GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Advanced Incident Response, Threat Hunting, and Digital Forensics (Forensics
at http://www.giac.org/registration/gcfa

Analysis of a Software Write Blocker - That Works?

GCFA practical Assignment v1.2

Suzanne Chevalier

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Contents List

© SANS Institute 2003, Author retains full rights.

Summary

This document contains all three sections of the GCFA Practical Assignment version 1.2.. Part 1 is the analysis of an unknown file. Part 2 is the evaluation of a propriety software program that claims to write block media attached to a Windows 2000 system. Part 3 is a description of how a system administrator at an ISP handled requests from a law enforcement officer and how specific user account information could legally be provided to that officer.

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Part 1. Analyze an Unknown Binary

This section contains the analysis of an unknown program that was seized from a computer. Unfortunately, the details of the computer and the computers' purpose were not provided.

The test environment for the analysis of the program consisted of four systems.

- IBM 600E Thinkpad with Pentium II
  OS: Linux
  Distribution Version: Red hat Linux release 7.1 (seawolf)
  OS Version: #1 Sun Apr 8 20:41:30 EDT 2001
  OS System Release: 2.4.2-2
  Processor Type: i686

- IBM T23 Thinkpad with Pentium III
  OS: Microsoft Windows 2000 Professional
  Version: 5.0.2195 Service Pack 2 Build 2195
  Processor: x86 Family 6 Model 11 Stepping 1 Genuine Intel

- Dell Precision 450 with XEON
  OS: Linux
  Distrubution Version: Red Hat Linux version 8.0 (Psyche)
  OS System Release: 2.4.18-14

- Dell Precision 450 with XEON
  OS: Linux
  Distrubution Version: Red Hat Linux version 7.2
  OS System Release: Undetermined
  250 Mhz Pentium II processor

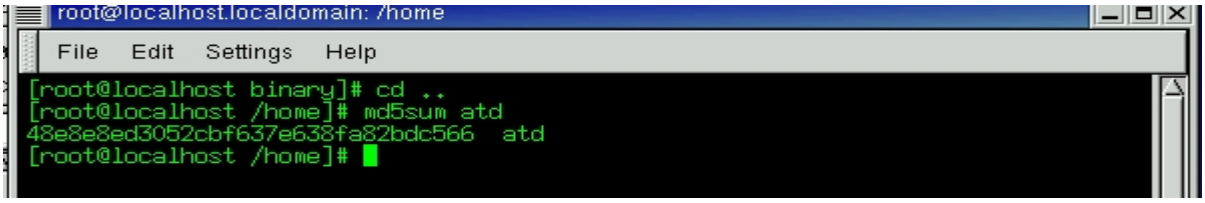
Binary Details

The seized program was delivered on a compact disc (CD). The CD contained the file sn.zip. A copy of the CD was made using the program IBM RecordNow[1] version 3.5 on the IBM laptop and the copy was used for analysis. The following md5sum was created using the Linux machine and the command # md5sum /mnt/cdrom/sn.zip: 5fea57f2a1546bc391c6b1bbfc452.

The file sn.zip copied to the home directory was double-clicked and unzipped. Two files, atd and atd.md5, were saved to the /home/ directory. The following md5sum was created using the command #md5sum /home/atd: 48e8e8ed3052cbf637e638fa82bdc566. This matched the hash located in the atd.md5 file.

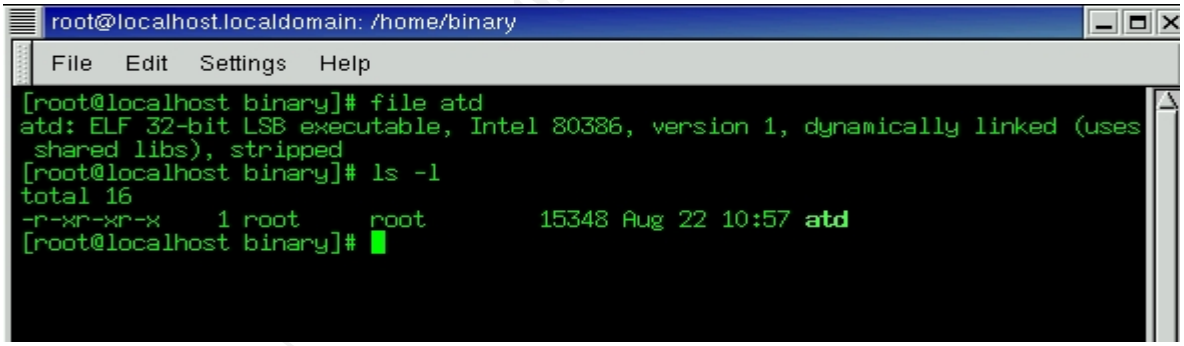
[1] Information on IBM RecordNow may be located at http://www.pc.ibm.com/support?page=6843CTO

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.



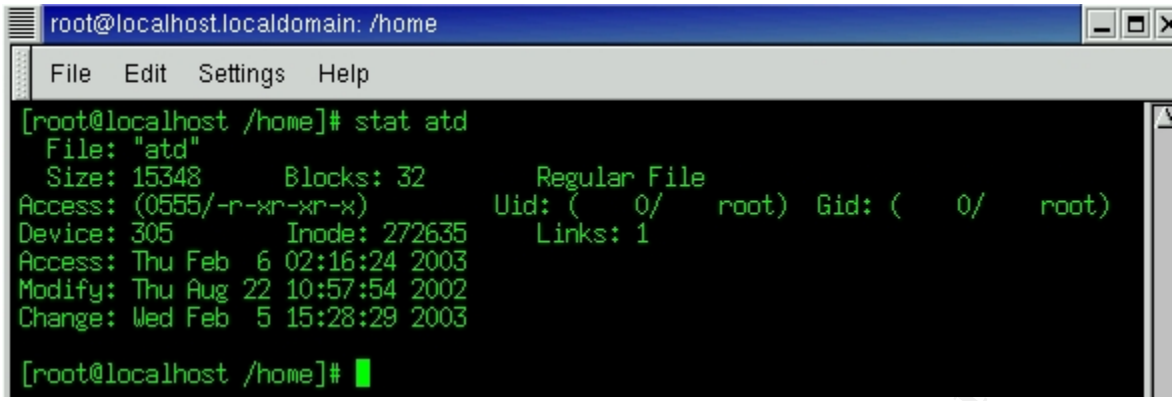
The atd file was copied into the /binary/ directory. The file command determines file type and shows information about the specified file – is it an executable, a binary, text, etc.. The file command run against the atd file showed that the file is dynamically linked (uses shared libs) and is an executable.

The ls command lists the contents of a directory and the -l parameter is a long listing format that displays the file permissions, number of links to the file, user and group information, file size in bytes, the last changed date and time, and the filename. ls -l was run to show that the file size is 15348 bytes and that both the owner and group user is root. Since the system from which the file was seized is not available there is no way to tell if it was built on a corporate system or a personal system. Had there been evidence to indicate the file was built on a corporate system then the user who created the file would had to have root access either legitimately, perhaps as a system administrator, or illegitimately as an attacker.

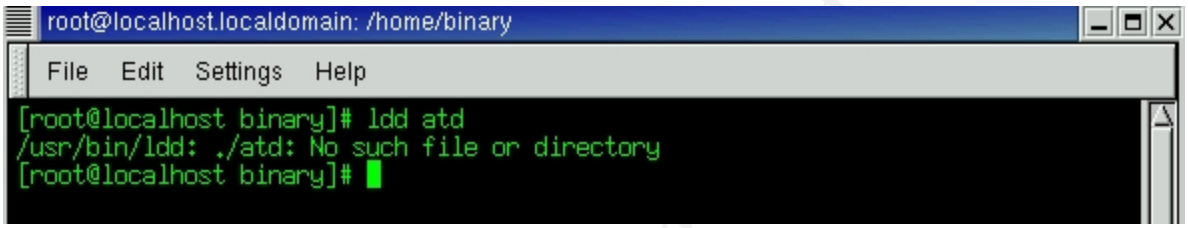


The stat command was run to determine the file's last modified, accessed, and changed (MAC) times. The modified time for this file is the only accurate depiction of when this file may have been used last.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

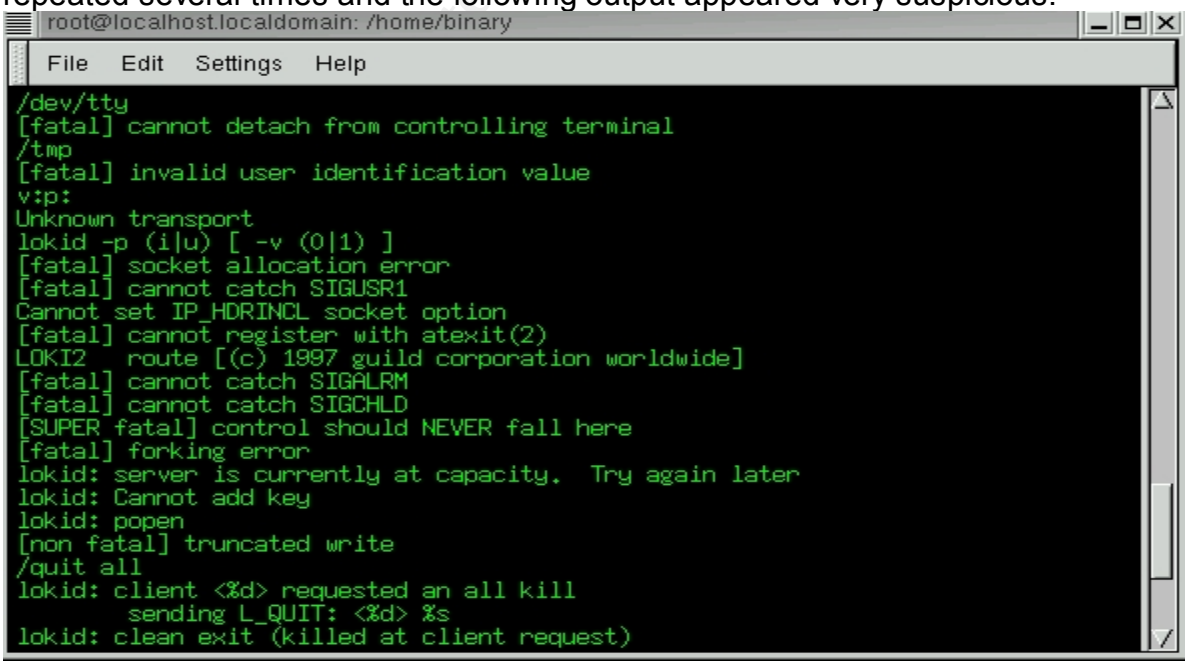


The ldd command that displays the shared library files of the specified file was used next, but did not assist with analysis.



The strings -a command was used next and output was sent to a file for easier review. It appears that the compiler is GCC GNU version 2.7.2.1. The strings command looks for printable strings with four or more characters in binary or object files. The -a parameter searches the entire file instead of just the initial data area of the file.

The strings output file was examined for unusual entries. LOKI was a term that was repeated several times and the following output appeared very suspicious.



© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Program Description and Identification

Searching the web for LOKI and LOKI2 produced plenty of information that indicated an ICMP echo-reply hack program called LOKI2 could be used maliciously. It appears that LOKI2 is the program within the atd file.

An article in Phrack magazine states "LOKI2 is an information-tunneling program. It is a proof of concept work intending to draw attention to the insecurity that is present in many network protocols."[2] Another article from iss.net states

> Loki is a covert-channel client/server program published in the online publication Phrack. This program is a working proof-of-concept to demonstrate that data can be transmitted somewhat surreptitiously across a network by hiding it in traffic that normally does not contain payloads. The example code can tunnel the equivalent of a Unix RCMD/RSH session in either ICMP echo request (ping) packets or UDP traffic to the DNS port. This is used as a back door into a Unix system after root access has been compromised. Presence of LOKI on a system is evidence that the system has been compromised in the past.[3]

LOKI2 allows covert data to travel within the data portion of an ICMP echo or echo-reply packet[4]. Since security policies in general do not block ICMP echo traffic (eg. the ping[5] command is used to see if a node on a network is alive) it passes through firewalls and packet-sniffers without problem. And therefore can contain data that could be used to execute commands on a system and leave open backdoors. LOKI2 also allows the use of encryption. Therefore, if a packet-sniffer would actually pick up a packet the contents would simply be garbled text.

LOKI2 contains two parts, a client that is used on the attacker machine and the server or listener that is placed on the compromised system. The attacker would have to obtain root access on the compromised system in order to compile LOKI2 on it. Once completed, the LOKI2 client is run on the attacker system and the compromised system is available to the attacker.

The source code of LOKI2 was located at the Phrack website[6] in Phrack 51. The article was downloaded. The extract.c file was compiled as per the directions in the article (page 17).

---

2 (Phrack Magazine) URL http://www.phrack.com/phrack/51/P51-06

3 (Internet Security Systems™) URL http://www.iss.net/security_center/advice/Intrusions/2000112/default.htm

4 (Phrack Magazine) URL http://www.phrack.org/show.php?p=49&a=6

5 PING is a reference to SONAR use as with submarines. A signal is sent through the water and if it hits another submarine a sound (or ping) is bounced pack to the sender. It is such with the PING command. The PING command sends a packet to a particlar node and if that node is alive it sends back a reply.

6 (Phrack Magazine) URL http://www.phrack.org/show.php?p=51&a=6

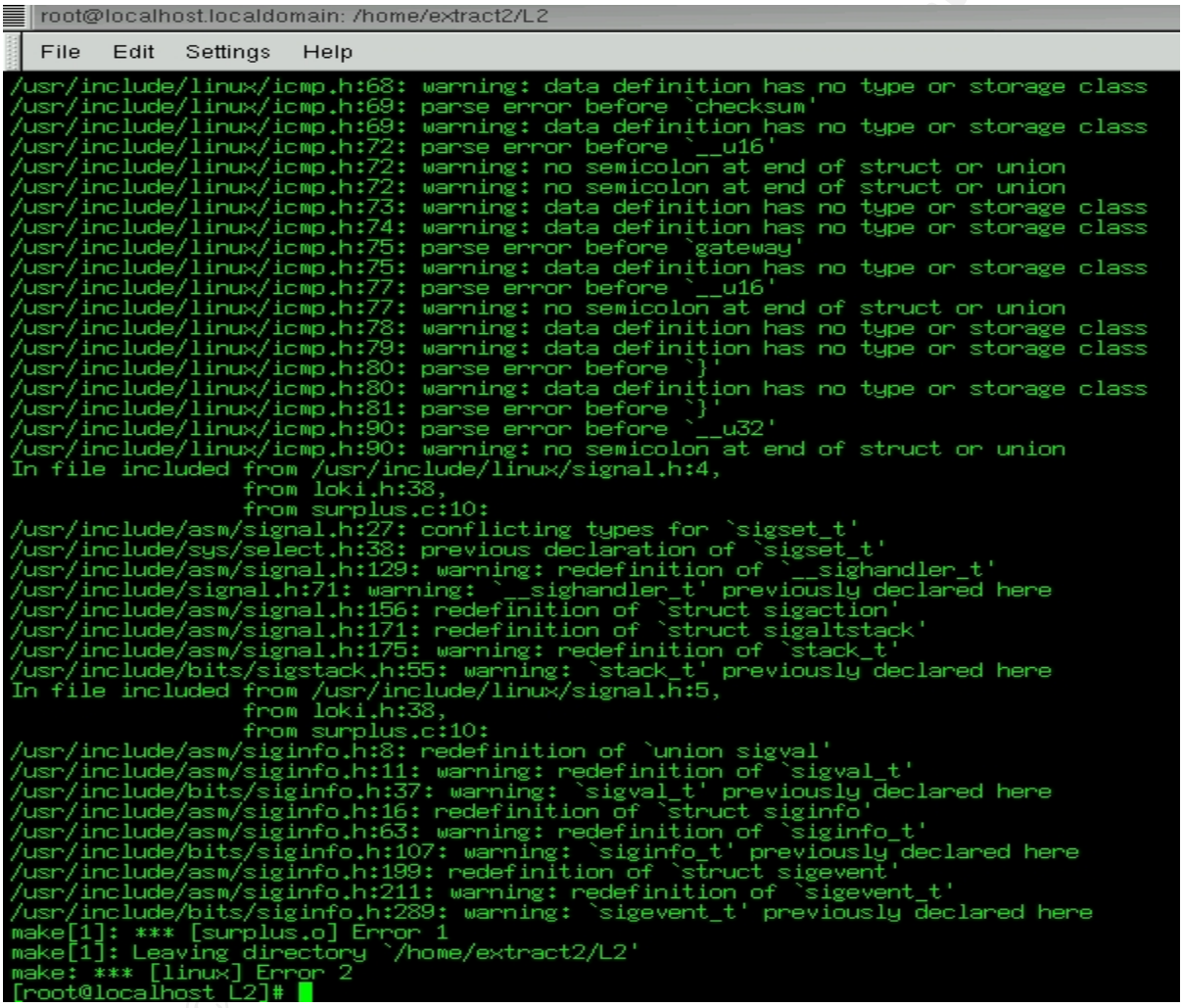
© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

gcc –o extract extract.c

The extract command was then used to remove the source code from the article. The output was automatically placed in the directory L2. Following the instructions in the article, the make command was executed.

#make linux

Errors were received.



The errors received show that there are problems with icmp.h and signal.h among others. Both icmp.h and signal.h are found in the loki.h file (as indicated in the errors displayed above) as #includes for the Linux build. One or both of these may need to be altered or removed in order for the source to compile properly. The surplus.c file has a #include for the loki.h file explaining the 'from surplus.c' in the errors above. Attempts were made to alter the loki.h file to no avail and a successful compile was still unattainable. Therefore, in order to prove that the atd file is LOKI2, I compare the strings output of the atd file to the source code of LOKI2. The following lines of code are found in the LOKI2 source code:

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

```
“fprintf(stderr, "DEBUG: stat_client nono\n");
        return (NOK);
    }
    n = sprintf(buf, "\nlokid version:\t\t%s\n", VERSION);
    n += sprintf(&buf[n], "remote interface:\t%s\n", host_lookup(rdg.iph.ip_dst));
    proto = getprotobynumber(prot);
    n += sprintf(&buf[n], "active transport:\t%s\n", proto -> p_name);
    n += sprintf(&buf[n], "active cryptography:\t%s\n", CRYPTO_TYPE);
    time(&now);
    n += sprintf(&buf[n], "server uptime:\t\t%.02f minutes\n", difftime(now, uptime) / 0x3c);
    locks();
    n += sprintf(&buf[n], "client ID:\t\t%d\n",      client[entry].client_id);
    n += sprintf(&buf[n], "packets written:\t%ld\n", client[entry].packets_sent);
    n += sprintf(&buf[n], "bytes written:\t\t%ld\n", client[entry].bytes_sent);
    n += sprintf(&buf[n], "requests:\t\t%d\n",       client[entry].hits);
    ulocks();”[7]
```

The following was located in the strings output of the atd file:

```
lokid: Client database full
DEBUG: stat_client nono
lokid version:          %s
remote interface:       %s
active transport:       %s
active cryptography:    %s
server uptime:          %.02f minutes
client ID:              %d
packets written:        %ld
bytes written           %ld
requests:               %d
```

The strings output of the atd file follows the source code as shown above and continues to follow it throughout the file. Through this comparison I believe the atd file to be the listener portion of LOKI2 or lokid. Actually compiling the source code and comparing the resulting files with the atd file would be preferred, however.

So far building LOKI2 has been unsuccessful and not knowing exactly how to alter the loki.h file makes things more difficult. The instructions state that Linux 2.0.x is supported. That is just slightly older than the versions that I am using. Therefore, I search the web again to attempt to locate a LOKI2 version compatible with my system. Unfortunately, I was not successful.

I tried the make command with OpenBSD and FreeBSD and received errors on both. I was unable to compile LOKI2 and therefore, unable to obtain an md5sum and compare with the seized atd file. Even if I could alter the file and have a successful build it would

---

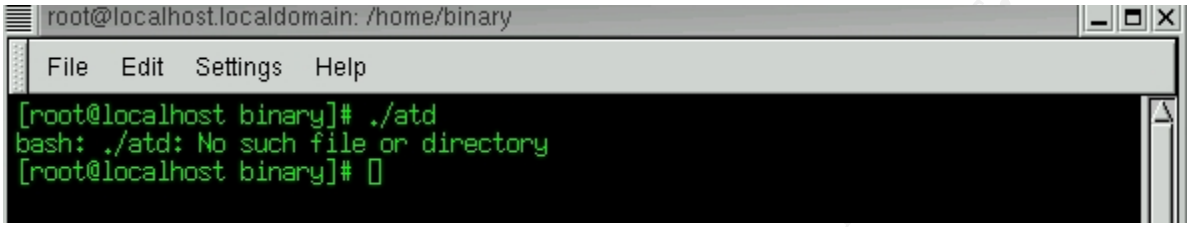
[7] (Phrack Magazine) URL http://www.phrack.org/show.php?p=51&a=6

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

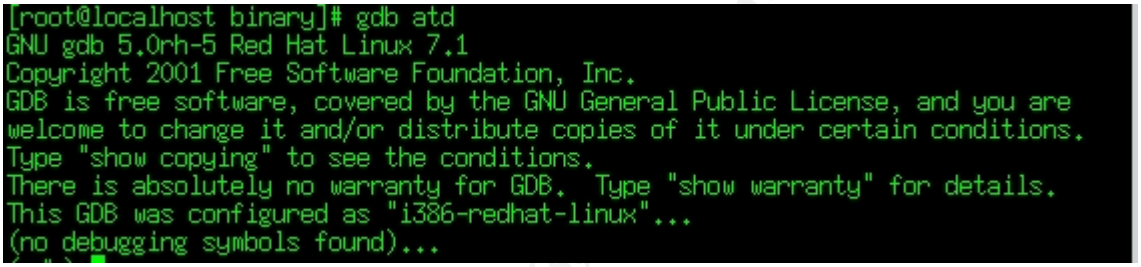
have to be the exact alterations made for use in the atd file in order to receive a matching md5sum - highly improbable.

Forensic Details

Having the seized atd file run on my test system would assist in obtaining forensic details of how the atd file affects the system on which it's run and what, if any, fingerprints are left behind. Since the Dell Precision 450 is a test machine I copy the atd file to it and try to run the program with the command ./atd.



That didn't work so I run the file through the debugger gdb.



Gdb tells me that no debugging symbols are found. I try gdb obscure, and I receive the output: "/home/binary/atd is not a core dump: File format not recognized.

The objdump command displays information from object files, so I run objdump –x to display the contents of all headers in the atd file.

Finally, output is received. The output of objdump indicates in the Dynamic Section that libc.so.5 is needed. I searched my IBM Thinkpad for libc.so.*. No libc.so.5 files were located, but libc.so.6 files were found in the /lib/ and /lib/i686 directories. That could be one reason why the file doesn't execute. It appears that there is nothing abnormal about the entry point 0x804, but I run the readelf –a command to see the entry point address more clearly. The readelf command displays elf-formatted files.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

```
[root@localhost binary]# readelf -a atd | more
ELF Header:
  Magic:   7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF32
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           Intel 80386
  Version:                           0x1
  Entry point address:               0x8048db0
  Start of program headers:          52 (bytes into file)
  Start of section headers:          14508 (bytes into file)
  Flags:                             0x0
  Size of this header:               52 (bytes)
  Size of program headers:           32 (bytes)
  Number of program headers:         5
  Size of section headers:           40 (bytes)
  Number of section headers:         21
  Section header string table index: 20
```

The entry point address appears normal.

A search of the Net for libc.so.5 sent me to the RedHat website. I downloaded and installed what I thought were the necessary library files in order to have older programs run on my version of Linux and tried the ./atd command again. I still received an error. Another search of the Net for libc.so.5 sent me to the website http://rpmfind.net/linux/rpm2html/search.php?query=libc.so.5. I then did a search for libc.so.5 with the system RedHat and the search produced results for RedHat 6.2, 5.2, and 5.1. Unfortunately, nothing was available for my Linux version. I continued to search and attempted several other downloads without success.

If I would be able to run the atd executable the netstat command may be of assistance in showing the network status and providing some additional forensic information. Hence, I attempt to run the program again. Typing . /atd provides me with the error 'bash: ./atd: No such file or directory. So, atd alone was typed at the command prompt. Viola! No error message. Now I am cautiously optimistic. I run the ps -aux command to show what processes are running along with the grep command to show only those processes with atd in them.

```
[root@localhost /home]# ps -aux | grep atd
rpcuser    429  0.0  0.0  1564    0 ?        SW   18:33   0:00 rpc.statd
daemon     574  0.0  0.0  1400    0 ?        SW   18:34   0:00 /usr/sbin/atd
root      6842  0.0  1.9  1620  592 pts/1    S    23:45   0:00 grep atd
[root@localhost /home]# []
```

It appears that atd is actually running, however it is an atd file found in the /usr/sbin directory and therefore a normal system file. This explains why the lokid program was renamed to atd.

Had the seized atd file actually been running, netstat -na might have been beneficial as it displays the network connections, the -n shows numerical addresses and the -a parameter shows all listening and non listening sockets. Of course, since I believe that the atd file is the listener portion of LOKI2 the netstat command might still look benign as shown below when run normally.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

```
[root@localhost /home]# netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:1024            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:6000            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
udp        0      0 0.0.0.0:1024            0.0.0.0:*
udp        0      0 0.0.0.0:605             0.0.0.0:*
udp        0      0 0.0.0.0:111             0.0.0.0:*
```

If the seized atd file were running the command strace would be used to provide additional details. Strace is a command that can capture calls made to the system. The -f parameter watches child processes. Strace can be configured to only look at particular types of system calls as well.

As an example, below is a partial screen shot of the strace –f output on the /usr/sbin/atd file.

```
[root@localhost binary]# strace -f atd
execve("/usr/sbin/atd", ["atd"], [/* 40 vars */]) = 0
uname({sys="Linux", node="localhost.localdomain", ...}) = 0
brk(0)                                  = 0x804c61c
old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x40017000
open("/etc/ld.so.preload", O_RDONLY)    = -1 ENOENT (No such file or directory)
open("/etc/ld.so.cache", O_RDONLY)      = 3
fstat64(3, {st_mode=S_IFREG|0644, st_size=58480, ...}) = 0
old_mmap(NULL, 58480, PROT_READ, MAP_PRIVATE, 3, 0) = 0x40018000
close(3)                                = 0
open("/lib/i686/libc.so.6", O_RDONLY)   = 3
read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0\200\302"..., 1024) = 1024
fstat64(3, {st_mode=S_IFREG|0755, st_size=5634864, ...}) = 0
old_mmap(NULL, 1242920, PROT_READ|PROT_EXEC, MAP_PRIVATE, 3, 0) = 0x40027000
mprotect(0x4014d000, 38696, PROT_NONE)  = 0
old_mmap(0x4014d000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED, 3, 0x125000) = 0x4014d000
old_mmap(0x40153000, 14120, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x40153000
close(3)                                = 0
munmap(0x40018000, 58480)               = 0
getpid()                                = 948
brk(0)                                  = 0x804c61c
brk(0x804ca34)                          = 0x804ca34
brk(0x804d000)                          = 0x804d000
socket(PF_UNIX, SOCK_STREAM, 0)         = 3
connect(3, {sin_family=AF_UNIX, path="
ocket"}, 110) = -1 ENOENT (No such file or directory)
close(3)                                = 0
open("/etc/nsswitch.conf", O_RDONLY)    = 3
fstat64(3, {st_mode=S_IFREG|0644, st_size=1750, ...}) = 0
mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x40018000
read(3, "#\n# /etc/nsswitch.conf\n#\n# An ex"..., 4096) = 1750
brk(0x804e000)                          = 0x804e000
read(3, "", 4096)                       = 0
close(3)                                = 0
```

Strace was also run on the atd process with the command strace –p 599 -e trace=read, write. The strace just runs and does not provide any output. Of course, nothing is being typed for strace to capture.

The strace output of the atd file does provide some useful information. The following is a brief explanation of what the strace output displayed.

- Executes the atd file[8] (execve)

[8] For a very good explanation of the argument output in strace visit http://www.informit.com/isapi/product_id~{DE6C301F-A7D0-4214-A06D-598313519AA3}/st~{BAC7BB78-22CD-4E1E-9387-19EEB5B71759}/content/index.asp

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

- Displays the operating system and the hostname
- Loads libc.so.6 (this atd file obviously runs with this c library as opposed to libc.so.5)
- Runs getpid to get the process ID
- Opens /etc/nsswitch.conf - Nsswitch.conf the Name Service Switch is used to control from where a machine gets data files.
- Control allowable accesses to memory (mprotect)
- Opens /etc/passwd as read only
- Opens /etc/group as read only
- Gets the user id of the current user (geteuid)
- Gets the group identity (getegid)
- Set real or effective group id (setregid)
- Set real or effective user id (setreuid)
- Changes directories to /var/spool/at

This provides some interesting information regarding the legitimate atd file found in the /usr/sbin directory, it is possible that lokid was renamed to atd because of the file similarities. If so, lokid would be accessing some sensitive system files.

At this point I know that LOKI2 can be used to send information via ICMP echo or echo-reply traffic after a back door is created on a system. However, I cannot determine exactly what an executed loki or lokid file does and cannot run the atd file on my system. The believe the atd file to be the listener of LOKI2, but without being able to actually run the seized atd file I am unable to prove that without doubt. It is also undeterminable what files are affected or accessed on the system when the program is run.

From the LOKI2 source code it is determinable that once the object files are created on the attacked system a command to 'clean' or remove them can be executed. Once that command was run and the lokid was renamed to atd I am uncertain if there would be any other fingerprints left behind.

The following are the facts regarding the seized atd file:

1) The file command indicates that the file is an ELF 32-bit executable, dynamically linked and stripped. The objdump command indicates that libc.so.5 is a shared library needed by the atd file in order to execute.
2) The file size is 15348 bytes.
3) Both the owner and group user of the file is root.
4) The file was last modified on Thursday, August 22, 2002 at 10:57:54. This could be when the file was executed, but I was unable to confirm whether or not it was executed.
5) The file consists of an ICMP echo-reply tunneling program called LOKI2 and more specifically the listener or lokid portion of LOKI2.
6) The file entry point appears normal.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Legal Implications

Unfortunately, I was unable to prove that this program was executed on the system because I could not perform analysis on the system from which the file was acquired. I was unable to find evidence within the file itself that would indicate it was executed.

Despite not having proof of file execution the existence of this file on a system could be and indication that the system was hacked or construed as malicious intent if an employee placed the file on the system. If an employee placed the file on the system it would most likely be a violation of company policy. Of course, it could have simply been there for research purposes as well or perhaps to indicate to management that ICMP traffic should be disallowed in order to prevent tunneling traffic to occur via ICMP.

Violating company policy may be the least of concern if the program was used to transfer company secrets. Then there would be a case to remove the employee and possibly bring up charges depending on what was compromised.

The file could have been used to bypass firewalls or other security measures on another network and therefore could have been used to gain unauthorized entry to another system. This could create legal problems for the company. It is possible that this program was used to gain access to a protected computer[9]. If that had occurred it could be prosecuted under the Computer Fraud and Abuse act and then possibly cause great damage to the reputation of the firm and possible fines and prison sentences (up to 20 years depending on the severity of the offense[10]) for those directly responsible.

If someone that hacked into the system placed the file there then a full investigation would need to be launched to determine what, if anything, was compromised.

Interview Questions

The interview is a sensitive process and what may work for one person in one interview may not work for the same person in any other interview. Nor is there a clear right way to conduct an interview. Each interview is unpredictable but the desired outcome is to acquire as much accurate information as possible, as quickly as possible and without causing any additional problems.

The following is a list of questions that may be asked by an investigator, system administrator, or manager to determine if a suspect actually used the malicious file on a system.

The supervisor of the interviewee was asking the questions and was not acting as a conduit for law enforcement. The interviewee had access to the system at the time the file was last modified and is therefore a suspect. The supervisor was attempting to befriend the interviewee and indicated that this issue would hopefully be resolved before upper management and possibly even law enforcement would have to be notified. This

---

9 (Federal Computer Fraud and Abuse Act). URL http://www4.law.cornell.edu/uscode/18/1030.html

10 ibid. Section (4)(c).

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

information could lead the interviewee to believe (if he/she knows anything) that this is the only opportunity to provide the requested information before things move to the next level.

It has been brought to my attention that a hacker type program was found on a system in this lab. I want to find out everything I can about it before I have to brief upper management. It's possible that it was used by one of our employees. Now, I don't understand anything about the program itself, but it is on this system that you use. Can you show me what it is on another system? Is it safe to look at on another system?

Other than maliciously are there other reasons why this program would be on your system? It may be nothing, but you had access to the system the last time the file was used, do you have any explanation for that?

Would you be able to explain how this type of program works and why it would be used to upper management?

If the program was placed on the system for legitimate reasons they there shouldn't be any problem. Can you explain why it could have been put there? What reasons are there to have this type of program on a system?
Are there any other files or programs on the system that could be construed as malicious?

I understand that of all the people working in this lab you have the most knowledge of hacker tools and techniques. Can you explain why this program would be used on this particular system? Is there something special about this system that would make it more vulnerable?

I have to report to upper management on this issue. Is there anything else I need to know before I go to them? They may want a full investigation if I don't provide them the right answers.

The interviewer should not be accusatory and yet allow the interviewee to believe that he/she knows more than what is actually being asked. If the interview is at all successful, the interviewee will understand that this is the best time to provide as much information as possible and will cooperate now rather than later.

Additional Information

Additional information can be found at the following links:

www.hackingexposed.com/tools/tools.html provides a listing of tools described in the book Hacking Exposed.

Linux.oreillynet.com/pub/a/linux/2001/12/14/rootkit.html provides rootkit information.

www.phrack.org

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

http://www.nagios.org/ "Nagios® is a host and service monitor designed to inform you of network problems before your clients, end-users or managers do".

http://www.rmlibrary.com/sites/legcompu.htm

http://www.mossbyrett.of.no/info/legal.html

http://www.cybercrime.gov/cccases.html Current and former computer fraud and intrusion cases.

© SANS Institute 2003, Author retains full rights.

Part 2 – Forensic Tool Verification

ABSTRACT

For the purposes of the test procedure described below, a recently acquired unknown file named sn.dat will be used in a few of the tests. The contents of the file are of no importance as will be shown. The original sn.dat file was not used for the test. Instead, an md5 hash of the original file was obtained and both the file itself and the sum results were placed in a safe in a secured area. Copies and images of the file were created for use in the testing and the md5 hash was printed for use in before/after test comparison.

I. Scope

Write Blocker for Windows 2000 is a software application that claims to enable a forensics analyst to perform examinations on evidentiary media without permitting any changes to that media.

The purpose of this test is to determine whether or not the software write blocker prevents all write attempts to the medium on which the sn.dat file resides. For example, if the sn.dat file is on a floppy disk without the manual read-only tab set on the floppy, will the write blocker software prevent any changes to the file sn.dat and block any other possible change to the floppy? Tests must be performed on multiple media types such as CD-RW, hard drives, Magneto Optical (MO) disks, Jazz disks, Zip disks, and remote drives such as those shared by a network file server.

II. Tool Description

Write Blocker for Windows 2000 version 4.10 is a proprietary, non-distributed software tool that is designed to prevent all intentional, unintentional, and system-initiated write operations to selected computer media. *Write Blocker for Windows 2000* can be configured to block all write attempts on any local computer storage media - including hard disk drives, floppy disks, and removable media - except for the system boot disk. *Write Blocker for Windows 2000* also claims to prevent write operations to connected network storage devices as well. The user interface allows a user to manually enable and disable write-blocking of any local or connected network storage device except the system boot volume.

Write Blocker for Windows 2000 could be of great use to a forensics examiner because it adds a new dimension to preserving evidence. If the tool were successful in its claims it would provide an investigator with the ability to ensure that no changes occur to original evidence while obtaining logical copies of media. For example, a search warrant is issued for Joe Schmo, who shares a computer with Alda Schmo; and therefore, Aldas' files cannot be collected as part of the search warrant. As a result, the computer itself cannot be confiscated and only files in Joe Schmo's "My Documents" folder are deemed as collectible as described in the search warrant. Using a Windows 2000 laptop with *Write Blocker for Windows 2000* installed the investigator would be able to attach to the suspect system and copy the desired files without worry that the

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

files would be changed or altered in any way. In addition, the tool provides the investigator with the ability to analyze multiple media types from one system. Once installed, *Write Blocker for Windows 2000* can be configured to automatically block all write attempts to all attached storage media and network storage devices, thus eliminating the need to install and configure hardware write-blockers for different media types and systems.

The tool is installed from CD and runs on systems with Windows 2000 Professional or Server Edition installed. The MSDIOM.DLL (provided with the software) proprietary system file is accessed when run and the tool is compiled statically as told to me by the author and verified by running the Linux ldd command against the .exe file with the result of 'not a dynamic executable'.

Write Blocker for Windows 2000 is a Boot driver that is loaded in the System Bus Extender driver group to start before any fixed or removable disk drivers regardless of the bus they're on (eg. Fixed SCSI/IDE disks, removable USB/FIREWIRE/SCSI disks, etc.).

Interrupt 13[11] which can be blocked to prevent write attempts in DOS based programs such as Windows95, is not applicable to *Write Blocker for Windows 2000* as Windows 2000 is not a DOS based system. Interrupt 13's equivalent within Windows 2000 is one or more commands destined for a physical device driver or entry point. I/O Request Packets (IRPs) sent from the OS are passed from driver to driver until the appropriate one is reached to perform the request. *Write Blocker for Windows 2000* is resident between the OS and the driver and prevents the driver from acknowledging/performing the command if the command would alter the associated write-blocked storage media.

The tool is not available for download as it is proprietary. The source code is unavailable as well (typical for most Windows-based computer forensics applications such as Guidance Software's EnCase and AccessData's Forensic ToolKit).

III. Test Apparatus

Write Blocker for Windows 2000 is designed to run on the Windows 2000 platform. Therefore, all tests were performed using a system drive configured with the following:

OS: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Various Windows 2000 System Disks were used during testing, as were several computers. Descriptions of all of the computers and media used during testing are found within each test procedure in Section VI.

The tests were completed in a controlled-access (cipher lock) lab environment; and, during the testing, I was the only one present. The network used for the network test is

[11] Interrupt 13 is the software interrupt for disk I/O used by DOS.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

a simple local area network with a Microsoft NT file server and no Internet connectivity. Twenty computers are attached to the network, but only two computers were used in the network testing - one Dell Dimension 4500 series and one Dell OptiPlex 150's. The remaining computers where not running at the time of the testing. There are several network shares with various levels of access control. Since the computers used during these tests have been successfully used over the course of several months prior to this particular testing, it is assumed that all computers and components are working properly and that no errors will be caused as a result of faulty equipment.

Outside forces that could affect the test results include the computer hardware itself. Because *Write Blocker for Windows 2000* is a software-only solution, many things can happen between the time the "on" button on the system is pressed and when the write blocker itself loads and becomes functional. Every adapter or controller card within a system machine can execute code during the boot process[12] and has the potential to modify media.

For example, during boot the SCSI card announces itself and its location and allows the user the option to enter its setup. Most hardware announces itself during boot so that the system knows it is available for use. Therefore, *Write Blocker for Windows 2000* relies on these boot software modules to be well behaved.

IV. Description of Test Procedures

The following is a description of test procedures performed to verify the correct operation of the write blocker application as well as its ability to prevent modification of write-protected storage media.

Write Blocker for Windows 2000 was tested with various different media types and interfaces. The Test Media/Devices included in the testing are listed below. A complete description of the test media disks and drives is located within the test procedures in Section VI.

- EIDE drives
- SCSI drives
- USB drives
- Firewire drives
- Floppy drives
- Zip drives
- MO drives
- Parallel port drives
- PCMCIA drives
- Recordable CD-ROM drives
- JAZ drives

---

12 (Howe 1995) URL http://dictionary.reference.com/search?q=bootstrap%20loader

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

The methods used to attempt to modify the contents of write-protected computer storage media include:

- Trying to copy/move/save a file in a write-protected partition
- Modifying properties/attributes of files on a write-protected partition
- Deleting a file in the write-protected partition
- Using WinHex to write bytes to unpartitioned space or to change/add/remove information on a Mac or Linux drive
- Attempt a format of the partition
- Attempt an upgrade to a Dynamic Disk

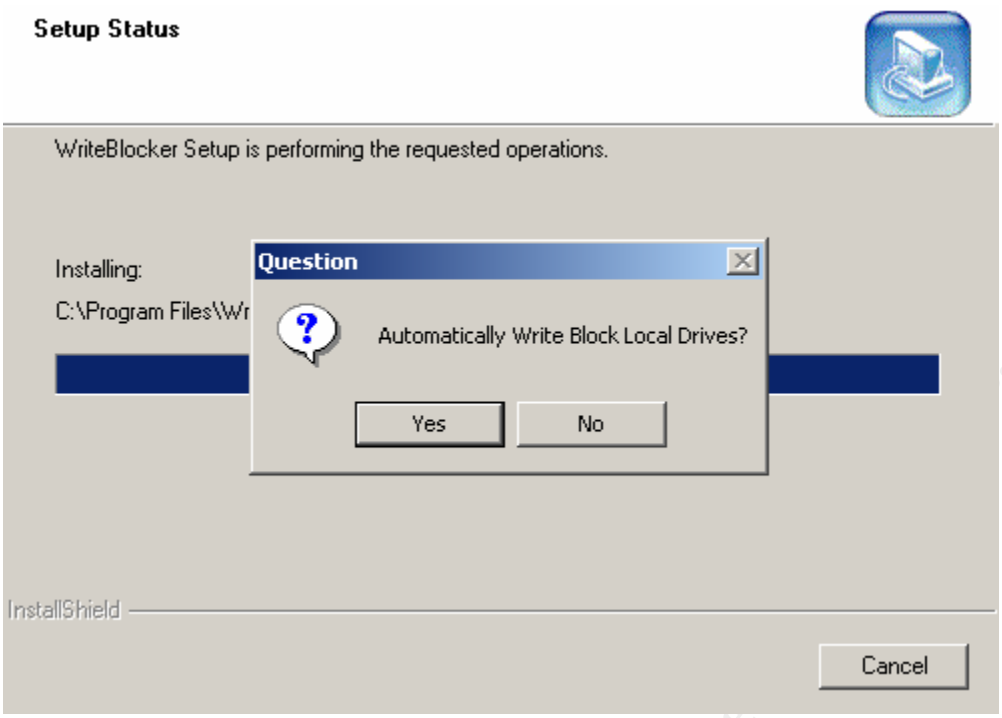
Specific Media Tests:

- Apply previously-mentioned write methods to different file systems by attaching (one at a time) a Linux, Mac, Windows 98, ME, XP, NT, or 2000 drive to the test system.
- Apply previously mentioned write methods to non-system partitions on the boot disk
- Have a hot-pluggable device connected at start-up and perform write tests
- Connect a hot-pluggable device after initial boot-up and perform write tests

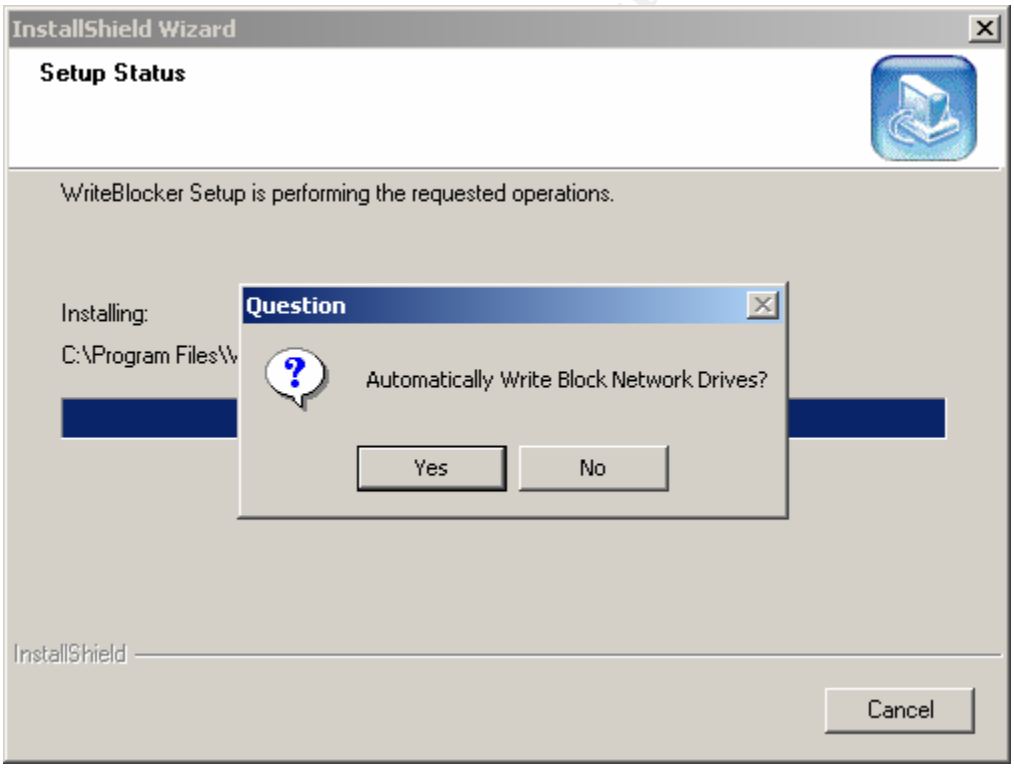
Before any tests are performed, *Write Blocker for Windows 2000* must be installed on all system disks.

The following is the installation process for *Write Blocker for Windows 2000.*
1. Insert the *Write Blocker for Windows 2000* CD.
2. Installshield® Wizard appears as a result of autorun.inf.
3. A message appears “do you wish to install *Write Blocker for Windows 2000*?”
4. Press the “yes” button.
5. The program is installed and then the following window appears

© SANS Institute 2003, Author retains full rights.

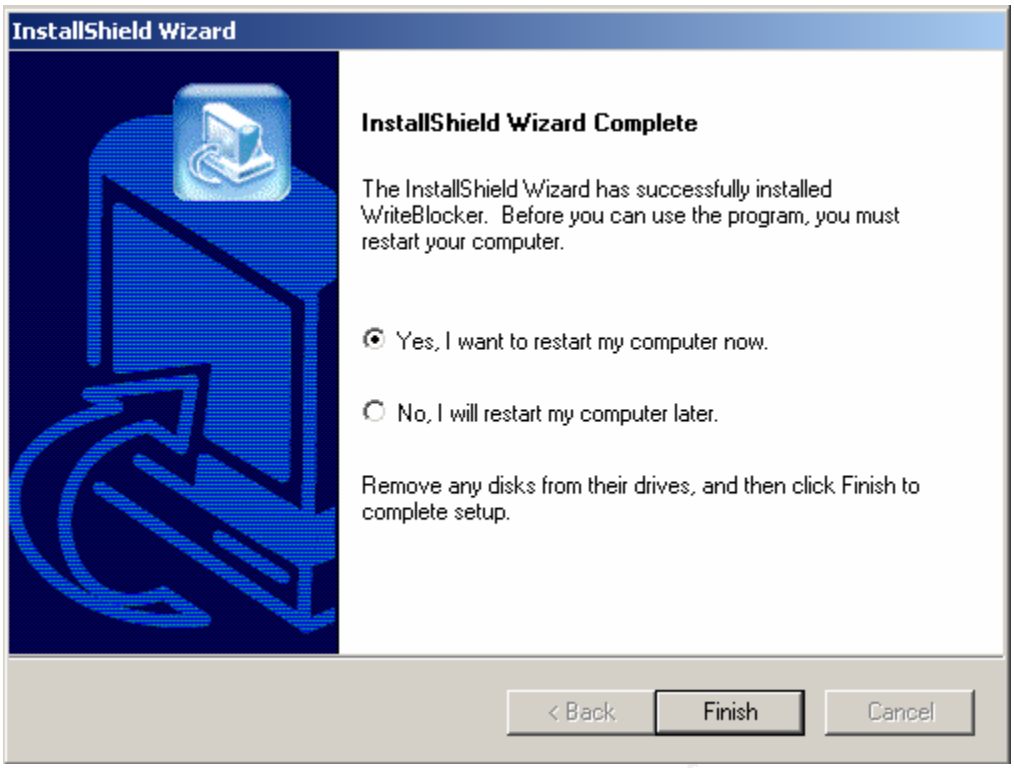


6. Click “Yes”
7. The following window appears:



8. Click “Yes”
9. The following window appears:

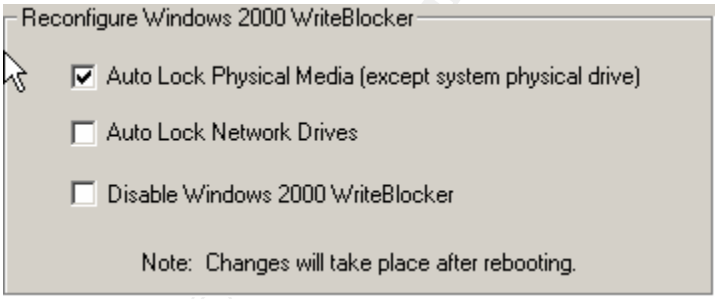
© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.



10. Click “Finish”

The system reboots and all local and networked drives are write-blocked.

Once installed, the user has the option to toggle on and off the write-blocked status of local and network drives in the Configuration window by double-clicking on a specific local or networked drive. Also, users have the ability to change the configuration as shown below.



The test results shown for the *Write Blocker for Windows 2000* consist of the MD5 and MD4 hashes collected before and after each test is completed.

V. Expected Results

By placing the sn.dat file or any data on media and obtaining before-test and after-test hashes of the media itself and of the sn.dat file solely on CD, the expected results are that both hashes will match. If the hashes do not match, then the data on the test disk has been modified and the tool would not be adequate for forensic investigation. If the hashes do match, then the documentation of the tests and the matching hashes could

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

be used to show that the tool did not allow any modifications to the test media and is a valuable forensic application. Thus, the results of the hashes could be used to verify that original and copies of evidence were not tampered with or changed inadvertently. The output of the hashes and explanations of MD5 (or other acceptable hash) could be used during presentation in court and is explained further in this paper.

VI. Test Procedures and Results

MD5 hashes listed under each test were created with the md5sum command in Linux from a Linux laptop and a Linux boot disk configured to not mount anything during boot.

- IBM 600E Thinkpad with Pentium II
  Distribution Version: Red Hat Linux release 7.1 (seawolf)
  OS Version: #1 Sun Apr 8 20:41:30 EDT 2001
  OS System Release: 2.4.2-2
  Processor Type: i686
  Stand-alone (no network connections)

MD5[13] has a 128-bit message digest and produces a 32-character display hash. The Linux md5sum command is used to calculate an md5sum hash. For the CD tests, the hash was calculated for the sn.dat file only. The Linux md5sum command does not calculate a hash for an entire CD. A hash of the CD device itself can be obtained by using the command md5sum /dev/sr0 on my Linux laptop, but this does not assist with the tests performed, as it does not hash the CD inserted in the CD-ROM drive. The CDs used during testing could have files added, removed or changed with the proper software as described in the CD-R and CR-RW tests.

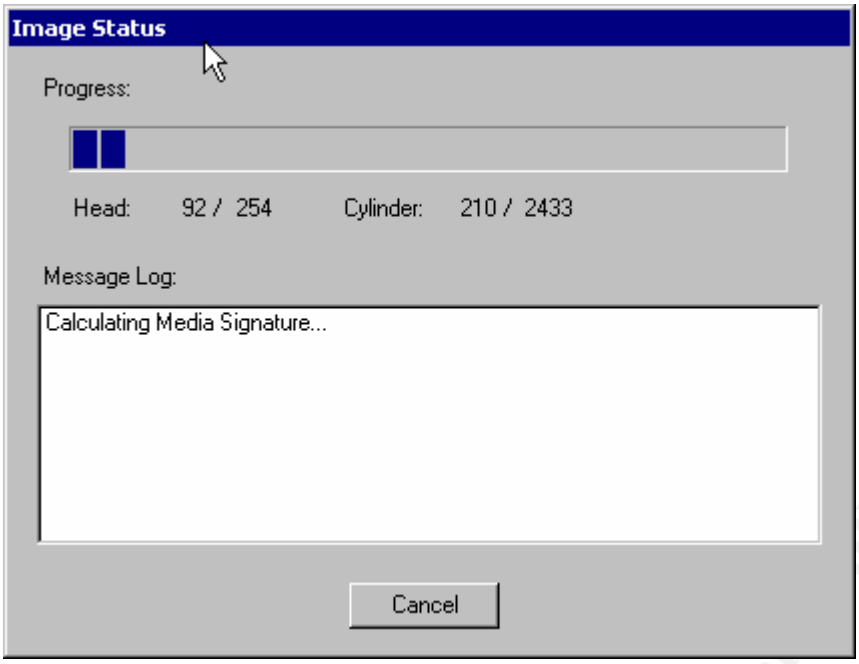
A proprietary MD4[14] hash utility is used for calculating hashes in some tests. The utility was created for Windows 2000 using the CryptoLib by Jack Lacy[15]. Examples of the utility creating a media signature and an image with media signature are shown below.

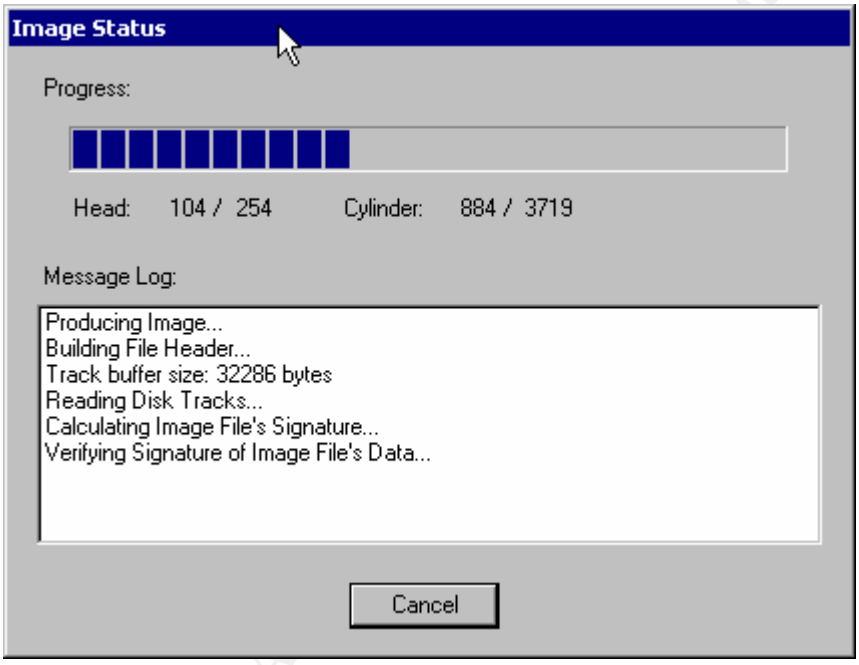
© SANS Institute 2003, Author retains full rights.

[13] (Rivest 1992) URL http://www.faqs.org/rfcs/rfc1321.html.

[14] (Rivest 1990) URL http://www.faqs.org/rfcs/rfc1186.html

[15] (Lacy, CryptoLib) URL
http://www.kanga.nu/~mirror/mirrors/www.homeport.org/%257Eadam/crypto/cryptolib.phtml





Tests Performed:

Write Blocker for Windows 2000 is designed to run on the Windows 2000 platform. Therefore, all tests were performed using a hard drive with Windows 2000 installed.

Test Case: 1) New drive verified

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a new out-of-the-box hard drive.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Test Number: Win2kWB.New.Drive.Test.1
Test Environment:
Hardware:
Dell OptiPlex GX 150
Processor: Intel Pentium III Processor 933Mhz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
Adaptec SCSI 2940
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
Manufacturer: Western Digital
Model: WD200BB-00DEA0
Manufacturer-Reported Capacity: 20 GB
Reported Capacity: 18.64 GB - 19,085 MB

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) IDE CD-ROM device 2) Diskette drive and 3) Hard-disk drive C:.
3. The test hard drive is in a removable caddy and the bay in which it is inserted is connected to the test system via the secondary IDE channel.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Attach the hard drive to the Host system.
4. Perform write attempts to the drive:
   - Attempt to format the attached drive
   - Attempt to write a signature to the attached drive by using the Disk Management utility within Windows 2000
   - Attempt to Upgrade to Dynamic Disk (using the Computer Management tool in Windows 2000 – right click on the desired drive and choose Upgrade to dynamic disk)
   - Attempt to create and modify partitions
   - Use WinHex© to attempt writes to unallocated space
   - Use the WinHex© Modify Data option (under Edit on the toolbar) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data
5. Reboot the system with the drive attached and retry the tests.
6. Calculate the hash again.
7. Compare the hashes.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Expected Result: The MD5 sum calculated after the testing is performed will match the original MD5 sum for the hard drive.

Actual Result: The sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing.
Before MD5sum Western Digital: 64955fcbc18b0746141e4c2d7ee508cb
After MD5sum Western Digital: 64955fcbc18b0746141e4c2d7ee508cb

© SANS Institute 2003, Author retains full rights.

Test Case: 2) Wipe drive verified

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent a disk wiping operation attempted by using WinHex© version 10.7 and the Fill Disk Sectors option.

Test Number: Win2kWB.WinHex.Wipe.Test.1
Test Environment:
Hardware:
- Dell OptiPlex GX 110
- Processor: Intel Pentium III Processor 733Mhz
- BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
- Adaptec SCSI 2930

Software:
- Operating System: Microsoft Windows 2000 Professional
- Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
- Manufacturer: 20GB Maxtor IDE hard drive
- Reported Capacity: 19595 MB
- Unallocated Space: 17595MB

Partition One:
- File System: FAT
- Capacity 1.95 GB - 2,097,119,232 bytes
- Used Space: 1.03 GB - 1,110,573,056 bytes
- Free Space: 940 MB - 986,546,176 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) IDE CD-ROM device 2) Diskette drive and 3) Hard-disk drive C:.
3. The Maxtor test hard drive is in a removable caddy and the bay in which it is inserted is connected to the test system via the secondary IDE channel.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the drive and each partition:
   - Attempt to wipe the hard drive using WinHex's "Fill Disk Sectors" option
4. Reboot the system with the drive attached and retry the test.
5. Calculate the hash again.
6. Compare the hashes.

Expected Result: The MD5 sum calculated after the testing is performed will match the original MD5 sum for the hard drive.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Actual Result: The sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing.
Before MD5sum: a1205a61eaa8371f74b6acc07aa1873c
After MD5sum: a1205a61eaa8371f74b6acc07aa1873c

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Test Case: 3) Multiple drives verified
Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to multiple hard drives attached to one system. The hard drives contain multiple partitions with various formats.

Test Number: Win2kWB.Multiple.Drives.1
Test Environment:
Hardware:
- Dell OptiPlex GX 150
- Processor: Intel Pentium III Processor 933Mhz
- BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
- Adaptec SCSI 2940

Software:
- Operating System: Microsoft Windows 2000 Professional
- Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive 1:
- Manufacturer: 20 GB Maxtor IDE hard drive
- Model: 2 F020J0
- Reported Capacity: 19876 MB
- Unallocated Space: 13382 MB

Partition One:
- Type: Primary
- File System: FAT32
- Capacity: 994 MB – 1,042,542,112 bytes
- Used Space: 280 MB – 294,481,920 bytes
- Free Space: 713 MB – 748,040,192 bytes

Partition Two:
- Type: Primary
- File System: NTFS
- Capacity: 1.95 GB – 2,097,444,864 bytes
- Used Space: 292 MB – 307,105,792 bytes
- Free Space: 1.66 GB – 1,790,339,072 bytes

Partition Three:
- Type: Primary
- File System: EXT2
- Capacity: 2.93 GB -
- Used Space: unavailable
- Free Space: unavailable

Test Drive 2:
- Manufacturer: 10.2 GB Conner Technology
- Model: CT210
- Reported Capacity: 9783 MB
- Unallocated Space: 2783 MB

Partition One:
- Type: Primary

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

File System: FAT
Capacity: 995 MB – 1,044,299,776 bytes
Used Space: 430 MB – 451,182,592 bytes
Free Space: 565 MB – 593,117,184 bytes

Partition Two:
Type: Primary
File System: NTFS
Capacity: 1.95 GB – 2,097,444,864 bytes
Used Space: 442 MB – 463,910,912 bytes
Free Space: 1.52 GB – 1,633,533,952 bytes

Partition Three:
Type: Logical Partition within an Extended Partition
File System: FAT32
Capacity: 1.93 GB – 2,081,136,640 bytes
Used Space: 429 MB – 450,672,640bytes
Free Space: 1.52 GB – 1,630,464,000 bytes

Partition Four:
Type: Logical Partition within an Extended Partition
File System: NONE - Empty
Capacity: 1.95 GB

Test Drive 3:
Manufacturer: Quantum
Model: Atlas V 3.5 Series
Reported Capacity: Unknown
Unallocated Space: 722 MB

Partition 1
Type: Primary
Volume Label: P1FAT
File System: FAT16
Volume Size: 1.46 GB - 1,570,766,848 bytes
Used Space: 1.39 GB - 1,500,020,736 bytes
Free Space: 67.4MB - 70,746,112 bytes

Partition 2
Type: Primary
Volume Label: P2FATAUTO
File System: FAT16 AUTO
Volume Size: 501 MB - 526,139,392 bytes
Used Space: 476 MB - 500,006,912 bytes
Free Space: 24.9 MB - 26,132,480 bytes

Partition 3
Type: Primary
Volume Label: P3FAT32
File System: FAT32
Volume Size: 2.93 GB - 3,152,330,752 bytes
Used Space: 2.32 GB - 2,500,009,984 bytes

© SANS Institute 2003, Author retains full rights.

Free Space: 622 MB - 652,320,768 bytes

Partition 4

Type: Logical Partition within an Extended Partition
Volume Label: P4NTFS
File System: NTFS
Volume Size: 1.47 GB - 1,587,445,760 bytes
Used Space: 1.40 GB - 1,504,765,952 bytes
Free Space: 78.8 MB - 82,679,808 bytes

Partition 5

Type: Logical Partition within an Extended Partition
Volume Label: P5FAT32
File System: FAT32
Volume Size: 1.46GB - 1,576,124,416 bytes
Used Space: 1.39 GB - 1,500,008,448 bytes
Free Space: 72.5 MB - 76,115,968 bytes

Test Notes:

1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) IDE CD-ROM device 2) Diskette drive and 3) Hard-disk drive C:.
3. Test hard drive 1 is attached to an added Promise Technology Inc. Ultra66 IDE Controller and is connected to IDE1 (primary) on the IDE controller. Test hard drive 2 is in a removable caddy and the bay in which it is inserted is connected to the test system via the secondary IDE channel. Test hard drive 3 is attached to an Adaptec 2940 SCSI adapter inside the system.

Procedures:

1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the drives:
    - Attempt to format the attached drives
    - Attempt to Upgrade to Dynamic Disk (using the Computer Management tool in Windows 2000 – right click on the desired drive and choose Upgrade to dynamic disk)
    - Attempt to create and modify partitions
    - Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drives
    - Use the WinHex© Modify Data option (under Edit on the toolbar) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data
4. Reboot the system with the drives attached and retry the tests.
5. Calculate the hash again.
6. Compare the hashes.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Expected Result: The MD5 sum calculated after the testing is performed will match the original MD5 sum for the hard drives.

Actual Result: All sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing for test-drives one through three.

Test Drive 1 Maxtor Before MD5sum: 428f6298177123bd6652385d542b2c4a
Test Drive 1 Maxtor After MD5sum: 428f6298177123bd6652385d542b2c4a

Test Drive 2 Conner Before MD5sum: e436f45abf3758c7adb9337d27fe9e51
Test Drive 2 Conner After MD5sum: e436f45abf3758c7adb9337d27fe9e51

Test Drive 3 Quantum Before MD5sum: 2748edca1ecf80c564e9536d611306a4
Test Drive 3 Quantum After MD5sum: 2748edca1ecf80c564e9536d611306a4

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Test Case: 4) Floppy disk verified - Dell OptiPlex GX 150
Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a floppy disk.

Test Number: Win2kWB.Floppy.Test.1
Test Environment:
Hardware:
Dell OptiPlex GX 150
Processor: Intel Pentium III Processor 933Mhz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
Adaptec SCSI 2940
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Floppy:
Manufacturer: Imation
Manufacturer-Reported Capacity: 1.44 MB
Reported Capacity: 1.38 MB (1,457,664 bytes)
Volume Label: Floppy
File System: FAT 12
Volume Size: 1.38 MB (1,457,664 bytes)
Used Space: 1.33 MB (1,400,832 bytes)
Free Space: 55.5 KB (56,832 bytes)

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C:. 2) IDE CD-ROM device and 3) Diskette drive

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts:
   - Attempt to format the attached disk
   - Try to delete files, add files, modify files, copy files, move files, and save files
   - Modify properties and attributes of files
   - Use WinHex© to attempt writes to end-of-file slack space
4. Reboot the system with the floppy disk inserted (the test system is configured to boot from the hard drive before looking for other bootable media) and retry the tests.
5. Calculate the hash again.
6. Compare the hashes.

© SANS Institute 2003, Author retains full rights.

Expected Result: The MD5 sum calculated after the testing is performed will match the original MD5 sum for the Imation 1.44 MB floppy disk.

Actual Result: The sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing.
Before MD5sum: 3436f45abf3758c7abd9337d27fe9e51
After MD5sum: 3436f45abf3758c7abd9337d27fe9e51

© SANS Institute 2003, Author retains full rights.

Test Case: 5) CD-RW verified - Dell OptiPlex GX 150
Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a CD-RW disk.

Test Number: Win2kWB.CD-RW.test.1
Test Environment:
Hardware:
- Dell OptiPlex GX 150
- Processor: Intel Pentium III Processor 933Mhz
- BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
- Adaptec SCSI 2940

Software:
- Operating System: Microsoft Windows 2000 Professional
- Version: 5.0.2195 Service Pack 2 Build 2195

Test CD-RW (Compact Disc – ReWritable) while blank
- Manufacturer: Maxell
- Manufacturer-Reported Capacity: 650 MB
- File System: Unknown
- Used Space: 0 bytes
- Free Space: 0 bytes

Test CD-RW (Compact Disc – ReWritable) after data has been written to the CD
- Manufacturer: Maxell
- Manufacturer-Reported Capacity: 650 MB
- Reported Capacity:
- Volume Label:020429_1616
- File System: CDFS
- Used Space: 36,864 bytes
- Free Space: 0 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C:. 2) IDE CD-ROM device and 3) Floppy drive
3. The Roxio Easy CD Creator was used in this test.

Procedures:

1. There is no hash to calculate on a new CD-RW disk, as no files are present in order to obtain a hash.
2. Perform write attempts to the CD-RW using Roxio Easy CD Creator (or similar utility).

© SANS Institute 2003, Author retains full rights.

- Attempt to record files to the CD using the record button within the Roxio Easy CD Creator after files were specified to be placed on the CD
- Choose the Format option within the directCD format utility

3. Reboot the system with the CD-RW inserted and retry the test.
4. Remove the CD-RW and insert in another system to check if the CD has data.
5. Toggle the Writeblocker off for the CD-ROM drive and write data to the CD using Roxio Easy CD Creator.
6. Toggle Writeblocker back on for the CD-ROM drive
7. Calculate a hash of the data on the CD-RW disk.
8. Perform write attempts to the CD-RW using Roxio Easy CD Creator (or similar utility).
    - Attempt to record files to the CD using the record button within the Roxio Easy CD Creator after files were specified to be placed on the CD
    - Choose the Format option within the directCD format utility
9. Calculate the hash again.
10. Compare the hashes.

Expected Result: After running the tests on the blank CD the CD would not display data when inserted into another system. The MD5 sum calculated on the file written to the CD-RW after the testing is performed will match the original MD5 sum for the Maxell 650MB CD-RW.

Actual Result: The CD contained no data after the original attempt to write data to the CD. After data had been written to the CD and hashes were calculated the sums matched. The sums listed show the actual MD5sums calculated before the testing and after the testing.
Before CD sn.dat file MD5sum: 0e954f43fd73f56e812a7285f32e41d3
After CD sn.dat file MD5sum: 0e954f43fd73f56e812a7285f32e41d3

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Test Case: 6) CD-R verified - Dell OptiPlex GX 150
Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a CD-R disk.

Test Number: Win2kWB.CD-R.test.1
Test Environment:
Hardware:
- Dell OptiPlex GX 150
- Processor: Intel Pentium III Processor 933Mhz
- BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
- Adaptec SCSI 2940

Software:
- Operating System: Microsoft Windows 2000 Professional
- Version: 5.0.2195 Service Pack 2 Build 2195

Test CD-R (Compact Disc – Recordable):
- Manufacturer: Memorex
- Manufacturer-Reported Capacity: 700MB
- Reported Capacity: 703MB – 737,271,808 bytes
- Volume Label: Test
- File System: CDFS
- Used Space: 25MB – 25,780,224 bytes
- Free Space: 679 MB - 711,491,584 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C:. 2) IDE CD-ROM device and 3) Floppy drive
3. The Test CD-R used in this test was created using IBM RecordNow v.3.5.
4. DirectCD has the option to Make Writable a CD-R that already contains data.

Procedures:

1. Create a hash of the file(s) that are present on the CD-R.
2. Record the hash(es).
3. Perform write attempts to the CD-R using Roxio directCD (or similar utility if desired).
   - Choose the Make Writable option within the directCD format utility
   - Choose the Format option within the directCD format utility
4. Reboot the system with the CD-R inserted and retry the tests.
5. Calculate the hash(es) again.
6. Compare the hashes.

Expected Result: The MD5 sum calculated after the testing is performed will match the original MD5 sum for the file(s) located on the Memorex 700 MB CD-R.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Actual Result: The sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing.
Before CD-R sn.dat file MD5sum: 0e954f43fd73f56e812a7285f32e41d3
After CD-R sn.dat file MD5sum: 0e954f43fd73f56e812a7285f32e41d3

© SANS Institute 2003, Author retains full rights.

Test Case: 7) CD-RW verified - IBM Thinkpad T-23
Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a CD-RW disk.

Test Number: Win2kWB.CD-RW.test.2
Test Environment:
Hardware:
IBM T23 Laptop
Processor: Intel Pentium III Processor
BIOS version: Version 1.00 PSRTTBL
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test CD-RW (Compact Disc – ReWritable) blank
Manufacturer: Maxell
Manufacturer-Reported Capacity: 650 MB
Volume Label:
File System: CDFS
Used Space: 0 bytes
Free Space: 0 bytes

Test CD-RW (Compact Disc – ReWritable) after data has been written to the CD
Manufacturer: Maxell
Manufacturer-Reported Capacity: 650 MB
Reported Capacity:
Volume Label: Test2
File System: CDFS
Used Space: 75,098 bytes
Free Space: 0 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C:. 2) IDE CD-ROM device and 3) Floppy drive
3. DirectCD has the option to Make Writable a CD-R that already contains data.

Procedures:

1. There is no hash to calculate on a new CD-RW disk, as no files are present in order to obtain a hash.
2. Perform write attempts to the CD-RW using Roxio Easy CD Creator (or similar utility).
    - Attempt to record files to the CD using the record button within the Roxio Easy CD Creator after files were specified to be placed on the CD

© SANS Institute 2003, Author retains full rights.

   - Choose the Format option within the directCD format utility
3. Reboot the system with the CD-RW inserted and retry the test.
4. Remove the CD-RW and insert in another system to check if the CD contains data.
5. Toggle the Writeblocker off for the CD-ROM drive and write data to the CD using Roxio Easy CD Creator.
6. Toggle Writeblocker back on for the CD-ROM drive
7. Calculate a hash of the data on the CD-RW disk.
8. Perform write attempts to the CD-RW using Roxio Easy CD Creator (or similar utility).
   - Attempt to record files to the CD using the record button within the Roxio Easy CD Creator after files were specified to be placed on the CD
   - Choose the Format option within the directCD format utility
9. Calculate the hash again.
10. Compare the hashes.

Expected Result: After running the tests on the blank CD the CD would not display data when inserted into another system. The MD5 sum calculated on the file written to the CD-RW after the testing is performed will match the original MD5 sum for the Maxell 650MB CD-RW.

Actual Result: The CD contained no data after the original attempt to write data to the CD. After data had been written to the CD and hashes were calculated the sums matched. The sums listed show the actual MD5sums calculated before the testing and after the testing.

Before CD-RW sn.dat file MD5sum: 0e954f43fd73f56e812a7285f32e41d3
After CD-RW sn.dat file MD5sum: 0e954f43fd73f56e812a7285f32e41d3

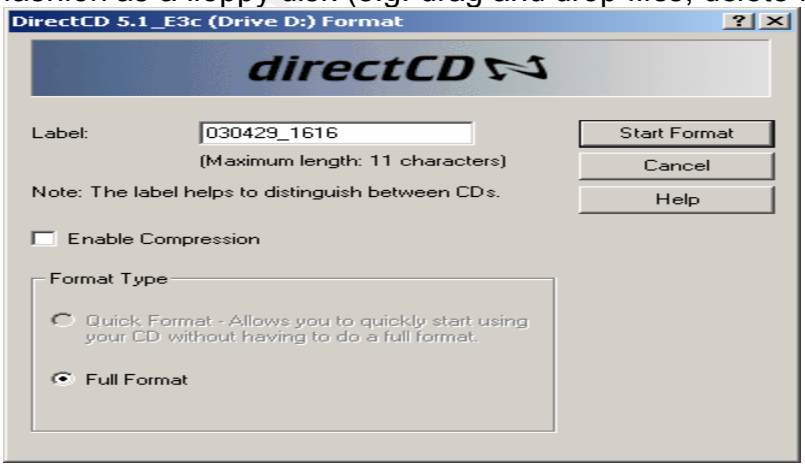
© SANS Institute 2003, Author retains full rights.

Test Case: 8) CD-R DirectCD formatted verified - IBM Thinkpad T-23
Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a CD-R disk.

Test Number: Win2kWB.CD-R.DirectCD.test.1
Test Environment:
Hardware:
IBM T23 Laptop
Processor: Intel Pentium III Processor
BIOS version: Version 1.00 PSRTTBL
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test CD-R (Compact Disc – Recordable)
Manufacturer: Maxell
Manufacturer-Reported Capacity: 700 MB
Reported Capacity: 737,271,808 bytes - 703 MB
Volume Label: lkj
File System: CDFS
Used Space: 39,923,712 bytes
Free Space: 697,348,096 bytes

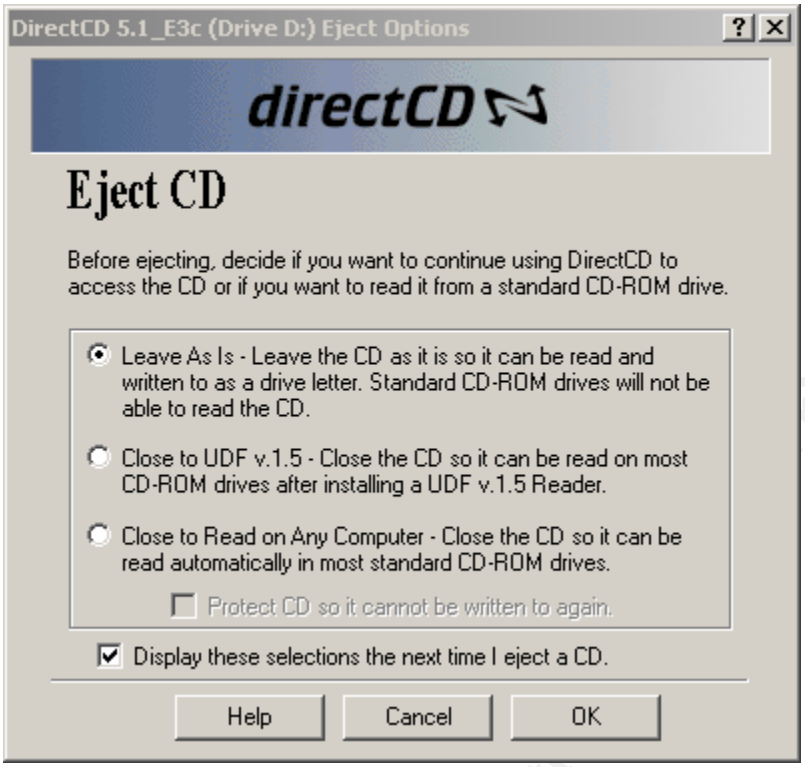
Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C:. 2) IDE CD-ROM device and 3) Floppy drive.
3. The Test CD-R was formatted with the Roxio directCD format utility within the Roxio easy CD creator 5 basic version to allow writing to the CD in the same fashion as a floppy disk (e.g. drag and drop files, delete files, move files, etc.)[16].



[16] Information on Roxio products may be found at http://www.roxio.com/en/products/ecdc/index.jhtml

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

4. When the CD-R was ejected from the system the directCD option to allow the CD-R to remain formatted as a directCD was chosen.



5. Shows that the formatted CD-R is ready for use.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Procedures:

1. Create a hash of the file(s) on the CD.
2. Record the hash(es).
3. Perform write attempts to the CD-R using Roxio directCD.
    - Try to delete files, add files, modify files, copy files, move files, save files
    - Modify properties and attributes of files
4. Reboot the system with the CD-R inserted and retry the tests.
5. Calculate the hash(es) again.
6. Compare the hashes.

Expected Result: The MD5 sum calculated after the testing is performed will match the original MD5 sum for the Memorex 700 MB CD-R.

Actual Result: The sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing.
Before CD-R file MD5sum: 40d28e6f8737da410ac20dcc1f814a2a
After CR-R file MD5sum: 40d28e6f8737da410ac20dcc1f814a2a

© SANS Institute 2003, Author retains full rights.

Test Case: 9) MO Drive verified - Dell OptiPlex GX 110

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any write attempts to a Magneto Optical (MO) Disk that is connected to the test system via the SCSI controller.

Test Number: Win2kWB.MO.test.1
Test Environment:
Hardware:
Dell OptiPlex GX 110
Processor: Intel Pentium III Processor 733Mhz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
Adaptec SCSI 2930

Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Disk:
Manufacturer: Verbatim
Manufacturer-Reported Capacity: 2.3 GB (1.15 GB per side)

Side A:
Reported Capacity: 1096 MB

Partition 1 (The only partition on the side)
Type: Primary
Volume Label: MONTFS
File System: NTFS
Volume Size: 547 MB - 574,602,240 bytes
Used Space: 372 MB - 390,291,456 bytes
Free Space: 175 MB - 184,310,784 bytes
Unpartitioned Free Space: 548 MB

Side B:
Reported Capacity: 1096 MB

Partition 1 (The only partition on the side)
Type: Primary
Volume Label: MOFAT
File System: FAT
Volume Size: 547 MB - 574,439,424 bytes
Used Space: 367 MB - 385,810,432 bytes
Free Space: 179 MB - 188,628,992 bytes
Unpartitioned Free Space: 548 MB

Test Notes:

© SANS Institute 2003, Author retains full rights.

1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C:. 2) IDE CD-ROM device and 3) Diskette drive

Procedures:

1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the MO Disk Side A :
    - Attempt to format the attached disk
    - Try to delete files, add files, modify files, copy files, move files, and save files
    - Modify properties and attributes of files
    - Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drive
    - Use the WinHex© Modify Data option (under Edit on the toolbar) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data
4. Turn off the MO Drive and turn the MO drive back on.
5. Reboot the system with the MO Disk inserted and retry the tests.
6. Remove the MO Disk and insert on Side B.
7. Turn off the MO Drive and turn the MO drive back on.
8. Reboot the system with the MO Disk inserted and retry the tests.
9. Calculate the hashes again.
10. Compare the hashes.

Expected Result: The MD4 sum calculated after the testing is performed will match the original MD4 sum for the Verbatim 2.3 GB Magneto-Optical (MO) Disk.

Actual Result: The sums match. The sums listed show the actual MD4sums calculated before the testing and after the testing.
Side A
Before MD4 hash MO: 6366a6e9f34e188ec40163d1341707d2
After MD4 hash MO: 6366a6e9f34e188ec40163d1341707d2

Side B
Before MD4 hash MO: d60051740c591fa9fdf8208f61a47529
After MD4 hash MO: d60051740c591fa9fdf8208f61a47529

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Test Case: 10) Jaz disk SCSI verified - Dell OptiPlex GX 110

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a Jaz disk connect by an Adaptec AHA-2930CU SCSI card.

Test Number: Win2kWB.Jaz.SCSI.test.1
Test Environment:
Hardware:
Dell OptiPlex GX 110
Processor: Intel Pentium III Processor 733Mhz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
Adaptec SCSI 2930
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
Manufacturer: Iomega
Model: Jaz 2GB
Manufacturer-Reported Capacity: 2.0 GB
Reported Capacity: 1.86 GB

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C: 2) IDE CD-ROM device and 3) Diskette drive.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the Jaz Disk:
   - Attempt to format the attached disk
   - Try to delete files, add files, modify files, copy files, move files, and save files
   - Modify properties and attributes of files
   - Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drive
   - Use the WinHex© Modify Data option (under the Edit file menu) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data
4. Reboot the system with the Jaz disk and retry the tests.
5. Calculate the Jaz disk hash again.
6. Compare the hashes.

© SANS Institute 2003, Author retains full rights.

Expected Result: The MD4 sum calculated after the testing is performed will match the original MD4 sum for the Jaz disk.

Actual Result: The sums match. The sums listed show the actual MD4sums calculated before the testing and after the testing.
Before MD4sum Jaz: 7e0674d28562778e037658de2bbce7cd
After MD4sum Jaz: 7e0674d28562778e037658de2bbce7cd

© SANS Institute 2003, Author retains full rights.

Test Case: 11) Zip disk verified - USB Dell Dimension 4550

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a Zip disk inserted in a Zip drive that is connected to the system by an Intel PCI to USB Enhanced Host Controller.

Test Number: Win2kWB.Zip.USB.test.1
Test Environment:
Hardware:
Dell Dimension 4550
Processor: Intel Pentium 4 Processor 2.4 GHz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A03
Adaptec SCSI 3.10.0
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
Manufacturer: Iomega
Model: Z100 USBS
Manufacturer-Reported Capacity: 100 MB
Reported Capacity: 95 MB

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C: 2) IDE CD-ROM device and 3) Diskette drive.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the Zip Disk:
   - Attempt to format the attached disk
   - Try to delete files, add files, modify files, copy files, move files, and save files
   - Modify properties and attributes of files
   - Use WinHex© to attempt writes to end-of-file slack space
4. Reboot the system with the Zip disk inserted (the System Disk is configured to boot from the hard drive before looking for other bootable media) and retry the tests.
5. Calculate the Zip disk hash again.
6. Compare the hashes.

Expected Result: The MD4 sum calculated after the testing is performed will match the original MD4 sum for the Zip disk.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Actual Result: The sums match. The sums listed show the actual MD4sums calculated before the testing and after the testing.
Before MD4sum USB Zip: 615dbcf6e3e367b8a5be9b4992cf3B4a
After MD4sum USB Zip: 615dbcf6e3e367b8a5be9b4992cf3B4a

© SANS Institute 2003, Author retains full rights.

Test Case: 12) Mapped drive verified

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a Network Drive while the host computer is connected to a file server using an Intel® Pro/100 VE ethernet network adapter.

Test Number: Win2kWB.Network.test.1
Test Environment:
Hardware:
Dell OptiPlex GX 110
Processor: Intel Pentium III Processor 733Mhz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
Adaptec SCSI 2930
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
Reported Capacity of Mapped Drive: 26.4GB

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. Map the test drive by using Map Network Drive in Microsoft Explorer.

Procedures:
1. Create a hash of the mapped network drive by using the software Advanced CheckSum Verifier ("ACSV")[17]. The software calculates a hash for every file on the specified mapped volume. There are 32,925 files for which the hash was calculated.
2. Record the hash(es).
3. Perform write attempts to the Mapped Network Drive:
   - Try to delete files, add files, modify files, copy files, move files, save files
   - Modify properties and attributes of files
4. Reboot the system and reconnect to the Mapped Network Drive.
5. Calculate the Mapped Network Drive's hash again using ACSV.
6. Compare the hashes by chosing several files to examine.

Expected Result: The MD5sum calculated after the testing is performed will match the original MD5sum for each file on the Mapped Network Drive.

Actual Result: The sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing.

[17] The Advanced CheckSum Verifier is copyrighted by author Irnis I.Haliullin and can be downloaded for a free trial at http://www.irnis.net/soft/acsv/?from=banner

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Before MD5sum for dfr7.bmp: 66e6f413e630ba05842bbd284f10f596
After MD5sum for dfr7.bmp: 66e6f413e630ba05842bbd284f10f596

Before MD5sum for ForensicStandAlone\lfa 2.0\1-31-03\data1.cab:
cf3876c5adb537219163483e72ff2660
After MD5sum for ForensicStandAlone\lfa 2.0\1-31-03\data1.cab:
cf3876c5adb537219163483e72ff2660

Before MD5sum for WBPort\Drivers\NTWBFS\SOURCES:
e868ab0bc71e874c5dcbf1ebf368288c
After MD5sum for WBPort\Drivers\NTWBFS\SOURCES:
e868ab0bc71e874c5dcbf1ebf368288c

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Test Case: 13) Floppy disk verified - Dell Dimension 4550

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a floppy disk.

Test Number: Win2kWB.Floppy.test.2
Test Environment:
Hardware:
Dell Dimension 4550
Processor: Intel Pentium 4 Processor 2.4 GHz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A03
Adaptec SCSI 3.10.0
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
Manufacturer: Imation
Manufacturer-Reported Capacity: 1.44 MB
Reported Capacity: 1.38 MB

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C: 2) IDE CD-ROM device and 3) Diskette drive.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum..
2. Record the hash.
3. Perform write attempts to the Floppy Disk:
   - Attempt to format the attached disk
   - Try to delete files, add files, modify files, copy files, move files, and save files
   - Modify properties and attributes of files
   - Use WinHex© to attempt writes to end-of-file slack space
4. Reboot the system with the floppy disk inserted (the System Disk is configured to boot from the hard drive before looking for other bootable media) and retry the tests.
5. Calculate the hash again.
6. Compare the hashes.

Expected Result: The MD5 sum calculated after the testing is performed will match the original MD5 sum for the floppy disk.

© SANS Institute 2003, Author retains full rights.

Actual Result: The sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing.
Before MD5sum Floppy: 71d75477b778b32efca8555765843618
After MD5sum Floppy: 71d75477b778b32efca8555765843618

© SANS Institute 2003, Author retains full rights.

Test Case: 14) IDE drive verified - Dell OptiPlex GX 110

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to an IDE hard drive.

Test Number: Win2kWB.IDE.test.2
Test Environment:
Hardware:
- Dell OptiPlex GX 110
- Processor: Intel Pentium III Processor 733Mhz
- BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
- Adaptec SCSI 2930

Software:
- Operating System: Microsoft Windows 2000 Professional
- Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
- Manufacturer: Maxtor
- Model: DiamondMax Plus 8 6E030L0510252
- Manufacturer-Reported Capacity: 30 GB
- Reported Capacity: 26.8 GB
- Unallocated Space: 0

Partition One:
- File System: NTFS
- Capacity: 28.8 GB – 30,704,967,680 bytes
- Used Space: 3.94 GB – 4,231,950,336 bytes
- Free Space: 24.6 GB – 26,473,017,344 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C: 2) IDE CD-ROM device and 3) Diskette drive. The test hard drive is in a removable caddy and the bay in which it is inserted is connected to the test system via the secondary IDE channel.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the IDE Disk:
   - Attempt to format the attached disk
   - Try to delete files, add files, modify files, copy files, move files, and save files
   - Modify properties and attributes of files
   - Attempt to Upgrade to Dynamic Disk (using the Computer Management tool in Windows 2000 – right click on the desired drive and choose Upgrade to dynamic disk)

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

- Attempt to create and modify partitions
- Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drive
- Use the WinHex© Modify Data option (under the Edit file menu) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data

4. Reboot the system with the IDE disk inserted and retry the tests.
5. Calculate the hash again.
6. Compare the hashes.

Expected Result: The MD5 sum calculated after the testing is performed will match the original MD5 sum for the IDE drive.

Actual Result: The sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing.
Before MD5sum Maxtor Diamond: 39f9674d16d7bef39dd2ec2f11f682e4
After MD5sum Maxtor Diamond: 39f9674d16d7bef39dd2ec2f11f682e4

© SANS Institute 2003, Author retains full rights.

Test Case: 15) IDE drive verified - Dell Dimension 4550

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to an IDE hard drive.

Test Number: Win2kWB.IDE.test.3
Test Environment:
Hardware:
- Dell Dimension 4550
- Processor: Intel Pentium 4 Processor 2.4 GHz
- BIOS version: Phoenix ROM BIOS Plus v. 1.10 A03
- Adaptec SCSI 3.10.0

Software:
- Operating System: Microsoft Windows 2000 Professional
- Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
- Manufacturer: Seagate
- Model: ST31276A
- Manufacturer-Reported Capacity: 1.2 GB
- Reported Capacity: 1.18 GB
- Unallocated Space: 0

Partition One:
- File System: FAT
- Capacity: 1.18 GB – 1,277,624,320 bytes
- Used Space: 434 MB – 455,278,592 bytes
- Free Space: 784 MB – 822,345,728 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C: 2) IDE CD-ROM device and 3) Diskette drive. The test hard drive is in a removable caddy and the bay in which it is inserted is connected to the test system via the secondary IDE channel.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the IDE Disk:
   - Attempt to format the attached disk
   - Attempt to create and modify partitions
   - Try to delete files, add files, modify files, copy files, move files, save files
   - Modify properties and attributes of files
   - Attempt to Upgrade to Dynamic Disk (using the Computer Management tool in Windows 2000 – right click on the desired drive and choose Upgrade to dynamic disk)

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

- Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drive
- Use the WinHex© Modify Data option (under Edit on the toolbar) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data

4. Reboot the system with the IDE drive inserted (the System Disk is configured to boot from the hard drive before looking for other bootable media) and retry the tests.
5. Calculate the IDE disk hash again.
6. Compare the hashes.

Expected Result: The MD5 sum calculated after the testing is performed will match the original MD5 sum for the IDE disk.

Actual Result: The sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing.
Before MD5sum Seagate: 306518e2c5dec02301a00c85f843fb0e
After MD5sum Seagate: 306518e2c5dec02301a00c85f843fb0e

© SANS Institute 2003, Author retains full rights.

Test Case: 16) External SCSI drive verified - Dell Dimension 4550

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to an external SCSI hard drive connected by an Adaptec AIC-7892 Ultra 160/M PCI SCSI card.

Test Number: Win2kWB.SCSI.test.1
Test Environment:
Hardware:
- Dell Dimension 4550
- Processor: Intel Pentium 4 Processor 2.4 GHz
- BIOS version: Phoenix ROM BIOS Plus v. 1.10 A03
- Adaptec SCSI 3.10.0

Software:
- Operating System: Microsoft Windows 2000 Professional
- Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
- Manufacturer: Fujitsu
- Model: MA J3182MP
- Manufacturer-Reported Capacity: 18.2 GB
- Reported Capacity: 17.01 GB

Partition One:
- File System: FAT
- Capacity: 1.95 GB – 2,097,897,472 bytes
- Used Space: 1.27 GB – 1,372,569,600 bytes
- Free Space: 691 MB - 725,327,872 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C: 2) IDE CD-ROM device and 3) Diskette drive. An Adaptec AIC-7892 Ultra 160/M PCI SCSI card connects the external test hard drive to the system.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the SCSI hard drive:
   - Attempt to format the attached disk
   - Attempt to create and modify partitions
   - Try to delete files, add files, modify files, copy files, move files, save files
   - Modify properties and attributes of files
   - Attempt to Upgrade to Dynamic Disk (using the Computer Management tool in Windows 2000 – right click on the desired drive and choose Upgrade to dynamic disk)

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

- Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drive
- Use the WinHex© Modify Data option (under Edit on the toolbar) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data

4. Reboot the system and retry the tests.
5. Calculate the SCSI hard drive hash again.
6. Compare the hashes.

Expected Result: The MD4 sum calculated after the testing is performed will match the original MD4 sum for the SCSI drive.

Actual Result: The sums match. The sums listed show the actual MD4sums calculated before the testing and after the testing.
Before MD4sum SCSI Fujitsu External: ee7ede1f6897ba9f187628d7a467536a
After MD4sum SCSI Fujitsu External: ee7ede1f6897ba9f187628d7a467536a

© SANS Institute 2003, Author retains full rights.

Test Case: 17) MO disk verified - Dell OptiPlex GX 110

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a Magneto-Optical disk connect by an Adaptec AHA-2930CU SCSI card.

Test Number: Win2kWB.MO.test.2
Test Environment:
Hardware:
Dell OptiPlex GX 110
Processor: Intel Pentium III Processor 733Mhz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
Adaptec SCSI 2930
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
Manufacturer: Sony
Model: RMO-S551
Manufacturer-Reported Capacity: 2.3GB
Reported Capacity: 2.14GB

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C: 2) IDE CD-ROM device and 3) Diskette drive.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the MO Disk:
   - Attempt to format the attached disk
   - Try to delete files, add files, modify files, copy files, move files, save files
   - Modify properties and attributes of files
   - Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drive
   - Use the WinHex© Modify Data option (under Edit on the toolbar) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data
4. Reboot the system with the MO disk inserted (the System Disk is configured to boot from the hard drive before looking for other bootable media) and retry the tests.
5. Calculate the MO disk hash again.
6. Compare the hashes.

© SANS Institute 2003, Author retains full rights.

Expected Result: The MD4 sum calculated after the testing is performed will match the original MD4 sum for the MO disk.

Actual Result: The sums match. The sums listed show the actual MD4sums calculated before the testing and after the testing.
Before MD4sum MO: 47bcbccd5f078da80ca53acb250bb87c
After MD4sum MO: 47bcbccd5f078da80ca53acb250bb87c

© SANS Institute 2003, Author retains full rights.

Test Case: 18) Jaz disk verified - SCSI Dell OptiPlex GX 110

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to a Jaz disk connect by an Adaptec AHA-2930CU SCSI card.

Test Number: Win2kWB.Jaz.SCSI.test.2
Test Environment:
Hardware:
Dell OptiPlex GX 110
Processor: Intel Pentium III Processor 733Mhz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
Adaptec SCSI 2930
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
Manufacturer: Iomega
Model: Jaz 2GB
Manufacturer-Reported Capacity: 2.0 GB
Reported Capacity: 1.86 GB

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C: 2) IDE CD-ROM device and 3) Diskette drive.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the Jaz Disk:
   - Attempt to format the attached disk
   - Try to delete files, add files, modify files, copy files, move files, save files
   - Modify properties and attributes of files
   - Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drive
   - Use the WinHex© Modify Data option (under Edit on the toolbar) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data
4. Reboot the system with the Jaz disk inserted (the System Disk is configured to boot from the hard drive before looking for other bootable media) and retry the tests.
5. Calculate the Jaz disk hash again.
6. Compare the hashes.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Expected Result: The MD4 sum calculated after the testing is performed will match the original MD4 sum for the Jaz disk.

Actual Result: The sums match. The sums listed show the actual MD4sums calculated before the testing and after the testing.
Before MD4sum Jaz: 7e0674d28562778e037658de2bbce7cd
After MD4sum Jaz: 7e0674d28562778e037658de2bbce7cd

© SANS Institute 2003, Author retains full rights.

Test Case: 19) Laptop hot-swap IBM T23

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to an IBM T23 laptop.

Test Number: Win2kWB.Hot.Swap.test.1
Test Environment:
Hardware:
IBM T23 Laptop
Processor: Intel Pentium III Processor
BIOS version: Version 1.00 PSRTTBL
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Drives:

Floppy:
Manufacturer: Imation
Manufacturer-Reported Capacity: 1.44 MB
Reported Capacity: 1.38 MB (1,457,664 bytes)
Volume Label: Floppy
File System: FAT 12
Volume Size: 1.38 MB (1,457,664 bytes)
Used Space: 1.33 MB (1,400,832 bytes)
Free Space: 55.5 KB (56,832 bytes)

CD-RW:
Manufacturer: Imation
Manufacturer-Reported Capacity: 650 MB
Volume Label: Test
File System: CDFS
Used Space: 1500 bytes
Free Space: 0 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. This test is performed on an IBM T23 Thinkpad laptop with Windows 2000 as the host operating system on the System Disk. The system is configured to boot from the hard drive before looking for other bootable media.

Procedures:
1. Create a hash of the file(s) on the CD-RW using a Linux Boot CD and the md5sum command.
2. Record the hash.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

3. Create a hash of the Floppy Disk using a Linux Boot CD and the md5sum command.
4. Record the hash.
5. Perform write attempts:
    - Attempt to format the attached disk
    - Try to delete files, add files, modify files, copy files, move files, save files
    - Modify properties and attributes of files
    - Use WinHex© to attempt writes to end-of-file slack space
6. With the write blocker on and blocking the attached drive a:, remove the floppy disk from the system while the system is still running.
7. Insert the hashed CD-RW disc into the CD-ROM drive.
8. Using a CD write program (e.g. Roxio direct CD and/or IBM Record Now v3.5)
    - Choose the Make Writable option within DirectCD
    - Choose the Format option within DirectCD
9. Reboot the system with the CD-ROM inserted and retry the tests.
10. Calculate the hash again.
11. Compare the hashes.
12. Remove the CD-RW disk from the system and reinsert the floppy drive with the floppy disk still inserted.
13. Reboot the system with the floppy inserted and retry the floppy disk tests.
14. Calculate the hash again.
15. Compare the hashes.

Expected Result: The MD5 sums calculated after the testing is performed will match the original MD5 sums for the disks.

Actual Result: The sums match. The sums listed show the actual MD5 sums calculated before the testing and after the testing.

Before MD5 hash Floppy: 3436f45abf3758c7abd9337d27fe9e51
After MD5 hash Floppy: 3436f45abf3758c7abd9337d27fe9e51

Before MD5 hash CD-RW: af4d866babe607cd98cf7be51b58582e
After MD5 hash CD-RW: af4d866babe607cd98cf7be51b58582e

© SANS Institute 2003, Author retains full rights.

Test Case: 20) Fire Wire verified - Dell OptiPlex GX 110

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to an IDE drive connected by Fire Wire.

Test Number: Win2kWB.firewire.test.1
Test Environment:
Hardware:
- Dell OptiPlex GX 110
- Processor: Intel Pentium III Processor 733Mhz
- BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
- Adaptec SCSI 2930

Software:
- Operating System: Microsoft Windows 2000 Professional
- Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive 1:
- Manufacturer: Conner
- Model: CFS1275A
- Manufacturer Reported Capacity: 1.28 GB
- Reported Capacity: 1.19 GB
- Unallocated Space: 696 MB

Partition One:
- Type: Primary
- File System: FAT32
- Capacity: 199 MB – 208,666,624 bytes
- Used Space: 138 MB – 145,608,704 bytes
- Free Space: 60.1 MB – 63,057,920 bytes

Partition Two:
- Type: Extended
- File System: FAT
- Capacity: 100 MB – 105,025,536 bytes
- Used Space: 1.03 MB – 1,081,344 bytes
- Free Space: 99.1 MB – 103,944,192 bytes

Partition Three:
- Type: Primary
- File System: NTFS
- Capacity: 180 MB – 188,858,368 bytes
- Used Space: 30.9 MB – 32,426,496 bytes
- Free Space: 149 MB – 156,431,872 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) IDE CD-ROM device 2) Diskette drive and 3) Hard-disk drive C:.

© SANS Institute 2003, Author retains full rights.

3. The test hard drive is attached to an external Pyro 1394 Drive Kit to an Adaptec AFW-4300 FCONN FireWire/1394.

Procedures:

1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the drive:
   - Attempt to format the attached disk
   - Attempt to create, modify, and delete partitions
   - Try to delete files, add files, modify files, copy files, move files, save files
   - Modify properties and attributes of files
   - Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drive
   - Use the WinHex© Modify Data option (under Edit on the toolbar) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data
4. Reboot the system with the drive attached and retry the tests.
5. Compare the hashes.

Expected Result: The MD4 sum calculated after the testing is performed will match the original MD4 sum for the hard drive.

Actual Result: The sums match. The sums listed show the actual MD4sums calculated before the testing and after the testing.
Before MD4sum: 3592a1764510d50e6394a432f7d95fa7
After MD4sum: 3592a1764510d50e6394a432f7d95fa7

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Test Case: 21) Fire Wire verified - Dell OptiPlex GX 110 Maxtor drive

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to an IDE drive connected through Fire Wire.

Test Number: Win2kWB.firewire.test.2
Test Environment:
Hardware:
Dell OptiPlex GX 110
Processor: Intel Pentium III Processor 733Mhz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A05
Adaptec SCSI 2930
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive 1:
Manufacturer: Maxtor
Model: D740X-6L
Manufacturer Reported Capacity: 20 GB
Reported Capacity: 19.1 GB

Partition One:
File System: FAT
Capacity: 1.95 GB – 2,097,897,472 bytes
Used Space: 1.27 GB – 1,372,569,600 bytes
Free Space: 691 MB - 725,327,872 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) IDE CD-ROM device 2) Diskette drive and 3) Hard-disk drive C:.
3. The test hard drive is attached to an external Pyro 1394 Drive Kit to an Adaptec AFW-4300 FCONN FireWire/1394.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts to the drive:
   - Attempt to format the attached disk
   - Attempt to create, modify, and delete partitions
   - Try to delete files, add files, modify files, copy files, move files, and save files
   - Modify properties and attributes of files

© SANS Institute 2003, Author retains full rights.

- Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drive
- Use the WinHex© Modify Data option (under Edit on the toolbar) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data

4. Reboot the system with the drive attached and retry the tests.
5. Compare the hashes.

Expected Result: The MD4 sum calculated after the testing is performed will match the original MD4 sum for the hard drive.

Actual Result: The sums match. The sums listed show the actual MD4sums calculated before the testing and after the testing.
Before MD4sum Conner FireWire: 9D7F5A22DBA2ACF710A5ED79C31E4603
After MD4sum Conner FireWire: 9D7F5A22DBA2ACF710A5ED79C31E4603

© SANS Institute 2003, Author retains full rights.

Test Case: 22) USB external hard drive verified - Dell Dimension 4550

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to an external hard drive connected by an Intel PCI to USB Enhanced Host Controller.

Test Number: Win2kWB.ExternalHardDrive.test.1
Test Environment:
Hardware:
Dell Dimension 4550
Processor: Intel Pentium 4 Processor 2.4 GHz
BIOS version: Phoenix ROM BIOS Plus v. 1.10 A03
Adaptec SCSI 3.10.0
Software:
Operating System: Microsoft Windows 2000 Professional
Version: 5.0.2195 Service Pack 2 Build 2195
Test Drive:
Manufacturer: Maxtor
Model: 6L020J1
Manufacturer-Reported Capacity: 20GB
Reported Capacity: 19595 MB
Unallocated Space: 17.18 GB – 17595 MB

Partition 1:
Type: Primary
Volume Label:
File System: FAT
Capacity: 1.95 GB – 2,097,119,232, bytes
Used Space: 1.03 GB – 1,110671,360 bytes
Free Space: 940 MB – 986,447,872 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C: 2) IDE CD-ROM device and 3) Diskette drive.
3. An Intel PCI to USB Enhanced Host Controller connects the Maxtor external test hard drive to the test system.

Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Perform write attempts:
   - Attempt to format the attached disk
   - Attempt to create, modify and delete partitions
   - Try to delete files, add files, modify files, copy files, move files, save files

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

- Modify properties and attributes of files
- Use WinHex© to attempt writes to end-of-file slack space and unallocated space and to alter data already present on the drive
- Use the WinHex© Modify Data option (under Edit on the toolbar) and Set Disk Parameters option (under Tools -> Disk Tools -> Set Disk Parameters) to change drive data

4. Reboot the system.
5. Calculate the hash again.
6. Compare the hashes.

Expected Result: The MD4 sum calculated after the testing is performed will match the original MD4 sum for the External Hard Drive.

Actual Result: The sums match. The sums listed show the actual MD4sums calculated before the testing and after the testing.
Before MD4sum Maxtor External: 39516987d42212f4ab4a0240db4f880j
After MD4sum Maxtor External: 39516987d42212f4ab4a0240db4f880j

© SANS Institute 2003, Author retains full rights.

Test Case: 23) Partition Magic verified - Dell Dimension 4550

Test Description:
This procedure will verify the ability of the *Write Blocker for Windows 2000* to prevent any writes to an IDE hard drive while using PowerQuest Partition Magic v.7.0.

Test Number: Win2kWB.IDE.PartitionMagic.test.1
Test Environment:
Hardware:
- Dell Dimension 4550
- Processor: Intel Pentium 4 Processor 2.4 GHz
- BIOS version: Phoenix ROM BIOS Plus v. 1.10 A03
- Adaptec SCSI 3.10.0

Software:
- Operating System: Microsoft Windows 2000 Professional
- Version: 5.0.2195 Service Pack 2 Build 2195

Test Drive:
- Manufacturer: Maxtor
- Model: 6L020J1
- Manufacturer-Reported Capacity: 20GB
- Reported Capacity: 19595 MB
- Unallocated Space: 17.18 GB – 17595 MB

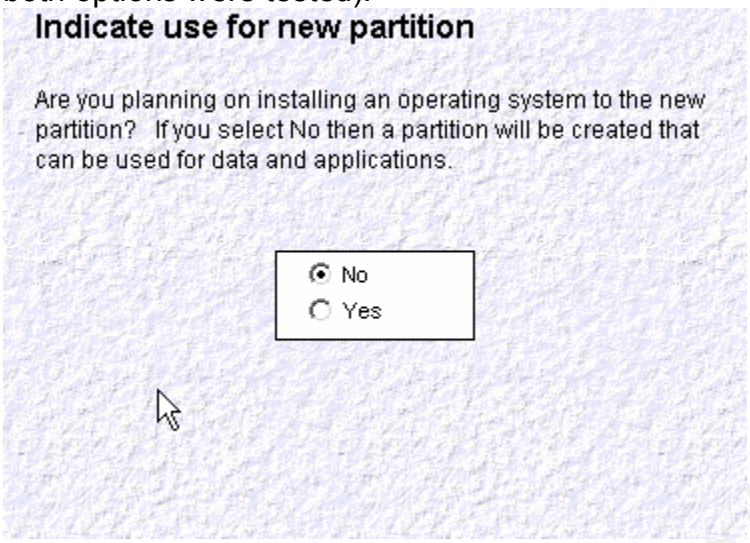
Partition 1:
- Type: Primary
- Volume Label:
- File System: FAT
- Capacity: 1.95 GB – 2,097,119,232, bytes
- Used Space: 1.03 GB – 1,110671,360 bytes
- Free Space: 940 MB – 986,447,872 bytes

Test Notes:
1. The *Write Blocker for Windows 2000* application is installed on the Windows 2000 Boot drive and configured to automatically write block all local and networked storage media.
2. The test system boot sequence is 1) Hard-disk drive C: 2) IDE CD-ROM device and 3) Diskette drive. The test hard drive is in a removable caddy and the bay in which it is inserted is connected to the test system via the secondary IDE channel.
3. PowerQuest Partition Magic v. 7.0 was used for this test.

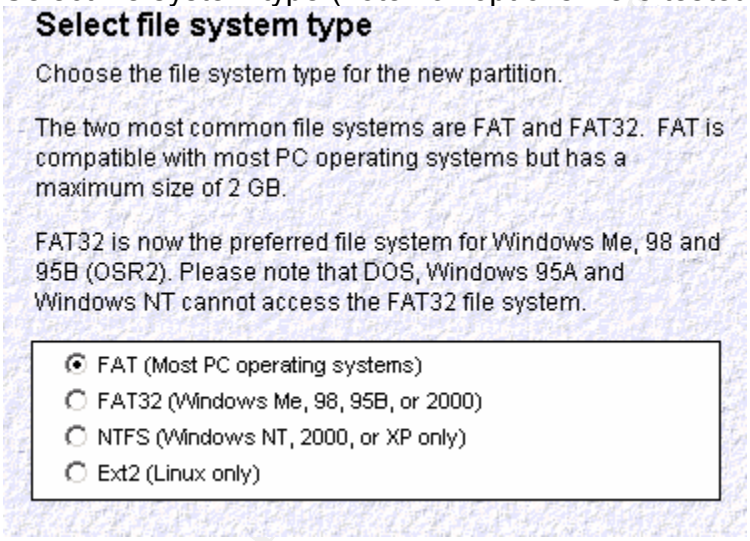
Procedures:
1. Create a hash of the attached media using a Linux Boot CD and the MD5sum command or by using the Ntimage Utility creating an MD4sum.
2. Record the hash.
3. Open Partition Magic.
4. Select the unallocated space and choose create new partition.

© SANS Institute 2003, Author retains full rights.

5. Choose an option for whether or not an operating system will be installed. (note: both options were tested).

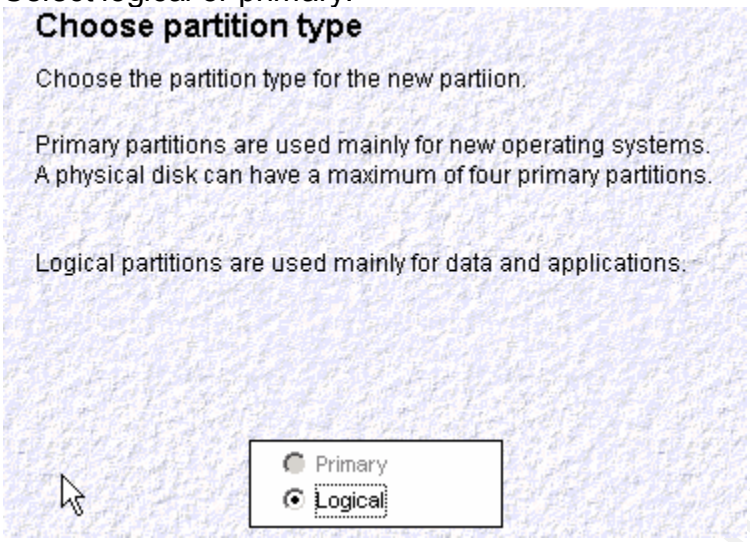


6. Select file system type (note: all options were tested).

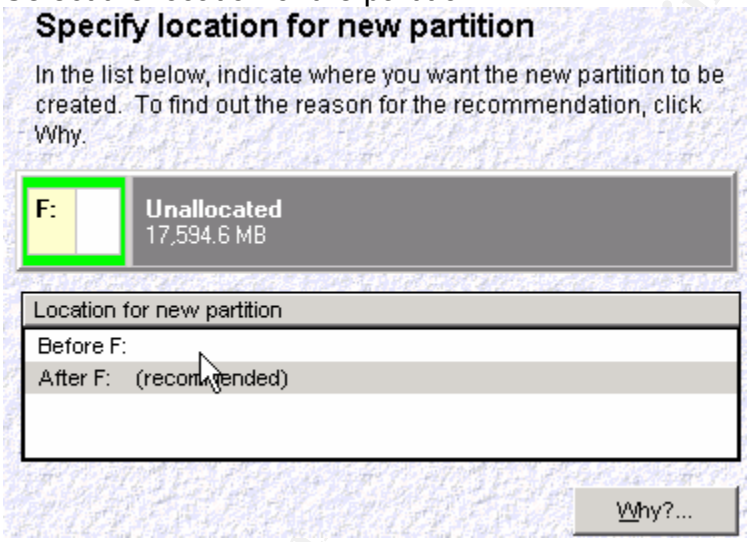


© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

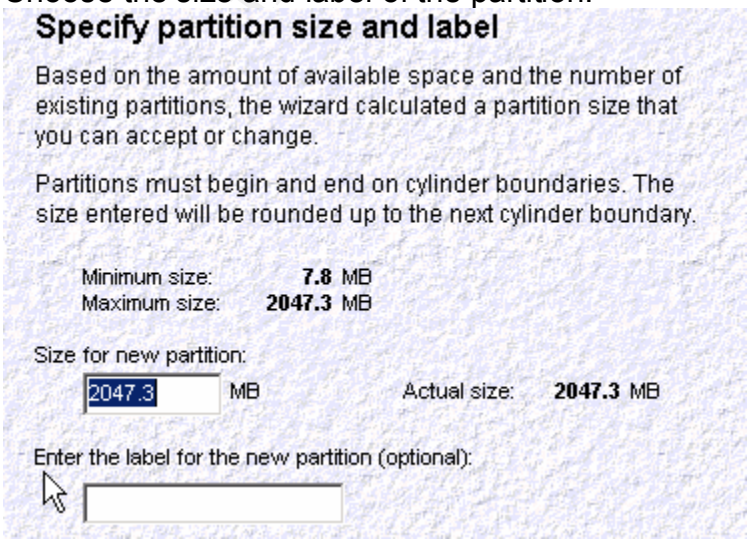
7. Select logical or primary.



8. Select the location of the partition.

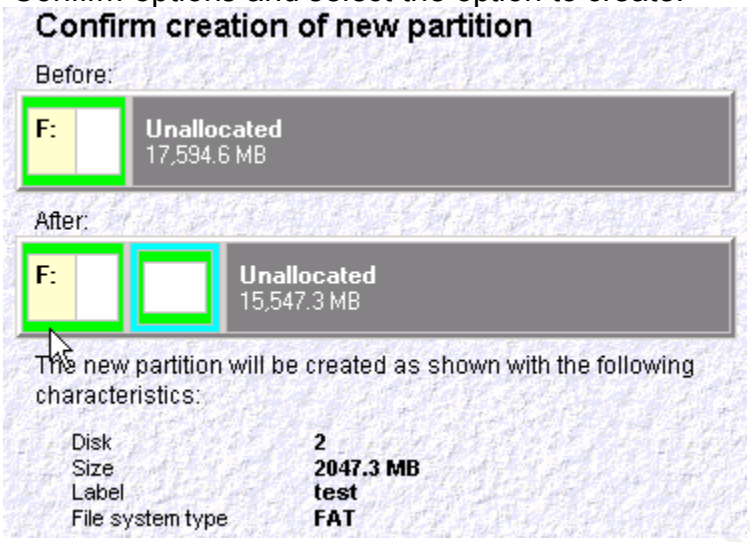


9. Choose the size and label of the partition.



© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

10. Confirm options and select the option to create.



An error is received stating that the partition could not be created.

11. Reboot the system.
12. Calculate the hash again.
13. Compare the hashes.

Expected Result: The MD5 sum calculated after the testing is performed will match the original MD5 sum for the Maxtor Hard Drive.

Actual Result: The sums match. The sums listed show the actual MD5sums calculated before the testing and after the testing.
Before MD5sum Maxtor: 39516987d42212f4ab4a0240db4f880
After MD5sum Maxtor: 39516987d42212f4ab4a0240db4f880

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

VII: Conclusions

As shown, all of the before and after hashes match for each test. As a result, the *Write Blocker for Windows 2000* successfully prevented all attempted changes to the test media and therefore to the sn.dat file. Nothing from the media was removed, deleted, formatted, nor altered in any fashion. The last-accessed times can be difficult to verify, however, they also do not change on the media itself. The last access times will update to the current time every time *properties* is viewed for a file in Windows. In DOS, the command dir /ta will display the accurate last accessed time. While the media is write-blocked, files may be opened on the media and copied from the media and still no writes are made to the media itself. Therefore, the media hash will not change.

The software can be used as the sole write blocker while analyzing media as it protects all local storage media including networked drives. There would be no need for a hardware write blocker to be attached to the system in addition to the software. Hardware write blockers work in the same fashion as the software write blocker examined in this paper in that they prevent write attempts. However, hardware write-blockers are physically attached to the media being analyzed. For instance, an *ACARD*[18] *SCSI to IDE Write Block Kit* has two connectors on one unit, one SCSI connector and IDE connector. If the SCSI connector is attached to the SCSI cable within a system, then the IDE connector must be connected to IDE hard drive and that IDE hard drive is then write blocked. Thus, only one drive is protected via that hardware write-blocker.

An investigator could easily use *Write Blocker for Windows 2000*, properly document all procedures used during an investigation, produce the before and after hashes, and therefore prove that nothing in the evidence has changed. This tool could be used to verify evidence in court case. Explaining how the software works to a jury could be done with simple analogies. For instance, when a videotape of a favorite movie is made, the person may want to make sure that the tape does not get taped over accidentally. So, the person removes the little tab on the tape that prevents the tape from being erased or taped over. The tape can still be watched, paused, rewound, and fast forwarded, but cannot be taped over. The write blocker software has the same premise. Everything on the media that is write-blocked can still be examined, but the actual contents cannot be changed.

Explaining the reliability of the software can be done by explaining how hashes work. If even one bit on a drive were altered, the MD5 or MD4sum would change. "It is conjectured that the difficulty of coming up with two messages having the same message digest is on the order of 2^64 operations, and that the difficulty of coming up with any message having a given message digest is on the order of 2^128 operations.[19]"

---

[18] The ACARD TECHNOLOGY Write Block Kit can be found at http://www.acard.com/eng/. Additional hardware write blockers may be located at http://www.wiebetech.com

[19] (Rivest 1992, 3)

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Most people have heard about DNA analysis being used in court. DNA is similar in premise to an MD5 hash. The chances of two people having the exact DNA are astronomical if not impossible. The same could be said for the MD5 hash. The chances of two files having the same hash are astronomical (but not impossible[20]). Having the same hash before and after the analysis of a system drive or removable media ensures that the media was not altered during the analysis.

Of course, having proper documentation that the hashes were calculated before the evidence was analyzed (and accessed in any fashion) and then again after is necessary. A good lawyer will undoubtedly point out in court that the analyst could have altered the evidence before an original hash was created.

Having an MD5 hash utility built into the write blocker would be of even more assistance to the investigator as it would ensure that the utility creating the sum would not change the evidence in any way.

© SANS Institute 2003, Author retains full rights.

[20] An interesting article on MD5 hashes may be found at http://www.forensics-intl.com/art12.html

Part 3 – Legal Issues

This section describes how I, a system administrator, responded to an incident that recently occurred at the company for which I work, an Internet Service Provider (ISP) that offers services to the paying public.

One of my employer's corporate lawyers contacted me saying that a law enforcement officer would be calling regarding some incident. The lawyer told me to cooperate with the officer to the best of my ability (because my employer, having attempted and repeatedly failed to fight law enforcement and it's attempts to "invade the privacy of our customers" (according to the owner), has decided to cooperate with law enforcement).

The lawyer provided me the name of the officer and assured me that the identity of the officer had been verified.

The officer phoned and informed me that a breach in a government computer was traced back to our system. The officer asked me to verify the activity by reviewing the system logs and to try to determine if the logs reflect whether the activity originated on our system or from another provider. After reviewing the logs I determined a valid user was logged in via dialup during the period of time specified by the officer.

At this point during the initial phone contact I only tell the officer that a valid user was logged in via dialup during the time frame in question.

According to the USA PATRIOT ACT (USAPA)[21], which was signed into law on October 26, 2001, ISP's are allowed to "voluntarily hand over all "non-content" information to law enforcement with no need for any court order or subpoena. sec. 212. Second, it expands the records that the government may seek with a simple subpoena (no court review required) to include records of session times and durations, temporarily assigned network (I.P.) addresses; means and source of payments, including credit card or bank account numbers. secs. 210, 211"[22].

Therefore, I cannot provide any additional information about the account such as the length of time that particular user was on the system, IP address information, and means and source of payment without a subpoena. Also, according the Email and Electronic Communications Privacy Act (ECPA)[23] I cannot inform the officer of the username, real name, and contact information of the subscriber. "It is important to note that, under the ECPA, the ISP cannot voluntarily provide the info to the LEO without

---

21 (USA PATRIOT) Act of 2001 in its entirety. URL http://www.politechbot.com/docs/usa.act.final.102401.html

22 (Electronic Frontier Foundation 2001) URLhttp://www.eff.org/Privacy/Surveillance/Terrorism_militias/20011031_eff_usa_patriot_analysis.html

23 *("ECPA"), 18 U.S.C. §§ 2701-2712*

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

"paper"[24]. "LEO" is a law enforcement officer and "paper" refers to legal documents used by law enforcement. It had appeared that, according to the USA PATRIOT ACT section 212(a)[25], since the subscriber had accepted our user agreement upon acquiring an account with our ISP, that I would be able to provide the name, username, address, and contact information of the user to the officer, but according to the ECPA I could not without a subpoena. Since, I am not a lawyer and am not sure which law I should follow or which would hold up in court, (for instance, would the user agreement hold up in court as "lawful consent"?) I err on the side of caution and provide no further information without proper "paper".

Of course, had the officer provided me with information that would lead me to believe that "there is an emergency involving immediate danger or death or serious physical injury to any person" then I would be required "without delay" to disclose the logs and any communications on our system "to the public" of that particular user.[26]

The officer was slightly dejected, but accepted my answer and told me that I needed to preserve all evidence I had regarding the user in question and that he would be contacting me rather soon.

According to the ECPA "the LEO can require the ISP to safeguard and sequester targeted info pending issuance of "paper" by making general requests to hold data."[27] Therefore, the officer simply needs to request that I safeguard the evidence.

After hanging-up with the officer I contacted the corporate lawyer about the events that took place and informed him that legal documents would be on their way. The lawyer informed me that, the USA PATRIOT ACT states that a simple subpoena (no court review required) can be issued to obtain records of "session times and durations, temporarily assigned network (IP) addresses: means and source of payments, including credit cards…."[28] The lawyer also said that even though it is acceptable that the officer requested I preserve the logs via the phone according to 18 USC 2703(f)[29], the ISP should request a letter stating so for its own protection. In addition, the ISP is required to preserve the logs for a period of 90 days, and the request may be extended for an additional 90 days.

As soon as I get off the phone with the lawyer the officer calls back wanting the logs to be released. The officer says that since the hacker was a legitimate user on the

---

24 (Lewis 2001) URL http://www.ipn.org/programs/programs2001/SpeakerNotes02_2001/Feb13Summary.PDF

25 (USA PATRIOT ACT) URL http://www.cdt.org/security/usapatriot/keyprovisions.pdf

26 (Bernstein (2001) URL www.gigalaw.com/articles/2001-all/bernstein-2001-11-all.html.

27 (Lewis 2001) URL http://www.ipn.org/programs/programs2001/SpeakerNotes02_2001/Feb13Summary.PDF

28 *Ibid*.

29 18 USC 2703(f) http://www.usdoj.gov/criminal/cybercrime/usc2703.htm

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

system, the hacker had to agree to the terms and conditions of use before accepting an account. Therefore, according to the ECPA Title 18, 2702(b)(3)[30], I can voluntarily disclose the system logs.

But, I say, the ECPA also says that I cannot provide this information without "paper". According to the ECPA the officer would need a court order to obtain "transactional information" (i.e. connection and transmission logs, email header info)[31]. So, in order to not violate the law myself, I wait until I receive the court order to release the logs.

While waiting for the appropriate "paper", I continue to do research to find out what else I can do with this case at this time.

The ECPA requires we notify users of our system that we may view electronic communication if necessary. That information is included in our user agreement that the hacker signed when obtaining an account. Since illegal activity occurred from our system, I am able to search previous logs to see what other sessions are captured in the system logs. In addition, I can preserve all stored content, including email and chat files (allowing the disclosure of the emails and chat files would require a search warrant from law enforcement if they wish to obtain that material[32]).

I still would not read any email or chat communications from the individual, as I was not an intended recipient of that communication.

Because of this incident I now realize just how much information system administrators need to know about the law (or, of course, need to know how to contact the company lawyers). Not exactly something we usually focus on. It's a good thing that our system wasn't hacked, but what if it was? What if the account used to access the government system was created on our system illegitimately? The unauthorized access to our system would create new problems. Now there is a security breach and it must be corrected. I would need to identify how the hacker gained access and determine how to patch that access point and when (law enforcement might want to observe the hacker in progress). I would need to identify what files were accessed, damaged, changed, created, and deleted as a result of the intrusion. In addition, meticulous logs would have to be kept about what was found and when, and what, if anything, was done to correct the problems.

---

30 *18 U.S.C. 2702(b)(3),* Electronic Communications Privacy Act.
http://www4.law.cornell.edu/uscode/18/2702.html

31 The ECPA requires a court order be issued in order to obtain transmission logs.
(Lewis 2001) URL http://www.ipn.org/programs/programs2001/SpeakerNotes02_2001/Feb13Summary.PDF

32 The ECPA requires a search warrant be issued in order to obtain Stored Content such as emails and chat files.
(Lewis 2001) URL http://www.ipn.org/programs/programs2001/SpeakerNotes02_2001/Feb13Summary.PDF

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

Also, Virginia Code § 18.2-152.4, Criminal Trespass, makes it illegal for "any person to use a computer or computer network without authority and with the intent to:

> 1. Temporarily or permanently remove, halt, or otherwise disable any computer data, computer programs, or computer software from a computer or computer network;
> 2. Cause a computer to malfunction, regardless of how long the malfunction persists;
> 3. Alter or erase any computer data, computer programs, or computer software…"[33]

Since my employer and our system servers are located in Virginia the Criminal Trespass code would apply, as the hacker would have altered computer data is some fashion in order to create an account on our system that was then used to hack into another system. The unauthorized access on our system constitutes criminal activity, however my actions to provide the law enforcement officer information would not change. I would keep records of every action taken in investigating the incident whether or not the hacker entered our system surreptitiously.

The hacker entered a government system that is a "protected computer" according to the Federal Computer Fraud and Abuse Act[34]. The act requires that "damage" resulted on the system, but since the officer contacted us I am guessing there is proof of damage. The law enforcement officer would still have to provide proper legal documents in order for me to provide detailed information regarding the user and the system logs.

© SANS Institute 2003, Author retains full rights

---

33 (Virginia Computer Crimes Act; Electronic Mail 1999 URL http://leg1.state.va.us/cgi-bin/legp504.exe?991+ful+CHAP0905

34 (Federal Computer Fraud and Abuse Act). URL http://www4.law.cornell.edu/uscode/18/1030.html

Reference List

1) “LOKI2 (the implementation).” Phrack Magazine.
Volume 7, Issue 51 (September 01, 1997): Article 6 of 17.
URL http://www.phrack.com/phrack/51/P51-06

2) Internet Security Systems™, “advICE: Intrusions: 2000112.”
URL http://www.iss.net/security_center/advice/Intrusions/2000112/default.htm

3) “Project Loki.” Phrack Magazine.
Volume 7, Issue 49 (August, 1996): Article 6 of 16.
URL http://www.phrack.org/show.php?p=51&a=6

4) “LOKI2 (the implementation).” Phrack Magazine.
Volume 7, Issue 51 (September 01, 1997): Article 6 of 17.
URL http://www.phrack.org/show.php?p=51&a=6

5) Ibid.

6) Howe, Denis, “Bootstrap Loader.” The Free Online Dictionary of Computing.
(November 27, 1995).
URL http://dictionary.reference.com/search?q=bootstrap%20loader

7) Federal Computer Fraud and Abuse Act. Title 18 sec. 1030. Provided by:
URL http://www4.law.cornell.edu/uscode/18/1030.html

8) Rivest, R., “The MD5 Message-Digest Algorithm. ” Network Working Group Request for Comments: 1321. (April 1992). URL http://www.faqs.org/rfcs/rfc1321.html.

9) Rivest, R., “The MD4 Message-Digest Algorithm. ” Network Working Group Request for Comments: 1186.
(October 1990). URL http://www.faqs.org/rfcs/rfc1186.html

10) Lacy, Jack, “CryptoLib”
URLhttp://www.kanga.nu/~mirror/mirrors/www.homeport.org/%257Eadam/crypto/cryptolib.phtml

11) Rivest, R., “The MD5 Message-Digest Algorithm. ” Network Working Group Request for Comments: 1321. (April 1992): 3. URL
http://www.faqs.org/rfcs/rfc1321.html.

12) Uniting and Strengthening America Act by Providing Appropriate Tools Required to Intercept and Obstruct Terrorism (USA PATRIOT)Act of 2001 (2001).
Listed by: Declan McCullagh’s Politechbot URL
http://www.politechbot.com/docs/usa.act.final.102401.html

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.

13) Electronic Frontier Foundation, "EFF Analysis Of The Provisions Of The USA PATRIOT Act: that Relates to Online Activities." Executive Summary, Item C. (October 31, 2001). URL http://www.eff.org/Privacy/Surveillance/Terrorism_militias/20011031_eff_usa_patriot_analysis.html

14) Electronic Communications Privacy Act ("ECPA"), 18 U.S.C. sec. 2701-2712.
13) Lewis, Amelia J., *"Cyber Crime: Playing on the Side of the Good Guys*." Part 1.2. (February 13, 2001). URL http://www.ipn.org/programs/programs2001/SpeakerNotes02_2001/Feb13Summary.PDF

15) "Uniting and Strengthening America Act by Providing Appropriate Tools Required to Intercept and Obstruct Terrorism (USA PATRIOT) Act of 2001. sec 212(a). Provided by:URL http://www.cdt.org/security/usapatriot/keyprovisions.pdf

16) Bernstein, Debra D., Winer, Johnathon (2001) Business Implications of the U.S. Anti-Terrorism Law. Page 3 URL www.gigalaw.com/articles/2001-all/bernstein-2001-11-all.html.

17) Lewis, Amelia J., *"Cyber Crime: Playing on the Side of the Good Guys*." Part 1.2. (February 13, 2001). URL http://www.ipn.org/programs/programs2001/SpeakerNotes02_2001/Feb13Summary.PDF

18) Ibid.

19) Electronic Communications Privacy Act. U.S.C. 2702(b)(3). URL http://www4.law.cornell.edu/uscode/18/2702.html

20) Lewis, Amelia J., *"Cyber Crime: Playing on the Side of the Good Guys*." Part 1.2. (February 13, 2001). URL http://www.ipn.org/programs/programs2001/SpeakerNotes02_2001/Feb13Summary.PDF

21) Ibid.

22) Virginia. Virginia Computer Crimes Act; Electronic Mail. Virginia Code sec. 18.2-152.4, Chapter 905, [H 1714]. (March 29, 1999) URL http://leg1.state.va.us/cgi-bin/legp504.exe?991+ful+CHAP0905

23) Federal Computer Fraud and Abuse Act. Title 18 sec. 1030. Provided by: URL http://www4.law.cornell.edu/uscode/18/1030.html

24) Requirements for Government Access. Title 18 Sec. 2073. Provided by:

© SANS Institute 2003, Author retains full rights.

http://www.usdoj.gov/criminal/cybercrime/usc2703.htm

© SANS Institute 2003, Author retains full rights.

© SANS Institute 2003, As part of GIAC practical repository. Author retains full rights.