Hal Berghel

Malware Month

August 2003: SoBig, W32/Blaster, and the malware month of the millennium.

August 2003 is officially the worst month on record for Internet malware according to vnunet.com. Of course, the creation and distribution of malware (malicious software) has been on a rapid rise for well over a decade. According to Carnegie Mellon's CERT Coordination Center (CERT/CC), the number of reported incidents rose from six in 1988 (the year of the Morris worm) to 82,094 in 2002, with 76,404 incidents reported in the first half of 2003 alone. The upward trend is unmistakable and frightening. But this past August exceeded everyone's wildest expectations and worst fears. Mi2g (Mi2g.com) estimates that $32.8 billion in economic damages were suffered in August 2003—the largest amount in the history of the Internet. These losses were produced by a variety of malware.

The table here, abridged from Symantec's Security Response Center online listing, illustrates Windows vulnerabilities. My focus in this column is on the two entries in the table that started and ended the week: W32/Blaster and SoBig. A brief analysis of these two exploits provides considerable insight into current hacker's modus operandi. It is important to emphasize that these exploits are Windows-centric because Symantic is a Windows security software and appliance vendor. The Unix world has its own cluster of vulnerabilities, although SoBig and W32/Blaster were not among them. However, Microsoft's hegemony in the desktop/workstation OS realm makes it the hacker target of choice.

W32/Blaster

W32/Blaster (aka LovSan, worm_msblast, Win32.Posa, W32Lovsan, MSBLASTER), in its myriad manifestations, is one of those exploits that will go down in the annals of Internet hacking as a giant thorn in the side of network security experts. Though the origin of this worm has yet to be identified, an 18-year-old Minnesota high school student has confessed to the FBI to releasing at least one of the modifications (W32/Lovesan.worm.b—see the table here) that infected more than 7,000 computers. Overall, more than 1.4 million computers worldwide have been affected by all W32/Blaster varieties since the original infection on August 11, 2003 according to Network Associates' Hackerwatch.org. Figure 1 depicts the daily spread of the

JASON SCHNEIDER

As bad as W32/Blaster was, it paled in terms of the number of computers affected by Sobig, which at its peak accounted for nearly 75% of all email traffic on the Internet.

infection during the week of August 11. The pattern is unforgettable and alarming.

However, the consequences of the W32/Blaster family of worms go far beyond the world of the Minnesota teenager. According to *Computerworld*, "The W32/Blaster worm may have contributed to the cascading effect of the August 14 blackout, government and industry experts revealed….On the day of the blackout, Blaster degraded the performance of several communications lines linking key data centers used by utility companies to manage the power grid…." (*Computerworld*, Aug. 29, 2003). Some have suggested that Blaster interfered with the network exchange of flow-control and load-balancing information the power grid control systems require to coordinate responses to grid anomalies. While Blaster hasn't been blamed for the cascading blackout, some industry analysts have stated that "it certainly compounded the problems."

The final word on the fate of the Minnesota high school student-cum-script kiddie has yet to be written. According to the *Kansas City Star*, Microsoft discovered this variant of the worm used a hard-coded download link to www.t33kid.com to download the primary malware executables. Internet registries linked this site to "teekid" who, at the time this column was written (fall 2003), faced 10 years in prison and a $250,000 fine.

So how does the W32/Blaster family of worms work? The ultimate objective was to launch a port 80 (the primary port for the Web) SYN flood distributed denial-of-service attack against Microsoft's windowsupdate.com site on August 16, 2003 based on a vulnerability discovered a month earlier in the Windows implementation of Remote Procedure Call (RPC). More specifically, the enabling vulnerability was a defect in Microsoft's interface between its Window's Distributed Component Object Model (DCOM) and RPC in Windows NT, 2000, XP, and 2003 Server. Like many OS vendors, Microsoft succumbed to the bête noir of modern programming: sloppy code. In this case, yet another instance of inadequate bounds checking lead to the reoccurrence of the now-familiar buffer overflow category of vulnerabilities.

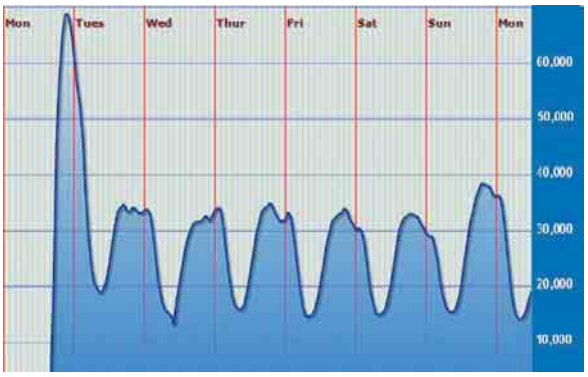

Figure 1. The spread of W32/Blaster-Lovsan during the week of Aug. 11, 2003. Note the peak of 68,000 newly infected IP addresses at 11 p.m. on Monday, Aug. 11. Source: Hackerwatch.org (hackerwatch.org/checkup/graph.asp).

I've discussed buffer overflows in previous columns ("The Code Red Worm," Dec. 2001), so I won't go into detail here but to say the typical OS inserts buffers in stream with instructions when it builds the execution stack. Thus, the full word after the last line of the buffer is presumed to be either an instruction, or a pointer thereto. If one can overflow a buffer (made possible by a lack of bounds checking on the input data), one can substitute a line of errant code (or pointer) into the instruction sequence that can serve as an access point for an exploit. One common variation of this hack is to put in a pointer right after the buffer's end that

points back into the previous buffer which has been overwritten with a "no op" sled followed by rogue code. In this case, one doesn't need know the exact location of the rogue code in the buffer, as the OS will hop through the no op commands until it reaches the first line of executable code. That line of code launches the exploit. In this case, since Microsoft's DCOM runs with local system privileges, the rogue code passed to it through hacked or "crafted" TCP/IP RPC packets will inherit those privileges. RPC is a protocol that enables cross-platform, interprocess communication. So if a crafted RPC packet from a hostile computer can corrupt the target's DCOM, the hostile computer can take over control of the target.

Microsoft released a technical bulletin and patch on July 16, 2003 that addressed the vulnerability. But a patch only fixes the problem if it is installed. Therein lies the rub. The evidence suggests there were at least 1.4 million computer users who didn't bother to install the patch. Of course, one could have protected one's computer even without the patch if one only knew how the exploit worked. But fewer people read Microsoft's technical bulletin than installed the patch. The situation migrated from bad to worse in a hurry.

The W32/Blaster infection sequence was pretty straightforward. The hacking relay sites use basic port scanning to find TCP Ports 135 open (see "URL Pearls" at the end of this column for port listings). Port 135 is the port used by Microsoft to support RPC and Windows Messenger, among other things. Blaster begins its work by port scanning computers to identify open TCP 135 ports and, if found, deposits a variation of the trojan horse program dcom.c which, in turn, executes a remote shell on TCP port 4444 to one of the compromised computers that warehouse the exploit. The warehouse computer then initiates a TFTP session request on UDP port 69, whereupon the target computer opens TFTP and downloads the actual malware. The Windows registry is then modified to autostart the exploit. At that point, the infected computer becomes an unwilling repeater in the distributed denial-of-service attack against the windowsupdate.com site.

Some of the code strings that suggest W32/Blaster infection include those shown in Figure 2. The point is this. Even if one did not patch one's computer as Microsoft recommended, the data in the previous paragraph is more than enough to prevent the exploit from taking root. For example, leaving Microsoft's Server Message Block and NetBios ports (135–139, 445) open is inherently risky. Standard security policy dictates closing them to all traffic at the firewall, or in the OS if no firewall is present.

In addition, blocking the ephemeral port 4444 prevents the initial shell script from executing. Ephemeral ports are negotiated between client and server, so blocking one should have no ill effect.

DATE	EXPLOIT	TYPE	TARGET OF ATTACK
August 11	**W32.Blaster.Worm**	**worm**	**Windows DCOM RPC**
August 11	Backdoor.WinShell.50.b	trojan horse	Windows OS
August 12	W32.Randex.E	worm/trojan	Windows/Internet Relay Chat
August 12	W32.HLLW.Habrack	worm	Windows file sharing networks
August 13	W32.Blaster.B.Worm	worm	Windows DCOM RPC
August 13	W32.Blaster.C.Worm	worm	Windows DCOM RPC
August 13	VBS.Lembra@mm	worm	Microsoft Outlook
August 13	Backdoor.Beasty.H	trojan horse	Internet Explorer
August 14	Backdoor.Graybird.E	trojan horse	Windows security settings
August 14	W32.Kuskus.Worm	worm	Windows file sharing networks
August 14	W32.Randex.F	worm	Windows/Internet Relay Chat
August 14	W32.Randex.G	worm	Windows/Internet Relay Chat
August 15	W32.Bugsoft	worm	Microsoft Outlook
August 15	PWSteal.Lemir.C	trojan horse	Windows online games
August 15	Trojan.Analogx	trojan horse	Windows spoofed proxy server
August 16	W32.HLLW.SShydy.B	worm	Windows file sharing networks
August 16	W32.Randex.H	worm	Windows/Internet Relay Chat
August 16	W32.Dumaru@mm	worm/trojan	Windows/Internet Relay Chat
August 16	BAT.Randren	virus	Windows OS
August 18	W32.Welchia.Worm	worm	Windows DCOM RPC and IIS
August 18	W32.Dinkdink.Worm	worm	Windows DCOM RPC
August 18	**W32.Sobig.F@mm**	**worm**	**SMTP mass mailing worm**

August 11–18, 2003: The bleakest week of malware month.

Another factor is that Blaster uses its own Trivial File Transfer Protocol (TFTP) on TCP/UDP Port 69 to download the exploit. TFTP is an inherent vulnerability, and so this port should be blocked anyway. Finally, recognition of dcom.c and many of the code signatures was already included in the major anti-virus programs prior to August 11. The lesson to be learned from W32/Blaster is that one really had to have one's head in the sand to get infected in the first place.

SoBIG

As bad as W32/Blaster was, it paled in terms of the number of computers affected by Sobig (aka W32.Sobig.X@mm, where X is one of the alphabet varieties). According to vnunet.com, at its peak SoBig accounted for nearly 75% of all email traffic on the Internet. Vnunet adds that any one of the top four viruses and worms dispatched in August 2003 would in itself have been the most significant exploit in an average month. To have four in one month, including W32/Blaster and SoBig, nearly brought some areas of the commercial Internet to a grinding halt. SoBig accounted for nearly 50% of the August 2003 malware exploits reported by many anti-virus vendors.

To make matters worse, SoBig has achieved the hacker holy grail of "most damaging malware in history" ($14.62 billion), surpassing Klez ($13.94 billion) and Love Bug ($8.75 billion) according to the Mi2g SIPS database.

Unlike W32/Blaster, the SoBig worm relies on email for propagation. The ubiquity of email makes SoBig especially pernicious. SoBig's modus operandi is a technique called "email spoofing," where the email addresses are "harvested" from target files with the following extensions: .dbx, .eml, .hlp, .htm, .html, .mht, .wab, and .txt. The email harvesting is performed with any of a number of simple approximate string matching algorithms in the public domain. These harvested email addresses on the compromised host are then used as return-addresses in subsequent mass mailings. SoBig also relies on its own internal SMTP mail server to propagate itself, so it doesn't have to concern itself about tightened security measures on the local SMTP servers. In this way, SoBig also produces two victims: the compromised host and the unsuspecting subject whose email address is contained in one of the files on the compromised host who subsequently receives hate email from the next target downstream.

```
<msblast.exe> (the primary executable of the exploit)
I just want to say LOVE YOU SAN!!
billy gates why do you make this possible ? Stop
making money and fix your software!!
windowsupdate.com
start %s
tftp -i %s GET %s
%d.%d.%d.%d
%i.%i.%i.%i
BILLY
windows auto update
SOFTWARE\Microsoft\Windows\CurrentVersion\Run
```

Figure 2. Example code indicating infection.

Here's how it works. SoBig sends out an email message with a worm in what appears to be a harmless attachment. Relying on four fundamental principles of hacker social engineering, the spoofed email encourages the unsuspecting recipient to open the attachment: 1) the email comes from an authentic email host (stolen from the previous victim's files); 2) the email uses innocuous-seeming subject lines like "Details," "Approved," "Thank You!," (and "Re: Thank You!") "Your Application," and of course pro-forma variations for the curious and devil-may-care among us such as "Wicked screensaver" and "That Movie;" 3) the email contains non-threatening message bodies like "Please see the attached file for details"; 4) and the attachments use harmless file names that appear to be non-executable such as your_document.pif, document_all.pif, details.pif, and wicked_scr.scr. Added together, the social engineering was obviously quite successful.

When the attachment-cum-worm executes, it loads itself in the Windows installation folder as the 72K executable winppr32.exe along with a datafile winstt32.dat.

The worm/install routine links this executable to the Windows registry by adding new values to registry keys within the HKEY_LOCAL_MACHINE and HKEY_CURRENT_USER registry groups so that the winppr32.exe autostarts, all of which leave behind easily detectable hacker trails that form the signatures used by the anti-virus software and intrusion detection systems.

Conclusion

If there's a single lesson in this, it's that eternal vigilance is the best defense against malware. Malware month didn't have to happen—the techniques used in the two most prominent exploits covered here involved nothing particularly innovative. Both were easily preventable by maintaining Windows update patches and hotfixes provided by Microsoft and by following reasonable security policies for Windows computers. Remember the eight M's: Malware month of the millennium made monkeys of many more than Microsoft. C

Hal Berghel (www.acm.org/hlb) is a professor and director of the School of Computer Science and director of the Center for Cybermedia Research at the University of Nevada, Las Vegas.

URL PEARLS

The intellectual magnet for Internet security for the past 15 years has been CERT/CC, a federally funded research center at Carnegie Mellon University. Originally a free-standing DARPA project, CERT/CC is now part of the University's Software Engineering Institute's Networked Systems Survivability Program. The CERT/CC Web site (www.cert.org/nav/index_main.html) is one of two core sites for anyone interested in network security vulnerabilities, incident handling, and reporting. The other mission-critical site is SANS Internet Storm Center (isc.sans.org/)—a virtual cornucopia of data, references, analyses, and alerts. Taken together, CERT/CC and SANS ISC are the points of first contact for network intrusion and detection specialists.

Semantec is one of the leading providers of Windows security software in the computer industry. Its Security Response Center (securityresponse.symantec.com/avcenter/vinfodb.html) contains up-to-date information on known exploits, with links to vendor alerts, patches, and hotfixes.

The *Kansas City Star* coverage of W32/Lovesan.worm.b is available at www.kansascity.com/mld/kansascity/news/breaking_news/6655970.htm.

Vnunet.com's assessment of August, 2003 as the worst month in history for virus and worm infection is available at www.vnunet.com/News/1143336, and www.vnunet.com/News/1143129.

The economic losses due to malware reported here are calculated by Mi2g (Mi2g.com) and reported on the Net-security Web site (net-security.org).

Another site to visit is Hackerwatch.org, which seems to be affiliated with or sponsored by McAfee Security of McAfee anti-virus renown. Special attention should be given to their animation of the spread of W32/Blaster_LovSan—a clever way of depicting the spread of the exploit. Their event maps are also of interest.

There are several databases of Internet port usage. You might find our version at ccr.i2.nscee.edu/port to be easier to use than most, so you might begin there to learn the nuances about Internet ports and services. It should be noted that the assignment of ports to services is based on an honor system to which the hackers do not subscribe.

Finally, technical information on the two exploits discussed here can be found on Carnegie Mellon's CERT Web site—Sobig.F Worm: www.cert.org/incident_notes/IN-2003-03.html; W32/Blaster worm—www.cert.org/advisories/CA-2003-20.html.

Similar information is also available on Windows security vendor's sites and SANS.ORG. C