Global Information Assurance Certification Paper

Building an Automated Behavioral Malware Analysis Environment using Open Source Software

GREM Gold Certification

Author: Jim Clausing, jac@att.com

Adviser: Charles Hornat

Accepted:

Jim Clausing   1

Outline

1.   **Abstract**

This paper describes how an automated behavioral malware analysis environment for analyzing malware targeted at Microsoft Windows can be built using free and open source software.   The environment described here is an evolving work-in-progress, but what is in place to date is the culmination of lessons learned over the last two years of development by the author.   This work was done as part of the author's employment with the AT&T Chief Security Office. While the author uses the royal "we" in places throughout the paper, the work is that of the author except where noted.

2.   **Motivation**

The first question one might ask is why build our own analysis environment when there are a growing number of services (free and otherwise) such as the Norman Sandbox, CWSandbox, Anubis, ThreatExpert, etc. that will do the analysis for us.   The primary answer is that, for privacy and policy reasons, there are some samples of malware that we encounter on a regular basis that we simply are not at liberty to share with other entities or organizations with which we do not have privacy or non-disclosure agreements.   This is a point that probably deserves more emphasis. Samples may be gathered from any number of sources ranging from honeypots to specific network monitoring to internal investigations or from other trusted sources, and in most of those cases, without careful examination of the sample, it is not clear in advance that there is not data in the sample that might be sensitive.   Policy therefore, is that, until proven otherwise, the samples must be treated as if they might contain personally identifiable information.

Jim Clausing   3

One might further ask, well, then Norman, the CWSandbox, and some of the others also come in commercial versions that one could purchase, why not just use them? These and other commercial products have been examined, and in some cases purchased, but we have also noted that these commercial tools sometimes give differing results for the same sample and different results from what our environment produces, for any number of reasons (including, but not limited to issues with virtualization or detection of the sandbox). So, even with these products, we will continue to develop and enhance the environment described here.

3.   Purpose

There are clearly weaknesses and limitations to the environment described herein, some of which may be addressed in the future and some by design or on which we choose not to spend time. The company is not an anti-virus or anti-malware company though security services are among those offered. The goal of this environment is not necessarily a complete understanding of absolutely everything that the malware does. Since the company's business is providing network services, the resources available for malware analysis are necessarily limited. The first purpose of this environment is the quick preliminary analysis of the malware sample to determine whether additional manual analysis is required or desired. The second purpose is to provide input into a security data system from which we can, for example, attempt to track bots, botnets, or network attacks in order to mitigate them and/or alert our customers.

This project was begun essentially without a budget, hence the reliance on free and open-source tools. This also means that it can be relatively easily be duplicated and adapted by others who can

Jim Clausing   4

benefit from the lessons the author has learned to this point. The
only money required to duplicate our environment would be for 2 PCs
(one could use some surplus older desktop machines that were recently
upgraded) and a legal copy of Windows 2000 or Windows XP. We started
the project using Windows 2000. We later added the capability to
choose between Windows 2000 or Windows XP SP2 (no SP3 yet, though it
wouldn't be hard to add) as the client under which to run the sample,
though to be honest, we rarely use the Windows 2000 image anymore
since there are fewer and fewer of those systems installed. Windows
Vista presents a few additional challenges (mostly requiring newer
hardware) which we have not yet addressed since we have yet to deal
with a sample that specifically targets Vista. We shall cross that
bridge when we come to it.

4.   Setup

The environment is built around Joe Stewart's TRUMAN sandnet.
As Joe explains (Stewart, 2005)(Stewart, 2006), the idea behind a
sandnet is that we allow the malicious software to execute in a
controlled environment where it cannot harm any other systems. This
is done by emulating the internet, not allowing connections to the
real thing. This too has its downside, since we cannot see what the
sample might do after it connects to its command and control (C&C)
channel or what might be contained in any additional software it may
download via HTTP, FTP, or some other protocol. Understanding these
limitations, we still feel that TRUMAN provides an excellent basic
framework for our automated behavioral analysis. Unfortunately, Joe
has stopped developing the environment, so we shall distribute our
modifications and bug fixes ourselves as patches on the author's
website (http://handlers.sans.org/jclausing/grem_gold). The TRUMAN

Jim Clausing   5

environment consists of a Linux server (flavor is unimportant, though we use Redhat Enterprise Linux 4 (RHEL4) since that was the corporate standard distribution when the project began, but pretty much any Linux distribution will work here) and the client machine which will at various times run Linux or Windows. David Bianco (Bianco, 2008) has put together some excellent notes on the NSMWiki on setting TRUMAN up for the first time, so we will not repeat any of that here. Our setup differs some from what David describes, for example, we use a client with 512MB of memory so that even with the memory dumps the entire disk image is 2GB. We are using fast ethernet (100MB/sec) NICs in both machines because they were too old to handle 1GB/sec NICs. We also have a much larger RAID array on the server so that we can store the images and analysis from quite a few samples. We use the userland NTFS-3g drivers (Wieers, 2008) to handle mounting NTFS file systems on the Linux server (see the explanation in the disk image analysis section below) and we have chosen to stick with the 4.5.6.x network scheme that Joe describes. The network between the client and the server can be through a real switch or simply a crossover ethernet cable. As stated earlier, we have the ability to choose between Windows 2000 or Windows XP, this is done by changing a softlink to the image that the server returns to the client. The Windows images themselves are basically vanilla Windows installs with just a couple of additional tools and scripts installed. Beyond what David describes in the Wiki, we have added *fport* from Foundstone (now part of McAfee) and dump the output of it and *netstat* to text files that become part of the disk image returned to the server. We have also installed a VNC server (we chose RealVNC, though a free one could be used just as easily) for those instances where some sort of interaction (such as clicking on a button in a dialog box) is required before the malware does anything malicious.

Jim Clausing   6

When Joe originally released TRUMAN, he provided a few scripts to emulate internet services, *pmodump.pl* - a Perl script to reconstruct the virtual memory space of a running process from a physical memory dump, *dumphive* - a program to dump the Windows registry hives, and he captured the network traffic to a pcap file. That was about it. The analyst was left on her/his own to figure out what was going on beyond running the Unix/Linux *strings* command on it and hunting through the registry dump. That quickly proved inadequate for our purposes, so the environment continued to evolve. As a need to extract additional data emerges, we determine where the data might be located and then we find and/or write tools/scripts to extract it. The pieces that we have written or modified to date are all available at the author's website (URL given above).

5.   **Analysis**

As the title of the paper suggests, we are interested in behavioral analysis of the malware we are examining. The question then, is what behavior and how do we analyze it? There are 4 major areas that we concern ourselves with in this analysis: a) network traffic; b) the disk image; c) the memory image; and d) static analysis of the binary. Appendix B shows the analysis flow at a high level.

a)   Network traffic

The network traffic is captured while the sample is being run and saved to a pcap file (well, the first 10,000 packets, we do not really need to capture all of the scanning of some of the noisier samples, like the recent Conficker.A). After the sample is run, the pcap is analyzed using a number of tools. First, we generate a brief

Jim Clausing   7

report showing source and destination IP addresses and ports, the amount of data (in bytes and packets) transferred between the two, and the start and end time of the communication. This is done with *ipaudit* (Rifkin, et al, 2005) (an example of the output from all these tools is in the sample report in Appendix A). The pcap is further analyzed to determine what other protocols are in use by running *tshark* (the command-line version of wireshark (Coombs, 2008)) against it to get the protocol summary, then *tcptrace* (Ostermann, 2003) is run against it to pull out any HTTP traffic on port 80. As mentioned earlier, TRUMAN comes with several scripts to emulate network services. We have instrumented and extended these scripts to have them generate log files that we include in the reports (again, the logs are shown in the report in Appendix A, the modified scripts are available from the author's website). These scripts implement fake DNS, SMTP, SMB, and IRC servers. We also take advantage of the Apache server we have on the server that uploads the malware and disk images to the client. In particular, we have installed a self-signed SSL certificate and use the Apache logs to see what HTTPS URLs the malware may be attempting to download. We then take advantage of the redirection capabilities of *iptables* (standard on most Linux distributions) on the server, to redirect traffic to most other TCP ports back to the IRC server since most of the malware that we have examined to date use IRC for command and control and the script that emulates IRC records all of the data that it receives in its log file. This does leave us with some problems when the malware uses some sort of peer-to-peer communication for command and control and that is an area we continue to investigate for better solutions. At the moment, this data ends up in the pcap file and in the IRC log which has worked for us, but there must be better solutions.

Jim Clausing 8

One of the results of this analysis phase is information on DNS names and/or IP addresses that may need to be investigated for the presence of either a botnet controller or additional malware. In some cases, it may also provide data on how the malware propagates and may suggest mitigation or detection steps that could be taken within the network.

b) Disk image

One of the behaviors that we are interested in is whether or not the malware creates or modifies any files on the disk. As part of the TRUMAN run process, the malware is placed in C:\Windows\system32\sandnet.exe on the client, so we run *AIDE* (Lehti, et al, 2006) against the disk image after the malware is run and compare it to the clean image that was uploaded to the client before the run and take note of files added, modified, or deleted during the malware run. We mentioned earlier that we run the NTFS-3g drivers. These are not the standard NTFS drivers on RHEL4 (though they are on some newer distributions), but some time back we began to worry that with the stock drivers we might not be able to detect changes the malware might make to NTFS Alternate Data Streams (ADS). After some research, we determined that the ADS were detectable under Linux if we ran the NTFS-3g drivers, so we made the switch. While the latest version of *AIDE* should be able to look for changes to extended attributes (which is how the NTFS-3g represents ADS), we additionally run *getfattr* against the image to look for the presence of ADS.

Another area of the disk image that we investigate is the Windows registry. For this effort, we use 3 different tools, 2 sets of Perl scripts and one binary. The first tool we tried for this (and we still keep this version of the output, but frankly don't look

Jim Clausing   9

at it too much anymore) is *dumphive* (version 07-31-2004-fpc) included
with the TRUMAN distribution. We ran this against the pristine image
and we run it against the image after the malware has been run and
compare using the Linux *diff -u* command. The biggest problem with
this (and the next tool we tried) is that many registry values and
timestamps change in the normal course of operation, so trying to
determine which changes/additions/deletions were malicious was
difficult, and frankly we know that some went unnoticed. The next
tool we tried was *regdiff.pl* by James McFarland (McFarland, 2008).
As with the previous tool, the problem was in attempting to determine
which changes were malicious and which were just part of the normal
course of operation. When we started using this tool it was version
0.30. The most recent versions of the package contain a couple of
additional scripts that we should probably take a look at, such as
regtimeline.pl. Most recently, we have been using Harlan Carvey's
regripper (Carvey, 2008) software to concentrate on any changes to
Run/RunOnce keys or Services (though this software has the ability to
do much more than we are doing with it). These are the primary
methods by which the malware might attempt to ensure that it runs
again after the system has been rebooted. One of the problems with
regripper is that it is primarily intended to run as a GUI
application under Windows itself, though we have recently become
aware of several methods for running the GUI on Linux. Since we wish
to do our analysis in an automated and scripted fashion under Linux,
we again made a few changes to the software. In particular, Harlan
expects that the memory image, script, and the plugins subdirectory
will all be in the same directory. We modified the command-line
script *rip.pl* to fix a minor bug and allow us to specify on the
command-line where the plugins directory is located. We have
submitted the patches back to Harlan, but he has, to this point,

Jim Clausing   10

chosen not to include them in the current version of *regripper*. Our patch can be found on the author's website. We also used a couple of Harlan's plugins as the basis for creating a couple of our own (and we changed the output to a pipe delimited format that worked better for our purposes). These, too, are available from the author's website. From this phase of analysis, we gather a great deal of data that can be useful for investigators who examine (potentially) infected systems.

c) Memory image

As noted earlier, one of the scripts that Joe Stewart includes with the TRUMAN package is *pmodump.pl*, a Perl script that searches through a memory dump for a particular process, then extracts the executable from the process' virtual memory. When Microsoft released Windows XP SP2, they introduced PEB randomization which changed the location of some of the process data structures in the memory dump (Whitehouse, 2007). When we added the Windows XP SP2 image to our environment, the original *pmodump.pl* was unable to locate some of the processes in memory in order to extract the executable. We have patched *pmodump.pl* to handle this PEB randomization and sent the patch back to Joe Stewart, but as he is no longer working on TRUMAN, he has not released an updated version of *pmodump.pl*. This patch can be found on the author's website. Once the executable is extracted, we run the Linux *strings* command on it and look for "interesting" ones. In particular, we look for IP addresses and DNS names of possible command and control servers. We also look for evidence that the malware attempts to subvert antivirus software by either shutting it down or adding entries to the hosts file to redirect update attempts to 127.0.0.1 (localhost).

Jim Clausing   11

One of the huge advantages of an environment that allows the malware to run on an actual machine, is that we can get the software to unpack itself, so that we do not need to necessarily identify (though we try, see static analysis section below) and find an unpacker for the particular packing/obfuscation used on the sample. We also use Jesse Kornblum's *ssdeep* (Kornblum, 2009) to do a fuzzy hash of the unpacked version to use for correlating samples to determine which samples might be related to which other samples.

For additional memory analysis, we rely primarily on the outstanding work of AAron Walters and friends on the Volatility framework (Walters, 2008) and anxiously await the new features and capabilities coming in version 2.0. We actually use the *vaddump* and *procdump* plugins to dump the virtual memory of running processes the same way as *pmodump.pl* does, and we do the same strings analysis of those dumps. We further, use the *pslist* and *psscan2* plugins and compare their output to each other and to the builtin *netstat* command to see if any of the running processes are attempting to hide themselves via some sort of rootkit. Similarly, we use the output from *fport* and the *connections* (and *sockets*) plugin(s) and compare them against the results of the *connscan2* and *sockscan2* plugins again looking for processes that are attempting to hide themselves from the system. The *connections* and *connscan2* plugins should show us established network connections, while the *sockets* and *sockscan2* plugins show ports on which the malware may be listening for incoming network connections. We encounter numerous malware samples that inject DLLs into the Windows Explorer process or get loaded as a running service via svchost.exe, but dumping those entire processes leaves us wading through a great deal of extraneous data, so we continue to investigate other methods of analyzing these samples (see

Jim Clausing   12

conclusions and future work section).

d) Static Analysis of the binary

The static analysis performed on the malicious software is fairly limited at this time. We begin by running the Linux *file* program against it which usually just tells us that the malware is a Windows binary, but sometimes it also tells us about packers. We run the binary through anti-virus software (currently using the corporate standard A/V software and ClamAV (Sourcefire, 2009) since it is free), with signatures updated nightly. We take multiple hashes of the binary, the industry standards, MD5 and SHA1, plus *ssdeep* as described earlier for correlation purposes. We also run the Linux *objdump* against the binary. This may show us some of the structure of the binary, what the various sections of the binary are, etc. We also run Chris Rohlf's *binhash* tool (Rohlf, 2007) which generates MD5 hashes of the headers and data in each of the sections of the PE file. The original intent was to use this data for correlation purposes, but we have gotten sidetracked from that effort, so at the moment, this data just sits in the report. We hope to return in the coming months to this avenue of investigation and see if this is a useful tactic for correlating related malware. We then use the author's *packerid.py* script (Clausing, 2007) to attempt to identify any packer that may have been used on the binary (this script utilizes Ero Carrera's *pefile* library (Carrera, 2008) and as such is capable of producing more data than we actually use it for at the moment). We further run *strings* against the original binary. If packers are involved, this last step does not usually yield us much of anything useful.

Jim Clausing   13

e) Pulling it all together

The entire analysis process is run from the *submit.sh* script (found on the author's website) which oversees the entire process. It takes as an argument, the name of a binary or a ZIP file. If the latter, it unzips the file. Then it copies the binary to the directory from which it will be uploaded to the client, starts up the fauxservers and then starts up *ngrep* and *tcpdump* which are displayed on the tty so that the analyst can monitor the network traffic as the sample runs. This is useful, in that it allows us to recognize when the sample is attempting to communicate on a port that is not being redirected or when it is not communicating at all which may be an indication that there is a dialog box on the client that requires a click before the malware can continue execution (in which case we can fire up VNC). When the 10 minute run on the client terminates and the disk image has been transferred back to the server, the *submit.sh* script, executes the *forensics.sh* script (also available on the author's website) which runs all the aforementioned tools on the disk, memory, and binary described in the preceding section. Finally, the *submit.sh* script runs the *summary.sh* script (also available on the author's website) which pulls the results of all the tools into a text report which summarizes the results of the analysis (see the example in Appendix A).

6. Conclusions, Future and Related Work

As stated at the beginning of this paper, the environment described here is very much a work-in-progress (as evidenced by the fact that this paper has taken 4 months to write because we continually went back and fixed/tweaked the environment as we were writing). We have noted limitations at various points as we have

Jim Clausing   14

discussed the environment and we will undoubtedly address some of them. The environment as it currently exists can process one sample every 20-25 minutes which includes 10 minutes of actual execution time on the client. The primary factor limiting throughput at the moment is the network hardware, since the automated analysis completes before the pristine image has been downloaded back to the client. We mentioned earlier that we had installed a VNC server on the client. Early on we discovered that some samples did not seem to do anything during the 10 minutes that they ran or in very rare cases, the execution did not terminate after 10 minutes. With the addition of the VNC server, we can, when necessary, connect to the client and click on dialog boxes. This means, however, that we have not achieved the ability for entirely hands-off analysis. There are some tools that we may consider adding in the future to overcome this limitation.

We have found the fauxserver scripts to be adequate in simulating the internet for most of our samples, but Tyler Hudak (Hudak, 2009) recently noted in his blog that he is using InetSim (INetSim, 2008), so we may take a look at that to see if it provides us any additional capabilities.

One area that we continue to work on is the memory analysis. We are beginning to look at Brendan Dolan-Gavitt's registry plugins (Dolan-Gavitt, 2009) to Volatility to see if the in-memory version of the Windows registry has been modified but not yet written back to disk. We also are looking at Michael Hale Ligh's *malfind.py* (Ligh, 2009) for finding malicious DLLs injected into other running processes.

One of the other huge limitations of this environment is that the analysis is all done after the execution of the malware. The

Jim Clausing   15

environment is not instrumented to collect information on system and library calls as the malware executes. The recently announced Zerowine project (Koret, 2009), however, appears very intriguing as it does exactly that, so we may look into augmenting the current environment by also running the sample through zerowine at some point in the future.

7. **Acknowledgements**

This work would not have been possible with out the support of our management, Cynthia Cama, Bill O'Hern, and Ed Amoroso, and the assistance and suggestions of the rest of the team. In particular, we'd like to thank John Hogoboom, Dave Gross, and Brian Rexroad for their thoughts and allowing the author to bounce ideas off them.

8.   References

Stewart, Joe (2005). TRUMAN - The Reusable Unknown Malware Analysis
    Net (Version 0.1) [Software]. Retrieved January 31, 2009 from
    http://www.secureworks.com/research/tools/truman.html

Stewart, Joe (2006). Behavioural malware analysis using Sandnets.
    *Computer Fraud and Security, Volume 2006, Issue 12, December
    2006, pp. 4-6.*

Bianco, David (2008). Truman Installation Notes. Retrieved January
    31, 2009 from NSMWiki:
    http://nsmwiki.org/Truman_Installation_Notes

Foundstone (n.d.). Fport (version 2.0) [Software]. Retrieved
    January 31, 2009 from
    http://www.foundstone.com/us/resources/proddesc/fport.htm

Rifkin, J., Pezzella, L., Khalil, H., and Hewlett, J. (2005). IPAudit
    (Version 1.0BETA2) [Software]. Retrieved January 31, 2009 from
    http://ipaudit.sourceforge.net

Wieers, Dag (2008). NTFS-3g (version 1.2310) [Software]. See also
    http://www.ntfs-3g.org. Retrieved October 1, 2008 from
    http://dag.wieers.com/rpm/packages/fuse-ntfs-3g/

Combs, G. (2008). Wireshark (version 1.0.5) [Software]. Retrieved
    January 31, 2009 from http://www.wireshark.org

Ostermann, Shawn (2003). Tcptrace (version 6.6.7) [Software].
    Retrieved January 31, 2009 from http://www.tcptrace.org

Lehti, R., Virolsinen, P., van den Berg, R. (2006). AIDE – Advanced
    Intrusion Detection Environment (version 1.13.1) [Software].
    Retrieved January 31, 2009 from
    http://sourceforge.net/projects/aide

Jim Clausing   17

Whitehouse, Ollie (2007). An Analysis of Address Space Layout
    Randomization on Windows Vista. Retrieved January 31, 2009 from
    http://www.blackhat.com/presentations/bh-dc-
    07/Whitehouse/Paper/bh-dc-07-Whitehouse-WP.pdf

Carvey, Harlan (2008). RegRipper (version 20080909) [Software].
    Retrieved October 21, 2008 from
    http://regripper.net/RegRipper/RegRipper/rr_20080909.zip

Kornblum, Jesse (2009). Ssdeep (version 2.1) [Software]. Retrieved
    January 31, 2009 from http://ssdeep.sourceforge.net/#download

Walters, AAron (2008). Volatility (version 1.3_beta) [Software].
    Retrieved August 31, 2008 from
    https://www.volatilesystems.com/volatility/1.3/Volatility-
    1.3_Beta.tar.gz

Dolan-Gavitt, Brendan (2009). Volatility Registry dump modules
    (version 0.2) [Software]. Retrieved January 31, 2009 from
    http://kurtz.cs.wesleyan.edu/%7Ebdolangavitt/memory/volreg-
    0.2.tar.gz

Ligh, Michael Hale (2009). Malfind.py (version 1.0) [Software].
    Retrieved January 31, 2009 from http://mhl-malware-
    scripts.googlecode.com/files/malfind.py

Sourcefire (2009). ClamAV (version 0.94.2) [Software]. Retrieved
    January 31, 2009 from http://www.clamav.net/download/

Clausing, Jim (2007). Packerid.py (version 1.3) [Software].
    Retrieved February 8, 2009 from
    http://handlers.sans.org/jclausing/packerid.py

McFarland, James (2008). Regdiff.pl (version 0.40) [Software].
    Retrieved September 30, 2008 from
    http://search.cpan.org/CPAN/authors/id/J/JM/JMACFARLA/Parse-
    Win32Registry-0.30.tar.gz

Rohlf, Chris (2007). Binhash (version 0.6.0) [Software]. Download

page has been removed, he asks that anyone interested in the software contact him through his blog at http://em386.blogspot.com

Carrera, Ero (2008). Pefile (version 1.2.10-60) [Software]. Retrieved January 31, 2009 from http://pefile.googlecode.com/files/pefile-1.2.10-60.tar.gz

Koret, Joxean (2009). Zerowine (version 0.0.2) [Software]. Retrieved January 31, 2009 from http://sourceforge.net/projects/zerowine

Hudak, Tyler (2009). InetSim Installation. Retrieved February 12, 2009 from http://secshoggoth.blogspot.com/2009/02/inetsim-installation.html

INetSim (2008). INetSim (version 1.1) [Software]. Retrieved February 12, 2009 from http://www.inetsim.org/index.html

Jim Clausing   19

9.   Appendix A

Summary report for 1.exe-200902041047-XPSP2-files created at Wed Feb 4 11:11:54 EST 2009

```
OS info>>>
kern - Determine OS from a Windows RAM Dump (v.0.1_20060914)
Ex: kern <path_to_dump_file>

File Description   : NT Kernel & System
File Version       : 5.1.2600.2180 (xpsp_sp2_rtm.040803-2158)
Internal Name      : ntoskrnl.exe
Original File Name  :
Product Name       : Microsoft  Windows  Operating System
Product Version    : 5.1.2600.2180

Registry Run Key changes>>>
+test|spoolsv.exe

Registry Service Key changes>>>

Packer info>>>
None

Alternate Data Streams>>>

Strings (if interesting-strings is non-zero see below)>>>
643 interesting-strings.txt
-rw-r--r--  1 root root  22205 Feb  4 11:11 0d529000-spoolsv.exe.strings.txt
-rw-r--r--  1 root root 573191 Feb  4 11:11 spoolsv.exe.2233da0.strings.txt
-rw-r--r--  1 root root 132042 Feb  4 11:11 System.23caa00.00010000-00033fff.strings.txt
-rw-r--r--  1 root root    653 Feb  4 11:11 System.23caa00.00060000-00060fff.strings.txt
-rw-r--r--  1 root root    712 Feb  4 11:11 System.23caa00.00070000-0016ffff.strings.txt
-rw-r--r--  1 root root  13995 Feb  4 11:11 System.23caa00.7c900000-7c9affff.strings.txt

Open Ports>>>
Local Address           Remote Address          Pid
4.5.6.7:1075            4.3.2.192:445           4
4.5.6.7:1030            4.3.2.133:18067         196

196    1030   6      Fri Feb 04 15:51:03 2005
4      1077   6      Fri Feb 04 16:01:05 2005
1844   1034   6      Fri Feb 04 15:51:11 2005

9,11c9,14
< 992                    -> 1032  TCP
> 196    spoolsv        -> 1030  TCP   C:\WINDOWS\system\spoolsv.exe
> 1844                   -> 1034  TCP
> 0      System         -> 1079  TCP
> 0      System         -> 1080  TCP
13,14c16,19
< 0      System         -> 123   UDP
< 992                   -> 138   UDP
> 1844                  -> 123   UDP
> 4      System         -> 123   UDP
> 196    spoolsv        -> 137   UDP   C:\WINDOWS\system\spoolsv.exe
> 0      System         -> 138   UDP
17,18c22
```

Jim Clausing   20

```
< 0      System        -> 1028   UDP

10c10,13
<   TCP   127.0.0.1:1031        0.0.0.0:0            LISTENING
---
>   TCP   4.5.6.7:1030          4.3.2.133:18067      ESTABLISHED
>   TCP   4.5.6.7:1079          4.5.6.7:445          TIME_WAIT
>   TCP   4.5.6.7:1080          4.5.6.7:445          TIME_WAIT
>   TCP   127.0.0.1:1034        0.0.0.0:0            LISTENING
13c16
<   UDP   0.0.0.0:1027          *:*
---
>   UDP   0.0.0.0:1029          *:*
20d22
<   UDP   127.0.0.1:1028        *:*


BinHash info>>>

File: [/forensics/exes/1.exe]    e07e68b63686ef7bdb20b23c0f192640
PE Phdr:     7cbfaa22c60abdbd2f50c91c8b6e617f
PE Opt Hdr:  e18da0d9900c279cb8dfe2e92e566e1c

ssdeep info>>>
3072:btvg4wwF+BlkOY4GkgSHct/UQOUSDJqBPOrCQc1Is:5vpwwF+Bc4RHcCQq8BPOePj,"/forensics/exes/1.ex
e"


6144:RvXnJkIlxAXuIwkcylqhrKnFAgSvpwwF+Bc4RHcCQq8Lpr8:ZnJkmAXokcywwnF90J+Bc4R8CX81r,"/data/fo
rensics/1.exe-200902041047-XPSP2-files/0d529000-spoolsv.exe"

Protocol Hierarchy Summary

===================================================================
Protocol Hierarchy Statistics
Filter: frame

frame                                frames:9431 bytes:6969218
  eth                                frames:9431 bytes:6969218
    arp                              frames:38 bytes:1974
    ip                               frames:9393 bytes:6967244
      tcp                            frames:461 bytes:161394
        http                         frames:25 bytes:6165
          data-text-lines            frames:7 bytes:2450
        data                         frames:7 bytes:758
        nbss                         frames:61 bytes:13278
          smb                        frames:61 bytes:13278
      udp                            frames:8894 bytes:6774094
        bootp                        frames:16 bytes:6960
        tftp                         frames:8820 bytes:6760419
        nbns                         frames:36 bytes:3582
        dns                          frames:12 bytes:1090
        ntp                          frames:2 bytes:180
        nbdgm                        frames:8 bytes:1863
          smb                        frames:8 bytes:1863
            mailslot                 frames:8 bytes:1863
              browser                frames:8 bytes:1863
        short                        frames:10 bytes:29740
      igmp                           frames:4 bytes:240
      icmp                           frames:24 bytes:1776
```

Jim Clausing   21

```
=====================================================================

DNS>>>
request: name=cc.republicofskorea.info, class=IN, type=A, peer=4.5.6.7
responseIP: 4.3.2.133
responseIP: 4.3.2.135
response: rcode=NOERROR, ans=Net::DNS::RR::A=HASH(0x9a5bdd4)
Net::DNS::RR::A=HASH(0x9a5d420), auth=, add=, aa=1
request: name=time.windows.com, class=IN, type=A, peer=4.5.6.7
responseIP: 4.5.6.1
response: rcode=NOERROR, ans=Net::DNS::RR::A=HASH(0x9a5d498), auth=, add=, aa=1
request: name=q.mwa.att.com, class=IN, type=A, peer=4.5.6.7
responseIP: 4.3.2.115
responseIP: 4.3.2.53
response: rcode=NOERROR, ans=Net::DNS::RR::A=HASH(0x9a5d498)
Net::DNS::RR::A=HASH(0x9a647a8), auth=, add=, aa=1
request: name=\194\184.mwa.att.com, class=IN, type=A, peer=4.5.6.7
responseIP: 4.3.2.156
responseIP: 4.3.2.178
response: rcode=NOERROR, ans=Net::DNS::RR::A=HASH(0x9a6833c)
Net::DNS::RR::A=HASH(0x9a52930), auth=, add=, aa=1
request: name=wpad.mwa.att.com, class=IN, type=A, peer=4.5.6.7
responseIP: 4.3.2.191
responseIP: 4.3.2.77
response: rcode=NOERROR, ans=Net::DNS::RR::A=HASH(0x9a68318)
Net::DNS::RR::A=HASH(0x9a5bc84), auth=, add=, aa=1
request: name=\195\165.mwa.att.com, class=IN, type=A, peer=4.5.6.7
responseIP: 4.3.2.192
responseIP: 4.3.2.60
response: rcode=NOERROR, ans=Net::DNS::RR::A=HASH(0x9a5bb04)
Net::DNS::RR::A=HASH(0x9a5bdec), auth=, add=, aa=1

IRC>>>
2009-02-04-10:50:28: Connection from 4.5.6.7
2009-02-04-10:50:28: NICK [USA|00||217496]
2009-02-04-10:50:29: USER XP-7867 * 0 :ATT
2009-02-04-10:50:29: MODE [USA|00||217496] -ix
2009-02-04-10:50:29: JOIN ##d0 d!
2009-02-04-10:57:08: QUIT Ping Timeout? (419-20)399/200

SMTP>>>

HTTP>>>

mod_http: Capturing HTTP traffic (port 80)
1 arg remaining, starting with '../small.pcap'
Ostermann's tcptrace -- version 6.6.7 -- Thu Nov  4, 2004

70 packets seen, 70 TCP packets traced
elapsed wallclock time: 0:00:00.014828, 4720 pkts/sec analyzed
trace file elapsed time: 0:07:28.438956
Http module output:
4.5.6.7:1048 ==> 4.3.2.115:80 (a2b)
  Server Syn Time:     Wed Feb  4 10:53:32.416062 2009 (1233762812.416)
  Client Syn Time:     Wed Feb  4 10:53:32.416021 2009 (1233762812.416)
  Server Fin Time:     Wed Feb  4 10:53:32.433715 2009 (1233762812.434)
  Client Fin Time:     Wed Feb  4 10:53:32.434261 2009 (1233762812.434)
    OPTIONS / HTTP/1.1
Response Code:       200 (OK)
```

Jim Clausing   22

```
Request Length:      135
Reply Length:        197
Content Length:      0
Content Type  :      httpd/unix-directory
Time request sent:   Wed Feb  4 10:53:32.416526 2009 (1233762812.417)
Time reply started:  Wed Feb  4 10:53:32.433628 2009 (1233762812.434)
Time reply ACKed:    Wed Feb  4 10:53:32.434012 2009 (1233762812.434)
Elapsed time:  17 ms (request to first byte sent)
Elapsed time:  17 ms (request to content ACKed)
4.5.6.7:1049 ==> 4.3.2.115:80 (c2d)
  Server Syn Time:       Wed Feb  4 10:53:32.435801 2009 (1233762812.436)
  Client Syn Time:       Wed Feb  4 10:53:32.435760 2009 (1233762812.436)
  Server Fin Time:       Wed Feb  4 10:53:32.436892 2009 (1233762812.437)
  Client Fin Time:       Wed Feb  4 10:53:32.437283 2009 (1233762812.437)
4.5.6.7:1052 ==> 4.3.2.191:80 (e2f)
  Server Syn Time:       Wed Feb  4 10:53:43.473343 2009 (1233762823.473)
  Client Syn Time:       Wed Feb  4 10:53:43.473295 2009 (1233762823.473)
  Server Fin Time:       Wed Feb  4 10:53:43.482897 2009 (1233762823.483)
  Client Fin Time:       Wed Feb  4 10:53:43.483289 2009 (1233762823.483)
    GET /wpad.dat HTTP/1.0
Response Code:       404 (Not Found)
Request Length:      109
Reply Length:        461
Content Length:      282
Content Type  :      text/html;
Time request sent:   Wed Feb  4 10:53:43.473794 2009 (1233762823.474)
Time reply started:  Wed Feb  4 10:53:43.482832 2009 (1233762823.483)
Time reply ACKed:    Wed Feb  4 10:53:43.483043 2009 (1233762823.483)
Elapsed time:  9 ms (request to first byte sent)
Elapsed time:  9 ms (request to content ACKed)
4.5.6.7:1061 ==> 4.3.2.156:80 (g2h)
  Server Syn Time:       Wed Feb  4 10:56:12.353494 2009 (1233762972.353)
  Client Syn Time:       Wed Feb  4 10:56:12.353443 2009 (1233762972.353)
  Server Fin Time:       Wed Feb  4 10:56:12.354455 2009 (1233762972.354)
  Client Fin Time:       Wed Feb  4 10:56:12.354943 2009 (1233762972.355)
    OPTIONS / HTTP/1.1
Response Code:       200 (OK)
Request Length:      135
Reply Length:        197
Content Length:      0
Content Type  :      httpd/unix-directory
Time request sent:   Wed Feb  4 10:56:12.353944 2009 (1233762972.354)
Time reply started:  Wed Feb  4 10:56:12.354371 2009 (1233762972.354)
Time reply ACKed:    Wed Feb  4 10:56:12.354696 2009 (1233762972.355)
Elapsed time:  0 ms (request to first byte sent)
Elapsed time:  1 ms (request to content ACKed)
4.5.6.7:1062 ==> 4.3.2.156:80 (i2j)
  Server Syn Time:       Wed Feb  4 10:56:12.356234 2009 (1233762972.356)
  Client Syn Time:       Wed Feb  4 10:56:12.356192 2009 (1233762972.356)
  Server Fin Time:       Wed Feb  4 10:56:12.357135 2009 (1233762972.357)
  Client Fin Time:       Wed Feb  4 10:56:12.357465 2009 (1233762972.357)
4.5.6.7:1081 ==> 4.3.2.192:80 (k2l)
  Server Syn Time:       Wed Feb  4 11:01:00.850749 2009 (1233763260.851)
  Client Syn Time:       Wed Feb  4 11:01:00.850710 2009 (1233763260.851)
  Server Fin Time:       Wed Feb  4 11:01:00.851744 2009 (1233763260.852)
  Client Fin Time:       Wed Feb  4 11:01:00.852209 2009 (1233763260.852)
    OPTIONS / HTTP/1.1
Response Code:       200 (OK)
Request Length:      135
```

Jim Clausing   23

```
Reply Length:        197
Content Length:      0
Content Type  :      httpd/unix-directory
Time request sent:   Wed Feb  4 11:01:00.851211 2009 (1233763260.851)
Time reply started:  Wed Feb  4 11:01:00.851658 2009 (1233763260.852)
Time reply ACKed:    Wed Feb  4 11:01:00.851963 2009 (1233763260.852)
Elapsed time:  0 ms (request to first byte sent)
Elapsed time:  1 ms (request to content ACKed)
4.5.6.7:1082 ==> 4.3.2.192:80 (m2n)
  Server Syn Time:       Wed Feb  4 11:01:00.853494 2009 (1233763260.853)
  Client Syn Time:       Wed Feb  4 11:01:00.853458 2009 (1233763260.853)
  Server Fin Time:       Wed Feb  4 11:01:00.854522 2009 (1233763260.855)
  Client Fin Time:       Wed Feb  4 11:01:00.854958 2009 (1233763260.855)

IP traffic>>>
4.5.6.7 4.3.2.133 6 1030 18067 697 617 8 9 2009-02-04-10:50:28.7052 2009-02-04-11:01:24.6877
1 1
4.5.6.7 224.0.0.22 2 0 0 0 216 0 4 2009-02-04-10:50:35.6915 2009-02-04-11:01:32.8117 1 1
4.5.6.7 4.3.2.115 1 2048 0 0 296 0 4 2009-02-04-10:50:52.4601 2009-02-04-10:51:04.8164 1 1
4.5.6.7 4.3.2.53 1 2048 0 0 296 0 4 2009-02-04-10:50:57.3166 2009-02-04-10:51:09.8163 1 1
4.5.6.7 4.3.2.115 6 1037 445 640 675 6 6 2009-02-04-10:51:02.3173 2009-02-04-10:51:32.3232 1
2
4.5.6.7 4.3.2.115 6 1040 445 640 675 6 6 2009-02-04-10:51:32.3244 2009-02-04-10:52:02.3297 1
2
4.5.6.7 4.3.2.115 6 1041 139 0 62 0 1 2009-02-04-10:51:32.3244 2009-02-04-10:51:32.3244 1 1
4.5.6.7 4.3.2.115 6 1042 445 640 675 6 6 2009-02-04-10:52:02.3305 2009-02-04-10:52:32.3366 1
2
4.5.6.7 4.3.2.115 6 1044 445 640 675 6 6 2009-02-04-10:52:32.3373 2009-02-04-10:53:02.3437 1
2
4.5.6.7 4.3.2.115 6 1046 445 640 675 6 6 2009-02-04-10:53:02.3447 2009-02-04-10:53:32.3503 1
2
4.5.6.7 4.3.2.115 6 1047 139 0 62 0 1 2009-02-04-10:53:02.3447 2009-02-04-10:53:02.3447 1 1
4.5.6.7 4.3.2.115 6 1048 80 477 415 5 5 2009-02-04-10:53:32.4160 2009-02-04-10:53:32.4342 1
2
4.5.6.7 4.3.2.115 6 1049 80 807 449 5 5 2009-02-04-10:53:32.4357 2009-02-04-10:53:32.4373 1
2
4.5.6.7 4.3.2.156 1 2048 0 0 296 0 4 2009-02-04-10:53:32.5784 2009-02-04-10:53:44.8152 1 1
4.5.6.7 4.3.2.178 1 2048 0 0 296 0 4 2009-02-04-10:53:37.3152 2009-02-04-10:53:49.8152 1 1
4.5.6.7 4.3.2.156 6 1050 445 640 675 6 6 2009-02-04-10:53:42.3154 2009-02-04-10:54:12.3205 1
2
4.5.6.7 4.3.2.191 6 1052 80 741 389 5 5 2009-02-04-10:53:43.4732 2009-02-04-10:53:43.4833 1
2
4.5.6.7 4.3.2.156 6 1053 445 913 675 7 6 2009-02-04-10:54:12.3220 2009-02-04-10:54:42.3284 1
2
4.5.6.7 4.3.2.156 6 1054 139 0 62 0 1 2009-02-04-10:54:12.3221 2009-02-04-10:54:12.3221 1 1
4.5.6.7 4.3.2.156 6 1055 445 640 675 6 6 2009-02-04-10:54:42.3291 2009-02-04-10:55:12.3350 1
2
4.5.6.7 4.3.2.156 6 1056 139 0 62 0 1 2009-02-04-10:54:42.3293 2009-02-04-10:54:42.3293 1 1
4.5.6.7 4.3.2.156 6 1057 445 640 675 6 6 2009-02-04-10:55:12.3359 2009-02-04-10:55:42.3413 1
2
4.5.6.7 4.3.2.156 6 1058 139 0 62 0 1 2009-02-04-10:55:12.3360 2009-02-04-10:55:12.3360 1 1
4.5.6.7 4.3.2.156 6 1059 445 640 675 6 6 2009-02-04-10:55:42.3423 2009-02-04-10:56:12.3482 1
2
4.5.6.7 4.3.2.156 6 1061 80 477 415 5 5 2009-02-04-10:56:12.3534 2009-02-04-10:56:12.3549 1
2
4.5.6.7 4.3.2.156 6 1062 80 807 449 5 5 2009-02-04-10:56:12.3561 2009-02-04-10:56:12.3574 1
2
4.5.6.7 4.3.2.192 1 2048 0 0 296 0 4 2009-02-04-10:58:21.0009 2009-02-04-10:58:33.3133 1 1
4.5.6.7 4.3.2.60 1 2048 0 0 296 0 4 2009-02-04-10:58:25.8132 2009-02-04-10:58:38.3132 1 1
```

Jim Clausing   24

```
4.5.6.7 4.3.2.192 6 1069 445 640 675 6 6 2009-02-04-10:58:30.8134 2009-02-04-10:59:00.8190 1
2
4.5.6.7 4.3.2.192 6 1071 445 640 675 6 6 2009-02-04-10:59:00.8203 2009-02-04-10:59:30.8259 1
2
4.5.6.7 4.3.2.192 6 1072 139 0 62 0 1 2009-02-04-10:59:00.8205 2009-02-04-10:59:00.8205 1 1
4.5.6.7 4.3.2.192 6 1073 445 640 675 6 6 2009-02-04-10:59:30.8269 2009-02-04-11:00:00.8325 1
2
4.5.6.7 4.3.2.192 6 1074 139 0 62 0 1 2009-02-04-10:59:30.8269 2009-02-04-10:59:30.8269 1 1
4.5.6.7 4.3.2.192 6 1075 445 640 675 6 6 2009-02-04-11:00:00.8335 2009-02-04-11:00:30.8396 1
2
4.5.6.7 4.3.2.192 6 1077 445 640 675 6 6 2009-02-04-11:00:30.8408 2009-02-04-11:01:00.8464 1
2
4.5.6.7 4.3.2.192 6 1078 139 0 62 0 1 2009-02-04-11:00:30.8409 2009-02-04-11:00:30.8409 1 1
4.5.6.7 4.3.2.192 6 1081 80 477 415 5 5 2009-02-04-11:01:00.8507 2009-02-04-11:01:00.8522 1
2
4.5.6.7 4.3.2.192 6 1082 80 805 441 5 5 2009-02-04-11:01:00.8534 2009-02-04-11:01:00.8549 1
2

AIDE>>>

Start timestamp: 2009-02-04 11:07:09

Summary:
  Total number of files:   8688
  Added files:                    11
  Removed files:          0
  Changed files:          46


---------------------------------------------------
Added files:
---------------------------------------------------

added: /mnt/new/System Volume Information/_restore{786D3857-363B-401B-B0E3-
5098A38A2814}/RP2/A0000088.ini
added: /mnt/new/System Volume Information/_restore{786D3857-363B-401B-B0E3-
5098A38A2814}/RP2/A0000089.ini
added: /mnt/new/System Volume Information/_restore{786D3857-363B-401B-B0E3-
5098A38A2814}/RP2/A0000090.ini
added: /mnt/new/System Volume Information/_restore{786D3857-363B-401B-B0E3-
5098A38A2814}/RP2/A0000091.cfg
added: /mnt/new/System Volume Information/_restore{786D3857-363B-401B-B0E3-
5098A38A2814}/RP2/A0000092.ini
added: /mnt/new/System Volume Information/_restore{786D3857-363B-401B-B0E3-
5098A38A2814}/RP2/change.log.1
added: /mnt/new/WINDOWS/Prefetch/NETSTAT.EXE-2B2B4428.pf
added: /mnt/new/WINDOWS/system/spoolsv.exe
added: /mnt/new/WINDOWS/system32/Microsoft/Protect/S-1-5-18/User/7dff95ec-cfc7-45d3-9e6d-
5124c7266e73
added: /mnt/new/WINDOWS/system32/sandnet.exe

---------------------------------------------------
Changed files:
---------------------------------------------------

changed: /mnt/new/System Volume Information/_restore{786D3857-363B-401B-B0E3-
5098A38A2814}/drivetable.txt
changed: /mnt/new/System Volume Information/_restore{786D3857-363B-401B-B0E3-
5098A38A2814}/RP2/change.log
```

Jim Clausing   25

```
changed: /mnt/new/System Volume Information/_restore{786D3857-363B-401B-B0E3-
5098A38A2814}/_driver.cfg
changed: /mnt/new/WINDOWS/Debug/UserMode/userenv.log
changed: /mnt/new/WINDOWS/Prefetch/NTOSBOOT-B00DFAAD.pf
changed: /mnt/new/WINDOWS/SoftwareDistribution/DataStore/DataStore.edb
changed: /mnt/new/WINDOWS/SoftwareDistribution/DataStore/Logs/edb.chk
changed: /mnt/new/WINDOWS/SoftwareDistribution/DataStore/Logs/edb.log
changed: /mnt/new/WINDOWS/system32/Microsoft/Protect/S-1-5-18/User
changed: /mnt/new/WINDOWS/system32/Microsoft/Protect/S-1-5-18/User/Preferred
changed: /mnt/new/WINDOWS/system32/wbem/Logs/wbemess.log
changed: /mnt/new/WINDOWS/system32/wbem/Logs/wmiprov.log
changed: /mnt/new/WINDOWS/system32/wbem/Repository/FS/INDEX.BTR
changed: /mnt/new/WINDOWS/system32/wbem/Repository/FS/INDEX.MAP
changed: /mnt/new/WINDOWS/system32/wbem/Repository/FS/MAPPING1.MAP
changed: /mnt/new/WINDOWS/system32/wbem/Repository/FS/MAPPING2.MAP
changed: /mnt/new/WINDOWS/system32/wbem/Repository/FS/OBJECTS.DATA
changed: /mnt/new/WINDOWS/system32/wbem/Repository/FS/OBJECTS.MAP
changed: /mnt/new/WINDOWS/system32/config/AppEvent.Evt
changed: /mnt/new/WINDOWS/system32/config/default
changed: /mnt/new/WINDOWS/system32/config/default.LOG
changed: /mnt/new/WINDOWS/system32/config/SAM
changed: /mnt/new/WINDOWS/system32/config/SAM.LOG
changed: /mnt/new/WINDOWS/system32/config/SECURITY
changed: /mnt/new/WINDOWS/system32/config/SECURITY.LOG
changed: /mnt/new/WINDOWS/system32/config/software
changed: /mnt/new/WINDOWS/system32/config/software.LOG
changed: /mnt/new/WINDOWS/system32/config/SysEvent.Evt
changed: /mnt/new/WINDOWS/system32/config/system
changed: /mnt/new/WINDOWS/system32/config/system.LOG
changed: /mnt/new/WINDOWS/system32/wpa.dbl
changed: /mnt/new/WINDOWS/WindowsUpdate.log
changed: /mnt/new/Documents and Settings/LocalService/Local Settings/Application
Data/Microsoft/Windows/UsrClass.dat
changed: /mnt/new/Documents and Settings/LocalService/Local Settings/Application
Data/Microsoft/Windows/UsrClass.dat.LOG
changed: /mnt/new/Documents and Settings/LocalService/NTUSER.DAT
changed: /mnt/new/Documents and Settings/LocalService/ntuser.dat.LOG
changed: /mnt/new/Documents and Settings/NetworkService/Local Settings/Application
Data/Microsoft/Windows/UsrClass.dat
changed: /mnt/new/Documents and Settings/NetworkService/Local Settings/Application
Data/Microsoft/Windows/UsrClass.dat.LOG
changed: /mnt/new/Documents and Settings/NetworkService/NTUSER.DAT
changed: /mnt/new/Documents and Settings/NetworkService/ntuser.dat.LOG
changed: /mnt/new/Documents and Settings/User/Local Settings/Application Data/IconCache.db
changed: /mnt/new/Documents and Settings/User/Local Settings/Application
Data/Microsoft/Windows/UsrClass.dat
changed: /mnt/new/Documents and Settings/User/Local Settings/Application
Data/Microsoft/Windows/UsrClass.dat.LOG
changed: /mnt/new/Documents and Settings/User/NTUSER.DAT
changed: /mnt/new/Documents and Settings/User/NTUSER.DAT.LOG

------------------------------------------------

Interesting strings>>>

_0014.jpeg-www.imageshack.com
#      102.54.94.97     rhino.acme.com          # source server
127.0.0.1    avp.com
127.0.0.1    ca.com
```

Jim Clausing   26

```
127.0.0.1     customer.symantec.com
127.0.0.1     dispatch.mcafee.com
127.0.0.1     download.mcafee.com
127.0.0.1     f-secure.com
127.0.0.1     kaspersky.com
127.0.0.1     kaspersky-labs.com
127.0.0.1     liveupdate.symantec.com
127.0.0.1     liveupdate.symantecliveupdate.com
127.0.0.1     mast.mcafee.com
127.0.0.1     mcafee.com
127.0.0.1     my-etrust.com
127.0.0.1     nai.com
127.0.0.1     networkassociates.com
127.0.0.1     rads.mcafee.com
127.0.0.1     scanner.novirusthanks.org
127.0.0.1     secure.nai.com
127.0.0.1     securityresponse.symantec.com
127.0.0.1     sophos.com
127.0.0.1     symantec.com
127.0.0.1     threatexpert.com
127.0.0.1     trendmicro.com
127.0.0.1     updates.symantec.com
127.0.0.1     update.symantec.com
127.0.0.1     us.mcafee.com
127.0.0.1     virscan.org
127.0.0.1     viruslist.com
127.0.0.1     virusscan.jotti.org
127.0.0.1     virustotal.com
127.0.0.1     www.avp.com
127.0.0.1     www.ca.com
127.0.0.1     www.f-secure.com
127.0.0.1     www.grisoft.com
127.0.0.1     www.kaspersky.com
127.0.0.1     www.mcafee.com
127.0.0.1     www.my-etrust.com
127.0.0.1     www.nai.com
127.0.0.1     www.networkassociates.com
127.0.0.1     www.scanner.novirusthanks.org
127.0.0.1     www.sophos.com
127.0.0.1     www.symantec.com
127.0.0.1     www.trendmicro.com
127.0.0.1     www.virscan.org
127.0.0.1     www.viruslist.com
127.0.0.1     www.virusscan.jotti.org
127.0.0.1     www.virustotal.com
32.dll
#      38.25.63.10     x.acme.com              # x client host
4:v15:v$C:v
accwiz.exe
aC:\WINDOWS\System32\rasctrs.dll
advapi32.dll
ADVAPI32.dll
ADVAPI32.DLL
advpack.dll
ADVPACK.DLL
aim.stop
ALLUSERSPROFILE=C:\Documents and Settings\All Users
alrpc:[DNSResolver,Security=Impersonation Dynamic False]
alrpc:[ntsvcs]
```

Jim Clausing   27

```
APPDATA=C:\Documents and Settings\User\Application Data
appHelp.dll
ATL.DLL
AUTHZ.dll
Autoloader.exe
AUTPRX32.DLL
BidLab.dll
browseui.dll
BypassFtpTimeCheck
BypassHTTPNoCacheCheck
BypassSSLNoCacheCheck
Can not run Unicode version of ATL.DLL on Windows 95.
c:\autoexec.bat
=C:=C:\Documents and Settings\User
{cc.republicofskorea.info
cc.republicofskorea.info
<CDispenser::GetActiveConnection|POOL|ERR> FreeResource failed, %p{.}, %p{IHolder*}, lpDbc:
%p{LPDBC}, 0x%08X{HRESULT}
<CDispenser::TryAllocResource|POOL|ERR> FreeResource failed, %p{.}, %p{IHolder*}, *ppDbc:
%p{LPDBC}, 0x%08X{HRESULT}
<CDispenser::TryAllocResource|POOL|RET> %p{.}, lpIDbc: %p{LPIDBC}, ppDbc*: %p{LPDBC}
C:\DOCUME~1\User\LOCALS~1\Temp\Perflib_Perfdata_c4.dat
C:\DOCUME~1\User\LOCALS~1\Temp\SQL.LOG
C:\Documents and Settings\All Users\Application Data
C:\Documents and Settings\User\Application Data\Micros
C:\Documents and Settings\User\Cookies
C:\Documents and Settings\User\Cookies\
C:\Documents and Settings\User\Cookies\index.dat
C:\Documents and Settings\User\Local Settings\History
C:\Documents and Settings\User\Local Settings\History\History.IE5\
C:\Documents and Settings\User\Local Settings\History\History.IE5\index.dat
C:\Documents and Settings\User\Local Settings\History\History.IE5\MSHist012008010920080110\
C:\Documents and Settings\User\Local Settings\Temporary Internet Files
C:\Documents and Settings\User\Local Settings\Temporary Internet Files\Content.IE5\
C:\Documents and Settings\User\Local Settings\Temporary Internet Files\Content.IE5\index.dat
cfgmgr32.dll
Choose OK to attempt to use the GT or Cancel to abort.HFailed to enlist in DTC: SQL state
%s, native error %d, error message %s
CLocator::GetWbemLocator: Load of ole32.dll failed
cmd.exe
```

Jim Clausing   28

10.   Appendix B

Jim Clausing   29