



Core Knowledge Foundation, www.coreknowledge.org

Sleuthkit, the Digital Forensic Toolkit

Detective Work

Sleuthkit performs forensic analysis on both Microsoft and UNIX filesystems, applying its sleuth-like skills from the command line to identify evidence, recover deleted files and reconstruct scenarios.

All of which play a vital role in the art of digital computer forensic work.

In this growing field it is better to prepare and practice for the problems that lie in the future.

BY RALF SPENNEBERG

- Which systems are affected?
- Has any of the information stored on your systems been modified and if so, what?
- What was the intruder aiming to do?

It is quite often possible to answer questions like these after examining the Firewall and IDS protocols – provided the admin implemented a security system and adequate logging. But comprehensive monitoring of systems and networks often conflicts with protection of data privacy policies.

Forensic analysis of the affected systems after the event is the only way to obtain clear evidence. The exact procedures are complicated and depend on a number of factors. If you are looking for a comprehensive introduction to the subject matter, there are a few documents – some of them available online – which may help [5]. The goal of any forensic investigation is to ascertain the events and their timeline: the series of activities that led to the compromise and any actions performed by the intruder on the victim machines. If the results of this analysis are to be used as evidence at a later stage, there is also a requirement for unambiguous documentation of the individual investigative steps.

Performing the individual investigative steps in a fixed order will ensure that the results are reproducible: “Secure and isolate the scene. Record the scene. Conduct a systematic search for evidence.” [6]

Secure and Isolate

The first step in forensic analysis is a subject of much controversy. Most experts recommend disconnecting the system from the active network immediately. This prevents the intruder from recognizing the analysis and covering his tracks.

Some experts disagree with this; the intruder might have installed a hidden script that automatically wipes the system if it is disconnected from the network. It is difficult to make a general recommendation, but I tend to recommend and use the former approach.

Recording the Scenario

When analyzing a compromised computer the forensic expert should first look to saving volatile information,

Firewalls and Intrusion Detection Systems alert administrative users and provide protection against most attacks. But intruders continue to find loopholes and attack machines. The administrator then acts like a coroner performing forensic tests on the living (i.e. running) or deceased (i.e. powered off) system to salvage vital evidence.

Tools like TCT, the classic forensic toolkit (The Coroner’s Toolkit [4]) or the newer Sleuthkit [1], which is the subject of this article, help solve these problems. After a compromise, there are a number of difficult questions to answer, such as:

- How did the compromise occur?
- What prevented the firewall from fending off the attack?
- Why did the IDS not recognize the attack (until it was too late)?

THE AUTHOR

Ralf Spenneberg is a freelance Unix/Linux trainer and author. Last year saw the release of his first book: “Intrusion Detection Systems for Linux Servers”. Ralf has also developed various training materials.



Enhanced Loopback Driver

Once the enhanced loopback driver and the modified `losetup` command have been installed you have to create the appropriate devices using the supplied `createdev` command. Then you can simply mount the image of a whole harddisk using the following commands:

```
# losetup -r /dev/loopa host_2
hda.dd
# fdisk -l /dev/loopa
```

Now you can access the individual partitions using `/dev/loopa1`, `/dev/loopa2`, etc. Since the `-r` option sets up the loopback device as read-only, even journaling filesystems may not write.

TCT and Sleuthkit

The first forensic toolkit for UNIX systems was written by Wietse Venema and Dan Farmer in 1999 [5]. The Coroner's Toolkit (TCT) is a collection of commands that admins can use to perform post-incidents forensics on UNIX computers. Wietse Venema still maintains this toolkit and regularly offers updates on his homepage [4]. TCT can analyze the following operating systems: Sun Solaris, SunOS, Linux, FreeBSD, OpenBSD, and BSD/OS. Brian Carrier from Atstake (typically spelled

@stake [3]) started adding additional tools to TCT at an early stage, releasing these additions as the TCT Utils. The TCT Utils offer a range of extended filesystem functions to those found in TCT.

Analyzing Non-UNIX Filesystems

Early on in 2002 Brian Carrier implemented FAT and NTFS support to provide analytical facilities for addition filesystem types. He also reworked the TCT sources and this was reason enough to change the name to TASK, The Atstake Sleuth Kit. Brian additionally

released the Autopsy Forensic Browser. Despite its name Autopsy it is in fact a Web server; admins performing such forensic work may find Autopsy useful as a simple GUI front-end for TASK commands. In April 2003 Brian Carrier again changed the name of the project to The Sleuthkit to mark the release of the current 1.61 version. This emphasizes the Open Source nature of the project and its independence of the Atstake corporation. The project files are available from Sourceforge [1, 2].

before moving on to non-volatile data. The following are volatile:

- Main memory
- Kernel messages
- Time/date
- Active processes
- Open files
- Network configuration
- Network connections

This information is lost, when you down the system, but it can be extremely valuable. Normal Linux commands are typically all you need to secure this data and the *grave-robber* command from tct is useful. The command is not part of Sleuthkit. It is important to use programs from trustworthy sources when securing the evidence; these could be statically compiled commands on a CD or a write-protected floppy.

Non-volatile data is next on the list. This means the filesystems and the swap memory. You should additionally look to recording any documents in the vicinity and the general appearance of the system. A camera can be a useful tool for documentation purposes.

A simple backup is not sufficient

A simple backup of the system is not sufficient. The filesystem copy must be

identical to the original; you might miss files deleted by the intruder otherwise. The backup should also include slack-space. The *dd* tool for UNIX can perform this task; a Windows version is also available from [7].

The admin should use *dd* to backup the whole hard disk and not just individual partitions. When doing so, it is important not to store any data on the original disk. Instead you should use an external hard disk, running the Netcat tool, or its cryptographic counterpart Cryptcat, or an SSH tunnel, to transfer the data to the forensic system on which you will test and examine.

Migrating the Data

If you use Netcat to transfer the content of the disk, the Netcat listener must be running on the target machine:

```
nc -l -p 3000 > host_hda.dd
```

And then feed the hard disk data from the system you need to back up to Netcat:

```
dd if=/dev/hda | nc Server 3000
```

Forensic investigators are well advised to calculate a checksum for the disk image immediately after backing up. The

checksum later provides evidence that the backup has not been manipulated.

```
md5sum host_hda.dd
```

The next task is to divide the hard disk into individual partitions, as the Sleuthkit can only handle individual partitions. The *fdisk -lu host_hda.dd* command lists the partitions (see Listing 1).

You can again use *dd* to extract the individual partitions. To do so, you will need to calculate the size of the partition using the start and end cylinders. In the case of *hda1* this amounts to $15\ 119\ 999 - 63 + 1 = 15\ 119\ 937$. *dd* can then use this information to store the first partition in a file of its own:

```
dd if=host_hda.dd of=host_hda1.
.dd bs=512 skip=63 count=
15119937.
```

If these steps sound too complicated, you can back up individual partitions from the outset, but this does imply the danger of missing information needed for forensic analysis if it is stored in non-partitioned areas. The enhanced loopback driver [8], which is capable of accessing the complete hard disk as a loopback device, is more practical

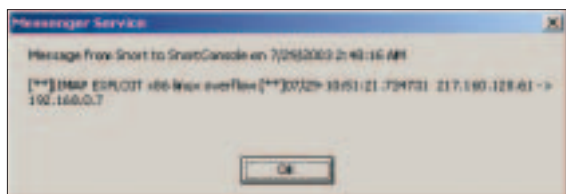


Figure 1: When the IDS reports an intruder (in this case Snort is using Win-Popup to alert on an IMAP exploit), it is the admin's task to initiate incident response procedures. Sleuthkit can help analyze the filesystems

Listing 1: Partitions for the first example

```
Disk /dev/hda: 0 heads, 0 sectors, 0 cylinders
Units = sectors of 1 * 512 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1		63	15119999	7559968+	83	Linux
/dev/hda2		15120000	80408159	32644080	5	Extended
/dev/hda5		15120063	16178399	529168+	82	Linux swap
/dev/hda6		16178463	24373439	4097488+	83	Linux
/dev/hda7		24373503	32568479	4097488+	83	Linux
/dev/hda8		32568543	80408159	23919808+	83	Linux

Listing 2: Deleted Files

```
class|host|device|start_time
ils|kermit.spenneberg.de|honeypot.hda5.dd|1052056153
st_ino|st_alloc|st_uid|st_gid|st_mtime|st_atime|st_ctime|st_dtime|st_mode|st_nlink|st_size|st_block0|st_block1
1|a|0|0|973385737|973385737|973385737|0|0|0|0|0|0
1890|f|17275|20|973695367|973695367|973695367|973695367|40755|0|0|6172|0
1891|f|17275|20|846630170|973695173|973695367|973695367|100644|0|1082|6173|0
1892|f|17275|20|973695367|973695367|973695367|973695367|40755|0|0|6174|0
1893|f|17275|20|846630171|973695173|973695367|973695367|100644|0|1458|6175|0
1894|f|17275|20|846630171|973695173|973695367|973695367|100644|0|1362|6176|0
2243|f|17275|20|846630171|973695173|973695367|973695367|100644|0|1465|6177|0
```

Table 1: File Systems in Sleuthkit

Filesystem	Command line Option
BSDi FFS	bsdi
FAT Filesystem FAT12	fat12
FAT Filesystem FAT16	fat16
FAT Filesystem FAT32	fat32
FreeBSD FFS	freebsd
Linux Filesystem EXT2	linux-ext2
Linux Filesystem EXT3	linux-ext3
NTFS	ntfs
OpenBSD FFS	openbsd
Solaris FFS	solaris

than manual separation of the individual images on the disk.

Systematic Searching

After storing the filesystem of the compromised computer, creating checksums and storing a copy in a safe place, the investigator can go on to analyze a copy of the filesystem, leaving the original untouched as evidence. Sleuthkit pro-

vides a number of commands for this step – we will be looking at some of the major tools in the following sections. The second part of this series describes a hands-on investigation using Autopsy.

Installation is extremely simple: download the package from the homepage, extract it, and call *make*. This places the tools in the *bin/* subdirectory. The Autopsy Forensic Browser is quite easy

to compile, however, you might prefer to use the RPM packages.

Sleuthkit tools are divided into four categories. The first category displays information for complete filesystems and contains only the *fsstat* command. The second group starts with *d*, and allows you to access data stored in files: *dcalc*, *dcat*, *dls*, and *dstat*. The Sleuthkit provides the following tools for meta-

Waking the Dead

Restoring deleted files is by no means a trivial task on UNIX-type filesystems. I am not aware of an *unerase* command for restoring deleted data without any danger of loss, no matter what UNIX operating system you use. Thus, some background knowledge of how the filesystem stores and deletes data is essential to run tools like TCT or Sleuthkit, and to cope with issues that may arise when restoring deleted files.

The Second Extended Filesystem, EXT2, is a useful example of a heritage Inode based UNIX filesystem. Each file is represented by a special structure, its Inode.

The Inode stores the meta-information belonging to the file. Additionally, data blocks are required to store the payload. Directories are simply special files that again comprise an Inode. Their data blocks store the directory list which contains filenames and links to the appropriate Inodes.

The Inode in EXT2

Inodes store the meta-information for a file except its name. This includes the file size, type, permissions, the owner and group, the refer-

ence counter and three UNIX timestamps (*ctime*, *atime* and *mtime*). The Inode also contains twelve direct references to data blocks, that is the addresses of the first twelve data blocks in the file.

Inodes use additional indirect references, the first of which points to a data block containing direct references to additional data blocks. The last two references point to a double or triple indirect block (see Figure 2).

When the filesystem deletes a file, it simply marks the directory entry and Inode as

deleted and frees up the space occupied by the Inode and the data blocks. The references to the data blocks and the link between the filename and the Inode tend to remain unchanged. Thus, it is possible to use tools such as *ils* to display deleted Inodes and *icat* to restore the file content. However, this only works if the filesystem has not already reassigned the Inode or data blocks to another file. The *fls* command shows the names of deleted files.

Modern Implementations Delete More

Many modern Linux distributions delete files

in a way that effectively prevents them from being restored. Unfortunately, this security feature tends to impede forensic analysis with tools such as *ils* and *icat*. In this case, the admin performing the investigation is typically forced to resort to the *dls* Sleuthkit command which reads the slackspace on the hard disk. The investigator can then use *grep* or the *sorter* tool to investigate the results. Autopsy is an elegant front-end that helps out with this more complex investigative step.

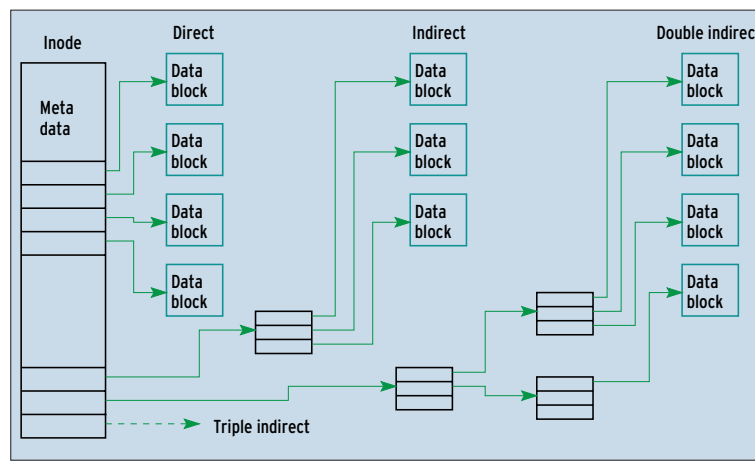


Figure 2: The Inode (left) stores meta-information for a file. It references the data blocks with the file contents. Larger files may also require the use of indirect references

information stored in Inodes: *icat*, *ifind*, *ils*, and *istat*. The commands in the fourth group start with *f*, and are designed for file level tasks such as creating file lists and searching for files.

The *mactime* command displays events chronologically on a system. The *file* and *sorter* commands sort the files in a hard disk image by type.

Supported Filesystems

Sleuthkit analyzes the filesystem types shown in Table 1. Unfortunately, it does not support typical Linux filesystems like Reiser FS, JFS, and XFS. The “Waking the Dead” insert describes how UNIX filesystems work, indicating the information on which tools are oriented, and pointing out their limitations.

EXT2 and EXT3 can produce some unpleasant side effects with more modern distributions. Depending on what patches you have applied, these

```

# find -l linux-ext2 -x / -x honeypot_hda8.dd > data/body
# find -l linux-ext2 -x /boot -x honeypot_hda1.dd >> data/body
# find -l linux-ext2 -x /home -x honeypot_hda6.dd >> data/body
# find -l linux-ext2 -x /usr -x honeypot_hda5.dd >> data/body
# find -l linux-ext2 -x /var -x honeypot_hda7.dd >> data/body
# ls -l linux-ext2 -x honeypot_hda8.dd >> data/body
# ls -l linux-ext2 -x honeypot_hda1.dd >> data/body
# ls -l linux-ext2 -x honeypot_hda6.dd >> data/body
# ls -l linux-ext2 -x honeypot_hda5.dd >> data/body
# ls -l linux-ext2 -x honeypot_hda7.dd >> data/body
# wact.log -b data/body -z CST 11/07/2000 > data/wact.log.11072000

```

Figure 3: In order to analyze the modification order across multiple filesystems, the investigator needs to collate file and Inode information in the *body* file

filesystems delete the reference to the data blocks from the Inode, and this prevents simple tools like *icat* from restoring the content of a deleted Inode.

Practical Hints: Reanimation

Now it is time to analyze the hard disk images you backed up. Our examples use the Forensic Challenge filesystems. This competition was organized by the Honeynet project in January 2001. A tarball containing the data is available for downloading from various addresses [9].

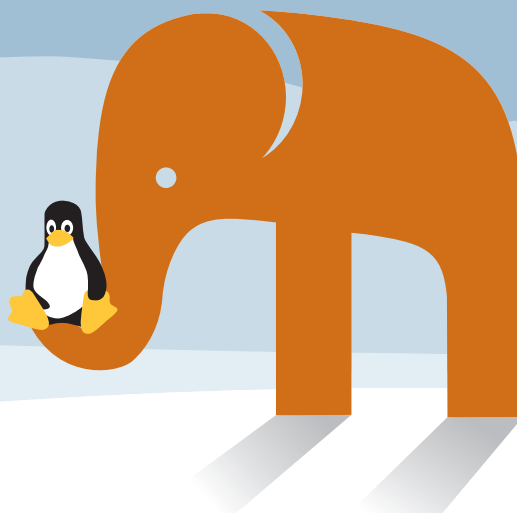
```

# tail +12 readme | head >
-6 | md5sum -c
honeypot.hda1.dd: 0k
honeypot.hda5.dd: 0k
honeypot.hda6.dd: 0k
honeypot.hda7.dd: 0k
honeypot.hda8.dd: 0k
honeypot.hda9.dd: 0k

```

The function of the individual partitions is supplied as background knowledge (see Table 2). We will be mounting

EASY TO BUY • EASY TO SET UP • EASY TO SEE



p-p-p-pick up

a dedicated linux server

EasyVserver solutions
Debian or RedHat O/S
True “root” access
4, 6 or 8GB raid space
1IP address
Highly secure

Make the most of Linux technology with the big name in Web registration and hosting packages. Our flexible, scalable, secure EasyVserver solutions start at just £39 per month. Back up by unrivalled support and know-how. If you want the best of Linux come along for the ride.

log on today at: www.easyspace.com

EasySpace 
your perfect partner for the web

Established in 1997 with over 1/2 million customers. Accredited ICANN registrar & nominet member. Prices exclude VAT.

Read-only, Write Sometimes

When performing forensic analysis it is often useful to mount the hard disk partition to look for files. This particularly applies to Reiser FS, as the Sleuthkit does not support the Reiser filesystem. To ensure data integrity, investigators should generate checksums for all filesystems and mount them as read-only.

Unfortunately read-only cannot always be taken at face value: non-journaling filesystems really don't change anything if they are mounted read-only. However, journaling filesystems like EXT3 or Reiser FS tend to update the journal even for read-only mount ops. Of course, this changes the MD5 checksum. There is a simple test for this: You will need a test image and its checksum for the test:

```
# dd if=/dev/zero
of=/tmp/testimage.orig >
bs=1024k count=50
```

```
# losetup /dev/loop0
/tmp/testimage.orig
# mkreiserfs /dev/loop0
# md5sum /tmp/testimage.orig
2ceed9e819bf4348a33a21f7697149c82
/tmp/testimage.orig
```

After mounting read-only and immediately unmounting, the MD5 sum has changed:

```
# mount -o ro -t reiserfs
/dev/loop0 /mnt
# umount /mnt
# md5sum /tmp/testimage.orig
c6ffc8a13a6821cd327d1db9ee351faf2
/tmp/testimage.orig
```

There are only three ways to avoid this at present:

- Modify the EXT3 or Reiser FS driver
- Save the filesystem on a CD-ROM
- Use the enhanced loopback driver [8]

the partitions as read-only using the loopback device to provide an initial overview – but this can be dangerous and may modify the data, see the “Read-only Write Sometimes” insert.

Sleuthkit commands provide a more in depth look. The deleted files are the first point of interest – if the attacker has attempted to cover his tracks, this is often the best way of uncovering them.

Inodes of Deleted Files

The *ils* command displays Inode information for a filesystem; the *-r* flag tells the command to concentrate on deleted Inodes (default). Listing 2 shows the results for *ils -f linux-ext2 -r honeypot.hda5.dd*.

Output is in table format, with line 1 containing the header and line 2 the corresponding data. The command was

obviously run on the host *kermit.spenneberg.de* for a file called *honeypot.hda5.dd*. Things start to become more interesting in line 3; this is the second header for the table and is used for any remaining lines. The first column, *st_ino* contains the Inode number, column two, (*st_alloc*), indicates whether the Inode is free (*f*) or allocated (*a*).

There are many approaches to discovering interesting files. The first approach assumes any files with a non-zero size (column 11, *st_size*). These files are investigated using the *file* command, in order to categorize the file content. To do this, the tool accesses a large database containing nearly every known file format (*magic**). The Sleuthkit version of *file* is no different from the version included with many distributions, although the database may be larger.

Interesting Files

Listing 3 shows the possible syntax. Line 1 cuts off the first four lines of the Inode information output by the *ils* command, and passes the rest to an *awk* command (line 2). This ascertains any Inodes with a non-zero size, and passes the Inode numbers to the *icat* command. The latter extracts the files from the image and stores them in a subdirectory *data/hda5.icat/* using the Inode number as the filename.

The *file* command is then used to ascertain the file type, and *egrep* filters out the interesting files (line 7). An investigation on the first tarball (line 15) shows that we have discovered SSH Version 1.27.

Logbook Stardate 2000-11-07

The second approach to analyzing deleted files involves the forensic investigator first creating a chronological journal of the file operations on the system. The journal will cover any files, rather than just deleted files. Additionally, it makes sense to cover all the filesystems with this analysis.

The admin first collects data from the filesystems (see Figure 3). In addition to the familiar *ils* command, this involves using *fls*. This command collects information on any files that still exist on the filesystem. The *-m* option creates output,

Table 2: Challenge Partitions

Partition	Filesystem
/dev/hda8	/
/dev/hda1	/boot
/dev/hda6	/home
/dev/hda5	/usr
/dev/hda7	/var
/dev/hda9	swap

Listing 4: Mactime summary

```
ed Nov 08 2000 14:51:53 17969 .a. -/-rwxr-xr-x 1010 100 109832 /usr/man/.Ci/scan/x/x
1760 .a. -/-rwxr-xr-x 1010 100 109829 /usr/man/.Ci/scan/bind/ibind.sh
15092 .a. -/-rwxr-xr-x 1010 100 109836 /usr/man/.Ci/scan/x/pscan
4096 .a. d/drwxr-xr-x 1010 100 109841 /usr/man/.Ci/scan/port/strobe
1259 .a. -/-rwxr-xr-x 1010 100 109834 /usr/man/.Ci/scan/x/xfil
4096 .a. d/drwxr-xr-x 1010 100 109831 /usr/man/.Ci/scan/x
[...]
Wed Nov 08 2000 14:52:09 9 m.c l/lrwxrwxrwx 0 0 46636 /root/.bash_history -> /dev/null
9 m.c l/lrwxrwxrwx 0 0 23 /.bash_history -> /dev/null
[...]
```

which in turn is processed by the *mactime* command. The original mount point is also output. The *-m* flag is also required for *ils*, although the mount point is not.

The file used to collect the mass of data is called *body* for historical reasons. Calling *mactime* for this file generates a logbook of filesystem modifications. You can restrict the output to a period you are interested in. In our example taken from the Forensic Challenge, we can work on the assumption that the attack took place on November 7 2000.

Consistent Timekeeping

To provide consistent timekeeping, you should use the *-z* flag to specify the time-zone used by the system that created the data. In this case, it is CST (GMT-0600). The output shows unusual activities at 14:51 on November 8 (see Listing 4): The */usr/man/.Ci/* directory should not really be there.

Files with content changes are also of interest. These files are easily recognized

by the *m* in the third column. An *a* indicates access to the file, and *c* shows that the meta-information (permissions, owner, ...) has been modified. You can see where compilers have been called and libraries linked, which allows you to reconstruct the steps the attacker took.

There is one drawback to this approach. If the attacker has modified a file more than once, the Inode will only save the date of the last modification. The *mactime* tool will thus display only the last modification.

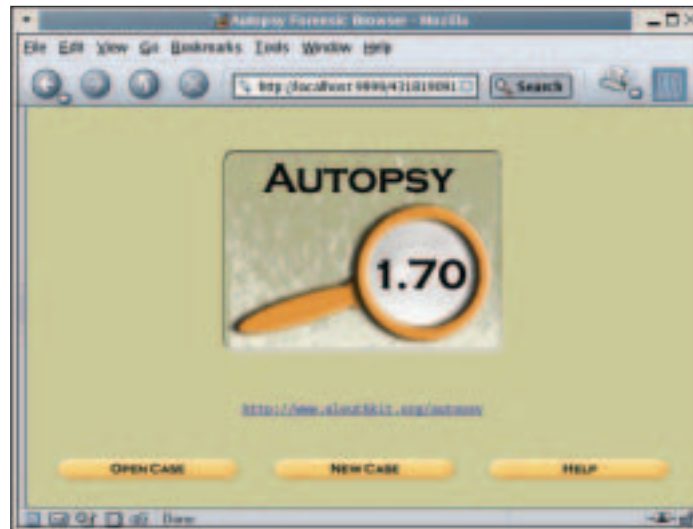


Figure 4: The Autopsy Forensic Browser provides a Web front-end for Sleuthkit and facilitates filesystem analysis

GUI to the Rescue

Sleuthkit commands can be quite difficult to use, and this makes it hard to keep an eye on the results. Searching for specific strings on all filesystems and analyzing the contents of deleted files can be tedious, and may require some extensive scripting.

Given these circumstances a GUI can be quite useful when performing forensic analysis. It provides an abstraction layer for the commands and presents only the results, meaning that forensic investigators does not need

to consult manpages to achieve results.

The Autopsy Forensic Browser (see Figure 4) provides a GUI of this type. It also allows the investigator to document and comment on data and results. Under some specific circumstance it may still make sense or be necessary to resort to the command line, but investigators will be able to perform major steps from within the GUI.

Part 2 of this series will be looking at the Autopsy Forensic Browser using the Forensic Challenge examples. ■

Listing 3: Discovering Interesting Data

```
# ils -f linux-ext2 -r honeypot.hda5.dd | tail +4 |
> awk -F '|' '{ $11 > 0 {print $1}' |
> while read in; do
>   icat honeypot.hda5.dd $in > data/hda5.icat/$in
> done

# file data/hda5.icat/* | egrep "tar|gzip|RPM|ELF"
data/hda5.icat/109791: GNU tar archive
data/hda5.icat/109861: GNU tar archive
data/hda5.icat/109865: RPM v3 bin i386 nfs-utils-0.1.9.1-1
data/hda5.icat/109866: RPM v3 bin i386 wu-ftpd-2.6.0-14.6x
data/hda5.icat/109943: ELF 32-bit LSB relocatable, Intel 80386, version
1 (SYSV), not stripped
data/hda5.icat/109944: ELF 32-bit LSB relocatable, Intel 80386, version
1 (SYSV), not stripped

# tar tf data/hda5.icat/109791
ssh-1.2.27/
ssh-1.2.27/COPYING
ssh-1.2.27/README
ssh-1.2.27/README.SECURID
[...]
```

INFO

- [1] Sleuthkit: <http://www.sleuthkit.org>
- [2] Autopsy Forensic Browser: <http://autopsy.sf.net>
- [3] Atstake: <http://www.atstake.com>
- [4] The Coroner's Toolkit: <http://www.porcupine.org/forensics/tct.html>
- [5] Dan Farmer and Wietse Venema, "Computer Forensic Analysis": <http://www.porcupine.org/forensics/>
- [6] Richard E. Saferstein, "Criminalistics: An Introduction to Forensic Science", Prentice Hall
- [7] dd for Windows – Unix-Utils: <http://unxutils.sf.net/>
- [8] Enhanced Loopback: ftp://ftp.hq.nasa.gov/pub/ig/ccd/enhanced_loopback/
- [9] Forensic Challenge files: <http://project.honeynet.org/challenge/images.html>