



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 permitted 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>

GIAC Certified Forensic Analyst Practical Assignment

Version 1.4.

**Presented by: Kevin B. Fiscus
January 30, 2004**

© SANS Institute 2004, Author retains full rights.

Kevin Fiscus – GCFA Practical v1.4

1. Abstract / Summary	1
2. Part 1 - Binary Analysis	1
2.1. Situational Overview	1
2.2. Findings Overview	2
2.3. Findings Process and Detail	4
2.3.1. Mounted Image Examination	5
2.3.2. Forensic Image Analysis	18
2.3.3. Legal Implications	22
2.3.4. Suggested Follow-up	25
3. Part 2 - Forensic Tool Validation	28
3.1. Tool Overview	28
3.2. Testing Methodology	30
3.3. Read-Only Testing	31
3.4. Read-Write Access Testing	35
3.5. Viewing Non-Registry Files	37
3.6. Presentation	38
3.7. Summary	40
4. Part 3 - Legal Issues	41
4.1. Question A – Broken Laws	41
4.2. Question B – Contraband on Corporate Systems	42
4.3. Question C – Evidence	44
4.4. Question D – Child Pornography	44
References	46

© SANS Institute 2004, All rights reserved. Author retains full rights.

1. Abstract / Summary

This is a three-part assignment. The first part involves a situation in which an employee is suspected of illegally distributing copyrighted material on a company computer. The only evidence collected was a floppy disk that contains, among others, a binary file of unknown origin or use. The “part 1” assignment (section 2 of this document) involves the analysis of the floppy disk (provided as a zipped image of the actual disk) and more specifically, the binary file located on the floppy disk. The analysis should identify the floppy, tie the floppy disk to the employee in question and locate any other evidence on the disk related to the case.

The second part of the assignment (section 3 of this document) involves validating a tool that could be used to conduct a forensic investigation by proving that the evidence it collects is both “verifiable and repeatable”. In this case, the MiTeC Registry File Viewer is reviewed to assess its use in viewing individual registry files from systems under investigation.

The third and final part of the assignment (section 4 of this document) is an extension of the first part and deals with the legal issues surrounding the part 1 case.

The following document details the approach, methodology and findings as directed by the GIAC Certified Forensic Analyst (GCFA) Practical Assignment, Version 1.4 (July 21, 2003).

2. Part 1 - Binary Analysis

2.1. *Situational Overview*

The following description of the situation was provided as part of the GCFA practical assignment:

An employee, John Price has been suspended from his place of employment when an audit discovered that he was using the organizations computing resources to illegally distribute copyrighted material. Unfortunately, Mr. Price was able to wipe the hard disk of his office PC before investigators could be deployed. However, a single 3.5 inch floppy disk was found in the drive of the PC. Although Mr. Price has subsequently denied that the floppy belonged to him, it was seized and entered into evidence. The floppy disk contains a number of files, including an unknown binary named “prog”. It is suspected that Mr. Price may have had access to other computers in the workplace.

2.2. Findings Overview

An analysis of the contents of the floppy disk revealed that the binary file, “prog” is, in reality a utility called “bmap”. The web site <http://build.inx-bbc.org/packages/fs/bmap.html> contains the following description of the utility:

“The blocksize of a typical file system varies from 1K to 4K. Every file takes at least one block. The unused space in that block is slack space. bmap can save data into this slack space, extract data from slack space, and delete data in slack space. The data cannot be accessed using tools unaware of slack space (ie. almost all other tools), does not change existing files, and therefore cannot be detected using checksums or access times.”

In summary, the bmap utility can hide data on a computer in such a way that it is undetectable without using specialized tools or the utility itself.

Further examination of the floppy drive image revealed several interesting files and directories. In addition to “prog”, the floppy drive contained the following files and directories:

- **/nc-1.10-16.i386.rpm..rpm** – this file is used to install the program “netcat” on a Linux (or similar) system. As stated on the web site http://freshmeat.net/projects/netcat/?topic_id=150, “Netcat is a simple Unix utility which reads and writes data across network connections, using TCP or UDP protocol. It is designed to be a reliable backend tool that can be used directly or easily driven by other programs time, it is a feature-rich network debugging and exploration tool, since it can create almost any kind of connection you would need and has several interesting built-in capabilities.” While this tool has many legitimate uses, it is frequently used by hackers to serve as a “backdoor” providing access to systems that have already been compromised.
- **/~5456g.tmp** – Apparently a temporary file. Part of a binary, compressed or encrypted file. Neither the origin nor the function of this data was able to be determined during the course of this investigation.
- **/Docs/DVD-Playing-HOWTO-html.tar** – an archive file containing a set of html (web) documents describing how to play DVDs on a Linux operating system.
- **/Docs/Kernel-HOWTO-html.tar.gz** – a compressed archive file containing a set of html (web) documents detailing how to modify a Linux operating system kernel.
- **/Docs/Letter.doc** – a blank Microsoft Word template for writing a formal business letter.
- **/Docs/Mikemsg.doc** – a Microsoft Word document containing a message to “Mike” from “JP”.
- **/Docs/MP3-HOWTO-html.tar.gz** – a compressed archive file containing a set of html (web) files that describe the “*hardware, software and procedures needed to encode, play, mix and stream MP3 sound files*”

under Linux.” (This description was taken from the MP3-HOWTO.html file contained within the archive.)

- **/Docs/Sound-HOWTO-html.tar.gz** – a compressed archive file containing a set of html (web) files describing sound support for Linux.
- **/John/sect-num.gif** – an image file illustrating how sectors are numbered on a computer disk.
- **/John/sectors.gif** – an image file illustrating sectors and tracks on a computer disk.
- **/lost+found/** - a directory used to store “bad” or corrupted data.
- **/May03/ebay300.jpg** – an image file that appears to be a picture of a portion of a screen display from the eBay (<http://www.ebay.com>) web site. The screen shot contains no details other than a general message stating that the eBay system is temporarily unavailable.

An analysis of the files contained on the floppy images discovered additional detail both tying the disk to John Price and providing insight into what he was doing and how he was doing it.

Using a “known good” version of the bmap utility, it was discovered that the unused disk space associated with the file /Docs/Sound-HOWTO-html.tar.gz contained the following hidden data:

Ripped MP3s - latest releases:

*www.fileshares.org/
www.convenience-city.net/main/pub/index.htm
emmpeethrees.com/hidden/index.htm
ripped.net/down/secret.htm*

****NOT FOR DISTRIBUTION****

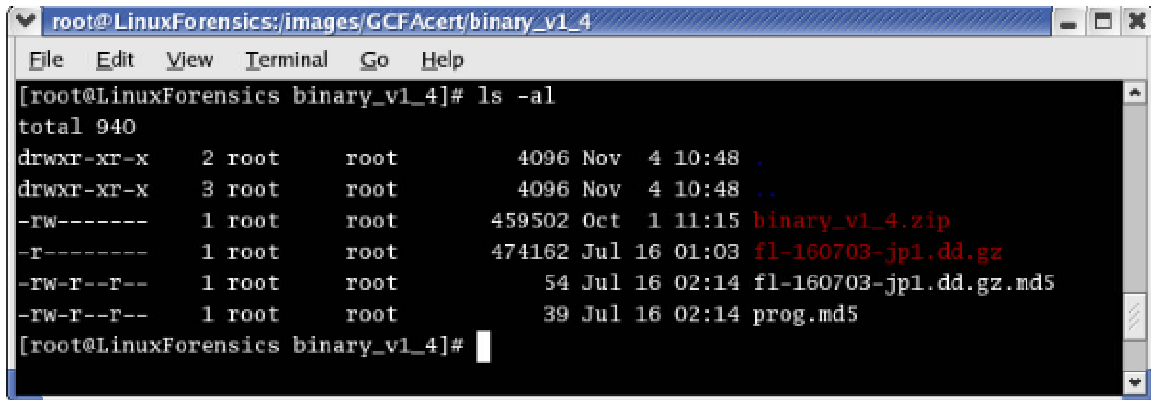
In addition, the directory entitled “John” and the document signed “JP” seem to refer to the suspect, John Price. A more detailed examination of those files more firmly established those ties. Viewing the “Mikemsg.doc” and “Letter.doc” using Microsoft Word found that the properties associated with both documents listed John Price as the author. Examining these documents using a hex editor identified numerous instances of the name “John Price”.

The evidence collected strongly indicates that John Price was responsible for the data on the floppy disk in question. It further indicates that Mr. Price was receiving and then distributing pirated MP3 music files. The locations where pirated files could be acquired were being stored in the slack space of files. The bmap utility, renamed “prog” to hide its identity, was used to hide and retrieve the information. It is likely that the netcat utility was used to send and receive information and/or data providing a “backdoor” communications channel. Bmap (prog) and netcat allowed the information to remain hidden from traditional surveillance or monitoring methods. The content of the majority of the remaining

files on the floppy disk related to playing music and movies on a Linux system further substantiating the case against Mr. Price.

2.3. Findings Process and Detail

To begin my analysis I downloaded the file “binary_v1_4.zip”. After unzipping the file into a directory named “binary_v1_4” I discovered 3 files as seen in the following image:

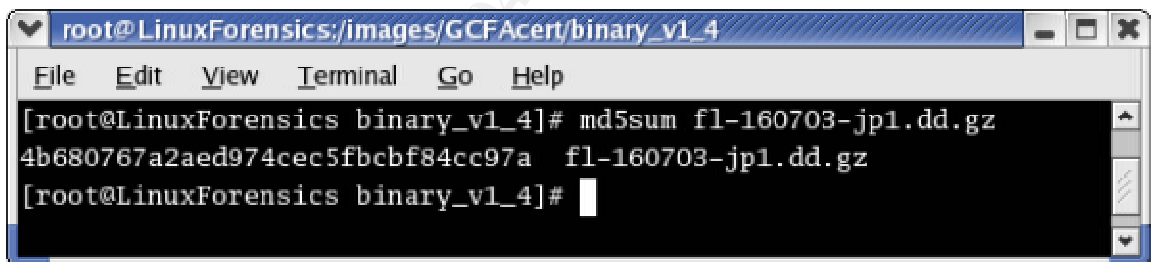


```

root@LinuxForensics:/images/GCFAcert/binary_v1_4
File Edit View Terminal Go Help
[root@LinuxForensics binary_v1_4]# ls -al
total 940
drwxr-xr-x  2 root  root    4096 Nov  4 10:48 .
drwxr-xr-x  3 root  root    4096 Nov  4 10:48 ..
-rw-----  1 root  root  459502 Oct  1 11:15 binary_v1_4.zip
-r-----  1 root  root  474162 Jul 16 01:03 fl-160703-jp1.dd.gz
-rw-r--r--  1 root  root    54 Jul 16 02:14 fl-160703-jp1.dd.gz.md5
-rw-r--r--  1 root  root    39 Jul 16 02:14 prog.md5
[root@LinuxForensics binary_v1_4]#

```

I attempted to verify the integrity of the fl-106703-jp1.dd.gz file by verifying its MD5 hash value against the value provided. The MD5 hash value provided with the file was “4b680767a2aed974cec5fbcfb84cc97a”. Running the md5sum utility against the fl-160703-jp1.dd.gz file resulted in the same value.



```

root@LinuxForensics:/images/GCFAcert/binary_v1_4
File Edit View Terminal Go Help
[root@LinuxForensics binary_v1_4]# md5sum fl-160703-jp1.dd.gz
4b680767a2aed974cec5fbcfb84cc97a  fl-160703-jp1.dd.gz
[root@LinuxForensics binary_v1_4]#

```

As the values match, we can thus be assured that file integrity has been maintained.

My assumption, based on the name of the extracted files was that the file fl-160703-jp1.dd.gz is a compressed version of the image of the floppy drive in question. Running the file command resulted in “gzip compressed data, was “fl-160703-jp1.dd”, from Unix” which verified this assumption. I then uncompressed the file with the gunzip command. Running the file command against the resulting file, fl-160703-jp1.dd indicated that the file was “Linux rev 1.0 ext2 filesystem data”.

I used two methods to work with the data in the image file. I mounted the image in read-only mode and I accessed the image file directly using the Autopsy Forensic Browser utility.

2.3.1. Mounted Image Examination

To begin the investigation I mounted the image file to the /mnt/hack/GCFAcert directory with the “ro”, “loop”, “noexec” and “noatime” options. The “ro” option mounted the image in read-only mode ensuring that no data or files could be written to or otherwise modified. The “loop” option mounted the image as a loopback device (required to mount image files). The “noexec” option does not allow any binary file on the mounted filesystem to be executed. This is helpful when there is the potential that the mounted file system contains malicious code. The “noatime” option ensures that the access time is not modified while reviewing the contents of the mounted image. Mounting the image with this set of options allowed me to analyze the contents of the image file without risking altering or modifying any evidence present.

When reviewing the contents of the mounted floppy drive image I discovered the following files:

- /nc-1.10-16.i386.rpm..rpm
- /prog
- /~5456g.tmp -
- /Docs/DVD-Playing-HOWTO-html.tar
- /Docs/Kernel-HOWTO-html.tar.gz
- /Docs/Letter.doc
- /Docs/Mikemsg.doc
- /Docs/MP3-HOWTO-html.tar.gz
- /Docs/Sound-HOWTO-html.tar.gz
- /John/sect-num.gif
- /John/sectors.gif
- /lost+found/
- /May03/ebay300.jpg

A cursory examination of the files discovered revealed the following:

File Name	Owner / Group	File Command Results	Apparent Description
/nc-1.10-16.i386.rpm..rpm	502 / 502	RPM v3 bin i386 nc-1.10-16	An installable Netcat program
/prog	502 / 502	ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux 2.2.5, statically linked, stripped	A executable binary file of unknown origin or function
./~5456g.tmp -	root / root	data	An unidentified file, probably a temporary file.
/Docs/DVD-Playing-HOWTO-html.tar	502 / 502	POSIX tar archive	An archive file containing a set of html (web) documents describing how to play DVDs on a Linux operating system.
/Docs/Kernel-HOWTO-html.tar.gz	502 / 502	gzip compressed data, was "Kernel-HOWTO-html.tar", from Unix	A compressed archive file containing a set of html (web) documents detailing how to modify an operating system kernel.
/Docs/Letter.doc	502 / 502	Microsoft Office Document	A blank template for writing a formal business letter.
/Docs/Mikemsg.doc	502 / 502	Microsoft Office Document	A document containing a message to "Mike" from "JP".
/Docs/MP3-HOWTO-html.tar.gz	502 / 502	gzip compressed data, was "MP3-HOWTO-html.tar", from Unix	A compressed archive file containing a set of html (web) files that describe the "hardware, software and procedures needed to encode, play, mix and stream MP3 sound files under Linux." (Description taken from the MP3-HOWTO.html file contained within the archive.
/Docs/Sound-HOWTO-html.tar.gz	502 / 502	gzip compressed data, was "Sound-HOWTO-html.tar", from Unix	A compressed archive file containing a set of html (web) files describing sound support for Linux.
/John/sect-num.gif	502 / 502	GIF image data, version 87a, 145 x 145	An image file illustrating how sectors are numbered on a computer disk.
/John/sectors.gif	502 / 502	GIF image data, version 87a, 282 x 131	An image file illustrating what sectors and tracks are on a computer disk.
/lost+found/	root / root	directory	A directory used to store "bad" or corrupted data.
/May03/ebay300.jpg	502 / 502	JPEG image data, JFIF standard 1.01, resolution (DPI), 96 x 96	An image file that appears to be a picture of a portion of a screen display from the eBay web site. The screen shot contains no details other than a general message stating that the eBay system is temporarily unavailable.

The initial examination of the mounted floppy drive image revealed some interesting information. While the root directory of the floppy disk, the "lost+found" directory and the "~5456g.tmp" file had owner and group information relating to "root", all other files and directories had an owner and a group of "502". The root directory and the lost+found directory are created by the system when the drive is formatted and thus an owner/group of root is to be expected. This is also likely the case for the "~5456g.tmp" file as it seems to be a temporary file. The common owner/group of 502 for the remaining files and directories

indicates that these files were all placed on the system by user 502. On a Linux system, the first user number assigned to non-default users is 500. As users are added, these numbers increment by one. While it is not possible to determine who user 502 was without access to the original system, we can state that user 502 was responsible for the placement of all files and directories associated with that owner on the floppy drive. The identification of that individual would require additional information.

Before examining any of the executable files (prog and nc-1.10-16.i386.rpm..rpm) I attempted to gather information from the other files on the system. A review of the two .gif files in the John directory did not provide any significant information however, as the subject of this investigation is named John Price, the presence of a “John” directory should be noted.

I then began to examine the files in the “Docs” directory. I extracted the three “tar.gz” files using the “zxvfps” options of the tar command. They all contained html or web page documents. I also extracted the “.tar” file using the tar command with the “zxvfps” options. It too contained html documents. Each of the four files contained a set of linked html documents that provided a tutorial on the configuration of a Linux system, including kernel modifications, to support playing MP3s and DVDs with sound. The sets of html documents associated with MP3s, DVDs and kernel modifications, when extracted had an owner of “901” and a group of “bin”. The set of html files associated with sound support had an owner of “6050” and a group of “wheel”.

The remaining two files, according to the file command, were Microsoft Office documents. To verify this, I attempted to open both files on a Windows computer with Microsoft Word. The files successfully opened. The file “Letter.doc” appeared to be an empty copy of a default Microsoft Word “contemporary letter” template. The file did not contain any non-default text however, as illustrated in the following image; the properties associated with the letter did contain valuable information.



As shown, the properties associated with the file indicate that the author of this document was “John Price”.

A review of the Mikemsg.doc file revealed a document containing the following text:

Hey Mike,

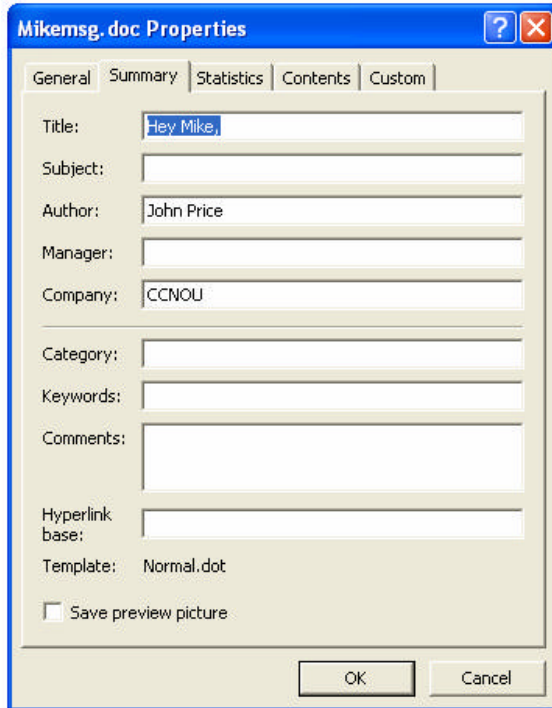
I received the latest batch of files last night and I'm ready to rock-n-roll (ha-ha).

I have some advance orders for the next run. Call me soon.

JP

This message indicates that someone with the initials of “JP” was corresponding with an individual named Mike. JP seems to be taking orders for a product that he receives from Mike. The product apparently consists of a “batch of files”.

Like the previous document, the properties associated with the document reveal additional information.



Like Letter.doc, Mikemsg.doc has an author of “John Price”. In addition, this set of properties has a company, CCNOU, listed.

To gather additional information, I reviewed the two .doc files using GHex, a utility that can view the file in hexadecimal format and display its associated ASCII text. This allowed me to view some of the information that is automatically written to the file by Microsoft Word. This hex review of the Letter.doc file revealed the words “John Price” four separate times. The review of Mikemsg.doc found “John Price” five times. As a result of this review, we can state with a high degree of certainty that these files were created by the subject of this investigation, John Price.

Having tied files on the floppy image to the subject of the investigation I began the detailed investigation of the binary file, “prog”. The file command indicated that “prog” is an ELF 32-bit LSB executable. This is a standard executable binary format for Linux. It also indicated that the file was compiled for an Intel 80383 version of the GNU/Linux 2.2.5 kernel.

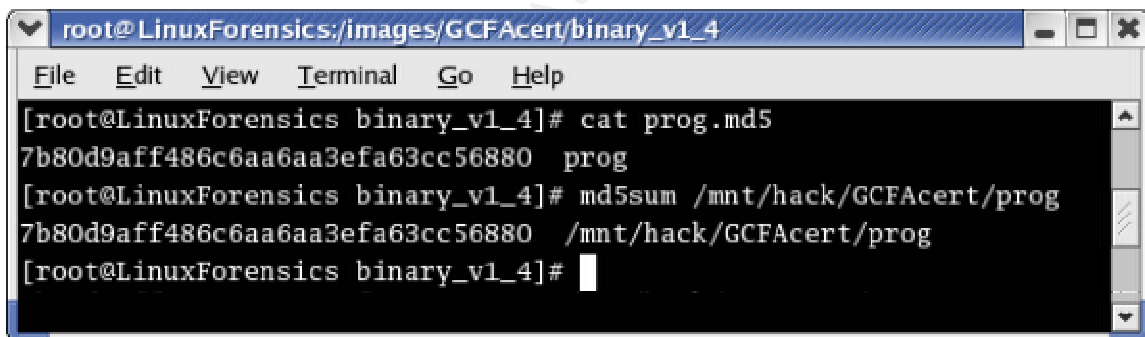
The file has been statically linked. This means that all code libraries required for the program to run have been compiled into the executable file itself. Many executable files are dynamically linked requiring the system on which they are run to provide many of the libraries. If a system is missing any of the required libraries, the program will fail to execute properly. A statically linked file has far fewer dependencies and is thus more portable than a dynamically linked file. This also makes the identification of the file more difficult. When a program is

dynamically linked, its use of dynamically linked libraries (DLLs) can provide hints as to the programs purpose. A statically linked file provides no such clues.

The file has also been “stripped” meaning all symbols have been removed from the program. This decreases the overall size of the program. It also makes debugging more difficult. When an unstripped file is run through a debugger, the debugger displays the names of the functions that are processed. These function names can assist in identifying the purpose of the program. Stripping a program removes these function names. When run through a debugger, the memory locations of the functions are displayed instead of the function names thereby eliminating one potential source of information about the file.

The fact that the program has been statically linked and stripped makes the program easier to distribute and run on a variety of systems as it is both small in size and has few dependencies. This configuration also makes it more difficult to determine the purpose of the program.

When I initially extracted the binary_v1_4.zip file, I was provided with the image file and a file called prog.md5. This file contained a hash value for the prog file. I ran the md5sum program against the mounted prog file and compared the results to the prog.md5 value. As seen in the following image, the results match indicating that the integrity of the prog file has been maintained.



```
root@LinuxForensics:/images/GCFAcert/binary_v1_4
File Edit View Terminal Go Help
[root@LinuxForensics binary_v1_4]# cat prog.md5
7b80d9aff486c6aa6aa3efa63cc56880 prog
[root@LinuxForensics binary_v1_4]# md5sum /mnt/hack/GCFAcert/prog
7b80d9aff486c6aa6aa3efa63cc56880 /mnt/hack/GCFAcert/prog
[root@LinuxForensics binary_v1_4]#
```

In an attempt to determine the function of the prog binary, I ran the strings command against it. The strings command returned 4760 lines of results. Of those, I selected some data that seemed most likely to provide me with a lead to the use or functionality of the file. The following is the list of the stings I selected:

- useless bogus option
- test for fragmentation (returns 0 if file is fragmented)
- checkfrag
- display fragmentation information for the file
- frag
- wipe the file from the raw device
- print number of bytes available
- test (returns 0 if exist)
- wipe
- place data

display data
extract a copy from the raw device
list sector numbers
operation to perform on files
generate SGML invocation info
generate man page and exit
display options and exit
display version and exit
1.0.20 (07/15/03)
newt
use block-list knowledge to perform special operations on files
+45 3325-6543
+45 3122-6543
keld@dkuug.dk
Keld Simonsen
ISO/IEC 14652 i18n FDCC-set
C/o Keld Simonsen, Skt. Jorgens Alle 8, DK-1615 Kobenhavn V

A web search on www.yahoo.com using the majority of these keywords resulted in either uninteresting information or no responses. Upon entering “use block-list knowledge to perform special operations on files”, however, I received the following response:

1. [LWN - Announcements](#) 
... Blur Scope MAX, 1.3, A visualization plug-in for XMMS. bmap, 1.0.16, Use block-list knowledge to perform special operations on files. ...
old.lwn.net/2000/0413/announce.php3-67k - [Cached](#) - [More pages from this site](#)

Following the link provided and then searching the resulting page for “block-list” revealed the following information:

[bmap](#)

It is interesting to note the version number associated with bmap; 1.0.16. A review of the keywords generated from running the strings command against the prog file had an entry of “1.0.20 (07/15/03)”. The apparent similarities in version numbering may indicate a connection between “prog” and “bmap”. Unfortunately, the provided link for “bmap” was dead. A subsequent search on www.yahoo.com for “bmap” revealed a large amount of unrelated information. Narrowing the search criteria to “bmap 1.0.16”, however, resulted in seemingly relevant responses including:

- http://www.scyld.com/pub/forensic_computing/bmap/
- http://www.hacker-archiv.de/page/betriebssysteme/unix_linux/systemprogramme.html
- http://linux4u.jinr.ru/LinuxArchive/Ftp/SCYld/forensic_computing/bmap/

Each of these URLs related to web sites where the bmap utility could be downloaded. In addition to the “1.0.16” version, a “1.0.20” version was also available. I then conducted additional searches for the bmap utility to find a clear definition of its functionality and discovered the web site <http://build.linux-bbc.org/packages/fs/bmap.html>. This site contains the following description of the utility:

“The blocksize of a typical file system varies from 1K to 4K. Every file takes at least one block. The unused space in that block is slack space. bmap can save data into this slack space, extract data from slack space, and delete data in slack space. The data cannot be accessed using tools unaware of slack space (ie. almost all other tools), does not change existing files, and therefore cannot be detected using checksums or access times.”

The bmap utility can hide data on a computer in such a way that it is undetectable without using specialized tools or the bmap utility itself. This utility seemed a likely candidate to be the prog binary as the subject of the investigation was suspected of illegally distributing copyrighted material. To test this hypothesis, I created a controlled test environment using vmWare (<http://www.vmware.com>).

The test environment consisted of a vmWare virtual RedHat 9.0 (<http://www.redhat.com>) computer configured in non-persistent mode. This allows the virtual computer to function as a normal Linux system; however, changes made to the virtual hard drive will not be maintained when the system is restarted. This allows for full and complete testing of the prog binary without risking the ongoing compromise of the virtual computer.

Before conducting any of the testing, I created a script that runs a variety of system functions allowing me to take a “snapshot” of the test environment at various points during the testing. The snapshot script, called “checker” runs the following commands:

© SANS Institute

```

root@localhost:/home/test/check
File Edit View Terminal Go Help
[root@localhost check]# cat checker
uname -a > uname.1
netstat -l > listen.ports.1
netstat -anp > process.to.ports.1
netstat -rn > routing.1
arp -v > arp.1
netstat -i > interfaces.1
ifconfig > ifconfig.1
who > who.1
ps -auxeww > processes.1
top -n 1 -b > top.1
env > env.1
cat /etc/passwd > passwd.1
cat /etc/shadow > shadow.1
last > last.1
df -k > drive.space.1
mount > mounts.1
lsof > lsof.1
du -chb > disk.utilization.1
[root@localhost check]# █

```

After each run of the program, I incremented the name of the output file for each command by 1 thus, after running the program 3 times, I would have an env.1, an env.2 and an env.3 file. Before running prog I ran checker with “.1” output files. After each run of prog, I ran checker with incremented output files.

I downloaded, unzipped and mounted the floppy image on the virtual computer verifying the integrity of the files with the md5sum utility. To attempt to confirm the suspicion that the prog files is actually a statically linked version of the bmap program, I attempted to use the program. Before running the program against the image files, I created a file called “test.1” and filled it with a repeating set of alpha-numeric characters. The repeating set of characters was organized so patterns could be easily detected visually.

I first created an MD5 hash of the test.1 file. I then ran the prog file against the test.1 file using a variety of switches. After each run of the prog file, I verified the integrity of the file using the MD5sum utility. The following illustration shows the results of this process.


```

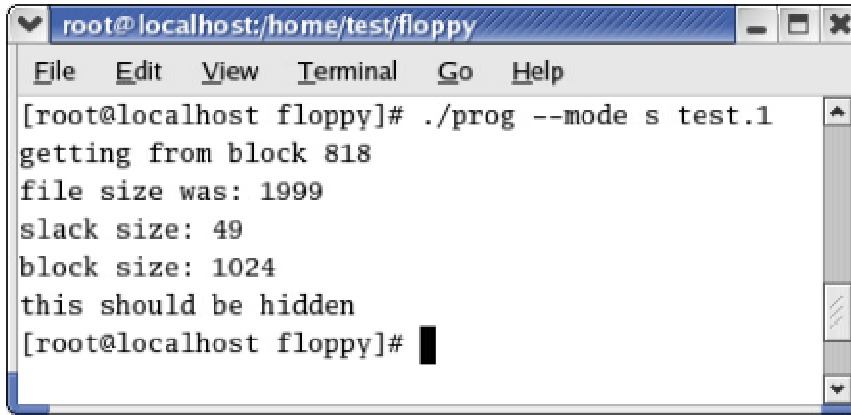
root@localhost:/home/test/floppy
File Edit View Terminal Go Help
[root@localhost floppy]# cat test.1.md5
d601042cb93a1784165184198394cbe4 test.1
[root@localhost floppy]# md5sum test.1
d601042cb93a1784165184198394cbe4 test.1
[root@localhost floppy]# ./prog --mode sb test.1
49
[root@localhost floppy]# cat test.1.md5
d601042cb93a1784165184198394cbe4 test.1
[root@localhost floppy]# md5sum test.1
d601042cb93a1784165184198394cbe4 test.1
[root@localhost floppy]# ./prog --mode m test.1
1634
1635      ----
1636
1637      ----
[root@localhost floppy]# cat test.1.md5
d601042cb93a1784165184198394cbe4 test.1
[root@localhost floppy]# md5sum test.1
d601042cb93a1784165184198394cbe4 test.1
[root@localhost floppy]# ./prog --mode s test.1
getting from block 818
file size was: 1999
slack size: 49
block size: 1024
" % ( + . [root@localhost floppy]#
[root@localhost floppy]# cat test.1.md5
d601042cb93a1784165184198394cbe4 test.1
[root@localhost floppy]# md5sum test.1
d601042cb93a1784165184198394cbe4 test.1
[root@localhost floppy]# █

```

As shown in the above illustration, file.1 seems to have data already in its slack space. As this file was only recently created, this slack space data is likely residual data from a file that previously occupied these disk blocks. To eliminate this data I ran “prog --mode w file.1” and verified that this action did not change the file’s hash value. I then ran prog with the “--mode s” option. The slack space of this file was empty.

To verify that prog could place data in slack space, I executed the following command: *echo “this should be hidden” | ./prog --mode p test.1*

The command executed successfully with no feedback. Running prog --mode s against test.1 however, revealed the following output:



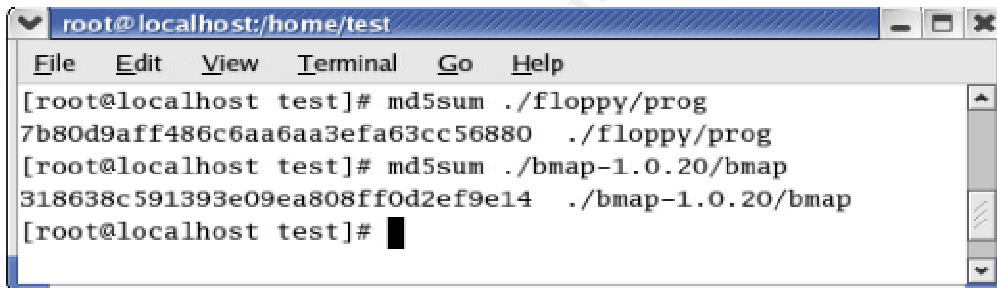
```

root@localhost:/home/test/floppy
File Edit View Terminal Go Help
[root@localhost floppy]# ./prog --mode s test.1
getting from block 818
file size was: 1999
slack size: 49
block size: 1024
this should be hidden
[root@localhost floppy]#

```

The results of this test indicate that the prog executable can successfully read and write to the slack space of files. To verify this binary file is, in reality, a version of the bmap utility, I repeated this testing process using the bmap utility in place of prog. The functionality was identical. I also successfully read data written by bmap with prog and read data written by prog with bmap.

To firmly establish the connection between bmap and prog I attempted to compare the MD5 hashes of the two binaries. The results of this can be seen in the following image:



```

root@localhost:/home/test
File Edit View Terminal Go Help
[root@localhost test]# md5sum ./floppy/prog
7b80d9aff486c6aa6aa3efa63cc56880 ./floppy/prog
[root@localhost test]# md5sum ./bmap-1.0.20/bmap
318638c591393e09ea808ff0d2ef9e14 ./bmap-1.0.20/bmap
[root@localhost test]#

```

As illustrated, the hash values do not match. This is expected. Prog is both statically linked and stripped. The bmap executable is dynamically linked and not stripped. If the bmap file were compiled as a statically linked, stripped file, the values still might not match as there could be differences in the library files or compiler software that would result in hash value differences. Even with the hash value differences, we can safely state that the prog binary is the bmap utility given the similarity of their help displays, their execution and their functionality.

Using the diff command to compare the results of the earliest checker output files (with .1) to the most recent revealed the following results:

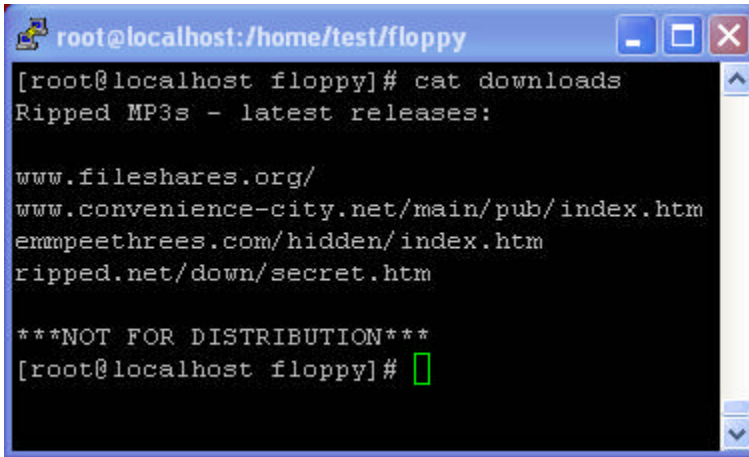
Uname	No Changes
Listen.ports	Changes Due to Reboot
Process.to.ports	Changes Due to Reboot
Routing	No Changes

Arp	No Changes
Interfaces	Changes to Network Statistics
Ifconfig	Changes to Network Statistics
Who	Changes Due to Reboot
Processes	Changes Associated with Normal System Operation
Top	Changes Associated with Normal System Operation
Env	Changes Due to Reboot
Passwd	No Changes
Shadow	No Changes
Last	Changes Due to Reboot
Drive.space	Changes Due to the Addition of Test Files and Data
Mounts	No Changes
Lsof	Changes Associated with Normal System Operation
Disk Utilization	Changes Due to the Addition of Test Files and Data

During the course of testing it was necessary to reboot the test system. These reboots caused discrepancies in a number of the checker files as their data would normally change after a system restart. A review of these changes did not reveal any data inconsistent with normal system operations or the system restart. Based on the results of a review of the checker output files, it does not appear that prog makes any significant modifications to the system other than those changes resulting from its use (i.e. data stored in or removed from swap space). The documentation included with the bmap program corroborates these findings.

After determining the identity and functionality of “prog” I attempted to determine if any of the slack space of the files on the image contained data. I ran the command “./prog --mode s *filename*” where filename was the name of a particular file on the image. This command was run against all of the files and directories found on the image. The slack space of the file ./Docs/Sound-HOWTO-html.tar.gz contained what appeared to be binary data. I then reran prog and redirected the output to a file called “Sound.test”. The effect was the creation of a new binary file. I ran the “file” command against the new file which revealed that the file was “*gzip compressed data, was “downloads”, from Unix*”. To identify the contents of the file I renamed “Sound.test” to “downloads.gz”. I then ran the command “gunzip downloads.gz”. The result was a file called “downloads”. Running the “file” command against the “downloads” file indicated that it was an ASCII text file.

To view the contents of the file, I used the cat command. The results are illustrated in the following image:



```
root@localhost:/home/test/floppy
[root@localhost floppy]# cat downloads
Ripped MP3s - latest releases:

www.fileshares.org/
www.convenience-city.net/main/pub/index.htm
emmpethrees.com/hidden/index.htm
ripped.net/down/secret.htm

***NOT FOR DISTRIBUTION***
[root@localhost floppy]#
```

Using the Sam Spade utility, I did a “whois” lookup on the domain names listed. No matches were found for the Fileshares.org, convenience-city.net and emmpethrees.com entries. The ripped.net domain returned the following information:

```
11/10/03 16:38:02 whois ripped.net
.net is a domain of Network services
Searches for .net can be run at http://www.crsnic.net/

whois -h whois.crsnic.net ripped.net ...
Redirecting to BULKREGISTER, LLC.

whois -h whois.bulkregister.com ripped.net ...

Softline Studios
PO Box 2004
Del Mar, CA 92014
US

Domain Name: RIPPED.NET

Administrative Contact
Loren Stocker: loren@800.net
Softline Studios
PO Box 2004
Del Mar, CA 92014
US
Phone +1.858.792.5000
Fax

Technical Contact
Loren Stocker: loren@800.net
Softline Studios
PO Box 2004
Del Mar, CA 92014
US
Phone +1.858.792.5000
Fax

Record updated date: 2003-06-29 18:36:07
Record created date: 2001-06-30
Record expires on date: 2004-06-30
Database last updated on: 2003-11-10 16:37:52 EST

Domain servers in listed order:

NS1.STEALTHREGISTRY.COM 64.175.161.93
NS2.STEALTHREGISTRY.COM 64.175.161.94
```

Web browsing to the domain (<http://ripped.net>) and the full URL (<http://ripped.net/down/secret.htm>) resulted in redirection to a travel site (1-800-Southbeach.net – <http://www.travelnow.com>).

These dead links seem to have been used for the purpose of this exercise only not relating to actual, existing web sites. If these had been real links, I would have conducted additional investigation to determine what the sites contained, who owned the sites and other data relevant to the investigation.

Finally, I performed an analysis of the `nc-1.10-16.i386.rpm..rpm` file. Using the same “non-persistent” vmWare virtual computer used to test prog and the checker script, I began the testing of the “nc” file. Checker was run first to establish a baseline. Then I ran the file command to verify that the `nc-1.10-16.i386.rpm..rpm` was, indeed an rpm file. Once confirmed, I ran the command `“rpm -i nc-1.10-16.i386.rpm..rpm”`. The file appeared to install correctly with no error messages. I then ran checker for the second time with a “.2” output file extension. To check the installation, I typed the command “nc” and the netcat command prompt was returned. At the command prompt I entered “-l -p 12345”. The command executed successfully. At this time I ran the checker file with a “.3” output file extension. Finally, I used “CTRL+C” to end the program and ran checker using “.4” for the output file extension.

All responses were consistent with the installation and execution of netcat. The results of the checker output files were also consistent with netcat including showing a process listening on port 12345 in the corresponding “.3” checker output files.

2.3.2. Forensic Image Analysis

After reviewing the contents of the floppy image as a mounted file system, I analyzed the image file itself. This was accomplished primarily using three forensics tools, the Autopsy Forensic Browser, Lazarus (part of The Coroner’s Toolkit) and Foremost.

Initial analysis was done using Autopsy. After creating a case I configured the tool to look at the `fl-160703-jp1.dd` file (`/images/GCFAcert/fl-160703-jp1.dd` on my forensics analysis system). The first action I performed using Autopsy was the creation of a MAC timeline for the files on the image. This process created a file (`/forensics/GCFAcert/floppy/output/floppy_timeline`) that provides a chronological representation of the MAC (Modify, Access, Change) times for all files found in the image. The results of this can be seen in the following image:

```

root@LinuxForensics:~/forensics/GCFacert/floppy/output
[root@LinuxForensics output]# cat floppy_timeline
Tue Jan 28 2003 10:56:00 20680 ma. -/fwwxr-xr-x 502 502 25 /mnt/floppy/John/sectors.gif
19088 ma. -/fwwxr-xr-x 502 502 24 /mnt/floppy/John/sect-num.gif
Mon Feb 03 2003 06:08:00 1024 m.. d/drwwxr-xr-x 502 502 12 /mnt/floppy/John
Sat May 03 2003 05:10:00 1024 m.. d/drwwxr-xr-x 502 502 14 /mnt/floppy/May03
Wed May 21 2003 05:09:00 29184 ma. -/fwwxr-xr-x 502 502 13 /mnt/floppy/Docs/DVD-Playing-HOWTO-html.tar
27430 ma. -/fwwxr-xr-x 502 502 19 /mnt/floppy/Docs/Kernel-HOWTO-html.tar.gz
Wed May 21 2003 05:12:00 32661 ma. -/fwwxr-xr-x 502 502 20 /mnt/floppy/Docs/MP3-HOWTO-html.tar.gz
Wed Jun 11 2003 08:09:00 29696 ma. -/fww----- 502 502 16 /mnt/floppy/Docs/Letter.doc
Mon Jul 14 2003 09:08:09 12288 m.c d/drwx----- 0 0 11 /mnt/floppy/lost+found
0 mac ----- 0 0 1 <fl-160703-jp1.dd-alive-1>
Mon Jul 14 2003 09:11:50 26843 ma. -/fwwxr-xr-x 502 502 21 /mnt/floppy/Docs/Sound-HOWTO-html.tar.gz
Mon Jul 14 2003 09:12:02 56950 ma. -/fwwxr-xr-x 502 502 22 /mnt/floppy/nc-1.10-16.i386.rpm..rpm
Mon Jul 14 2003 09:12:15 100430 ma. -rwxr-xr-x 0 0 23 <fl-160703-jp1.dd-dead-23>
Mon Jul 14 2003 09:12:48 13487 ma. -/fwwxr-xr-x 502 502 26 /mnt/floppy/May03/ebay300.jpg
Mon Jul 14 2003 09:13:13 546116 m.. -rwxr-xr-x 502 502 27 <fl-160703-jp1.dd-dead-27>
Mon Jul 14 2003 09:13:52 2592 m.c -/fwr-r-r-- 0 0 28 /mnt/floppy/~.5456g.tmp
Mon Jul 14 2003 09:19:13 100430 .c -rwxr-xr-x 0 0 23 <fl-160703-jp1.dd-dead-23>
Mon Jul 14 2003 09:22:36 1024 m.. d/drwwxr-xr-x 502 502 15 /mnt/floppy/Docs
Mon Jul 14 2003 09:24:00 487476 m.. -/fwwxr-xr-x 502 502 18 /mnt/floppy/prog
Mon Jul 14 2003 09:43:44 1024 .c d/drwwxr-xr-x 502 502 15 /mnt/floppy/Docs
26843 .c -/fwwxr-xr-x 502 502 21 /mnt/floppy/Docs/Sound-HOWTO-html.tar.gz
13487 .c -/fwwxr-xr-x 502 502 26 /mnt/floppy/May03/ebay300.jpg
Mon Jul 14 2003 09:43:57 56950 .c -/fwwxr-xr-x 502 502 22 /mnt/floppy/nc-1.10-16.i386.rpm..rpm
Mon Jul 14 2003 09:45:48 29184 .c -/fwwxr-xr-x 502 502 13 /mnt/floppy/Docs/DVD-Playing-HOWTO-html.tar
Mon Jul 14 2003 09:46:00 27430 .c -/fwwxr-xr-x 502 502 19 /mnt/floppy/Docs/Kernel-HOWTO-html.tar.gz
Mon Jul 14 2003 09:46:07 32661 .c -/fwwxr-xr-x 502 502 20 /mnt/floppy/Docs/MP3-HOWTO-html.tar.gz
Mon Jul 14 2003 09:47:10 546116 .a. -rwxr-xr-x 502 502 27 <fl-160703-jp1.dd-dead-27>
Mon Jul 14 2003 09:47:57 29696 .c -/fwr----- 502 502 16 /mnt/floppy/Docs/Letter.doc
Mon Jul 14 2003 09:48:15 19456 mac -/fwr----- 502 502 17 /mnt/floppy/Docs/Mikemesg.doc
Mon Jul 14 2003 09:48:53 19088 .c -/fwwxr-xr-x 502 502 24 /mnt/floppy/John/sect-num.gif
Mon Jul 14 2003 09:48:53 20680 .c -/fwwxr-xr-x 502 502 25 /mnt/floppy/John/sectors.gif
Mon Jul 14 2003 09:49:25 1024 .c d/drwwxr-xr-x 502 502 12 /mnt/floppy/John
Mon Jul 14 2003 09:50:15 1024 .c d/drwwxr-xr-x 502 502 14 /mnt/floppy/May03
Wed Jul 16 2003 01:03:00 546116 .c -rwxr-xr-x 502 502 27 <fl-160703-jp1.dd-dead-27>
Wed Jul 16 2003 01:03:13 1024 m.c -/drwwxr-xr-x 0 0 2 /mnt/floppy/John/ (deleted-realloc)
Wed Jul 16 2003 01:05:33 487476 .c -/fwwxr-xr-x 502 502 18 /mnt/floppy/prog
Wed Jul 16 2003 01:06:15 12288 .a. d/drwx----- 0 0 11 /mnt/floppy/lost+found
Wed Jul 16 2003 01:09:35 1024 .a. d/drwwxr-xr-x 502 502 12 /mnt/floppy/John
Wed Jul 16 2003 01:09:49 1024 .a. d/drwwxr-xr-x 502 502 14 /mnt/floppy/May03
Wed Jul 16 2003 01:10:01 1024 .a. d/drwwxr-xr-x 502 502 15 /mnt/floppy/Docs
Wed Jul 16 2003 01:11:36 2592 .a. -/fwr-r-r-- 0 0 28 /mnt/floppy/~.5456g.tmp
Wed Jul 16 2003 01:12:39 1024 .a. -/drwwxr-xr-x 0 0 2 /mnt/floppy/John/ (deleted-realloc)
Wed Jul 16 2003 01:12:45 487476 .a. -/fwwxr-xr-x 502 502 18 /mnt/floppy/prog
[root@LinuxForensics output]#

```

As seen in the timeline, the prog file was modified on July 14, 2003 at 9:24. This is likely the time the file was placed on the floppy disk. It was then changed on July 16, 2003 at 1:04. This is likely the time the file name was changed. Finally the file was accessed on July 16, 2003 at 1:12. As the file was only accessed at this time, this is likely the time that the file was last executed.

In addition, the timeline data reveals a large amount of additional information. The floppy disk was created on July 14, 2003 at 9:08. All of the files contained in the image were placed on the floppy disk between 9:08 and 9:48 on the 14th of July. Interestingly, the file “Mikemesg.doc” was modified, accessed and changed at 9:48 and was the last file added to the floppy disk. The fact that much of the activity seen on image files occurs within minutes of the placement of a file tied to John Price strongly indicates that he was responsible for the data found on the floppy disk.

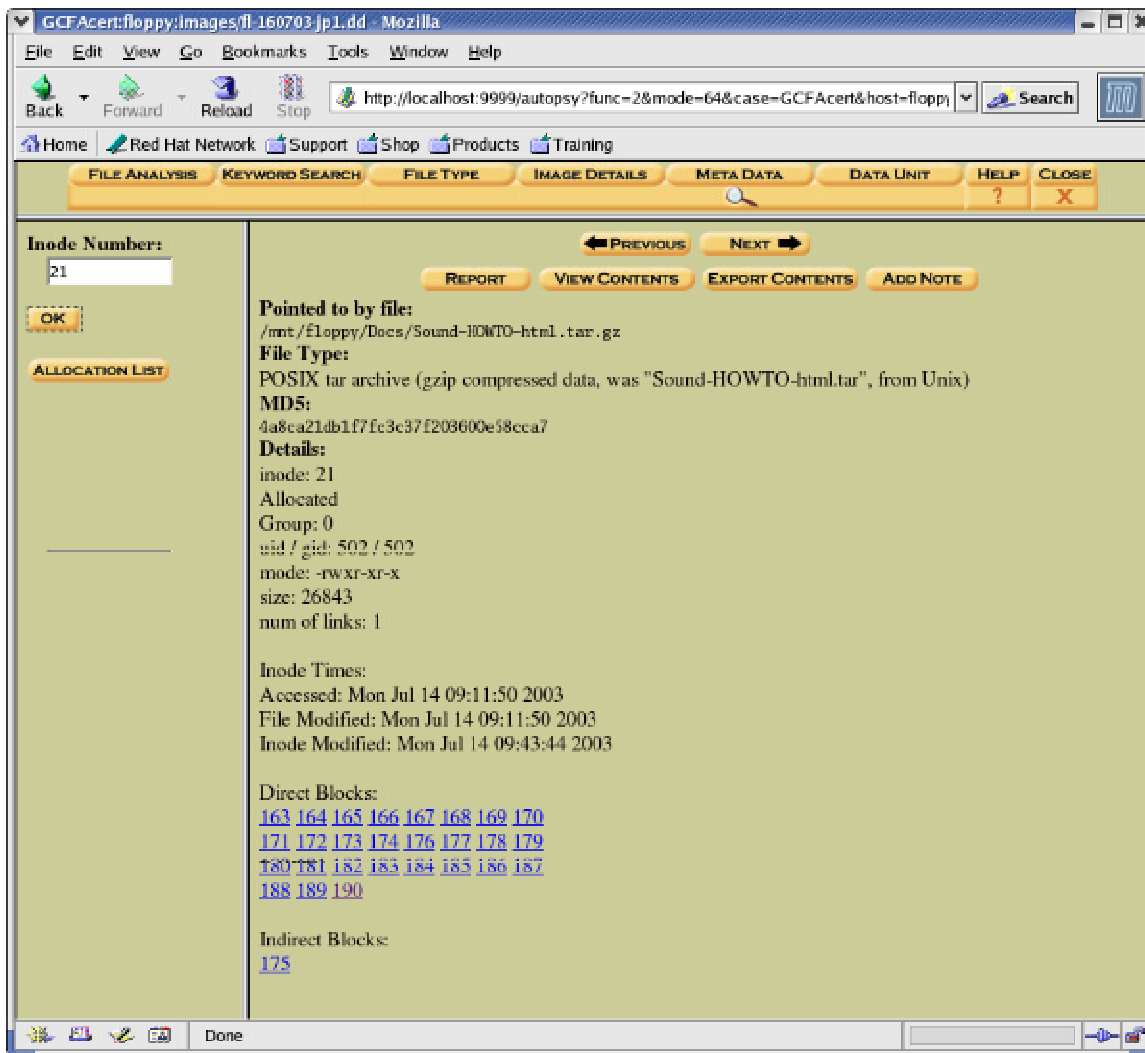
The timeline analysis reinforces information collected during the review if the mounted image. The owner and group for most of the files on the image were listed as “502”. This number relates to information stored in the /etc/passwd and /etc/group files on the original Linux computer from which the floppy was created. As a result, without that original information, this number cannot be tied back to a specific user or group. The timeline file confirmed the file size as 487,476 bytes.

As discussed earlier in this document, the timeline indicated that the last access time on the prog file was July 16, 2003 at 1:12. At this time, the access time was updated but the modified and change times were not. This can occur if the file is viewed (i.e. using cat, strings, etc.) or, in the case of an executable, if the file is executed. The prog file, as a binary, reveals no interesting detail when viewed using cat or strings. As the review of the mounted imaged discovered data loaded in slack space, it can be reasonably concluded that the file was executed at 1:12 on July 16, 2003.

I then began an analysis of the floppy image using the Autopsy forensics browser. Having set up the case when creating the timeline, I opened the floppy image and did a key word search on “download”. This word was specifically selected to find the data hidden in the swap space. When the swap space data was initially discovered and exported, the resulting file was determined to be a gzip compressed data. Furthermore, the file command indicated that the file “was downloads”. For this information to be revealed using the file command, “downloads” must be in clear, uncompressed text.

A search on “download” provided 5 “hits” in four files. Three of the files contained instances of “download” that were expected. One of the hits included the word “downloads” surrounded by what appeared to be binary data. This data was found on fragment 190 of the floppy image. Using Autopsy, I displayed the data unit information for fragment 190. The display matched the results of the keyword search. I then displayed the inode information associated with fragment 190. Autopsy revealed that fragment 190 is associated with inode 21. The following image shows the details revealed for this inode.

© SANS Institute 2004



As displayed, the data for inode 21 is stored in data blocks 163 through 190. It is important to note that the last data block for the file is block 190. This is also the block where the text “downloads” from the key word search was found. This further validates the assumption that “downloads” was, in fact, data stored in the slack space of a file as slack space exist only in the last block of a file.

Again using Autopsy, I then exported data block 190 to a file on my forensic analysis system. Once exported, running the file command identified the newly created file only as “data”. The file associated with inode 21 is Sound-HOWTO-html.tar.gz. The data presumably stored in the slack space is also a gzip file making the identification of where one file ends and the other begins more difficult. To discover what the beginning of a gzip file looks like, I created a test file filled with garbage data and compressed it using gzip. I then opened the file using a hex editor program. The first hex characters of the gzip file were “1F 8B”. I then opened the block 190 file using the hex editor and performed a search on “1F 8B”. That specific set of hex characters was discovered shortly before the “downloads” string. Using the hex editor, I deleted all of the characters leading

up to “1F 8B”. I then saved the file as “test.gz”. Running the file command against test.gz identified the file as “gzip compressed data, was ‘downloads’, from Unix”. This result matches the results of the file command when run against the data discovered in slack space using both bmap and prog. Uncompressing test.gz and running the cat command against the resulting “test” file revealed data identical to that discovered in the slack space file.

This process of testing confirmed that hidden data was indeed stored on the floppy image. The ability to access the data with a utility other than bmap or prog in a predictable location verified the functionality of those utilities. Furthermore, this testing confirmed that the data contained in slack space was not residual data. Data blocks are filled sequentially from beginning to end before the next available block is populated. Therefore, residual data would be comprised of data from the middle or the end of a file and not from the beginning as it is not possible for a file to begin in the middle of a data block. The fact that the slack space data contained a complete gzip file including its initial characters prove that the data was intentionally placed in the slack space of the Sound-HOWTO-html.tar.gz file after that file was placed on the floppy disk.

A review of the image file using the Foremost and Lazarus utilities revealed much of the same information discovered during the mounted image review, the timeline analysis and the Autopsy review.

2.3.3. Legal Implications

The executing of the prog file does not violate any laws as it simply stores data in a location that is both difficult to detect and difficult to access. This program, like any tool, can be put to uses both good and bad. If the program is used to hide information relating to the commission of a crime or is used to conceal contraband, its use may, indirectly, violate laws or regulations. In this case, prog was used to conceal links to web sites appearing to contain copies of MP3 files in violation of copyright laws. As such, prog was used to facilitate the illegal distribution of copyrighted material. Many of the files on the floppy support this conclusion.

According to United States copyright law, all music and recordings of music are by default, copyrighted. There is no requirement for the placement of any copyright notice or symbol or for any formal registration of the copyright. Therefore, the distribution of “ripped” MP3s violates copyright law.

There are four laws affecting copyrights of music. The following excerpt from the Recording Industry Association of America (RIAA) web site (<http://www.riaa.com/issues/copyright/laws.asp>), outlines these:

U.S. Copyright Law {Title 17 U.S.C. Section 101 et seq., Title 18 U.S.C. Section 2319} Federal law protects copyright owners from the unauthorized reproduction, adaptation, performance, display or distribution of copyright protected works.

Penalties for copyright infringement differ in civil and criminal cases. Civil remedies are

Kevin Fiscus – GCFA Practical v1.4

generally available for any act of infringement without regard to the intention or knowledge of the defendant, or harm to the copyright owner. Criminal penalties are available for intentional acts undertaken for purposes of "commercial advantage" or "private financial gain." "Private financial gain" includes the possibility of financial loss to the copyright holder as well as traditional "gain" by the defendant.

Where the infringing activity is for commercial advantage or private financial gain, sound recording infringements can be punishable by up to five years in prison and \$250,000 in fines. Repeat offenders can be imprisoned for up to 10 years. Violators can also be held civilly liable for actual damages, lost profits, or statutory damages up to \$150,000 per work.

The Federal Anti-Bootleg Statute {18 USC 2319A} prohibits the unauthorized recording, manufacture, distribution, or trafficking in sound recordings or videos of artists' live musical performances. *Violators can be punished with up to 5 years in prison and \$250,000 in fines.*

Two important legal concepts, especially pertaining to the Internet, should be kept in mind—contributory infringement and vicarious liability.

Contributory infringement may be found where a person, with knowledge of the infringing activity, induces, causes, or materially contributes to the infringing conduct of another. For example, a link site operator may be liable for contributory infringement by knowingly linking to infringing files.

Vicarious liability may be imposed where an entity or person has the right and ability to control the activities of the direct infringer and also receives a financial benefit from the infringing activities. Vicarious liability may be imposed even if the entity is unaware of the infringing activities. In the case of a site retransmitting infringing programs, providing direct access to infringing works may show a right and ability to control the activities of the direct infringer, and receiving revenue from banner ads or e-commerce on the site may be evidence of a financial benefit.

Fair Use Doctrine {USC Title 17, Sec 107} The "fair use doctrine" of federal law is a complicated area. Basically, it limits the extent of property interest granted to the copyright holder. For example, this might allow citizens to cite a quotation from copyrighted material when the excerpt is used for teaching, research, news reporting, comment, criticism or parody.

There are some limitations. Whether the court allows you to reproduce, distribute, adapt, display and/or perform copyrighted works depends upon the nature of the use (commercial purposes, non-profit, educational), the length of the excerpt, how distinctive the original work is, and how the use will impact the market for the original work.

Generally speaking, one is not allowed to take the "value" of a song without permission, and sometimes that value is found even in a three-second clip. When in doubt, it is always wise to check with the copyright owner, because in many cases even a small clip of a song may not be "fair use."

The Sonny Bono Copyright Term Extension Act This law extends U.S. copyright from life of the author plus 50 years, to life of the author plus 70 years. For "works made for hire," the term is extended from 75 to 95 years. This law should end the discrimination against U.S. works abroad, where countries applied a copyright to U.S. works which resulted in American creators receiving less protection than their foreign counterparts.

No Electronic Theft Law (NET Act) sets forth that sound recording infringements (including by digital means) can be criminally prosecuted even where no monetary profit or commercial gain is derived from the infringing activity. *Punishment in such instances includes up to 3 years in prison and/or \$250,000 fines.* The NET Act also extends the criminal statute of limitations for copyright infringement from 3 to 5 years.

Additionally, the NET Act amended the definition of "commercial advantage or private financial gain" to include the receipt (or expectation of receipt) of anything of value, including receipt of other copyrighted works (as in MP3 trading). Punishment in such instances includes up to 5 years in prison and/or \$250,000 fines. *Individuals may also be civilly liable, regardless of whether the activity is for profit, for actual damages or lost profits, or for statutory damages up to \$150,000 per work infringed.*

As stated in the RIAA web site excerpt, if convicted of violating copyright laws, Mr. Price is subject to both civil and criminal actions. If the distribution was done for "commercial advantage" or "private financial gain" the punishment could include up to 5 years in prison and up to \$250,000 in fines. If the distribution involved no monetary or commercial gain, the punishment could involve up to 3 years in prison and up to \$250,000 in fines. Civil liability could be up to \$150,000 per copyrighted work.

Additional information about United States copyright law can be found at the following:

US Code – Title 17 – Copyrights

<http://clea.wipo.int/clea/lpext.dll?f=templates&fn=main-h.htm&2.0>

Title 37, Code of Federal Regulations – Patents, Trademarks and Copyrights

<http://clea.wipo.int/clea/lpext.dll?f=templates&fn=main-h.htm&2.0>

The Federal Anti-Bootleg Statute

<http://www.usdoj.gov/criminal/cybercrime/18usc2319A.htm>

Sonny Bono Copyright Term Extension Act

<http://www.techlawjournal.com/courts/eldritch/pl1105-298.htm>

No Electronic Theft Act (NET Act)

<http://www.gseis.ucla.edu/iclp/hr2265.html>

The Copyright Law Center

<http://www.copyright-laws.com>

2.3.4. Suggested Follow-up

The evidence collected from the floppy image indicates that John Price was illegally distributing “ripped” MP3 files in violation of copyright laws. This was being accomplished with the assistance of an unknown individual named “Mike”. We can assume that Mike was providing John with the links to web resources where the MP3 files could be found. John was storing that information in the slack space on the discovered floppy drive. The letter “Mikemsg.doc” also indicates that John was taking orders for specific MP3 requests. These orders may have come from other employees. The presence of the netcat rpm file on the floppy image provides a key as to how the distribution of the MP3’s or links to the MP3’s may have been performed. John may have distributed and/or installed netcat on the computer of those interested in illegal MP3’s. Netcat provides the capability to transfer data over a network using any selected port. This provides a method for transmitting data that is difficult to detect and that does not generate log records.

To determine if other employees were involved in these activities, organization administrations should take the following actions:

- Review network traffic logs (if present) to detect anomalous traffic between John Price’s computer and other systems on the network. This could include traffic on unusual ports or traffic on “normal” ports that would not be expected from one workstation to another (i.e. web traffic – port 80 or 443, DNS – port 53, etc.)
- Port scan organizational workstations looking for suspicious open ports (see previous bullet item)
- Scan suspect computers for files with an MD5 hash value equal to the MD5 hash of the prog file.
- Scan suspect computers for files with an MD5 hash value equal to the MD5 hash of the netcat executable.
- Review processes running on suspect computers for netcat
- Review process to listening port mappings looking for unusual processes listening on unusual ports
- Review network traffic looking for web access to the sites listed in the file found in the slack space of the floppy

If other organization computers are identified that may be involved in the trafficking of MP3s (or other illegal material), the administrators should seize the system as quickly as possible to reduce the risk of the user wiping the hard drive thereby deleting evidence. At a minimum, the hard drives of suspect systems should be imaged and the images should be stored in a secure location for forensic analysis.

Based on the information collected, John Price should be questioned. The goal of the questioning is threefold.

1. Get Mr. Price to admit exactly what he was doing and how he did it
2. Identify the “Mike” from Mikemsg.doc.
3. Discover the identity of any other individuals involved in the requesting, illegal distribution or receipt of copyrighted material.

If I was conducting the interview I would ask the following questions:

Question 1:

The questions asked should be as follows:

Ask Mr. Price if he wrote or edited the Mikemsg.doc file.

If Mr. Price denies writing the message, point out the “JP” signature, show him a printout of the message properties with his name highlighted and show him a printout of the hexadecimal representation of the document with all instances of his name highlighted. Inform Mr. Price that Microsoft Word automatically records data about the author of messages and of each person who modified them and re-ask the first question.

If Mr. Price continues to deny writing Mikemsg.doc, show him the same detail (document properties and hexadecimal representation) of Letter.doc. Indicate that this file was found with Mikemsg.doc and re-ask the question.

Question 2:

Present Mr. Price with the contents of the html files contained in the archives in the “Docs” directory. Ask Mr. Price if he is familiar with these documents. If he indicates that he is, ask him what he was using the tutorials for.

Question 3:

Show Mr. Price the data discovered in the slack space on the floppy drive. Ask him if he is familiar with the information?

Question 4:

Ask Mr. Price is he is familiar with the program “netcat”.

Question 5:

Ask Mr. Price how his hard drive came to be erased in a manner that precludes forensic review and why, if it was erased accidentally, he did not immediately contact technical support or his manager.

Question 6:

Ask Mr. Price how with his hard drive erased, a floppy disk was found in his computer floppy drive containing:

- Documents that indicate he created them
- A directory called “John”
- Instructional files describing how to play MP3’s on a Linux system
- A letter to “Mike” signed “JP”
- A copy of netcat
- A program that hides data in slack space
- A file containing links to “ripped” MP3s hidden in slack space of one of the instructional files

If Mr. Price has not yet confessed, I would indicate that we are not interested in him but are interested in finding out who Mike is and furthermore, discovering the identity of anyone else involved. I would inform Mr. Price that we may be able to avoid turning the matter over to the FBI if he cooperates.

Once Mr. Price has confessed, I would attempt to discover the identity of all other individuals involved with the following questions:

- Mikemsg.doc talked about taking orders. How were these orders placed?
- What form did the orders come in?
- How did you receive them?
- Once you got them (the orders), what did you do with them?
- Who is Mike?
- What did he do for you?
- How did you receive information from Mike?
- How did you get information to Mike?
- Who was placing orders?
- How did you distribute product to those placing orders

Finally, Mr. Price’s organization should review its set of information security and acceptable use policies to ensure they provide adequate protection and direction. The policies should specifically and directly state that the acquisition, possession and distribution of copyrighted material in violation of applicable laws is expressly prohibited. In addition, the organization should state that the use of “hacker” utilities is prohibited unless such use has been specifically allowed to perform a legitimate business function.

3. Part 2 - Forensic Tool Validation

3.1. Tool Overview

MiTeC Registry File Viewer (RFV) is a “viewer for standalone files containing Windows registry hives of all MS Windows platforms except MS Windows 3.x”. These files contain the registry information for Windows 95/98/ME, Windows 2000, Windows XP and Windows 2003 operating systems. Together, the files make up the registry.

In the Microsoft Windows operating systems beginning with Windows 95, the registry is a single place for keeping such information as what hardware is attached, what system options have been selected, how computer memory is set up, and what application programs are to be present when the operating system is started. The registry is somewhat similar to and a replacement for the simpler INI (initialization) and configuration files used in earlier Windows (DOS-based) systems. INI files are still supported, however, for compatibility with the 16-bit applications written for earlier systems.

In general, the user updates the registry indirectly using Control Panel tools, such as TweakUI. When you install or uninstall application programs, they also update the registry. In a network environment, registry information can be kept on a server so that system policies for individuals and workgroups can be managed centrally. (http://searchwin2000.techtarget.com/sDefinition/0,,sid1_gci212883,00.html)

Additionally, user activity such as file searches and a history of certain user activity is stored in the registry.

The RVF utility allows the user to view data directly from any one of the individual files that make up the registry. This is useful as it provides an easy method for viewing registry data from an imaged system. Additionally, it allows for the viewing of registry data when one or more individual registry files are unavailable. RFV also allows for searching within these registry files and the exporting of data from these files into formats viewable by other utilities.

The RFV software is relatively simple consisting of four menu items; file, view, tools and help.

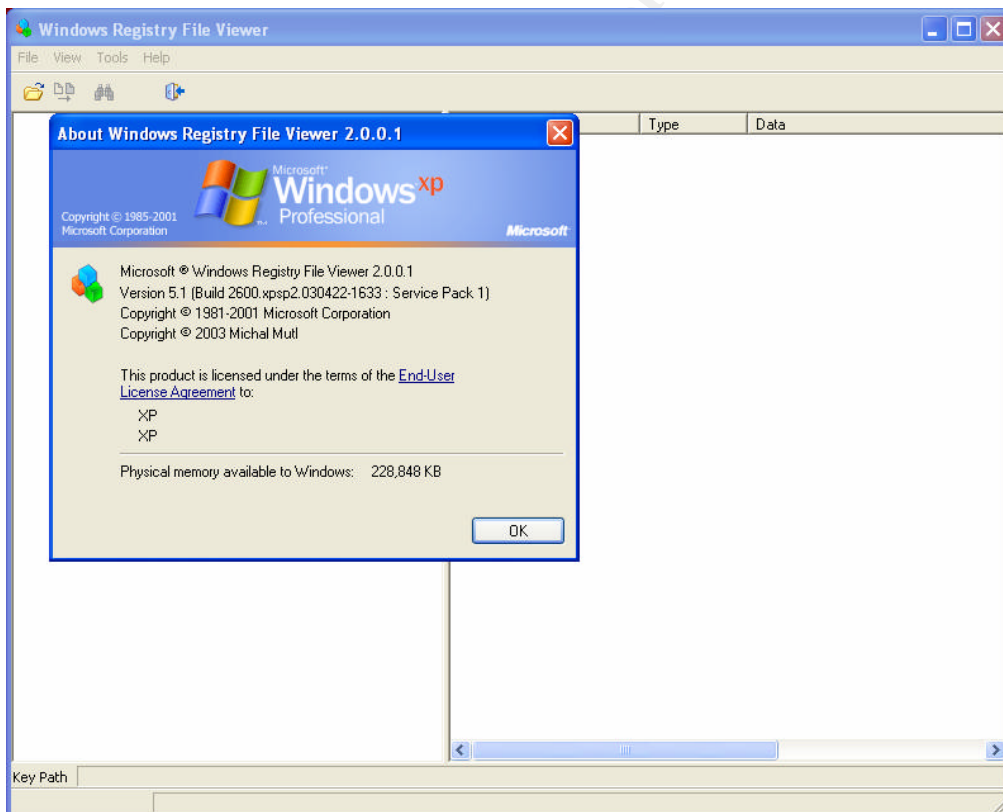
The file menu allows the user to open a registry file. Once the file is open, the user can save the data from an opened file as a text dump (.dmp) file or export the data in a format readable by REGEDIT4.

The view menu allows the user to copy a specific registry key path to the clipboard. It also provides a method for viewing specific information about the registry data. The data view provides a hexadecimal view of registry key data. “Key Information” provides detail about the selected registry key including the key name, the number of sub-keys and the date last modified. The “Security Key View” displays key related security information including the owner, group and

related access control lists (ACLs). “Hash Records View” provides hash information about the selected key and all sub-keys. “Root Key Information” provides similar information about the root key including the name of the HIV file, the number of keys within the file and the data and time the key was last modified. “Root Security Key Information” provides owner, group and ACL information about the selected file. “Root Key Hash Records” provides hash data for the file and all level-one keys. “File Information” provides information about the registry file being viewed including the location of the file, the modification date of the file, the format (Windows version) and the number of keys contained in the file. Finally, the view menu allows the user to switch between the normal two-pane format to a three-pane view where search results are displayed.

The tools menu is relatively simple providing an interface for the user to query the registry file for specific information. It also provides the option of viewing the security record of the file showing the record owner, group and any access control lists associated with the file.

The help menu provides information about the utility as displayed in the following image:



This menu also provides a link to the home page (<http://www.mitec.cz/>) for the product and a mail link for contacting the product’s creator (michal.mutl@mitec.cz).

3.2. Testing Methodology

During a forensics investigation this tool can be expected to be used in one of two situations as follows:

- Analysis of a read-only copy of the registry files. This may be the result of analyzing a read-only image of a system or having the specific files stored in some other read-only manner (i.e. CD-R, write protected floppy, etc.)
- Analysis of a writable copy of the registry files. This may be the result of the analysis of a system image or individual files where providing read-only protection is not possible or not reasonable.

(Note: The use of this tool to analyzed live data on a system is not possible. In the event that the analysis of a live system is required, the author of RFV recommends the use of the free utility “ERUNT – The Emergency Recovery Utility NT” by Lars Hederer. This utility can copy all registry files from a live system to a specific location. These copied files can then be reviewed by RFV. The ERUNT utility can be located at the following link: <http://home.t-online.de/home/lars.hederer/erunt>.)

To test the MiTeC Registry File Viewer utility as a forensic tool it is necessary to ensure that it accurately and reliably provides access to the data contained within registry files and that it does not modify the data contained within these files while accessing them. This testing must be performed on all types of registry file formats an investigator will likely encounter. Currently, there are two main registry formats, Windows 95 and Windows NT. Windows 95, Windows 98 and Windows ME utilize the Windows 95 format. Windows NT, Windows 2000, Windows XP and Windows 2003 use the Windows NT format.

This testing was conducted using a Windows 2000 Professional system, a Windows 2000 Server system, a Windows XP system and a Windows 98 system. Each of these systems was created as a virtual vmWare computer running on a RedHat Linux 9.0 host operating system.

The Windows 2000 (Pro and Server) and the Windows 98 systems were new, clean installations installed specifically for the purpose of this testing. No service packs were added to these systems. The systems were in their default state. The Windows XP system was a virtual vmWare system with all current patches, hot fixes and service packs installed as of the time of the testing. The XP system had numerous software packages installed and had been in use for a period of time. This allowed test searches of the registry files to be conducted on “production” system files.

3.3. Read-Only Testing

Testing was first performed to determine the effects of using the RFV utility on “read-only” files. Using read-only controls will eliminate the possibility of inadvertent modification of the files being reviewed however, it is important to ensure that the utility will function effectively when reviewing files that it cannot write to.

After creating each of the virtual systems (with the exception of the WinXP system), the systems were imaged using the dd utility and the images were transferred to the base Linux operating system using the NetCat utility as follows:

On the host operating system, the following command was executed:

```
nc -l -p 12345 > w2kpro.dd
```

(Note: the name of the output file was modified based on the operating system of the system to be imaged.)

On the guest operating system, the following command was executed:

```
dd if=\\.\c: | nc_orig.exe 192.168.2.1 12345
```

After the each system was imaged, the image was mounted on the Linux system using the following command:

```
Mount -t ntfs -o loop,ro,noexec,noatime,umask=0222,uid=forensic,gid=users  
/sourcedir/image.dd /mnt/mntdir/
```

(Note: the previous command was entered as a single line from the Linux system’s command prompt. The directory names were modified in this document to decrease the complexity of the statement. In addition, the file name “image.dd” was replaced by the actual name of the image file transferred to the system via NetCat.)

The Linux system had been configured to allow communication with a Windows computer via Samba. Support for accessing the NTFS file system was compiled into the kernel. The directory to which the images were mounted was shared and made accessible to a vmWare Windows system via a network connection. The Samba share was configured to allow only read-only access to further preserve data integrity.

To provide a baseline, the md5sum utility was used to generate a hash of each of the registry files with the following results:

Windows 2000 Server

File

	Original MD5 Hash
c:\winnt\system32\config\sam	48cbb562d744a799956a2951d03fc3cd
c:\winnt\system32\config\default	7d0eba1acc6fe952d58ae45933358d3f
c:\winnt\system32\config\security	8b3b305c8b93e745d5a38ce3cd0b4f7b
c:\winnt\system32\config\software	242f455b397a622f9e679f86d4c71e8e
c:\winnt\system32\config\system	93b22bedecb69097316b3994e8adde26
c:\documents and settings\administrator\ntuser.dat	fb80d4cf873d1443ed4b94e0fea124d5
c:\documents and settings\default user\ntuser.dat	945dd0ac35b2c2c3b785e1a2f96f4929

Windows 2000 Professional

File

	Original MD5 Hash
c:\winnt\system32\config\sam	bd32779fc073d7ef69a7bb82af26e2d7
c:\winnt\system32\config\default	f8f1cb44d352aa29eb272fa521c52536
c:\winnt\system32\config\security	4c669365e1b25cd2ae673d27c5f39b18
c:\winnt\system32\config\software	2b316a5ae7fbc99e55e287e469aa0f53
c:\winnt\system32\config\system	98cec64207577d5d36b51c20e144d3b3
c:\documents and settings\administrator\ntuser.dat	07252ecfad73b7a675c3ea91144d4502
c:\documents and settings\default user\ntuser.dat	d3a686e91d3eb27465ec9d43fbedaacb

Windows XP

File

	Original MD5 Hash
c:\winnt\system32\config\sam	bbd1e46f7f9ce5133e34cc9976a58fc7
c:\winnt\system32\config\default	33e7a697f87df568fde3929f58a4e2a7
c:\winnt\system32\config\security	2492f6ff42c92a47c0c5252329cdb390
c:\winnt\system32\config\software	ebca3ba3ed40ecd54b475911d409918c
c:\winnt\system32\config\system	ee4a48ae1c02be2afeb7c8b71ee057fe
c:\documents and settings\kfiscus\ntuser.dat	7f2606fa98f9f31c1323083d2dcf579c

Microsoft Windows 98 SE

File

	Original MD5 Hash
system.dat	2b690eb79bad75100804987727773cc4
user.dat	ddb6137f04bb3a09cf1ca42a06ef4df1

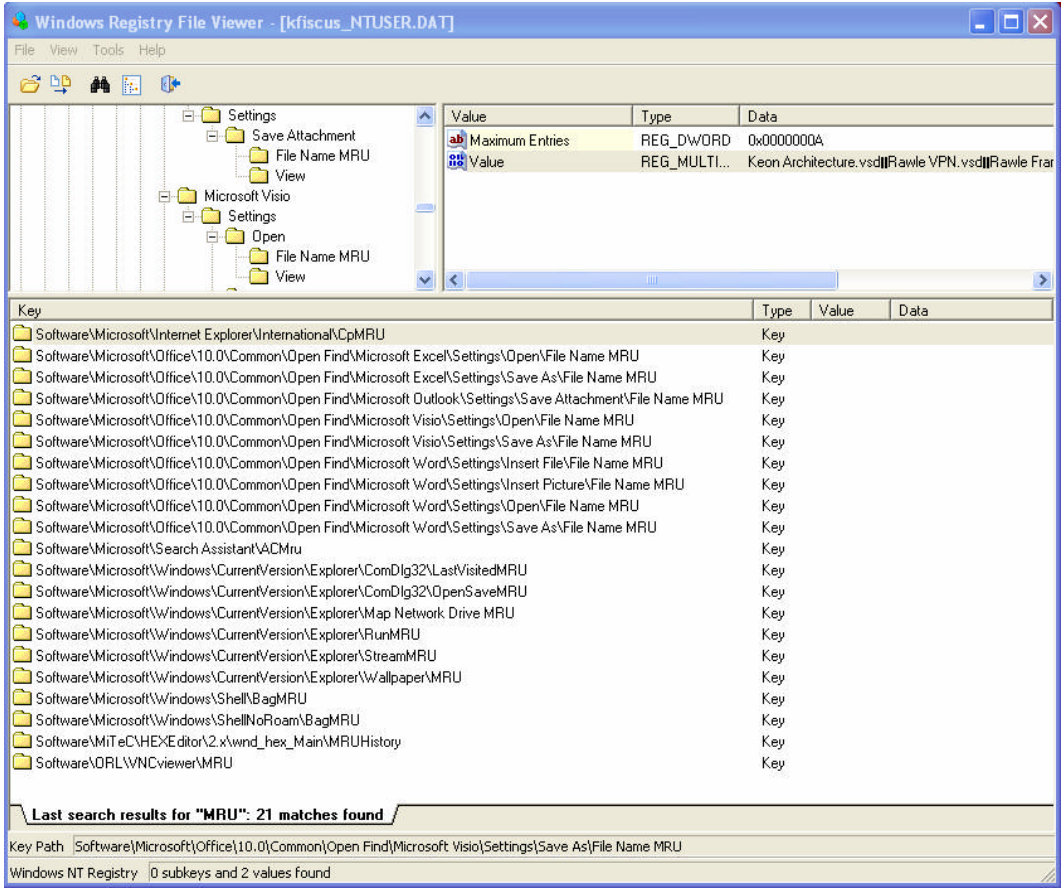
A virtual vmWare Windows XP computer was used to perform the analysis of the mounted images. The Samba shares were mapped to the virtual Windows systems and thus functioned as logical local read-only hard drive partitions. The RFV utility was installed on the local C drive of the XP system. The installation consisted of extracting a zip file into a directory on the hard drive. The utility was run directly from the extracted files with no additional installation required. In addition, copies of the utility files were written to a CD. The utility ran effectively from both the local drive and from the CD.

Using RFV, each of the registry files was opened and the following actions were performed:

- File – Save Dump
- File - View – Copy Key Path to Clipboard
- View – Data View
- View – Key Information
- View – Security Information
- View – Hash Records
- View – Root Key Information
- View – Root Key Security Information
- View – Root Key Hash Records
- View – File Information
- View – File Information – Properties
- View – Search Results
- Tools – Security Record Explorer
- Tools – Search (Password)

In addition, for the “production” Windows XP, an additional search for “MRU” was performed. The following image illustrates the results of the search:

© SANS Institute 2004, Author retains full rights.



The RFV utility functioned with no errors or unexpected results when operating on read-only files. Also, as expected, given the read only protections used, the hash values of the registry files remained constant as seen in the following:

Windows 2000 Server

File	Original MD5 Hash	Final MD5 Hash
c:\winnt\system32\config\sam	48cbb562d744a799956a2951d03fc3cd	48cbb562d744a799956a2951d03fc3cd
c:\winnt\system32\config\default	7d0eba1acc6fe952d58ae45933358d3f	7d0eba1acc6fe952d58ae45933358d3f
c:\winnt\system32\config\security	8b3b305c8b93e745d5a38ce3cd0b4f7b	8b3b305c8b93e745d5a38ce3cd0b4f7b
c:\winnt\system32\config\software	242f455b397a622f9e679f86d4c71e8e	242f455b397a622f9e679f86d4c71e8e
c:\winnt\system32\config\system	93b22bedecb69097316b3994e8adde26	93b22bedecb69097316b3994e8adde26
c:\documents and settings\administrator\ntuser.dat	fb80d4cf873d1443ed4b94e0fea124d5	fb80d4cf873d1443ed4b94e0fea124d5
c:\documents and settings\default user\ntuser.dat	945dd0ac35b2c2c3b785e1a2f96f4929	945dd0ac35b2c2c3b785e1a2f96f4929

Windows 2000 Professional

File	Original MD5 Hash	Final MD5 Hash
c:\winnt\system32\config\sam	bd32779fc073d7ef69a7bb82af26e2d7	bd32779fc073d7ef69a7bb82af26e2d7
c:\winnt\system32\config\default	f8f1cb44d352aa29eb272fa521c52536	f8f1cb44d352aa29eb272fa521c52536
c:\winnt\system32\config\security	4c669365e1b25cd2ae673d27c5f39b18	4c669365e1b25cd2ae673d27c5f39b18

Kevin Fiscus – GCFA Practical v1.4

c:\winnt\system32\config\software	2b316a5ae7fbc99e55e287e469aa0f53	2b316a5ae7fbc99e55e287e469aa0f53
c:\winnt\system32\config\system	98cec64207577d5d36b51c20e144d3b3	98cec64207577d5d36b51c20e144d3b3
c:\documents and settings\administrator\ntuser.dat	07252ecfad73b7a675c3ea91144d4502	07252ecfad73b7a675c3ea91144d4502
c:\documents and settings\default user\ntuser.dat	d3a686e91d3eb27465ec9d43fbedaacb	d3a686e91d3eb27465ec9d43fbedaacb

Windows XP

File	Original MD5 Hash	Final MD5 Hash
c:\winnt\system32\config\sam	bbd1e46f7f9ce5133e34cc9976a58fc7	bbd1e46f7f9ce5133e34cc9976a58fc7
c:\winnt\system32\config\default	33e7a697f87df568fde3929f58a4e2a7	33e7a697f87df568fde3929f58a4e2a7
c:\winnt\system32\config\security	2492f6ff42c92a47c0c5252329cdb390	2492f6ff42c92a47c0c5252329cdb390
c:\winnt\system32\config\software	ebca3ba3ed40ecd54b475911d409918c	ebca3ba3ed40ecd54b475911d409918c
c:\winnt\system32\config\system	ee4a48ae1c02be2afeb7c8b71ee057fe	ee4a48ae1c02be2afeb7c8b71ee057fe
c:\documents and settings\kfiscus\ntuser.dat	7f2606fa98f9f31c1323083d2dcf579c	7f2606fa98f9f31c1323083d2dcf579c

Microsoft Windows 98 SE

File	Original MD5 Hash	Final MD5 Hash
system.dat	2b690eb79bad75100804987727773cc4	2b690eb79bad75100804987727773cc4
user.dat	ddb6137f04bb3a09cf1ca42a06ef4df1	ddb6137f04bb3a09cf1ca42a06ef4df1

3.4. Read-Write Access Testing

As part of an investigation it may be necessary to use the RFV utility to examine files that aren't write protected or files where the write protection provided is suspect. As a result, it is necessary to test the effects of the use of RFV on a file that can be modified. To accomplish this, the read-only files from the previous testing were copied to a writable drive on the Windows XP test system. The read-only property was removed from the files and the md5sum utility was used to generate a hash for each of the files. The results of this were as follows:

Windows 2000 Professional

07252ecfad73b7a675c3ea91144d4502	Admin_NTUSER.DAT
f8f1cb44d352aa29eb272fa521c52536	default
d3a686e91d3eb27465ec9d43fbedaacb	Default_NTUSER.DAT
bd32779fc073d7ef69a7bb82af26e2d7	SAM
4c669365e1b25cd2ae673d27c5f39b18	SECURITY
2b316a5ae7fbc99e55e287e469aa0f53	software
98cec64207577d5d36b51c20e144d3b3	system

Windows 2000 Server

fb80d4cf873d1443ed4b94e0fea124d5	Admin_NTUSER.DAT
7d0eba1acc6fe952d58ae45933358d3f	default
945dd0ac35b2c2c3b785e1a2f96f4929	Default_NTUSER.DAT
48cbb562d744a799956a2951d03fc3cd	SAM
8b3b305c8b93e745d5a38ce3cd0b4f7b	SECURITY
242f455b397a622f9e679f86d4c71e8e	software
93b22bedecb69097316b3994e8adde26	system

Windows XP

33e7a697f87df568fde3929f58a4e2a7	default
7f2606fa98f9f31c1323083d2dcf579c	kfiscus_NTUSER.DAT
bbd1e46f7f9ce5133e34cc9976a58fc7	SAM
2492f6ff42c92a47c0c5252329cdb390	SECURITY
ebca3ba3ed40ecd54b475911d409918c	software
ee4a48ae1c02be2afeb7c8b71ee057fe	system

Windows 98 SE

2b690eb79bad75100804987727773cc4	System.dat
ddb6137f04bb3a09cf1ca42a06ef4df1	User.dat

(Note: the names of c:\Documents and Settings\Administrator\NTUSER.DAT and c:\Documents and Settings\Default User\NTUSER.DAT were changed to Admin_NTUSER.DAT and Default_NTUSER.DAT respectively to allow for all files from each test system to be placed in a single directory.)

Upon opening each of the writable files using RFV, the following actions were performed:

- File – Save Dump
- File - View – Copy Key Path to Clipboard
- View – Data View
- View – Key Information
- View – Security Information
- View – Hash Records
- View – Root Key Information
- View – Root Key Security Information
- View – Root Key Hash Records
- View – File Information
- View – File Information – Properties
- View – Search Results
- Tools – Security Record Explorer
- Tools – Search (Password for all files, Password & MRU for XP files)

After opening all test files and performing all listed actions, the md5sum utility was used to generate hash values for each file. The results showed that none of the hash values had changed for any of the tested files. These results indicate that this utility does not modify any registry file it is used to analyze. It is important to note, however, that when analyzing files that have not been write protected, care should be taken to avoid compromising those files by other means as, when not write protected, these files can be modified or deleted by other utilities or operating system functions.

3.5. Viewing Non-Registry Files

When forensically analyzing a system or specific files from a system file other than registry files may be involved. These include other system files, documents, executable files, etc. They may also include files that intentionally or accidentally appear to be registry files but are, in fact, not. To determine the effects of attempting to access these files with RFV, the following actions were performed:

The following files were copied to a writable directory called “test”

- **System.dmp** - A dump file created by RFV
- **Test** - An empty file created by the Linux touch command
- **Alert2a.xls** - A Microsoft Excel spreadsheet
- **System Security Engineer.doc** - A Microsoft Word document
- **Infosec Roadmap Executive Summary.pdf** - An Adobe PDF file
- **SAM** - A text file named to replicate a registry file
- **Putty.exe** - An binary executable file

An md5 hash was generated for each of the files. The RFV utility was then used to attempt to access each of the test files with the following results.

File	Results
system.dmp	Access violation at address 004A0A85 in module 'RFV.exe'. Write of address 00000024.
test	Stream read error.
alert2a.xls	Access violation at address 004A0A85 in module 'RFV.exe'. Write of address 00000024.
System Security Engineer.doc	Access violation at address 004A0A85 in module 'RFV.exe'. Write of address 00000024.
Infosec Roadmap Executive Summary.pdf	Access violation at address 004A0A85 in module 'RFV.exe'. Write of address 00000024.
SAM	Stream read error.
Putty.exe	Access violation at address 004A0A85 in module 'RFV.exe'. Write of address 00000024.

Md5 hashes for each of the files were again generated to determine if the files had been modified by the access attempt. In all cases, the hash values of the files after the access attempt were identical to those before the attempt indicating that no modifications were made to the test files. Although cryptic, the errors had no harmful effect on the operations of RFV.

The results of this testing indicate that non-registry files cannot be opened or reviewed by RFV. Also, attempts to access non-registry files will not modify or otherwise harm the RFV utility or the file in question. It is important to note that not all types of files were tested. It is therefore possible that RFV may be able to open a file of a specific untested format however, as the utility does not include functionality to modify files, the integrity of these non-registry files would be maintained.

3.6. Presentation

The RFV utility does not contain any integrated functionality for outputting data in a format that would be easily interpreted by others. This is due to the fact that registry information is, by nature, cryptic, complex and difficult to decipher. The utility does allow the investigator to view a wide variety of information about the registry file being viewed including its keys, subkeys, key values and data. This data may be displayed on the screen of a computer. The use of a screen capture tool such as “Screen Seize” by Ziff-Davis Media (<http://www.pcmag.com/article2/0,4149,22140,00.asp>) can be used to convert the data displayed by the utility to image files which can then be used as needed.

The following information can be displayed using MiTeC Registry File Viewer:

Main View

- Hierarchical display of all registry keys, subkeys, key values and value data.

File Information

- Format (Windows NT Registry or Windows 95 Registry)
- Last Modified
- Number of keys
- Number of HBINs
- Loading time

Root Key Information

- Key Name
- Relative Offset
- Number of Subkeys
- Security Key Offset
- Date Modified

Root Key Security Information

- Usage Counter
- Record Size

- Previous Record Offset
- Next Record Offset
- Owner SID
- Group SID
- Number of SACL ACEs
- Number of DACL ACEs
- Listing of SACLs including flags and permissions
- Listing of DACLs including flags and permissions

Root Key Hash Records

- A hash of the root key
- Names of all first level subkeys
- Hash values for all first level subkeys
- Key offsets for all first level subkeys

Key Information

- Key Name
- Idx
- Relative Offset
- Number of Subkeys
- Security Key Offset
- Date Modified

Root Key Security Information

- Usage Counter
- Record Size
- Previous Record Offset
- Next Record Offset
- Owner SID
- Group SID
- Number of SACL ACEs
- Number of DACL ACEs
- Listing of SACLs including flags and permissions
- Listing of DACLs including flags and permissions

Hash Values

- A hash of the selected key
- Names of all next level subkeys
- Hash values for all next level subkeys
- Key offsets for all next level subkeys

Security Record Exporter

- Listing of all security records
- For each security record
 - Usage Counter
 - Record Size
 - Previous Record Offset
 - Next Record Offset
 - Owner SID

- Group SID
- Number of SACL ACEs
- Number of DACL ACEs
- Keys affected by the security record
- Listing of SACLs including flags and permissions
- Listing of DACLs including flags and permissions

3.7. Summary

MiTeC Registry File Viewer provides the forensic investigator with the capability of viewing individual registry files in circumstances where the use of other tools, such as regedit, is not feasible. RFV provides a view of the registry files that is similar to that provided by regedit. Unlike regedit which provides a view of the entire registry, RFV provides a view of a subset of the registry at any given time.

The RFV utility is designed to be “read-only” in nature containing no functionality to modify the individual registry files. The only write actions taken by this tool are the ability to export data into dump (test file) or regedit4 formats. In neither case are the original files modified. Additionally, tests have verified that the use of RFV under normal conditions does not modify the files view in any way. This maintenance of file integrity occurs whether the files are “writable” or write protected.

The utility ran effectively from both a CD and from the local hard drive of an analysis system requiring no installation. This allows a forensic investigator to compile a CD of investigation tools including RFV to be used when a traditional forensic investigation workstation.

RFV operates on both registry files formats currently used by Windows operating systems, the Windows 95 format and the Windows NT format.

The MiTeC Registry File Viewer displays data in a manner that can be viewed easily but due to the nature of the Windows registry, the results must be interpreted to be meaningful to an individual unfamiliar with Microsoft Windows registry details. The utility does not contain any functionality to export the data it displays. As a result, the utility must be viewed “live” or another utility must be used to capture screen images and save them to a more usable format (i.e. bitmap images).

The results of this testing indicate that this utility can be used when conducting a forensic investigation; however, it has limited use as an incident response tool on a live system. On a live system, this utility must be used in conjunction with other tools. On captured data or imaged files, this tool can be used without risking data compromise. It can be run from either the local hard drive of a forensic system or from a CD as needs dictate. While this utility does not directly modify data it is used to view, it is strongly recommended that evidence always be write-protected to reduce the risk of inadvertent compromise.

4. Part 3 - Legal Issues

4.1. Question A – Broken Laws

Based on the type of material John Price was distributing, what if any, laws have been broken based on the distribution?

There are a number of factors affecting which laws have been broken. If Mr. Price were operating within the United States, he would have broken the U.S. Copyright Law (Title 17 U.S.C. Section 101 et seq., Title 18 U.S.C. Section 2319) the Federal Anti-Bootleg Statute (18 USC 2319A), the Fair Use Doctrine (USC Title 17, Sec 107) and the No Electronic Theft Law (NET Act). In addition, he would likely have broken one or more laws of the state from which he was operating.

More information about these laws and regulations can be found at the following sites:

17 U.S.C. §§ 101 et. seq.
<http://www4.law.cornell.edu/uscode/17/>

18 U.S.C. §§ 2319
<http://www4.law.cornell.edu/uscode/18/2319.html>

18 U.S.C. §§ 2319A
<http://www4.law.cornell.edu/uscode/18/2319A.html>

17 U.S.C. §§ 107
<http://www4.law.cornell.edu/uscode/17/107.html>

If Mr. Price was involving associates in other countries, depending on the country, he could have been breaking the Berne Convention for the Protection of Literary and Artistic Works, the Rome Convention, the Trade Related Aspects of Intellectual Property Rights (TRIPS) Agreement, The World Intellectual Property Organization Copyright Treaty (WCT) and the WIPO Performances and Phonograms Treaty (WPPT) in addition to local laws of the countries involved.

More information about these laws and regulations can be found at the following sites:

The Berne Convention for the Protection of Literary and Artistic Works
http://www.belipo.bz/e_library/articles/copyrightlaw.pdf

The Rome Convention
<http://www.wipo.int/clea/docs/en/wo/wo024en.htm>

The Trade Related Aspects of Intellectual Property Rights (TRIPS) Agreement

<http://www.uspto.gov/web/offices/com/doc/uruguay/finalact.html>

The World Intellectual Property Organization Copyright Treaty (WCT)

<http://www.wipo.int/clea/docs/en/wo/wo033en.htm>

The WIPO Performances and Phonograms Treaty (WPPT)

<http://www.wipo.int/clea/docs/en/wo/wo034en.htm>

4.2. Question B – Contraband on Corporate Systems

What would the appropriate steps be to take if you discovered this information on your systems? Site specific statutes.

17 U.S.C. Chapter 5, Section 503

(<http://www4.law.cornell.edu/uscode/17/503.html>), states that under certain circumstances, the illegally copyrighted material may be ordered impounded by a court. A final court judgment may call for the destruction of the material in question. Should legal, administrative or civil proceedings be involved, the collection of evidence should be conducted as well.

If mp3 files were discovered on corporate systems, assuming they served no legitimate business purpose, the system on which they were found should be forensically imaged. Initial incident response actions should be performed to collect any dynamic information about the system such as running processes, network connections, etc. After collecting all available information in a forensically sound manner, the files in question should be wiped from the system.

Once the evidence has been secured and the files in question have been removed, an attempt should be made to locate similar files on other corporate systems. Initial attention should be paid to any systems that had open connections to the suspect system and to any systems with a trust relationship with that system.

Before taking any additional action, it must be determined whether the files in question were illegally copyrighted or if they were legitimate copies.

Circumstances where these files may be legitimate include the following:

- The individual who placed the files there had already purchased the material, damaged it and is using these files as a replacement
- The files contain material not protected by a copyright
- The copyright owner gave the file “owner” permission to duplicate the files

If the investigation could determine that the files were illegally copied and tied to a specific individual, appropriate administrative disciplinary action should be taken. In addition, the company should send out information reminding its employees that such activity is illegal and will not be tolerated by the organization.

The potential exists that the activity in question can involve criminal prosecution. If so, law enforcement should be contacted. 17 U.S.C. Chapter 5, Section 506 (<http://www4.law.cornell.edu/uscode/17/506.html>) defines the conditions for such activity to be considered a criminal offense as follows:

(a) Criminal Infringement. -

Any person who infringes a copyright willfully either -

(1) for purposes of commercial advantage or private financial gain, or

(2) by the reproduction or distribution, including by electronic means, during any 180-day period, of 1 or more copies or phonorecords of 1 or more copyrighted works, which have a total retail value of more than \$1,000,

shall be punished as provided under section 2319 of title 18, United States Code. For purposes of this subsection, evidence of reproduction or distribution of a copyrighted work, by itself, shall not be sufficient to establish willful infringement.

(b) Forfeiture and Destruction. -

When any person is convicted of any violation of subsection (a), the court in its judgment of conviction shall, in addition to the penalty therein prescribed, order the forfeiture and destruction or other disposition of all infringing copies or phonorecords and all implements, devices, or equipment used in the manufacture of such infringing copies or phonorecords.

(c) Fraudulent Copyright Notice. -

Any person who, with fraudulent intent, places on any article a notice of copyright or words of the same purport that such person knows to be false, or who, with fraudulent intent, publicly distributes or imports for public distribution any article bearing such notice or words that such person knows to be false, shall be fined not more than \$2,500.

(d) Fraudulent Removal of Copyright Notice. -

Any person who, with fraudulent intent, removes or alters any notice of copyright appearing on a copy of a copyrighted work shall be fined not more than \$2,500.

(e) False Representation. -

Any person who knowingly makes a false representation of a material fact in the application for copyright registration provided for by section 409, or in any written statement filed in connection with the application, shall be fined not more than \$2,500.

(f) Rights of Attribution and Integrity. -

Nothing in this section applies to infringement of the rights conferred by section 106A(a).

In the event that a criminal offense is involved, the organization should contact law enforcement to determine if criminal prosecution will be involved.

4.3. Question C – Evidence

In the event that your corporate counsel decides to not pursue the matter any further at this point, what steps should you take to ensure any evidence you collect can be admissible in proceedings in the future should the situation change?

The first and most important step would be to store the floppy disk and an image of the floppy disk with an associated md5 hash in a secure location. Chain of custody should be maintained and the number of individuals with access to the disk, image and hash should be as limited as is possible. An email of the md5 hash should be sent with a description of what the hash is to a number of trusted individuals including the investigator. The email should be sent through a service that timestamps and certifies the message such as ReadNotify (<http://www.readnotify.com>).

After the raw evidence has been secured, all investigation notes, documents and findings should be duplicated and stored in a secure location. Electronic data should be hashed to prevent tampering.

4.4. Question D – Child Pornography

How would your actions change if your investigation disclosed that John Price was distributing child pornography?

While the possession of illegally distributed copyrighted material does violate law, the possession of child pornography is generally considered a more serious issue. The Protection of Children from Sexual Predators Act of 1998 makes the possession of even a single image an offence punishable by fines and imprisonment of up to 30 years. U.S. law (18 U.S.C. § 2252 and 2252A states that if child pornography is discovered, the discoverer must take “reasonable steps” to destroy all images or report the matter to law enforcement. The discoverer must also take steps to ensure that people, other than law enforcement do not access or copy any image.

Based on the laws covering child pornography, I would immediately contain and control any discovered images restricting access to the images to all but investigation personnel. I would also collect any data that could be potentially relevant to the investigation including network, system and email logs, network traffic capture data, hardware and media. After collecting as much evidence as possible in a reasonable time frame, I would involve law enforcement. Once law enforcement became involved, I would then be acting under “color of law” and would be subject to additional restrictions such as the requirements for court orders for wiretaps and subpoenas for searches. Law enforcement, however, could expand the scope of the investigation to include assets and property owned by the suspect.

Depending on the instructions from law enforcement and corporate counsel, I would either turn over all collected evidence to the investigating law enforcement officer(s) or I would begin a detailed forensic investigation.

Information about the laws referenced in this section can be found at the following sites:

Protection of Children from Sexual Predators Act of 1998
<http://www.antichildporn.org/fedstat-cac.htm>

18 U.S.C. § 2252
<http://www4.law.cornell.edu/uscode/18/2252.html>

18 U.S.C. § 2252A
<http://www4.law.cornell.edu/uscode/18/2252A.html>

© SANS Institute 2004, Author retains full rights.

References

SANS/GIAC "GIAC Certified Forensic Analyst (GCFA) Practical Assignment – Version 1.4." 21 July 2003.

URL: http://www.giac.org/GCFA_assignment_print.php

Freshmeat.net – “netcat – Default Branch”

30 November 1999

URL: http://freshmeat.net/projects/netcat/?topic_id=150

bmap – bmap forensic tool

URL: <http://build.inx-bbc.org/packages/fs/bmap.html>

University Technology Services – Ohio State University “Unix System Administration” by Frank G. Fiamingo - 1996

URL: http://wks.uts.ohio-state.edu/sysadm_course/html/sysadm-82.html

ITsecurity.com – “Slack Space”

25 September 2003

URL: <http://www.itsecurity.com/dictionary/slack.htm>

PC World “Sam Spade v1.14”

URL: http://www.pcworld.com/downloads/file_description/0,fid,4709,00.asp

VMware “VMware Workstation 4”

URL: <http://www.vmware.com>

Brian Carrier “The Autopsy Forensic Browser”

URL: <http://www.sleuthkit.org/autopsy/index.php>

Dan Farmer and Wietse Venema “The Coroner’s Toolkit”

URL: <http://www.porcupine.org/forensics/tct.html>

United States Air Force Office of Special Investigations “Foremost”

URL: <http://sourceforge.net/projects/foremost/>

Jupitermedia Corporation “inode”

9 November 1999

URL: <http://www.webopedia.com/TERM/I/inode.html>

AntiChildPorn.org

URL: <http://www.antichildporn.org/fedstat-cac.htm>

Legal Information Institute - 18 U.S.C. § 2252

URL: <http://www4.law.cornell.edu/uscode/18/2252.html>

Legal Information Institute - 18 U.S.C. § 2252A
URL: <http://www4.law.cornell.edu/uscode/18/2252A.html>

Legal Information Institute - 17 U.S.C. Chapter 5, Section 506 URL:
<http://www4.law.cornell.edu/uscode/17/506.html>)

Legal Information Institute - 17 U.S.C. Chapter 5, Section 503
URL: <http://www4.law.cornell.edu/uscode/17/503.html>

Legal Information Institute - 17 U.S.C. §§ 101 et. seq.
URL: <http://www4.law.cornell.edu/uscode/17/>

Legal Information Institute - 18 U.S.C. §§ 2319
URL: <http://www4.law.cornell.edu/uscode/18/2319.html>

Legal Information Institute - 18 U.S.C. §§ 2319A
URL: <http://www4.law.cornell.edu/uscode/18/2319A.html>

Legal Information Institute - 17 U.S.C. §§ 107
URL: <http://www4.law.cornell.edu/uscode/17/107.html>

Alhaji Tejan-Cole: The Berne Convention for the Protection of Literary and
Artistic Works - 1886
URL: http://www.belipo.bz/e_library/articles/copyrightlaw.pdf

The Rome Convention - 1961
URL: <http://www.wipo.int/clea/docs/en/wo/wo024en.htm>

The Trade Related Aspects of Intellectual Property Rights (TRIPS) Agreement
April 1995
URL: <http://www.uspto.gov/web/offices/com/doc/uruguay/finalact.html>

The World Intellectual Property Organization Copyright Treaty (WCT)
URL: <http://www.wipo.int/clea/docs/en/wo/wo033en.htm>

The WIPO Performances and Phonograms Treaty (WPPT)
December 20, 1996
URL: <http://www.wipo.int/clea/docs/en/wo/wo034en.htm>