systems at current version levels. Anti Virus is deployed and it is the responsibility of the administrator to ensure that all systems have been updated with the latest virus definition files.

Incident response procedures have been established to deal with any incident that may affect the network. The incident response team consists of the system administrator, the technical services manager and support personnel. This incident response team has sufficient knowledge about all the systems within the organization.

Users are made aware of the procedure to be undertaken when an incident has or is occurring. Their first point of contact is the helpdesk, which will in turn alert the system

administrator and technical services manager if it is felt that an incident is or has taken place.

Access to all the systems is ensured through the presence on the incident response team of the system administrator. Although there maybe a conflict of interest in having the System administrator on the IR team it was felt that his presence would ensure that a response to an incident would be much faster with his participation.

Materials for the purpose of responding to an incident are available including log books, CD burner, spare HD's backup tapes, laptops and software. It was felt within the organization that the steps that were taken should enable the organization to avoid any incident and to adequately deal with one should it occur.

Identification.

The process of identification stated from an employee reporting that a printer was printing page after page of unintelligible text. Support investigated this and found that there was activity on the network, which involved a connection to a share for this printer.

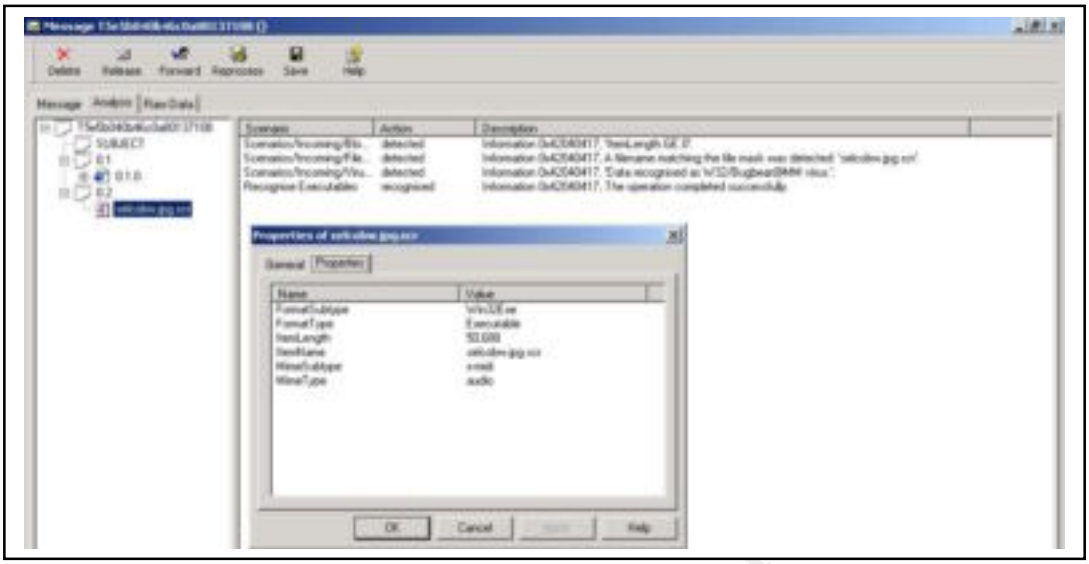
Subsequent investigation revealed that a new worm was spreading on the Internet and that one of the symptoms was unintelligible print jobs occurring. It was at this point that the support staff called for the incident response team to be assembled.

The countermeasures employed by the organization appear to have failed in this instance because:

1) The virus signature file in place on the systems on this date September 30th 2002 did not recognize this worm because it was literally hours old at this time and there were no updates available.
2) Not all copies of IE on the organizations systems had been patched or upgraded. It actually transpired that the majority of the systems still had the vulnerability.

Investigation of the system that appeared to be producing the print jobs revealed other signatures of infection by this worm.

1 There was a new program in the startup directory called yao.exe
2 Examination of the users mailbox revealed that she had received a message with an attachment called selcobw.jpg.scr, which was 50,688 bytes long. This email was from a spoofed source.
3 Other mailboxes had similar emails with similar attachments.
4 The particular infected system was running a service that was listening on port 36794.



This is a screenshot of another instance of the virus, which was detected after virus definitions had been brought up to date. It serves to illustrate the payload size and its identification as W32/Bugbear by the McAfee command line scanner.

The team was fairly confident that this was an infection by this worm BugBear. The initial description of the worm meant that it was a concern because of the various security issues that arose from key loggers , backdoors and mail engines on the system. It was also a concern that the worm could start to deplete system and network resources if it attempted to propagate from all nodes on the network.

This first system was taken offline and removed from the network for further analysis.

Containment

Once it has been established that an incident has or is taking place then containment becomes the number one priority. Lets not let it get any worse. An issue for an incident handler when approaching the containment phase is how do you contain the incident without compromising the evidence. A quick reaction may be to disconnect the affected system and to scrub it clean, but this will destroy evidence and therefore drastically reduce the handler's ability to establish what has occurred. If a case was to go to court then evidence must be preserved in pristine condition to be even considered. This is why the incident handler needs to proceed very carefully at this stage.

In this particular incident the team have so far established that it is likely a system or systems have become infected with the BugBear worm. Initial analysis of this worm indicates that it has several levels of functionality. It propagates itself through SMTP and NetBIOS shares. It has a payload that includes a backdoor, a key logger, a password sniffer etc. The presence of these elements gives early cause for concern regarding how

to proceed in the containment phase. If the worm has not been fully analysed at this stage and contains additional functionality, there may other implications for containment.

Many viruses, Trojans, worms etc contain code that is designed to detect when they have been discovered. A Trojan that is say sniffing the network may be monitoring the network connection to establish that the node has not been disconnected. If it were to detect that the network was suddenly gone it may have code built in designed to remove evidence of its existence, for example it may kill all its own processes and delete itself. A more malicious infection may attempt to hide its existence by wiping the system, which it has infected. It is with this in mind that the team proceed cautiously.

Since the original infection we have found 6 other systems on the network that were infected. These were all workstations within the LAN. We had put a sniffer on the LAN at the segment where the infection occurred to watch for new traffic on the NetBIOS ports, which would indicate any newly infected nodes. A hub was setup and all the infected workstations were transferred directly from the switch to the hub to mimic network connectivity. The mail server was also temporarily transferred to the hub. The mail server didn't exhibit any evidence of infection itself but this was considered to be prudent until we knew more. It also meant that nobody on the network could pick up mail, but also wouldn't get infected from newly delivered mail.

At the mail gateway we didn't have a virus definition file that could detect the presence of the worm yet so we had to put in some filters to quarantine all incoming mails with attachments. So far it appeared that the attachment was a standard size as defined by the AV companies so it was felt that it wouldn't be too difficult to identify all those mails containing the virus at a later time and to release legitimate email into the system. It soon became apparent that this was a prudent course of action as it was observed that there were between 30 and 40 infected messages arriving into the mailboxes per hour.

In the meantime the organization had received several calls from customers and partners suggesting that we had sent them the BugBear Virus. This was very significant in that

1 There was an infection of BugBear on the Network and it was quite possible that mails were sent to customers containing the worm .
2 The worm is known to spoof source addresses and therefore could have come from an entirely different source.

Establishing the facts became a priority. There was a log of mail through the gateway and this had to be analysed to see if mail was sent out during the last few hours containing a suspicious attachment. The team also requested that a copy of the mail including the header be sent over as soon as possible so that they could establish if the mail was spoofed.

Management was informed that some customers felt that they may have been infected with a worm from our network and that the team was establishing the facts.

The systems that were infected had evidence taken from them in case there was a need for future analysis or investigation.

Firstly a list of connections was taken from each machine before transferring to the hub.

```
Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    test:http              test.test.local:0      LISTENING
  TCP    test:epmap             test.test.local:0      LISTENING
  TCP    test:microsoft-ds      test.test.local:0      LISTENING
  TCP    test:1025              test.test.local:0      LISTENING
  TCP    test:1026              test.test.local:0      LISTENING
  TCP    test:1031              test.test.local:0      LISTENING
  TCP    test:1173              test.test.local:0      LISTENING
  TCP    test:1214              test.test.local:0      LISTENING
  TCP    test:1603              test.test.local:0      LISTENING
  TCP    test:1711              test.test.local:0      LISTENING
  TCP    test:1713              test.test.local:0      LISTENING
  TCP    test:1748              test.test.local:0      LISTENING
  TCP    test:1765              test.test.local:0      LISTENING
  TCP    test:1786              test.test.local:0      LISTENING
  TCP    test:1791              test.test.local:0      LISTENING
  TCP    test:1824              test.test.local:0      LISTENING
  TCP    test:1826              test.test.local:0      LISTENING
  TCP    test:1844              test.test.local:0      LISTENING
  TCP    test:1845              test.test.local:0      LISTENING
  TCP    test:1942              test.test.local:0      LISTENING
  TCP    test:3372              test.test.local:0      LISTENING
  TCP    test:netbios-ssn       test.test.local:0      LISTENING
  TCP    test:1603              pc2-grnk1-3-cust114.renf.cable.ntl.com:2500  ESTABLISHED
  TCP    test:1711               host217-42-205-65.range217-42.btcentralplus.com:339910:20
09/30/200210:20 09/30/2002

ESTABLISHED
  TCP    test:1713              pc5-cdif2-4-cust100.cdf.cable.ntl.com:1061  ESTABLISHED
  TCP    test:1748              0x50c6336c.abnxx5.adsl-dhcp.tele.dk:2361  ESTABLISHED
  TCP    test:1765              195.214.128.40:3883    ESTABLISHED
  TCP    test:1770              p50816E2D.dip.t-dialin.net:1214  TIME_WAIT
  TCP    test:1778              80.230.132.220:2943    TIME_WAIT
  TCP    test:1780              pc3-mapp1-3-cust52.nott.cable.ntl.com:3315  TIME_WAIT
  TCP    test:1782              195.158.106.90:1214    TIME_WAIT
  TCP    test:1783              d150-156-118.home.cgocable.net:1746  TIME_WAIT
  TCP    test:1784              dsl-80-46-133-97.access.uk.tiscali.com:2701  TIME_WAIT
  TCP    test:1786              80.230.204.240:1214    ESTABLISHED
  TCP    test:1788              modem-678.llama.dialup.pol.co.uk:1214  TIME_WAIT
  TCP    test:1789              p50816E2D.dip.t-dialin.net:1214  TIME_WAIT
  TCP    test:1790              dijon-1-a7-62-147-210-215.dial.proxad.net:1214  TIME_WAIT
  TCP    test:1791              x1-6-00-08-0e-33-c8-9f.k93.webspeed.dk:1214  ESTABLISHED
  TCP    test:1792              hly-68-112-49-38.nc.charter.com:1576  TIME_WAIT
  TCP    test:1795              dsl-80-46-133-97.access.uk.tiscali.com:2701  TIME_WAIT
  TCP    test:1815              prisoner.iana.org:domain  TIME_WAIT
  TCP    test:1819              prisoner.iana.org:domain  TIME_WAIT
  TCP    test:1820              prisoner.iana.org:domain  TIME_WAIT
  TCP    test:1822              dsl-80-46-133-97.access.uk.tiscali.com:2701  TIME_WAIT
  TCP    test:1823              fia52-115.dsl.hccnet.nl:2867  TIME_WAIT
  TCP    test:1824              rhrh01060.res.utk.edu:1202  ESTABLISHED
  TCP    test:1825              d150-156-118.home.cgocable.net:1746  TIME_WAIT
  TCP    test:1826              ACB34B4A.ipt.aol.com:1214  ESTABLISHED
  TCP    test:1838              dsl-80-46-133-97.access.uk.tiscali.com:2701  TIME_WAIT
  TCP    test:1844              pc2-bary1-4-cust226.cdf.cable.ntl.com:2405  ESTABLISHED
  TCP    test:1845              ip-81-211.evc.net:1853  ESTABLISHED
  TCP    test:1846              ALyon-203-1-3-83.abo.wanadoo.fr:1214  TIME_WAIT
  TCP    test:11767             test.test.local:0      LISTENING
  TCP    test:36794             test:0                 LISTENING
  UDP    test:80                *:*
  UDP    test:epmap             *:*
```

```
  UDP    test:microsoft-ds      *:*
  UDP    test:isakmp            *:*
  UDP    test:1027              *:*
  UDP    test:1030              *:*
  UDP    test:1214              *:*
  UDP    test:1942              *:*
  UDP    test:1071              *:*
  UDP    test:1130              *:*
  UDP    test:1172              *:*
  UDP    test:1558              *:*
  UDP    test:1720              *:*
  UDP    test:10000             *:*
  UDP    test:netbios-ns        *:*
  UDP    test:netbios-dgm       *:*
  UDP    test:11678             *:*
```

Secondly a list of running processes was taken:

```
Process           PID  User
Idle              0
System            8
smss.exe          144  NT AUTHORITY\SYSTEM
csrss.exe         172  NT AUTHORITY\SYSTEM
winlogon.exe      192  NT AUTHORITY\SYSTEM
services.exe      220  NT AUTHORITY\SYSTEM
lsass.exe         232  NT AUTHORITY\SYSTEM
svchost.exe       392  NT AUTHORITY\SYSTEM
SPOOLSV.EXE       420  NT AUTHORITY\SYSTEM
Avsynmgr.exe      448  NT AUTHORITY\SYSTEM
svchost.exe       464  NT AUTHORITY\SYSTEM
regsvc.exe        500  NT AUTHORITY\SYSTEM
mstask.exe        516  NT AUTHORITY\SYSTEM
explorer.exe      684  TEST\Administrator
VSStat.exe        720  NT AUTHORITY\SYSTEM
vshwin32.exe      780  NT AUTHORITY\SYSTEM
Mcshield.exe      796  NT AUTHORITY\SYSTEM
Avconsol.exe      872  NT AUTHORITY\SYSTEM
msiexec.exe       920  NT AUTHORITY\SYSTEM
msimn.exe         672  TEST\Administrator
VAJC.EXE          480  TEST\Administrator
cmd.exe           1044 TEST\Administrator
pulist.exe        1040 TEST\Administrator
```

Once these two pieces of information was recorded we decided to take the system down so as to preserve the contents of the disk. There are two schools of thought about how this should be done.

1 The system should simply have its power disconnected thus preserving everything apart from volatile memory.

2 The system should be shutdown normally which will obviously alter the system as it goes through the shutdown process.

The team decided to use the first approach because these were workstations and can be easily re-imaged. There is no valuable data stored locally so in the event of the system being lost there would not be a problem.

Once the system was brought down an image of the disk was taken using DD. An additional disk was added to the system and it was then booted with Trinux from a CD. Once booted an MD5 sum was taken of the disk and it was then imaged.

dd if=/dev/hda of=/dev/hdb

after the imaging process was completed an additional MD5 sum was taken to verify integrity. An advantage of using Trinux like this was that it was possible to image all the systems simultaneously by using their own hardware thus speeding up the process.

Analysis of the emails that customers and partners had received showed that these were in fact spoofed messages.

Received: from gateway.ourdomain.ie (X.X.1.55 [X.X.1.55]) by mailserver.ourdomain.ie with SMTP (Microsoft Exchange Internet Mail Service Version 5.5.2650.21)

In the example above the domain does not match the IP address. Similarly for the customers that had received the worm a lookup of 'ourdomain.ie' mx record revealed it was in fact not the same address as the one detailed in the header thus showing the mail source to have been forged. The problem however still exists. Customers believe that you have in fact infected their systems with the worm. The only reason that you are aware of this is that they have informed you. What about all those customers that will not inform you and will just accept it ? This could lead to damage to the organizations reputation. In view of this it was felt that there should be a proactive approach taken. Customers would be contacted in advance and informed that they may receive a worm from what appeared to be ourdomain.ie but it was in fact not from us. We would also inform them of what we had learnt so far and how best to deal with the situation. This would we hope contain the worms effects both physically and psychologically.

Eradication

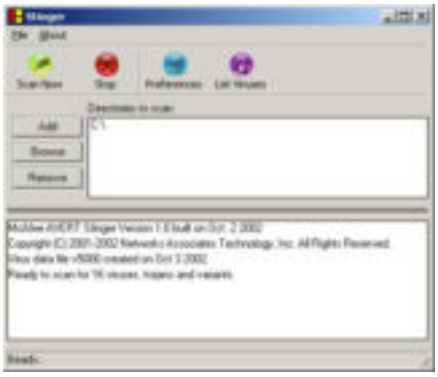
We could simply have wiped the workstations and concentrated our efforts on the email server, but we felt that the image files that we had still contained the MIME header vulnerability anyway and we would have to patch the systems. We decided to eradicate the virus from the systems through restoration from backup, which would retain the systems in their most current configuration, and to then patch the systems to prevent re-infection. The mail server would need a manual clean up as we could not afford to restore from backup for this system.

Each workstation was restored from backup and the systems were patched. An upgrade to IE 6 would be performed later when time was more plentiful. When a virus definition file became available it was applied to the each system.

The mail server didn't appear to have an infection itself as it had no shares and also no mail client viewing messages. We therefore upgraded the system to IE 6.

To prevent further infection of the network from mail within the system it was decided that the mailserver would not be put back on the live network until all workstations were patched and a new virus definition file was installed. This process was completed in around 4 hours by which time a virus definition file had been issued. Two days later a

cleanup tool known as stinger was issued. Stinger was designed to detect and cleanup BugBear, Klez and other such worms and not to be affected by the worms process killing elements. This proved an extremely useful tool.



New copies of the worm were attempting to enter the system all the time so the filter on our gateway prevented re-infection. This gateway was a Mailsweeper 4.2_11 system, which could identify many different file types. It does not rely on extension for the most common data types including exe. This was important here because it was possible the worm would arrive with an extension other than exe.

A virus definition update became available a few hours into the incident response process. We pushed out this new file to all workstations via McAfee EPO. A major advantage of using this product was that it would report on the status of each and every machine. Once each machine was updated we were then in a position to reconnect the mailserver to the network. The AV software on the workstation would catch any mail that had slipped by us. We were confident that there were no systems already infected apart from the 6 already identified. This was of paramount importance because of the worms ability to terminate the processes of AV products.

The cause of the incident was the release into the wild of the BugBear worm, this combined with the systems having unpatched versions of software combined to allow the worm to enter the network and to begin multiplying. The symptoms of this were the heightened NetBIOS activity coupled with the erroneous print jobs and the constant flow of messages with the same attachment size.

Recovery

Each system was restored to a known good state from backup and was immediately patched to prevent further infection. Each systems virus definition file was brought up to date when that file became available.

The network itself was protected from further infection initially through the use of a filter on the mail gateway and then by a valid virus definition file. Once this definition file was in place it was possible to identify all messages that contained BugBear by running them

through the Mailsweeper Policy again. These infected messages were deleted and all others that had been stopped by the filter were released to the mailserver.

After this was completed all systems were reconnected to the network including the original six on the hub and the network was deemed to be functioning normally.

The process of contacting customers continued and an alert was posted on our website.

The image disks were copied to an archive on a file server and the disks themselves were stored for future inspection should this be necessary.

We tested the system with a controlled attempt to get the virus past the gateway and into the network and the provisions that we had put in place seemed to contain it.

Finally we issued information to staff regarding the outbreak of BugBear. We advised them of its ability to spoof and if they received any complaints to contact support.

Lessons Learned

This infestation was caused by a breakdown of the security policy. It was company policy to patch all systems and apply upgrades as and when they became available. It was felt that there was adequate provision for the people responsible to receive information regarding system vulnerabilities in a timely and efficient manner. This was ignored. The policy had been in place for a long time but lacked an element of positive reinforcement, which would have removed the complacency regarding its implementation. Microsoft issues security alerts every week and they were not being acted upon. Although the internal network was well protected from the outside by two firewalls, an IDS, content filtering and anti virus it was still vulnerable. Keeping systems up to date would have prevented this worm from spreading from within the network. Copies of the worm would still have entered the network from outside but they would not have been able to execute and to start propagating themselves. There would have been a small cleanup required when new virus definitions became available but it would not have been necessary to take systems such as the email server offline. Once the team was aware of the worm a filter at the gateway would prevent it entering the network.

The incident concentrated the management's collective mind on the need to reinforce the policy and to make periodic checks of its implementation. Steps that were taken following the incident:

1. A bi-monthly review of the security policy by management and technical staff.
2. A bi-monthly statement of effectiveness of the security policy from the technical services manager.
3. Random checks of systems for un-patched software, missing virus definitions or unauthorised software.

4 Vulnerability testing of the networks defences on an ongoing basis to prepare for other forms of attack.

It was felt that the incident response effort was effective and that the approach taken was the correct one. The loss of the email server for several hours was the most significant negative impact of the incident response process. Upon review it was agreed that it was the right thing to do to prevent the worm from spreading via email. In the absence of a virus definition file there was no guaranteed way to control the spread of the worm.

Extras

Snort:
In addition to the steps taken to detect the worm and to prevent it reinfecting the network there was also the addition of a Snort rule to detect the worm and to monitor the level of activity that the worm was generating in the days after it was initially seen.

```
alert tcp any any -> any 25 (msg:"Bugbear@MM virus in SMTP";
content:"uv+LRCQID7dIDFEECggDSLm9df8C/zSNKDBBAAoGA0AEUQ+FEN23f7doqAT/dC
Qk/xWcEQmDxCTD";
sid:900001; classtype:misc-activity;
rev:1;)
```

Shane Williams devised this rule and details can be found at:
http://www.geocrawler.com/lists/3/SourceForge/6752/0/9765751/

Conclusion

The constant merry go round of new virus – panic – system recovery –new virus…. appears to have life in it yet. Those responsible for the systems that ultimately become infected are simply fighting fires. Once the incident has been completed it is a matter of waiting for the next one to come along. This could be next month, six month who knows. The flaw in this approach is obvious. It accepts that damage will occur and its emphasis is on limiting that damage and then recovering. Preventing the incident in the first place is where our efforts should be. An organization can spend thousands of dollars on a security infrastructure comprising of firewalls, IDS systems, Authentication systems etc. but then totally ignore the most fundamental practices such as patching a system. Information Security magazine in their November 2002 issue list the five worst cyber attacks of all time:

1. Code Red
2. Nimda
3. Melissa & LoveLetter
4. Distributed Denial of Service
5. Remote Control Trojans 1998-2000

Lack of system maintenance is a factor in the top 3, in fact if systems had been properly patched after Nimda then BugBear would not have been effective. The most effective

way that any administrator can protect their network is though proper and timely maintenance of all the systems that they control. The network that was the victim of Bugbear as described in this paper had many security features including firewalls, IDS, AV etc but the vulnerability lay in the systems and it appears that this is what is being exploited again and again in the current environment.

References used in preparation:

Mandia, Kevin and Proisse, Chris, Incident Response, McGraw Hill Publishing, 2001.

Stevens, W Richard, TCP/IP Illustrated, Addison Wesley, 1994.

Microsoft security Bulletin: MS01-020
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-020.asp

Trend Micro Analysis of BugBear Worm
http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_BUGBEAR.A

Sophos AV analysis of BugBear
http://www.sophos.com/virusinfo/analyses/w32bugbeara.html

GCIH Practical Submission by Lloyd L. Conner, August 2002
http://www.giac.org/practical/Lloyd_Conner_GCIH.doc

Snort BugBear detection Rule:
http://www.geocrawler.com/lists/3/SourceForge/6752/0/9765751/

Cert Advisory – CA-2001-06
www.cert.org/advisories/CA-2001-06.html

Vulnerability Note VU#980499
www.kb.cert.org/vuls/id/980499

CVE-2001-0154
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0154