Global Information Assurance Certification Paper

Sys Admins and Hackers

GCFA Practical Assignment Version 1.4
Submission 2nd attempt
2nd July 2004

Lars Fresen

1 Content

2 Abstract

This document is part of the GIAC GCFA certification process. According to the GCFA practical assignment version 1.4 the following options were chosen:

1. Part1: Analysis of an unknown binary
2. Part2: Option1: Forensic analysis on a system
3. Part3: Legal issues of incident handling

Part 1 analyses the given case by the GIAC Team for this practical. It deals with the usage of an unknown binary and an employee suspected to have used it on computer equipment owned by his employer.
Part 2 is the case description of a real compromised system at my company and it describes not just the forensic analysis, additionally all the things which happened around the investigation were discussed.
Part 3 discusses the legal implications of the violation of the laws concerning the illegal distribution of copyright protected material and in special the situation of illegal distribution of child pornography.

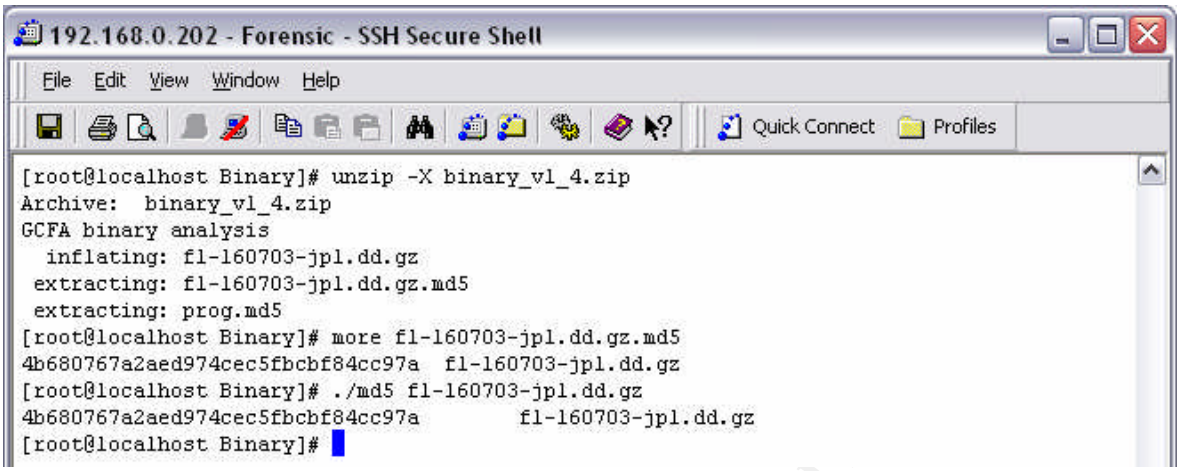
3 Part I: Analysis of an unknown binary

3.1 Floppy Disk Image Details

The image of the floppy disk was downloaded to the forensic workstation (see Part II of this Practical for a detailed description of the forensic workstation) from the GIAC web site in a ZIP file named “binary_v1_4.zip”. The following information of the seized disk is given on the GIAC web site:

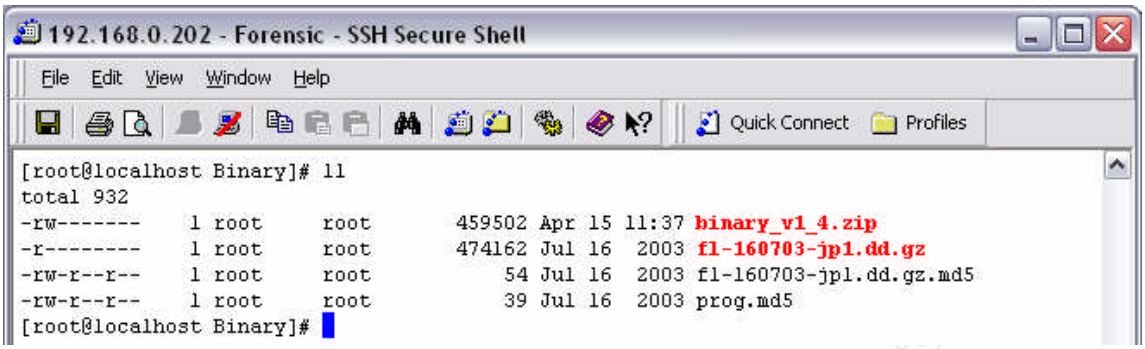
Tag# fl-160703-jp1
3.5 inch TDK floppy disk
MD5: 4b680767a2aed974cec5fbcbf84cc97a
Fl-160703-jp1.dd.gz

First the downloaded Zip file was unzipped to receive the compressed *dd* image of the disk mentioned in the evidence list above.



To uncompress the zip file the *unzip* command with the –X option was used. This option stands for the extraction of a zip archive and additionally it preserves the UID and GID of the extracted files. This is done surely having in mind that this is unimportant in this case because it could only reveal the UID and GID of the administrator or investigator who made the image of the floppy disk. After that the md5 checksum of the uncompressed ZIP file was generated to proof that the downloaded file was not altered. This is done by using the *more* command to view the content of the file "fl-160703-jpl.dd.gz.md5" which is the md5 checksum of the floppy disk image generated by the investigator seizing the evidence. Additionally I generated an md5 sum of the uncompressed file "fl-160703-jpl.dd.gz" using the *md5* command. An md5 checksum comparison[1] is a very powerful and simple method to proof that compared files are the same or not. The basic principle relies on a mathematical function which generates on an input a unique number as the result of the function. This unique number will always stay the same if the input stays the same and can be reproduced endlessly. As a result you can say the following. If two inputs result in the same unique number the input must had been equal. So in this case the input is the computer file and the unique number is the md5 hash value as shown in the above screen shot. As we can see the given md5 hash and the newly generated are the same so the files must have been received unaltered.

[1] A more comprehensive describtion on md5 could be found at this URL http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci527453,00.html

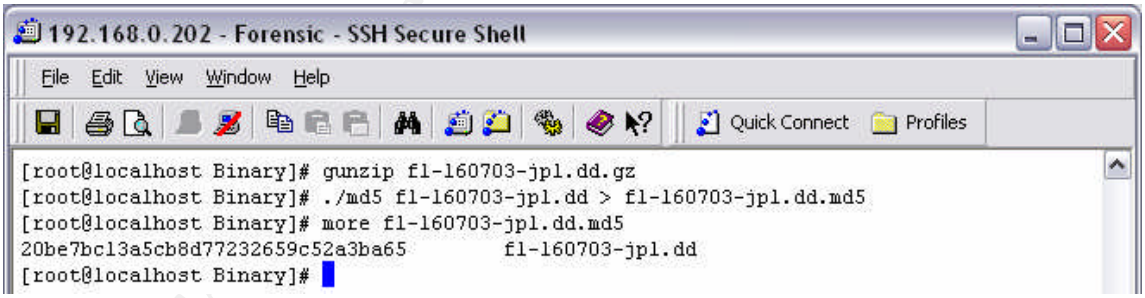


As you can see in the screenshot I received three files. Two files with md5 hashes. One of the floppy disk image and one of the still to investigate unknown binary which still has to be extracted from the floppy disk image. In table 3-1 the names and bit sizes of the files were shown.

Filename	Size
fl-160703-jp1.dd.gz	474162
fl-160703-jpl.dd.gz.md5	54
prog.md5	39

Table 3-1 the give files and their file sizes

The image was uncompressed using *gunzip* because the ".gz" ending of the file indicates that this file is an archive generated by *gzip*[2]. The result of this uncompressing activity is a file named "fl-160703-jpl.dd". The ending ".dd" indicates that this image of the floppy disk was generated by using the *dd* command. More information on the *dd* command can be found here[3]. After that an md5 hash of the uncompressed *dd* image was taken and compared against the given md5 hash to check the integrity of the file "fl-160703-jpl.dd". As these two md5 hashes are identically it is proofed that the evidence with the tag number Tag# fl-160703-jp1was not altered from the point it was seized to the point this investigation started.



[2] A brief decription of the gzip file format http://www.gzip.org/format.txt
[3] http://publibn.boulder.ibm.com/doc_link/en_US/a_doc_lib/cmds/aixcmds2/dd.htm

3.2 The Timeline

For the following investigations Sleuthkit 1.68[4] and Autopsy 2.0[5] were used. Sleuthkit and Autopsy are essential tools for a forensic computer investigator. A more detailed description of these tools can be found on the mentioned web sites and in part 2 of this practical. First a new case in Autopsy was added with the name “Mr. Price”. Than the floppy disk image was added including the md5 check option which was successfully completed and by this again proofed by another system the integrity of the evidence. Now the timeline was created using the corresponding option in Autopsy. Always if available the option “generate md5 hash” of any generated output or file is activated in Autopsy. The timeline of the floppy disk image is shown beneath in table 3-2:

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

Table 3-2 timline of the disk image

Actually in the information from the investigators of the Company (respective the information on the GIAC website) no information is given in which time zone the computer had been used and therefore all times can differ to the real time of the actions. The times

[4] www.sleuthkit.org
[5] www.sleuthkit.org/autopsy

you can see in table 3-2 were based on the chosen setting for the investigation which is the CET time zone.

All files were owned by a user with the UID 502 and the GID 502. Depending on the investigated system this might proof if these files belong to the suspect Mr. Price or not. Although we do not have the corresponding system we might come to the conclusion that UID 502 is equivalent with Mr. Price because of another fact. There is a sub directory which is called “John” which is the surname of Mr. Price. This might be a first indicator for the proof that the questioned evidence with Tag# fl-160703-jp1 belongs to the suspect. But at this point of the investigation we have to have in mind that this is by now just an assumption.

As you can see the timeframe within actions were taken ranges from the 28th January 2003 to the 16th of July, the day the evidence was taken. From the different file names like “DVD-Playing-HOWTO”, “SOUND-HOWTO”, “Kernel-HOWTO” and “prog” there is no clear hint in which direction the further investigation on the questioned binary “prog” should be directed. Nevertheless there are a few quite interesting information’s which later might lead to a valuable solution. A detailed analysis of the found files will follow in the next chapter.

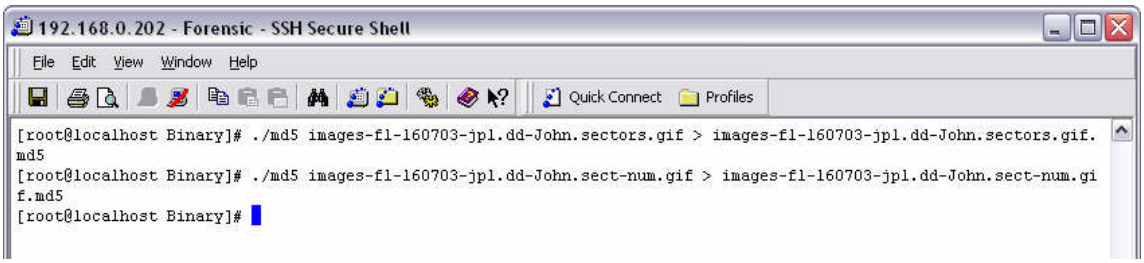
The first appearance of the binary “prog” in the timeline is on Monday, July the 14th at 14:24:00 with a size of 487476 bytes. Because of the set “m” flag in the Mactime table this binary there was first written to the floppy disk. The binary was changed on July the 16th at 06:05:33 indicated by the set “c” flag and then accessed (maybe executed) at 06:12:45 indicated by the set “a” flag. Interestingly the size did not change during the actions taken on July 16th at 06:05:33. Maybe the file was just opened and then saved again without any changes.

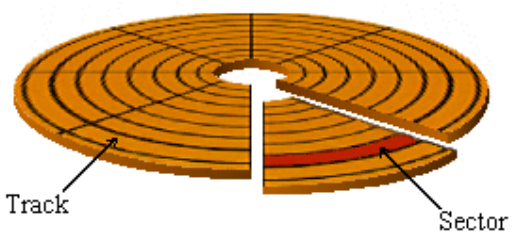
The other information in the timeline does not give any clue to what kind of fraud I am dealing with. It might have to do something with illegal distribution of copyright protected materials like mp3’s or DVD’s because of the files “DVD-Playing-HOWTO”, “SOUND-HOWTO” and “MP3-HOWTO-html.tar.gz”. But this is more some kind of speculation than knowledge. So let’s continue the investigation.

3.3 Analysis of the different files from the disk image

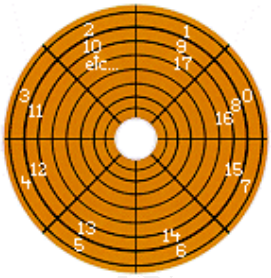
3.3.1 The files: sectors.gif and sect-num.gif

The files “sectors.gif” and “sect-num.gif” first appearance in the timeline is on January the 28th. The pictures were extracted from the disk image using the export option from Autopsy in the file analysis module. Then the md5 hashes from both files were taken as shown in the screenshot. Autopsy classifies the files as GIF Ver 87a image files. So I tried to view the files with the Netscape web browser and the corresponding option “display” in Autopsy. The results are shown in picture 3-1 and 3-2.





Picture 3-1 the Image sectors.gif

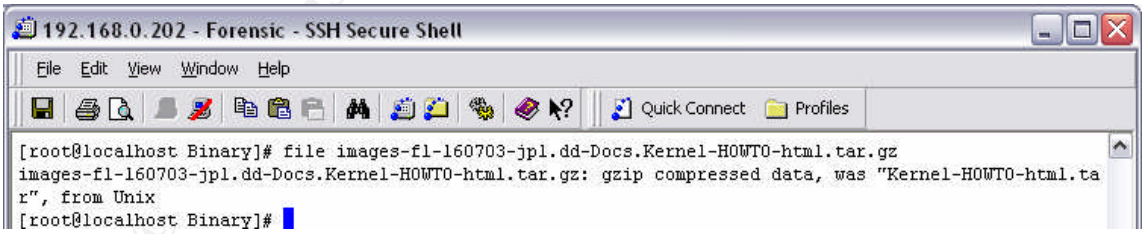


Picture 3-2 the Image sect-num.gif

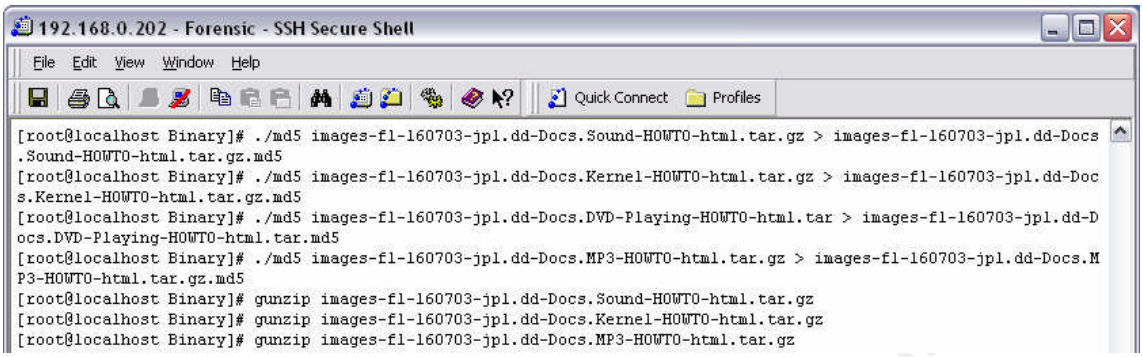
From these pictures no further conclusion can be made at this moment. The pictures seem to show a layout of disk or hard drive and the corresponding sector layout and the dependency between “Tracks” and “Sectors” on the corresponding media. Maybe this information should be considered when investigating the unknown binary.

3.3.2 The files “DVD-Playing-HOWTO-html.tar”, “Kernel-HOWTO-html.tar.gz”, “MP3-HOWTO-html.tar.gz” and “SOUND-HOWTO-html.tar.gz”

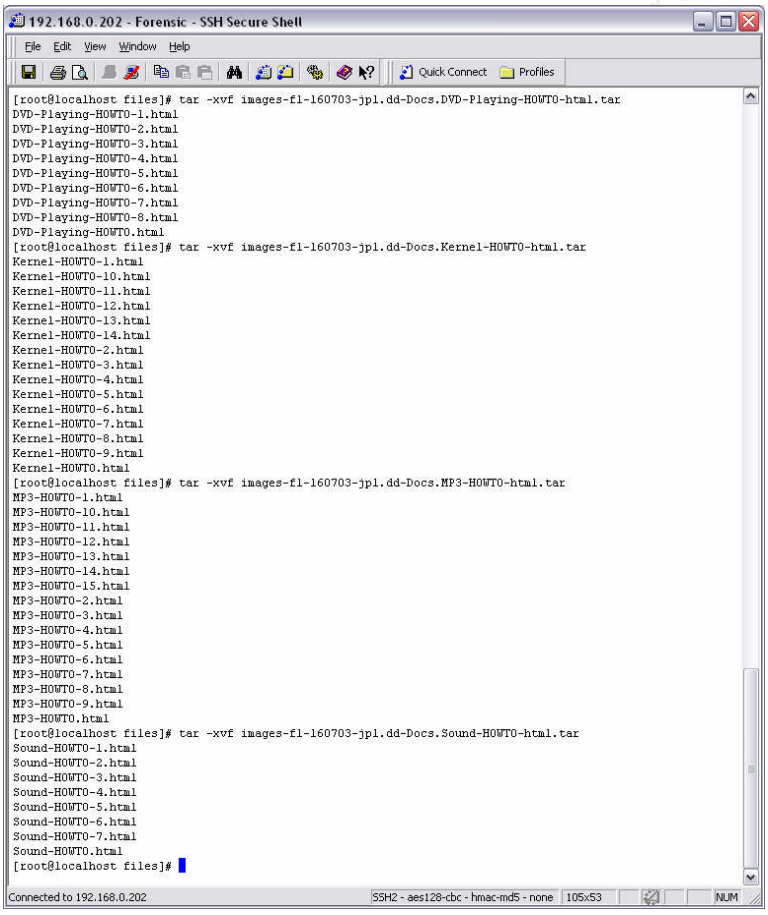
First of all md5 hashes of the files were taken as shown in the screen shot. After that the 3 files which indicate by their file ending “.gz” that these files were compressed gzip archives were checked with the *files* command to proof they were really gzip archives. This is exemplarily shown in the following screenshot for the file “images-fl-160703-jpl.dd-Docs.Kernel-HOWTO-html.tar.gz”.



After that the files were unzipped using the *gunzip* command.

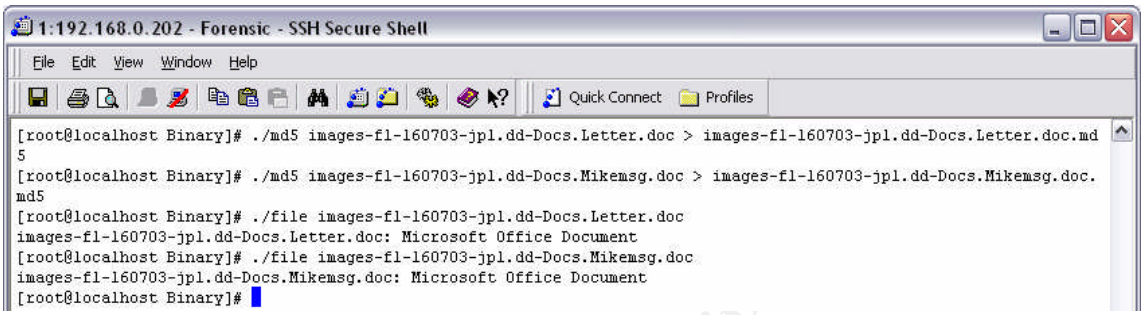


After that all tar archives were extracted as shown in the screen shot using the before named steps and procedures. After a short review of the files I can say the following: all files are HOWTO guides around the topics of playing DVD's on Linux systems, generating MP3 files from Audio CD's and getting sound cards working on Linux systems. There seems to be nothing unusual or illegal with these files. All documents have an author with contact information's and various references to the Linux community which still work. Nevertheless the picture of what the case might deal with is getting a small focus on the topic of illegal copying or distributing music and movies.



3.3.3 The files “Letter.doc” and “Mikemsg.doc”

Again the files were extracted from the floppy disk image using Autopsy and then the md5 hashes were created. Afterwards using the *file* command from the Sleuthkit I checked the file type of the documents. Both were identified as normal Microsoft Word documents. Now it was time for a string analysis because Microsoft Word documents normally keep quite a lot of metadata about the document e.g. the author, creation date and other things in the file.



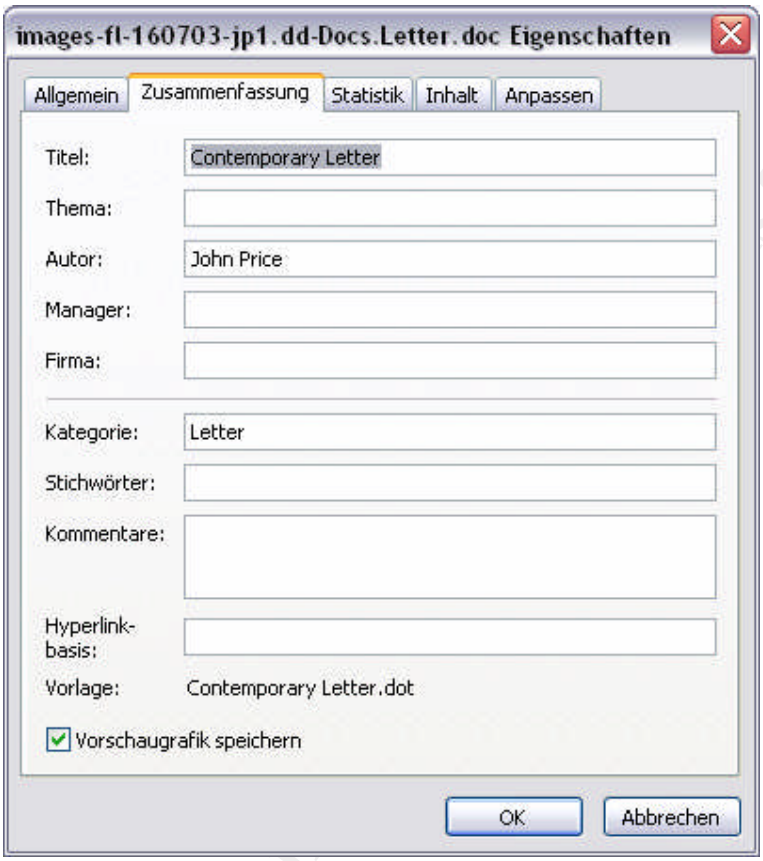
Here we can see some of the interesting output of the *strings* command to the file Letter.doc

```
Contemporary Letter
John Price
Contemporary Letter
John Price
Microsoft Word 8.0
Word
Microsoft Word
"System
Times New Roman
Times New Roman
. . . . . .
Times New Roman
Word
Microsoft Word
Times New Roman
Word
Microsoft Word
Times New Roman
Word
Microsoft Word
Times New Roman
Times New Roman
Company Name Here
14 July, 2003
[Click r
Times New Roman
here
 and type recipient
s address]

--- not so relevant parts cutted ----

 Letter
Contemporary Letter
Title
_PID_GUID
Microsoft Word Document
MSWordDoc

Word.Document.8
```

Table 3-3-3 *strings* output of the file “Letter.doc”

From the *strings* output we get the following information. The letter was written using Microsoft Word© 8.0, the name of the document could be “Contemporary Letter” and the name of the suspect “John Price” is mentioned in the *strings* output twice (marked in yellow). This might be the author entry which every Microsoft Word document contains in

its properties dialog. To prove this theory I have to open the document with Microsoft Word. To ensure that nothing unwanted happens during the file opening process like macro viruses I transferred the document to a special VMware©[6] machine with a clean installation of Microsoft Windows XP Professional© and Microsoft Office XP© running without any patches. Before doing anything I took a snapshot (functionality of VMware) of the system so I would easily be able to recover to the initial state. Additionally I disabled the network connection of the VMware machine to prevent that malicious code can spread across the network. The output of the documents properties dialog can be found in the following picture 3-3.



Picture 3-3 properties dialog of Letter.doc

And here we are. The name in the field author is John Price. This seems to be the second link between the evidence and the suspect.

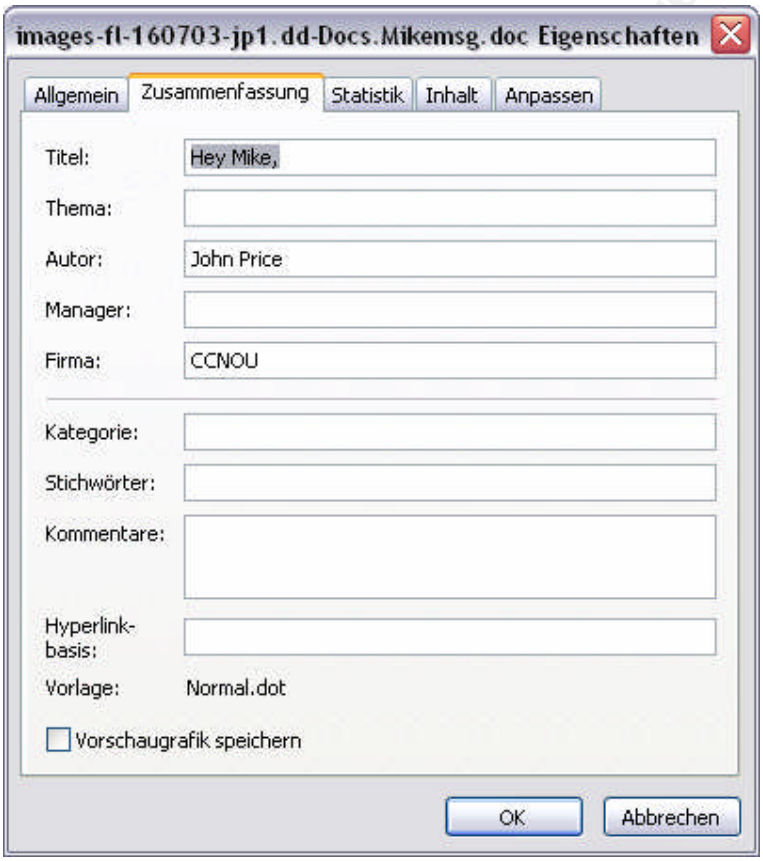
[6] www.vmware.com

Now let's see if I can find a third connection in the document "Mikemsg.doc". The procedure to discover this is the same as described before for the file "Letter doc". First let's check the output of the *strings* command. Again we find the name of the suspect in it.

```
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.
Hey Mike,
John Price
Normal
John Price
Microsoft Word 8.0
CCNOU
Hey Mike,
Title
_PID_GUID
Microsoft Word Document
MSWordDoc

Word.Document.8
```

Table 3-4 *strings* output of the file "Mikemsg.doc"

And again we make the dialog test on the VMware Machine as described above. The result is shown in the next picture. So here I have found a third connection between evidence and the suspect.



Picture 3-4 properties dialog of Mikmsg.doc

3.3.4 The file “prog”

No we come to the primarily questioned binary named “prog”. First of all the binary was extracted from the image using the corresponding option in Autopsy (File Analysis -> Extract). I received a file with the name “images-fl-16703-jpl.dd.prog”. The file is named this way because of the naming conventions of Autopsy. The received file has a byte size of 487476 bytes (as shown in table 3-5). Then I did a *file* command to the binary to detect its binary type. As shown in the screenshot it is a Linux executable for Linux systems with a 2.2.5 kernel.

```
[root@localhost Binary]#ll
-rw-------  1 root  root  487476 Jun 23 14:02 images-fl-160703-jp1.dd-.prog
[root@localhost Binary]# file images-fl-160703-jp1.dd-prog
images-fl-160703-jp1.dd-prog: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux 2.2.5, statically linked, stripped
[root@localhost Binary]#
```

Table 3-5 usage of the *file* command against the questioned binary

Next I did a strings command to the binary to see if there is any interesting ASCII information in it which will lead me to the real identity of the binary. A first look at the enormous list of entries of the strings command output did not lead to any direct connection. So the only thing which could be done now is “active googling” with interesting parts of the output. First I started with the most obvious part, some kind of name and address shown beneath.

```
+45 3325-6543
+45 3122-6543
keld@dkuug.dk
Keld Simonsen
ISO/IEC 14652 i18n FDCC-set
C/o Keld Simonsen, Skt. Jorgens Alle 8, DK-1615 Kobenhavn V
```

Table 3-6 first google search for information about the binary

After a few searches in combination with other found “keywords” I canceled this search pattern. I found quite an enormous amount of search results regarding the name and address but nothing in combination with the binary or something which could lead me to the searched binary or made sense in any other way.

The next interesting search pattern I found were the text parts which look like some kind of help output of the binary describing how to use it (table 3-7). I always searched with a combination of a few lines of the suspected help information of the binary. This took quite a lot of tries but then it was time for the right combination "getting from block" and "file size was". The first match for this search by Google was a link[7] to a “Linux Community’s Center for Security” article about “Linux Data Hiding and Recovery” Within this article there is shown how data can be hidden and recovered in file systems with a tool called “bmap”. Also there is direct link to an ftp site where this tool could be downloaded.

[7] http://www.linuxsecurity.com/feature_stories/data-hiding-forensics.html

Sorrowfully this link is dead. So again I tried Google to find another location where to download bmap to make an md5 hash of it to check if the still unknown binary is “bmap”. So I did a Google with “bmap” and a string which looks like the version of the binary “1.0.20” found in the strings output. This led me directly to a bunch of download locations. So I chose the first one and downloaded the gzipped and tared file to my forensic machine. Then I decompressed (gunzip, tar –xvf) it. It was the source code archive so I first had to compile it before I could make any comparison. After a “make” on the forensic workstation in the source code directory I received a binary called “bmap” which correctly was executable. But it did not have the same size nor was the md5 hash equivalent to the “prog” md5 hash given with the evidence floppy disk. But this was nearly expected because the chance that the compiler, the libraries and so on were the same on the forensic workstation and the machine the suspect worked with was nearly zero.

So let’s go again deeper in the analysis. First I did a *strings* command on the received binary and compared it to the output of the *strings* command against the original bmap (table 3-7 and table 3-8). There are many passages which are exactly the same and a few ones which are slightly changed. The equivalent passages could be found comparing the outputs in table 3-7 and 3-8.

```
%s: %s
<table bgcolor=%s><tr><td>%s: %s</td></tr></table><br>
<table bgcolor=%s><tr><td>%s</td></tr></table><br>
<table bgcolor=%s><tr><td></td></tr></table><br>
Brazil
.TH %s "%d" "%s" "%s" "%s"
.SH NAME
%s \- %s
.SH SYNOPSIS
.B %s
[\fIOPTION\fR]...
.SH DESCRIPTION
\fB\-\-%s\fR %s
\fB\-\-%s\fR \fIARG\fR %s
\fB\-\-%s\fR \fIINT\fR %s
\fB\-\-%s\fR \fIFILENAME\fR %s
\fB\-\-%s\fR \fIVALUE\fR %s
\fIVALUE\fR can be one of:
                \fB%s\fR
 | \fB%s\fR
                \fBSHORTHAND INVOKATION:\fR
Any of the valid values for \fB--%s\fR can be supplied directly as options. For instance, \fB--%s\fR can be used in place of \fB--%s=%s\fR.
                \fB%s\fR %s
--%s %s
.SH REPORTING BUGS
Report bugs to %s.
Usage: %s [OPTION]...
 [<%s-filename>]
--%s            %s
--%s <arg> %s
--%s <int> %s
--%s <filename> %s
--%s <
 | %s
> %s
--%s VALUE
  where VALUE is one of:
  %s  %s
<tt>%s</tt> invocation
<tt>%s [&lt;OPTIONS&gt;]
 [&lt;%s-filename&gt;]
</tt>
Where <bf>OPTIONS</bf> may include any of:
<descrip>
<tag>--%s</tag>              %s
<tag>--%s &lt;arg&gt;</tag> %s
<tag>--%s &lt;int&gt;</tag> %s
<tag>--%s &lt;filename&gt;</tag> %s
<tag>--%s &lt;
&gt;</tag> %s
<tag>--%s VALUE</tag>
<tag>%s</tag>  %s
</descrip>
<tag>--%s</tag> %s
%s:%s %s
operate on ...
```

target
entryexit
progress
branch
info
error
fatal
none
logging threshold ...
log-thresh
be verbose
verbose
name
useless bogus option
label
write output to ...
outfile
test for fragmentation (returns 0 if file is fragmented)
checkfrag
display fragmentation information for the file
frag
wipe the file from the raw device
print number of bytes available
test (returns 0 if exist)
wipe
place data
display data
extract a copy from the raw device
list sector numbers
operation to perform on files
mode
generate SGML invocation info
sgml
generate man page and exit
display options and exit
help
display version and exit
version
autogenerate document ...
1.0.20 (07/15/03)
newt
use block-list knowledge to perform special operations on files
prog
main
off_t too small!
07/15/03
invalid option: %s
try '--help' for help.
how did we get here?
no filename. try '--help' for help.
target filename: %s
Unable to stat file: %s
%s is not a regular file.
%s has multiple links.
Unable to open file: %s
Unable to determine blocksize
target file block size: %d
unable to raw open %s
Unable to determine count
Unable to allocate buffer
%s has holes in excess of %ld bytes...
error mapping block %d (%s)
nul block while mapping block %d.
seek failure
read error
write error
%s fragmented between %d and %d
%d %s
getting from block %d
file size was: %ld
slack size: %d
block size: %d
seek error
File: %s Location: %Ld size: %d
stuffing block %d
%s has slack
%s does not have slack
%s has fragmentation
%s does not have fragmentation
bmap_get_slack_block
NULL value for slack_block
Unable to stat fd
Unable to determine blocksize
error getting block count
fd has no blocks
mapping block %lu
error mapping block %d. ioctl failed with %s
error mapping block %d. block returned 0
bmap_get_block_count
unable to stat fd
unable to determine filesystem blocksize
filesystem reports 0 blocksize
computed block count: %d
stat reports %d blocks: %d

```
bmap_get_block_size
bmap_map_block
nul block while mapping block %d.
bmap_raw_open
NULL filename supplied
Unable to stat file: %s
%s is not a regular file.
unable to determine raw device of %s
unable to stat raw device %s
device mismatch 0x%x != 0x%x
unable to open raw device %s
raw fd is %d
bmap_raw_close
```

Table 3-7 strings on the binary prog

So what went wrong? If the help texts are nearly the same, both binaries should have the same source and the same goal. Go back to the first paragraph of this chapter. The output of the *file* command said about the binaries that the necessary libraries were statically linked. This means that the necessary libraries were included in the final compiled binary. So let's see what the *file* command says about the newly compiled binary on the forensic workstation.

```
examining a filename or url!
%s is a well-formed argument
checking against %s
flag-
flagized option invokation
examining an enum!
matched against an enum val
examining a venum!
matched against an venum val
arg matches against %s
process_match
true
matches against %s
invalid value for enum
mft_log_init
nbd-server
MFT_LOG_THRESH
none
fatal
error
info
branch
progress
entryexit
mft_log_shutdown
unspecified
enter
exit
%s: %s
violet
blue
green
yellow
orange
white
%s: %s
<table bgcolor=%s><tr><td>%s: %s</td></tr></table><br>
<table bgcolor=%s><tr><td>%s</td></tr></table><br>
<table bgcolor=%s><tr><td></td></tr></table><br>
Brazil
.TH %s "%d" "%s" "%s" "%s"
.SH NAME
%s \- %s
.SH SYNOPSIS
.B %s
[\fIOPTION\fR]...
.SH DESCRIPTION
\fB\-\-%s\fR %s
\fB\-\-%s\fR \fIARG\fR %s
\fB\-\-%s\fR \fIINT\fR %s
\fB\-\-%s\fR \fIFILENAME\fR %s
\fB\-\-%s\fR \fIVALUE\fR %s
\fIVALUE\fR can be one of:
                \fB%s\fR
 | \fB%s\fR
                \fBSHORTHAND INVOKATION:\fR
Any of the valid values for \fB--%s\fR can be supplied directly as options. For instance, \fB--%s\fR can be used in place of \fB--%s=%s\fR.
                \fB%s\fR %s
--%s %s
.SH REPORTING BUGS
Report bugs to %s.
Usage: %s [OPTION]...
 [<%s-filename>]
--%s            %s
```

--%s <arg> %s
--%s <int> %s
--%s <filename> %s
--%s <
 | %s
> %s
--%s VALUE
  where VALUE is one of:
  %s %s
<tt>%s</tt> invocation
<tt>%s [<OPTIONS>]
 [<%s-filename>]
</tt>
Where <bf>OPTIONS</bf> may include any of:
<descrip>
<tag>--%s</tag> %s
<tag>--%s <arg></tag> %s
<tag>--%s <int></tag> %s
<tag>--%s <filename></tag> %s
<tag>--%s <
></tag> %s
<tag>--%s VALUE</tag>
<tag>%s</tag> %s
</descrip>
<tag>--%s</tag> %s
%s:%s %s
operate on ...
target
entryexit
progress
branch
info
error
fatal
none
logging threshold ...
log-thresh
be verbose
verbose
name
useless bogus option
label
write output to ...
outfile
test for fragmentation (returns 0 if file is fragmented)
checkfrag
display fragmentation information for the file
frag
wipe the file from the raw device
wipe
print number of slack bytes available
slackbytes
test for slack (returns 0 if file has slack)
checkslack
wipe slack
wipeslack
place data into slack
putslack
display data in slack space
slack
extract a copy from the raw device
carve
list sector numbers
operation to perform on files
mode
generate SGML invocation info
sgml
generate man page and exit
display options and exit
help
display version and exit
version
autogenerate document ...
1.0.20 (05/29/00)
newt@scyld.com
use block-list knowledge to perform special operations on files
bmap
main
off_t too small!
05/29/00
invalid option: %s
try '--help' for help.
how did we get here?
no filename. try '--help' for help.
target filename: %s
Unable to stat file: %s
%s is not a regular file.
%s has multiple links.
Unable to open file: %s
Unable to determine blocksize
target file block size: %d
unable to raw open %s
Unable to determine count
Unable to allocate buffer

```
%s has holes in excess of %ld bytes...
error mapping block %d (%s)
nul block while mapping block %d.
seek failure
read error
write error
%s fragmented between %d and %d
%d %s
getting from block %d
file size was: %ld
slack size: %d
block size: %d
seek error
# File: %s  Location: %Ld  size: %d
stuffing block %d
%s has slack
%s does not have slack
%s has fragmentation
%s does not have fragmentation
bmap_get_slack_block
NULL value for slack_block
Unable to stat fd
Unable to determine blocksize
error getting block count
fd has no blocks
mapping block %lu
error mapping block %d. ioctl failed with %s
error mapping block %d. block returned 0
bmap_get_block_count
unable to stat fd
unable to determine filesystem blocksize
filesystem reports 0 blocksize
computed block count: %d
stat reports %d blocks: %d
bmap_get_block_size
bmap_map_block
nul block while mapping block %d.
bmap_raw_open
NULL filename supplied
Unable to stat file: %s
%s is not a regular file.
unable to determine raw device of %s
unable to stat raw device %s
device mismatch 0x%x != 0x%x
unable to open raw device %s
raw fd is %d
bmap_raw_close
/.../image
bogowipe
```

Table 3-8 *strings* on the original bmap

And now we see the reason for the big difference in file size. The binary from the evidence floppy disk comes with static linked shared libraries, the one I compiled with dynamically linked ones. This leads to the conclusion that the suspect not just had altered the source code oft the “bmap” file, but in addition he has altered the “Makefile” of the provided source code package.

```
[root@localhost Binary]# file images-fl-160703-jp1.dd-prog
images-fl-160703-jp1.dd-prog: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux 2.2.5, statically linked, stripped
[root@localhost Binary]# file bmap
bmap: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux 2.2.5, dynamically linked (uses shared libs), not stripped
[root@localhost Binary]#
```

Table 3-9 usage of the *file* command against the binary “prog” and “bmap”

Another try with changing the options in the “Makefile” (Adding the”–static” option to the LDFLAGS line) of the source package has the wanted effect but shows that I do not have the right version of libraries to come close to the exact file size.
Here I will stop this part of the investigation for the part of the file size. To proof this I would need the system the binary was compiled on with its libraries and further on for a total proof I will need all changes the suspect applied to the source code of it.

This all in mind I make the assumption that than an md5 hash comparison would be positive. Further on lead these new information’s to a new addition to this whole chapter (3.4). After finishing the “normal” analysis of the left files I will have to investigate the

"prog" binary in connection with all files on the disk. Maybe the tool was used for its purpose and there are hidden information's on the evidence disk.

3.3.5 The file: nc-1.10-16.i386.rpm..rpm

The next file I am looking at is the file "nc-1.10-16.i386.rpm..rpm. It was exported from the image using the "Export" functionality of Autopsy. Its first appearance in the timeline is on July the 14th at 14:12:02. It looks like a normal rpm file despite the unusual double dot before the "rpm" file ending. From the name it could be the well known *netcat*. But first let's do a *file* to it.

```
[root@localhost Binary]# ./file images-fl-160703-jp1.dd-.nc-1.10-16.i386.rpm..rpm
images-fl-160703-jp1.dd-.nc-1.10-16.i386.rpm..rpm: RPM v3 bin i386 nc-1.10-16
[root@localhost Binary]#
```

Table 3-10 *file* command against nc-1.10-16.i386.rpm..rpm

From the output of the *file* command we can now say that it is an rpm package with a program called "nc-1.10-16" to install. Still from its name it looks like *netcat* version 1.10 release 16. To proof if this file is the original version of *netcat* with the questioned version I downloaded the corresponding rpm package at rpmfind.net [8]and made an md5 hash comparison.

```
[root@localhost Binary]# ./md5 nc-1.10-16.i386.rpm
535003964e861aad97ed28b56fe67720        nc-1.10-16.i386.rpm
[root@localhost Binary]# ./md5 images-fl-160703-jp1.dd-.nc-1.10-16.i386.rpm..rpm
535003964e861aad97ed28b56fe67720        images-fl-160703-jp1.dd-.nc-1.10-16.i386.rpm..rpm
[root@localhost Binary]#
```

Table 3-11 *file* command against nc-1.10-16.i386.rpm

We can see from the md5 checksum comparison of both packages that they must be identically because the generated md5 hashes were the same. So this rpm package is *netcat* 1.10 release 16 as mentioned on the rpmfind.net page. Additionally we get the information that this rpm package comes for redhat 8.0 systems.

3.3.6 The file: ebay300.jpg

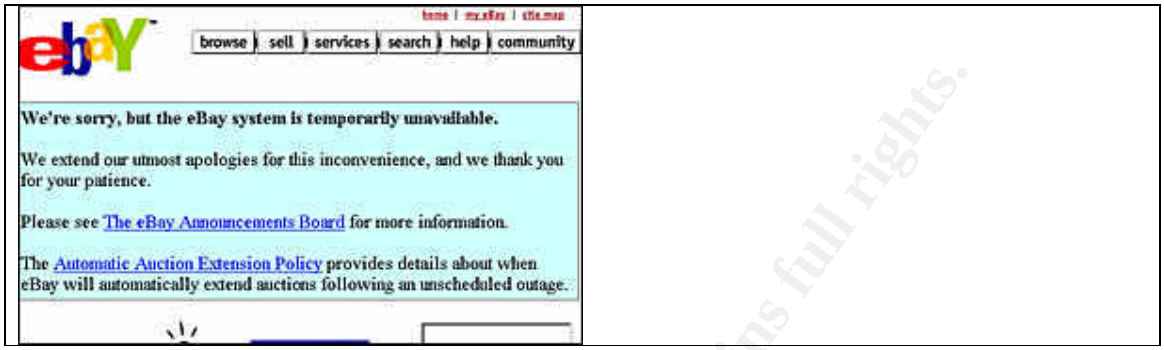
The file was exported from the image using the "Export" functionality of Autopsy. First of all let's do a *file* command against the file "ebay300.jpg" It appears first in the timeline on July the 14th at 14:12:14. The output of the *file* command says that the file is a normal jpeg. So let's view it. The jpeg shows a part of a screenshot of the eBay[9] webpage saying that the eBay webpage is temporarily down. Because of the fact that the message in the screenshot is written in English it could be the eBay webpage of a country with English as the native language e.g. USA, Canada or Great Britain. No additional information is given directly in the screenshot. At this point there is no direct connection to the suspect or any

[8] http://rpmfind.net/linux/RPM/redhat/8.0/i386/nc-1.10-16.i386.html
[9] www.ebay.com

other discovered information in the case. Maybe the later investigations using the binary "prog" will discover interesting information's in this file.

```
[root@localhost Binary]# file images-fl-160703-jp1.dd-.May03.ebay300.jpg
images-fl-160703-jp1.dd-.May03.ebay300.jpg: JPEG image data, JFIF standard 1.01, resolution (DPI), 96 x 96
[root@localhost Binary]#
```

Table 3-12 *file* command against May03.ebay300.jpg



Picture 3-5 the file ebay300.jpg

3.3.7 The file: .~5456g.tmp

This file first appears in the timeline on July the 14th at 14:13:52 and was exported again using Autopsy. The *file* command says that this file contains plain data. The output of the *strings* command gives no valuable information.

```
[root@localhost Binary]# file images-fl-160703-jp1.dd-...5456g.tmp
images-fl-160703-jp1.dd-...5456g.tmp: data
[root@localhost Binary]# strings images-fl-160703-jp1.dd-...5456g.tmp
FfM}8
qQJ9
Q9]Y
*t,v
oQGn5%<
$:g=
%!MX
&vTZ
&(ms
V|T^
]r-s
i:$(
=oXYaDf9$
O*f0
J%o@
iOgYa
DMnH
7X?`
H%sSd}
LIzo
KW"o
      )SZ
G@ho
glR3;
OH1M
.,A2
      >H5
~(2N##
ANQs
`N?:S*
D>QW
```

Table 3-13 *file* and *strings* against 5456g.tmp

3.4 Investigation of the floppy disk content with the prog binary

As we discovered in chapter 3.3.4 the binary “prog” is used to hide and unhide information in file systems. Therefore it is now necessary to go on with the investigations using the “prog” tool and check if there is any information hidden in the file system of the floppy disk of the suspect and how the program works. So first I made a floppy disk from the given dd image. Beforehand I made a copy of the given image file and renamed it to floppy.dd

```
[root@localhost Binary]# dd if=/GCFA/Binary/floppy.dd of=/dev/fd0
2880+0 records in
2880+0 records out
[root@localhost Binary]# mount -a /dev/fd0 /mnt/floppy
[root@localhost Binary]# cd /mnt/floppy
 [root@localhost floppy]# ./prog --mode chk nc-1.10-16.i386.rpm..rpm
nc-1.10-16.i386.rpm..rpm does not have slack
[root@localhost floppy]# ./prog --mode chk ./Docs/DVD-Playing-HOWTO-html.tar
./Docs/DVD-Playing-HOWTO-html.tar does not have slack
[root@localhost floppy]# ./prog --mode chk ./Docs/Kernel-HOWTO-html.tar.gz
./Docs/Kernel-HOWTO-html.tar.gz does not have slack
[root@localhost floppy]# ./prog --mode chk ./Docs/Letter.doc
./Docs/Letter.doc does not have slack
[root@localhost floppy]# ./prog --mode chk ./Docs/Mikemsg.doc
./Docs/Mikemsg.doc does not have slack
[root@localhost floppy]# ./prog --mode chk ./Docs/MP3-HOWTO-html.tar.gz
./Docs/MP3-HOWTO-html.tar.gz does not have slack
[root@localhost floppy]# ./prog --mode chk ./Docs/Sound-HOWTO-html.tar.gz
./Docs/Sound-HOWTO-html.tar.gz has slack
```

Table 3-14 creating a floppy disk from the disk image

To check if a file has hidden data the modified “bmap” tool found as “prog” on the evidence disk Tag# fl-160703-jp1 has to be used with the following command line “# ./prog --mode chk filename”. The knowledge of the usage of the binary “prog” was the result of executing the binary with the known help parameter (# prog --h”) to view the help output which was already discovered during the *strings* analysis of the binary “prog”. If a file has so called “slack” it will give the corresponding information on the standard output as seen in the command sequence above (table 3-14).

So how does this program work? What is “slackspace” You will find this answer again in part of the case conclusion in this chapter. Every physically data carrier is before it could store any data logically divided into tracks, sectors and blocks. Blocks are the smallest logically entity and what is more important fixed in the size and set during the initialization process of a data carrier. In the Linux ext2 file system the typical block sizes are 1, 2 and 4 KB. What the “prog” binary does is the following. If a file is smaller then the fixed minimum block size there will still remain space in this block to store information which is called “slack” or “slackspace”. This information is totally transparent to normal file operations and could not be discovered without special tools or procedures and by this very, very hard to detect. The usage of the “prog” binary will not alter any of the normal information’s like used or free disk space, file size and even the mactime attributes stay unchanged! To check the following procedure was used. I first copied the extracted binary to the /tmp directory of the forensic workstation from the GCFA path. Now I had to add the executable attribute to the file to make it executable. Afterwards I did a directory listing to show the directory and the files with their attributes like size, last date of change and so on. Then I took an already there copied file named “timeline” and checked it for available slackspace (prog -- mode sb timeline). The result says that the file “timeline” has a slackspace of 4053 bit. So now added the data “this is a test” to slack space and afterwards I did a check with the tool to see if the slack was added and then I did a

directory listing to look if something changed or not. Ay you can see in table 3-15 nothing changed. Neither the file size nor the access time has changed. So let's now check if the injection of the information did work correctly. The entered information can be displayed using the "s" option. The last thing I did was to clear the used slackspace. This is achieved by using the "w" option on the file and the checked by using the "chk" option to check if the file has slack. At the end I again checked via doing a directory listing if anything did change. As you can see nothing changed.

```
[root@localhost tmp]# cd /GCFA/Mr_Price/host1/output/
[root@localhost output]# cp images-fl-160703-jp1.dd-.prog /tmp
[root@localhost output]# chmod +x images-fl-160703-jp1.dd-.prog
[root@localhost output]# ll
total 1144
-rwx--x--x    1 root     root       487476 Jun 23 14:02 images-fl-160703-jp1.dd-.prog
-rw-r--r--    1 root     root         4139 Apr 15 12:21 timeline
[root@localhost output]# ./images-fl-160703-jp1.dd-.prog --mode sb timeline
4053
[root@localhost output]# ./images-fl-160703-jp1.dd-.prog --mode p timeline
stuffing block 25723459
file size was: 4139
slack size: 4053
block size: 4096
this is a test
[root@localhost output]# ./images-fl-160703-jp1.dd-.prog --mode chk timeline
timeline has slack
[root@localhost output]# ll
total 1144
-rwx--x--x    1 root     root       487476 Jun 23 14:02 images-fl-160703-jp1.dd-.prog
-rw-r--r--    1 root     root         4139 Apr 15 12:21 timeline
[root@localhost output]# ./images-fl-160703-jp1.dd-.prog --mode s timeline
getting from block 25723459
file size was: 4139
slack size: 4053
block size: 4096
this is a test
[root@localhost output]# ./images-fl-160703-jp1.dd-.prog --mode w timeline
stuffing block 25723459
file size was: 4139
slack size: 4053
block size: 4096
write error
write error
write error
[root@localhost output]# ./images-fl-160703-jp1.dd-.prog --mode chk timeline
timeline does not have slack
[root@localhost output]# ll
total 1144
-rwx--x--x    1 root     root       487476 Jun 23 14:02 images-fl-160703-jp1.dd-.prog
-rw-r--r--    1 root     root         4139 Apr 15 12:21 timeline
 [root@localhost output]#
```

Table 3-15 test and usage of the binary "prog"

So after I discovered how the program works it is time to operate the binary "prog" on the files on the seized evidence floppy disk. The first file with "slack" I found was the file "Sound-HOWTO-html.tar.gz". To show or extract hidden data the "prog" tool needs the following command line "# ./prog --mode s filename". Using this command line I extracted the hidden data and redirected the output to a new file. Now it was time to see what I had extracted. It was a gzip archive according to the test with the file command shown in table 3-16.

```
[root@localhost floppy]# ./prog --mode s ./Docs/Sound-HOWTO-html.tar.gz > /GCFA/Binary/extract
getting from block 190
file size was: 26843
slack size: 805
block size: 1024
[root@localhost floppy]# cd /GCFA/Binary
[root@localhost Binary]# file extract
extract: gzip compressed data, was "downloads", from Unix
[[root@localhost Binary]# mkdir extract-1
[root@localhost Binary]# mv extract extract-1/
[root@localhost Binary]# cd extract-1
[root@localhost extract-1]# mv extract extract.gz
[root@localhost extract-1]# gunzip extract.gz
[root@localhost extract-1]# ll
total 4
-rw-r--r--    1 root     root          185 Apr 18 13:31 extract
```

```
[root@localhost extract-1]# file extract
extract: ASCII text
[root@localhost extract-1]# more extract
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***
[root@localhost extract-1]#
```

Table 3-16 hidden data in the file "Sound-HOWTO-html.tar.gz"

So I created a new subdirectory on the forensic workstation named "extract-1", moved the file there, renamed it to extract.gz because of the *gunzip* command and unzipped it. Then again I did a *file* command to the received file to see what kind of document I had received. As you can see in the command sequence in table 3-15 *file* said that it was an ASCII document so I executed the *more* command to it. The output can be seen in table 3-15. The given outputs were URL's of different web pages. Because of their names it seems that they were dealing with the distribution of files. A short check if any of the mentioned web pages is still online led to no additional information. The three URL's did not have a current DNS entry. The fourth and last entry is redirected to a URL[10] of a travel agency.

Now I will continue with the remaining files to check if there is other hidden data in it. As you can see in table 3-17 none of the other files did contain any "slack".

```
[root@localhost floppy]# ./prog --mode chk ./John/sect-num.gif
./John/sect-num.gif does not have slack
[root@localhost floppy]# ./prog --mode chk ./John/sectors.gif
./John/sectors.gif does not have slack
[root@localhost floppy]# ./prog --mode chk ./May03/ebay300.jpg
./May03/ebay300.jpg does not have slack
[root@localhost floppy]# ./prog --mode chk nc-1.10-16.i386.rpm..rpm
nc-1.10-16.i386.rpm..rpm does not have slack
[root@localhost floppy]# ./prog --mode chk prog
prog does not have slack
[root@localhost floppy]# ./prog --mode chk .~5456g.tmp
.~5456g.tmp does not have slack
```

Table 3-17 checking files for hidden data

3.5 Conclusion

3.5.1 Case information and conclusion

Now let's sum up what we found out. First of all the time frame within all actions on the questioned evidence the floppy disk with evidence tag # fl-160703-jp1 did take place is from January the 28th 15:56:00 to July the 16th 06:12:45. Most of the action within the file system is located on July the 14th.
The user which had access to the disk, modified it, added files, accessed files and so on had the UID 502 and belongs to a user group with the GID 502. If there are any written documents, backups or any other information which contains a clear and reliable relation

[10] http://www.travelnow.com/index.jsp?cid=58623&home=truedown/secret.htm

between UID, GID and real names it could be proofed if this disk belongs to the suspect Mr. John Price or not (see chapter 3.2).
In the file and directory structure of the floppy disk image a subdirectory with the name "John" was found. Because of the fact that the suspect's first name is John this might be another indicator that the questioned disk belongs to Mr. Price. This assumption is based on the common knowledge that users often put private documents in subfolders named like themselves (see chapter 3.2).
The content of the found pictures on the disk named "sectors.gif" and "sect-num.gif" fit in the overall picture and the purpose of the binary "prog". Additionally the other investigated files "DVD-Playing-HOWTO-html.tar", "Kernel-HOWTO-html.tar.gz", "MP3-HOWTO-html.tar.gz" and "SOUND-HOWTO-html.tar.gz" fit in the topic about copyright protected material. Nevertheless the files themselves do not contain any illegal content.
The files "Letter.doc" and "Mikemsg.doc" (chapter 3.3.3) play a special role. First of all both files state in the document properties dialog a person called "John Price" as the author of these documents. Because of this fact and the fact that these files were found on the evidence they are a very good and clear link between the evidence and the suspect. Second the file "Mikemsg.doc" informs us that the person who wrote the letter is not working on its own. He has partner or accomplice named "Mike" which is the only known information about this person.
As shown in chapter 3.3.4 the questioned binary "prog" is a modified version of the binary "bmap". What exactly is changed could not be evaluated without the source code of the "prog" program despite of the obvious changes to the help output. What was shown is that the functionality of both programs is the same. Both aim for the functionality to hide information or files in the file system itself. Therefore they use a special characteristic of the file system, the technically called "slack" or "slackspace". So what is this? To understand it I will shortly jump into this part of a file system. Every physically data carrier is before it could store any data logically divided into tracks, sectors and blocks. Blocks are the smallest logically entity and what is more important fixed in the size and set during the initialization process of a data carrier. What the "prog" or "bmap" binary do is the following. If a file is smaller then the fixed minimum block size there will still remain space in this block to store information which is called slack. This information is totally transparent to normal file operations and could not be discovered without special tools or procedures as described in the next chapter 3.5.2.
Additionally the change of the source code of the bmap program, the fact that a redhat 8.0 system which was used for the compilation of the "prog" binary and the modification of the "make" file gives us a small part of the profile of the person who did this all. This person must have had access to a redhat 8.0 workstation and has brief programming skills and knowledge.
For the next found file "nc-1.10-16.i386.rpm..rpm" it was shown that this is the a regular and unchanged version of *netcat*, the well known Swiss army knife for networking purposes. The last two files "ebay300.pjg" and ".~5456g.tmp" did not add any valuable information to the case. All 3 files have one thing in common. No additionally information could be gathered from them.

In one of the files on the floppy disk information was hidden (chapter 3.4). This was the file "Sound-HOWTO-html.tar.gz". The content which was found there were information's

leading the investigation to a crime related to the illegal distribution of copyrighted material. This is especially indicated by the found words “Ripped MP3s – latest releases:”

Most of the information found during this investigation lead to a crime related to the prohibited distribution of copyright protected material, especially material like MP3 files from copyright protected music. How far the distribution reaches is unknown. The possibilities range from just within the internal network owned by the company the suspect is employed at or even further up to connections up to he whole Internet. To evaluate the grade of the violation more information is needed. Especially about the network and the infrastructure of the company the suspect is employed at and the measurements which are in place protecting employees from a free connection to the Internet. If the network is protected by systems like IDS, firewalls and so on maybe in their log files interesting information could be found and the logging of the IP address of the workstation of the suspect might give additionally links between the suspect Mr. Price and the IP address of his workstation and by this a link between Mr. price and the case.

3.5.2 How to defend against the unwanted usage of slackpace

We learned during the investigation of this case that it is quite easy for someone to hide information and files in space on a data carrier which is under normal circumstances not detectable because a special kind of space is used which is “not” existent from the point of view of the file system.
So how can we defend against this? Sorrowfully it is something which can only be detected on the local workstation. So there is no centralized attempt? What else choices do we have? Why not use the tool ”bmap” itself for the detection? I am using the original and not the discussed “prog” because the investigator and administrator can only use tools which he could gain access to or retrieve information about.
As we found out by the help file the command “bmap -- mode checkslack *FILENAME*” will give us the information about a file if it has slack or not. Sorrowfully the “bmap” tool is not capable of doing this check recursively and using wildcards either. So lets put around this command a view more commands and we receive exactly what we need. A script that recursively checks from a defined starting point in the directory hierarchy any file beneath for slack. So here is the necessary command line:

```
find / -exec ./bmap --mode checkslack {} \; 2>&1 | grep 'has slack'
```

The command line consists out of a combination of the commands “find”, “bmap” and “grep”. The command works quite simple. “find” simply recursively checks the file system. The output of find is uses as input for the “bmap” command and “grep” is used as a filter to show only the positive findings.
So additionally to make this a bit more comfortable for a company with a quite a lot of workstations this script could be run on any workstation timed by a cron job e.g. every night at 12 pm and the output is transferred to an central logging server which again checks the received log files for the occurrences of “has slack” and mails the corresponding log file to the responsible administrator.

3.5.3 Interview Questions

The following interview questions were based on the information gathered during the investigation of the questioned floppy disk with the evidence tag# fl-160703-jp1 or the information given in the case description by GIAC. The questions were aiming to help to make a clear connection between the suspect Mr. Price and the floppy disk to direct him into contradictions and thereby find out the truth.

Questions:

1. Mr. Price, can you please give us a short description of your job? What role does your computer play in your daily work?
2. Mr. Price, where did you spend the afternoon of July the 14th and the morning of July the 16th? Do you have any witnesses who can testify your answers? When do you normally arrive at your workplace? When did you arrive at your workplace on July the 16th? Can you proof your answer somehow?
3. Did you ever have access to a redhat Linux system? What did you do with this system? Do you know what version of redhat was installed on that or the various systems you used?
4. Mr. Price, could you please explain how the floppy disk with evidence tag # fl-160703-jp1 did get into the floppy drive of your office PC? If you do not have any idea, do you have any explanation how this floppy disk got into your workplace computer or by whom?
5. Mr. Price, do you use email as method of communication? How often do you use it? What are the typical person's you mail with?
6. Mr. Price, what is the normal text editing program you use for writing letters? Do you also know the version of this program? When did you last use this program?
7. Mr. Price, can you please explain the moment when the hard disk drive of your office PC crashed or was accidentally erased? Do you have any idea how this happened? Did this happen before? What kind of operating system do you use on your computer? Do you also have a computer at home? If yes, what kind of operating system do you use there?
8. Mr. Price did you ever try to write a computer program by your own? What programming language did you use? Do you have any experience programming on a Linux platform?
9. Mr. Price, on the floppy disk we found two Microsoft Word documents which clearly state you as the author of these documents. Are you the author of these documents? Furthermore one of the documents is written to a person called "Mike" and is signed with your initials "JP". Do you know any person named Mike? Did you ever write an email to a person called Mike?
10. Mr. Price, what is the way you normally name your folders and subfolders. How do you organize your stuff? Do you have any naming convention your normally use? Do you have any idea why there is a directory on the questioned floppy disk which is named like your surname "John"?
11. Mr. Price, can you explain how the documents named "Letter.doc" and "Mikemsg.doc" got onto the questioned floppy disk and are you the author of these documents as it is shown in the corresponding Microsoft Word dialogue?

3.6 Legal implications

As I am a German citizen the whole consideration of the legal implications will be done according to the German law.

First of all depending on the fact if there is one or more Company policies which deal with the above mentioned crimes the suspect might have committed these policies have to be observed if they were violated by the employee. Normally an employee of a company is legally bound to these policies due to the fact that he signed his employment contract. Additionally within these policies the necessary procedures are declared how to react on the violation of one or more rules of the policy. This might have consequences of civil law like extraordinary cancellation of the employment, a liability law suit against the employee forcing him to face the full responsibility for all consequences of his actions and so on. These are the direct consequences which arise due to a violation of company policies between an employee and an employer.

For the assumed violation of the German copyright protection law (Urheberrecht[11], UrhG) we will first have to check if music is protected by the named law. The protected work is declared in § 2[12] of the German UrhG. In article 2 of § 2 of the German UrhG it is said that any kind of music work is protected by that law (§ 2, article 2, Werke der Musik) and therefore we have to observe the UrhG for the investigation of the legal implications of the assumed crime.

For the fact that in this case we are dealing with the assumed illegal distribution of copyright protected material we have to consult the referring paragraphs of the law. The right of distribution of copyright protected work is defined in § 53[13] of the UrhG (Vervielfältigungen zum privaten und sonstigen Gebrauch). In article 6 it is said that any distribution of protected material is illegal. Duplication of copyright protected work is just allowed for private backup reasons and only by the legitimate owner of the work. According to the UrhG § 69f[14] the aggrieved party (the owner of the protected work) has the right that any illegal copy of the work is eliminated according to § 98[15], article 2 and 3 of the UrhG. In § 97[16] article 1 the claim for damages is defined. The amount of the damage will be decided by court. In addition to § 98, article 1 also § 100[17] has to be observed because as in this case if an employee of a company commits violation according to § 53 of UrhG the aggrieved party can also claim the company for damage

[11] http://bundesrecht.juris.de/bundesrecht/urhg/index.html
[12] http://bundesrecht.juris.de/bundesrecht/urhg/__2.html
[13] http://bundesrecht.juris.de/bundesrecht/urhg/__53.html
[14] http://bundesrecht.juris.de/bundesrecht/urhg/__69f.html
[15] http://bundesrecht.juris.de/bundesrecht/urhg/__98.html
[16] http://bundesrecht.juris.de/bundesrecht/urhg/__97.html
[17] http://bundesrecht.juris.de/bundesrecht/urhg/__100.html

payments. The next paragraph which is relevant is § 99[18] (Anspruch auf Vernichtung oder Überlassung der Vorrichtungen) in which is defined that the aggrieved party has the right to claim the violator for the destruction or the disposal of all used equipment in the relevant case used for the illegal duplication and distribution of the questioned material. The limitation of this crime is defined in § 102[19] of the UrhG which refers to article 5 of the first book of the BGB[20] (Bürgerliches Gesetzbuch) Did the accused party gain any profit due to the violation of § 53 of the UrhG then § 852[21] of the BGB has to be observed.

So for the company it could become very important whether there are any policies in place which Mr. Price is bound to or not. Very important in that case is an agreement that Mr. Price personally has to face all consequences of his actions. Otherwise the company itself will have to face law suit with a claim for damages according to § 100 UrhG.

[18] http://bundesrecht.juris.de/bundesrecht/urhg/__99.html
[19] http://bundesrecht.juris.de/bundesrecht/urhg/__102.html
[20] http://bundesrecht.juris.de/bundesrecht/bgb/index.html
[21] http://bundesrecht.juris.de/bundesrecht/bgb/__852.html

4 Part II: Analysis of a hacked system

4.1 The Synopsis

On March the 5th 2004 the responsible system administrators of an Internet/Intranet web server at our company were running a few system audits. Goal of these audits was to check the system for unwanted changes and stability. During this audit the administrators came across a view abnormal behaviors of the system. One was quite obvious, the lack of performance. First the administrators began to do online updates to the system but with no success. Than due to the fact that the output of normal system information tools like the "top" command did not show any load on the machine or any other unusual information the administrators came after a few hours of investigation to the clue that the machine might had been compromised and so decided to analyze the system by themselves. During their investigation the administrators installed quite a lot of software like system updates, Snort[22], chkrootkit[23] and further. With the tool chkrootkit they found out that the system was compromised with the "SuckIt" rootkit in version 1.3. Meanwhile the weekend passed by and the administrators continued "their work" on Monday, March the 8th. One of them found on the Internet a tool called "skdetect 0.4b[24]" In hope that this tool will "disable" the rootkit the software was also installed and started. After the first run the tool stated out that it found the "SuckIt" rootkit, disabled and deleted it. At this time the administrators partly began to realize that they were losing control of what they were doing. Sorrowfully what they did not realize is that they were messing up forensic evidence with enormous speed now for about three days. Nevertheless their main interest was to get the system back online with a reliable performance. So in the late afternoon on March the 8th the security team was "accidentally" informed by mail. Shortly before the administrators began a total new installation of the machine we could stop them from doing it and talked to their superior. During the talk he clearly stated out that his main goal was to bring the system back online because of the service level agreements he is responsible for, so the only thing which I was allowed was to make an online image of the machine and afterwards the machine will be rebuild immediately. This was an exclusive admission I was given. This meant I was allowed to make the image but not allowed to do anything on the console of the compromised system. This was quit a big drawback because by this order I lost a lot of forensic information's like open network connections, running processes, memory and so on.

4.2 The forensic hardware

First of all let me describe the hardware which was used in this case. For the imaging purpose I have used a small Dell x86 workstation named Optiplex GX 260, 512 MB RAM,

[22] http://www.snort.org/
[23] http://www.chkrootkit.org/
[24] http://www.gnu.org/directory/network/security/skdetect.html

onboard Intel 10/100 Mbit Ethernet card, Intel onboard graphic card and a cdrom to boot from. The serial number of this machine is 1KGZL0J, no inventory number. As operating system on this machine I used the bootable Linux cdrom distribution fire[25]. In this case version 0.4a of the fire distribution was used. The imaging process is described in chapter 4.4.
The forensic workstation which was used is a HP x86 workstation VL Series 420, 512 MB RAM, onboard Intel 10/100 Mbit Ethernet card, ATI Rage 128 Pro graphic card, DVD ROM, 3,5" floppy drive and a 200 GB Western Digital IDE hard drive with a clean installation of RedHat Fedora Core 1[26], Sleuthkit 1.68[27] and Autopsy 2.0[28]. The serial number of the forensic workstation is "NL22510214" and the internal inventory number is "WS986790". During this investigation Sleuthkit 1.69 was released on April the 20th and the forensic workstation was upgraded with it. Every described step of the investigation of part II of this document was done with Sleuthkit 1.69. Part I of this practical was done with Sleuthkit 1.68.
The whole investigation takes place in our newly built up CERT lab. It is a special room which is normally and automatically locked and only selected people have access to it and the access is logged. This is also the place were the collected evidences and the corresponding case documentation will be stored in a locked cabinet.

4.3 Hardware of the compromised system

The hardware of the compromised system consists of a Compaq Proliant ML370 server. The serial number of the system is "8151JSS11262" and the internal inventory number is "LSY 7111538". The system is equipped with 2048 MB of RAM, a Intel Pentium III 1,13 Ghz CPU, 1 onboard 10/100 Mbit network interface and 1 3COM TX985 10/100 Mbit network card (eth0 and eth1), 4 x 18,2 GB Ultra SCSI 320 hot swap hard disk drives, 1 x 146,8 GB Ultra SCSI 320 hot swap hard drive and a onboard graphic card. The serial numbers of the hard disk drives could not be checked because I was not allowed to take the system down by the managing directors (to check the serial numbers I would have had to pull out the hard disks out of its hot swap rack).

4.4 The purpose of the compromised system

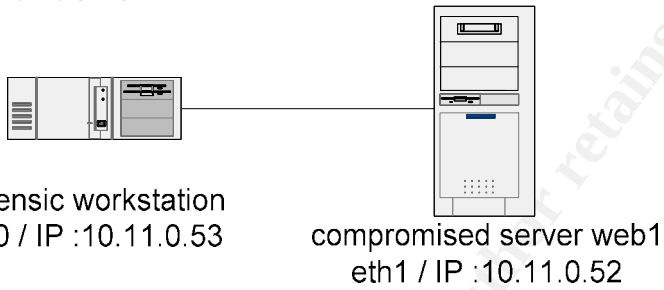
The main purpose of the compromised system is as an Intranet web server which is also available via the Internet. This is quite an unusual situation for an Intranet server but because of the fact that the company has a headquarter with a centralized data center

[25] http://fire.dmzs.com/ FIRE is a bootable cdrom based distribution with the goal of providing an immediate environment to perform forensic analysis, incident response, data recovery, virus scanning and vulnerability assessment.
[26] http://fedora.redhat.com/
[27] http://www.sleuthkit.org/sleuthkit/index.php
[28] http://www.sleuthkit.org/autopsy/index.php

and many, many traveling salesman which need regularly access to the Intranet via an Internet connection. Why for better security not a VPN is used is unclear but definitely would be an over all improvement.
As far as the administrators were able to tell me the system is based on a standard installation of rehat Linux 7.2. What actual patch level the system has is unknown. The used web server is an Apache[29] with PHP installed and a connected MySQL database for information retrieval.

4.5 Imaging of the compromised system

As already said in the synopsis I was not allowed to take the system down, check the system online for interesting information's like open network connections, memory, running processes neither make a real clone of the hard drives! So the only thing I was allowed to was to image the system online via the network. So I and the system administrator of the compromised server built up the following network configuration for the imaging purpose:



Picture 4-1 network configuration during backup

On the compromised system the network interface eth1 was used because this one only is used for backup needs and by this the system is still online via interface eth0. First I connected two brand-new Samsung 160 GB IDE hard disk drives to the DELL workstation for the backup purpose. The serial numbers of the hard disk drives are "S01FJ10X242734" and "S01FJ10X242737". The first hard disk (serial number S01FJ10X242734) was labeled with the evidence tag "buderus_2004_03_08_0001" and the second hard disk (serial number S01FJ10X242737) was labeled with the evidence tag "buderus_2004_03_08_0002". Then I booted the backup workstation using the fire 0.4a CD. After the initial configuration (setting up the network) of the newly booted system I choose the command line option of the fire boot menu. Within there I first checked that the hard disks were correctly recognized and fully functional. Now I formatted the newly installed hard disks using the shown commands in table 4-1 on the DELL backup workstation using the booted Linux from the fire 0.4a CD.

[29] http://www.apache.org

```
[root@FIRE] ~> dd if=/dev/zero of=/dev/hdc
312576642 +0 records in
312576642 +0 records out
[root@FIRE] mkfs -t ext2 /dev/hdb1
mke2fs 1.27 (08-Mar-2002)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
19546112 inodes, 39072080 blocks
1953604 blocks (5.00%) reserved for the super user
First data block=0
1193 block groups
32768 blocks per group, 32768 fragments per group
16384 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872

Writing inode tables: done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 36 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
[root@FIRE] ~> dd if=/dev/zero of=/dev/hdd
312576642 +0 records in
312576642 +0 records out
[root@FIRE] mkfs -t ext2 /dev/hdc1
mke2fs 1.27 (08-Mar-2002)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
19546112 inodes, 39072080 blocks
1953604 blocks (5.00%) reserved for the super user
First data block=0
1193 block groups
32768 blocks per group, 32768 fragments per group
16384 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872

Writing inode tables: done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 36 mounts or
```

Table 4-1 formatting the hard disk of the backup workstation

After formatting the hard disks I created a directory on the first hard disk called “buderus-web1-1” for the first images. Afterwards I set up a *netcat* server listening on port 31337. Again the *netcat* binary from the fire 0.4a CD was used.

Because of the fact that the compromised system was or still is rootkitted there is the possibility that some binaries are not doing what they should do. So to ensure that the imaging process is not somehow affected by compromised binaries a clean and reliable binary of *netcat* and *dd* had to be used. This was achieved by using a reliable *netcat* and *dd* binary from the fire 0.4a CD. Therefore first the binaries were copied form the fire CD to a fabric new floppy disk which is first formatted and then the “nc” and “dd” binaries were copied. The corresponding commands are shown in table 4-2.

```
[root@FIRE] mkfs –t ext2 -c /dev/fd0
mke2fs 1.27 (08-Mar-2002)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
184 inodes, 1440 blocks
72 blocks (5.00%) reserved for the super user
First data block=1
1 block group
8192 blocks per group, 8192 fragments per group
184 inodes per group

Writing inode tables: done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 23 mounts or
```

```
180 days, whichever comes first.  Use tune2fs -c or -i to override.
[root@FIRE] mount /dev/fd0 /mnt/floppy
[root@FIRE] cp nc /mnt/floppy
[root@FIRE] cp dd /mnt/floppy
```

Table 4-2 copying a reliable version of netcat to a floppy disk

I started the imaging process using the *dd* and *netcat* command for each partition. The command sequence is shown in the table below. Important is that only the reliable versions of *netcat* and *dd* from the floppy disk mentioned above were used. This is ensured by first mounting the floppy disk to "/mnt/floppy", changing to this directory and then via executing the binaries with the pre-sequence "./" which is a shell option that ensures that the file which is called is taken from the local directory and not via the global set path variable. For the last and biggest image I switched to the second hard disk drive and created a directory "buderus-web1-2". As you can see one important partition is missing, the swap partition. This is quite an interesting partition and often very useful in combination with using the *strings* command against it for gathering information. Sorrowfully at the point of the incident our CERT policy was not in place and between me and the responsible administrator came up a harsh discussion if imaging the swap partition might cause the running system to fail or not. He insisted that it would harm the system and therefore not allowed me to do it. I tried to convince him the best I could but even without an Internet connection and by this the possibility to show him that there is no known issue with this action I failed. The administrator called his managing director and he decided that I was not allowed to image the swap partition.

```
First the start of netcat server listening on port 31337 on the forensic workstation machine. Beneath the corresponding command line sequence

[root@FIRE] ~> nc –l –p 31337 > c0d0p3.dd

[root@FIRE] ~> nc –l –p 31337 > c0d0p1.dd

[root@FIRE] ~> nc –l –p 31337 > c0d2p1.dd

[root@FIRE] ~> nc –l –p 31337 > c0d0p6.dd

[root@FIRE] ~> nc –l –p 31337 > c0d0p7.dd

[root@FIRE] ~> nc –l –p 31337 > c0d1p1.dd

Beneath the command line sequence on the hacked system during the hard disk imaging

[web1] # mount /dev/fd0 /mnt/floppy
[web1] # cd /mnt/floppy
[web1] # ./dd if=/dev/cciss/c0d0p3 | ./nc 10.11.0.53 31337
4096320+0 records in
4096320+0 records out
[web1] # ./dd if=/dev/cciss/c0d0p1 | ./nc 10.11.0.53 31337
65248+0 records in
65248+0 records out
[web1] # ./dd if=/dev/cciss/c0d2p1 | ./nc 10.11.0.53 31337
286734208+0 records in
286734208+0 records out
[web1] # ./dd if=/dev/cciss/c0d0p6 | ./nc 10.11.0.53 31337
514048+0 records in
514048+0 records out
[web1] # ./dd if=/dev/cciss/c0d0p7 | ./nc 10.11.0.53 31337
54141568+0 records in
54141568+0 records out
[web1] # ./dd if=/dev/cciss/c0d1p1 | ./nc 10.11.0.53 31337
35553088+0 records in
35553088+0 records out
```

Table 4-3 imaging the hacked system

After the imaging process was finished and all partitions were copied I created md5 checksums of each image file for later checks of the integrity of the images. A comparison with the md5 sums of the hacked system at this time was not done because it would had been useless because of the fact that the system is still online and a lot of changes were made due to the fact that the system is still running and operating.

```
[root@FIRE] ~> md5sum c0d0p3.dd
c7fa3ea4ddb8f7ca257893558f25903b  c0d0p3.dd
[root@FIRE] ~> md5sum c0d0p1.dd
868cee5fbeaa12f6a5a35367d90b6026  c0d0p1.dd
[root@FIRE] ~> md5sum c0d2p1.dd
432c603189fa78af32955fa0484df072  c0d2p1.dd
[root@FIRE] ~> md5sum c0d0p6.dd
f6146639f33fb8df9f13c450a6006127  c0d0p6.dd
[root@FIRE] ~> md5sum c0d0p7.dd
1a6505fb82287d323d34867e33f9a661  c0d0p7.dd
[root@FIRE] ~> md5sum c0d1p1.dd
20131b9380ee4c05a0c6b3ec1e328d11  c0d1p1.dd
```

Table 4-4 generating md5 sums of the images

4.6 The Analysis

4.6.1 The initial tasks before the analysis can start

First of all the hard disks with the case tags #web1-0001 (Samsung IDE hard Drive # S01FJ10X242734) and #web1-0002 (Samsung IDE hard Drive # S01FJ10X242737) were installed in the forensic workstation. Afterwards the forensic system is booted. Now two directories (imagehd1 + imagehd2) were created in the mount path (/mnt) on the forensic workstation and the hard disks with the images of the partitions of the hacked system were mounted to these two directories as shown in table 4-5.

```
[root@localhost root]# cd /mnt
[root@localhost root]# mdkir imagehd1
[root@localhost root]# mkdir imagehd2
[root@localhost root]# mount /dev/hdc1 /mnt/imagehd1
[root@localhost root]# mount /dev/hdb1 /mnt/imagehd2
```

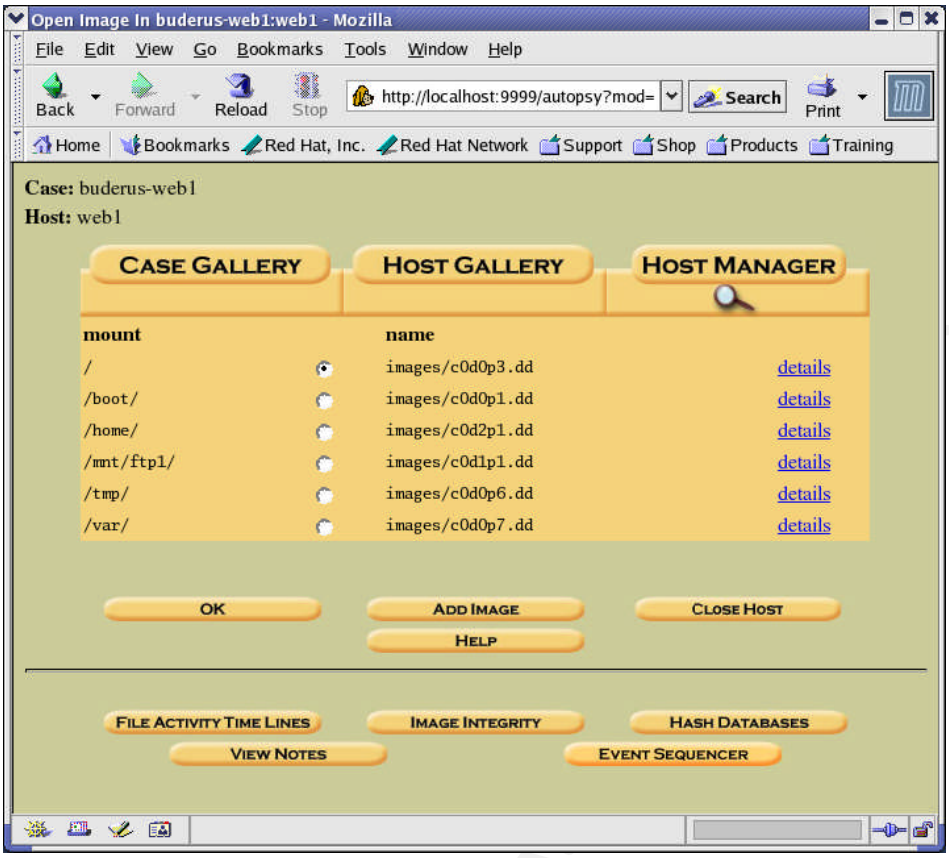
Table 4-5 mounting the hard disk images

After Autopsy is started using the corresponding command (./autopsy) the start page of Autopsy is requested using the mozilla web browser. A new case is opened called “buderus-web1” and a new host named “web1” is added. Now it is time to add the 6 images of the different partitions of the hacked system to Autopsy with the corresponding mount points as shown in table 4.7. Now for every added partition the unallocated space and the timeline have been created.

Image name/ partition	Mount point	Size
c0d0p3.dd	/	2 GB
c0d0p1.dd	/boot	0,0 3 GB
c0d2p1.dd	/home	140 GB
c0d0p6.dd	/tmp	2,4 GB
c0d0p7.dd	/var	26 GB
c0d1p1.dd	/mnt/ftp1	17 GB

Table 4-6 the partition table and the mount points

The result of opening a new case, importing all images, creating timeline and unallocated space is shown in picture 4-2.



Picture 4-2 the ready to investigate autopsy case

4.6.2 The history of the hacked system

As none of the interviewed administrators were able to proof the date when the system was first installed I first tried to figure out the installation date. This could be done via checking the first appearance of the vmlinuz file in the timeline. vmlinuz is the kernel file of a Linux system. Its first appearance in the timeline indicates the date a Linux kernel was firstly copied or compiled to a system. The next interesting information to seek for is to check the version of a redhat system which is stored in the “redhat-release” file located in the “/etc” directory. The content of the file is shown in Table 4-7. As I know from the interview with the system administrator the system should be using a 2.4.7-10 enterprise kernel.

Red Hat Linux Release 7.2 (Enigma)

Table 4-7 redhat-release file

Now let’s check the installation date and kernel type via checking the timeline. As we can see in table 4-8 the vmlinuz file was first written and compiled to the system on September the 6th 2001 at 21:05:13 so probably the system was installed first on the 6th September 2001 because there is no earlier appearance of the vimlinuz file in the history. In the timeline you can see the first appearance of the vmlinuz files and afterwards a lot of kernel drivers were written to the system. At 21:21:09 the kernel is again touched, modified and other drivers were added to the system.

```
Thu Sep 06 2001 21:05:13   862199 m.. -/-rw-r--r-- root     root     31       /boot/vmlinuz-2.4.7-10enterprise
                            13598 ma. -/-rw-r--r-- root     root     30       /boot/module-info-2.4.7-10enterprise
                           452303 m.. -/-rw-r--r-- root     root     29       /boot/System.map-2.4.7-10enterprise
Thu Sep 06 2001 21:05:22    15652 m.. -/-rw-r--r-- root     root     210864   /lib/modules/2.4.7-
10enterprise/kernel/drivers/block/paride/pt.o;3ce1089d (deleted-realloc)
                           350088 m.. -/-rw-r--r-- root     root     210826   /lib/modules/2.4.7-10enterprise/kernel/drivers/atm/ambassador.o
                             8000 m.. -/-rw-r--r-- root     root     210868   /lib/modules/2.4.7-10enterprise/kernel/drivers/bluetooth/hci_uart.o
                             6868 m.. -/-rw-r--r-- root     root     210852   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/paride/fit3.o
                            10304 m.. -/-rw-r--r-- root     root     210857   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/paride/on20.o
                            10288 m.. -/-rw-r--r-- root     root     210849   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/paride/epat.o
                             6588 m.. -/-rw-r--r-- root     root     210867   /lib/modules/2.4.7-10enterprise/kernel/drivers/bluetooth/hci_emu.o
                           261756 m.. -/-rw-r--r-- root     root     210834   /lib/modules/2.4.7-10enterprise/kernel/drivers/atm/suni.o
                           411527 m.. -/-rw-r--r-- root     root     210829   /lib/modules/2.4.7-10enterprise/kernel/drivers/atm/fore_200e.o
                            22404 m.. -/-rw-r--r-- root     root     210861   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/paride/pd.o
                           410480 m.. -/-rw-r--r-- root     root     210833   /lib/modules/2.4.7-10enterprise/kernel/drivers/atm/nicstar.o
                            15652 m.. -/-rw-r--r-- root     root     210864   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/paride/pt.o
                            11132 m.. -/-rw-r--r-- root     root     210863   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/paride/pg.o
                             4524 m.. -/-rw-r--r-- root     root     210851   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/paride/fit2.o
                            24396 m.. -/-rw-r--r-- root     root     210865   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/xd.o;3ce1089d
(deleted-realloc)
                             8392 m.. -/-rw-r--r-- root     root     210847   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/paride/comm.o
                             9008 m.. -/-rw-r--r-- root     root     210850   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/paride/epia.o
                            11504 m.. -/-rw-r--r-- root     root     210846   /lib/modules/2.4.7-10enterprise/kernel/drivers/block/paride/bpck6.o
                           268124 m.. -/-rw-r--r-- root     root     210827   /lib/modules/2.4.7-10enterprise/kernel/drivers/atm/atmtcp.o


Thu Sep 06 2001 21:21:09   858769 ma. -/-rw-r--r-- root     root     18       /boot/vmlinuz-2.4.7-10smp
                           452106 m.. -/-rw-r--r-- root     root     16       /boot/System.map-2.4.7-10smp
                            13598 ma. -/-rw-r--r-- root     root     17       /boot/module-info-2.4.7-10smp
Thu Sep 06 2001 21:21:11    11616 m.. -/-rw-r--r-- root     root     225329   /lib/modules/2.4.7-10smp/kernel/drivers/block/paride/kbic.o
                           343448 m.. -/-rw-r--r-- root     root     49184    /lib/modules/2.4.7-10smp/kernel/drivers/atm/eni.o
                            13292 m.. -/-rw-r--r-- root     root     225319   /lib/modules/2.4.7-10smp/kernel/drivers/block/paride/bpck.o
                            27744 m.. -/-rw-r--r-- root     root     209475   /lib/modules/2.4.7-10smp/kernel/drivers/block/cciss.o
                            11504 m.. -/-rw-r--r-- root     root     225320   /lib/modules/2.4.7-10smp/kernel/drivers/block/paride/bpck6.o
                            10304 m.. -/-rw-r--r-- root     root     225331   /lib/modules/2.4.7-10smp/kernel/drivers/block/paride/on20.o
                             6868 m.. -/-rw-r--r-- root     root     225326   /lib/modules/2.4.7-10smp/kernel/drivers/block/paride/fit3.o
                            11860 m.. -/-rw-r--r-- root     root     209478   /lib/modules/2.4.7-10smp/kernel/drivers/block/nbd.o
                            10468 m.. -/-rw-r--r-- root     root     193277   /lib/modules/2.4.7-10smp/kernel/drivers/bluetooth/hci_usb.o
                           350056 m.. -/-rw-r--r-- root     root     49182    /lib/modules/2.4.7-10smp/kernel/drivers/atm/ambassador.o
                            11132 m.. -/-rw-r--r-- root     root     225337   /lib/modules/2.4.7-10smp/kernel/drivers/block/paride/pg.o
                            17160 m.. -/-rw-r--r-- root     root     209477   /lib/modules/2.4.7-10smp/kernel/drivers/block/loop.o
                            21880 m.. -/-rw-r--r-- root     root     225332   /lib/modules/2.4.7-10smp/kernel/drivers/block/paride/on26.o
```

Table 4-8 timeline of the first installation of the system

As you can also see the kernel which is created has the version number 2.4.7-10. So the information given by the system administrators is correct. A short search in the timeline for any changes to the kernel until the date of the imaging shows that the kernel from then on never was updated again.

4.6.3 Search for anomalies in the timeline

So first lets make a sum of what we know from the few information's the administrators told me in the interview before I run across the timeline. This should lead into a bit more coordinated way of the timeline analysis because I am dealing with a timeline on about 190 GB of hard disk space and timeline file size of 155 MB! Additionally I am facing the challenge to see if still evidences have been left by the system administrator and their analysis untouched. This will be one of the main questions of this investigation.

According to the information from the administrators they noticed that something is wrong with their system on the 5th March 2004. During their investigation they used tools like chkrootkit 0.43 and skdetect-0.4b because chkrootkit found the suckit rootkit v1.3 installed. The chkrootkit was downloaded to the hard disk at 13:55:14 as a gzip compressed tar archive which was decompressed at 13:55:24 and then executed. Afterwards the administrator executed the skdetect tool and rebooted the machine at 14:40. Sorrowfully the administrator did not save any of the output he received from the tools. Than he saved a few files he found or which were identified by the used tools to a directory called "/root/quarantaene". I will come to an analysis of these files later.

After saving these files the administrator finished his work. There is no interesting activity by the next two days (a weekend, Saturday and Sunday). On Monday, March the 8th the administrator returned to the machine and tried again by to doing some kind of

investigation. Again they used the skdetect tool. Than the administrator did something which is from a forensic point of view a disaster. He installed/updated Snort and activated it. With this action a lot of information for sure is gone because Snort is a very resource consuming tool which needs quite a lot of disk space in combination with its database (mysql). In addition this is a very unusual constellation. Normally an intrusion detection system (e.g. Snort) is installed in front of a web server on a separate machine and not on the server itself. All the information in the above paragraph can be found within the timeline extracts in chapter 4.6.6.

So far I know the following. The system was presumably compromised and the SuckIt rootkit was installed. Nevertheless I am facing two parties destroying evidence on the system, the system administrator and if the hacker was not a total rookie he for sure also tried his best to delete his evidences.

Due to the fact that there were no evidences in the timeline around the assumed date of compromise on January the 6th 2004 which lead to a direct hint how or when the system was hacked and I was not allowed to gather any other information's than the hard disk images I have only a few options on how to continue.

1. First I can examine the files found by the administrator in the timeline, when and how they first appeared. Maybe this will lead to a date when the system was compromised.
2. Secondly I can search the compromised system for log files of the different services running on it, for bash-history files with interesting information in it, for hidden directories and so on.
3. Thirdly I can do a check of the total timeline for any interesting anomalies. This is some kind of what I will call the last choice of a forensic analyst, because this is a very time consuming procedure; its success is quite unpredictable and in this case because of the huge amount of data in the timeline (the timeline is a 155 MB file) very, very time consuming.

4.6.4 Analysis of the files found by the system administrators

So let's first start with files the system administrators identified and saved as parts of the rootkit. These are the following files which had been saved to the locations in the file system shown in table 4-8.

Path	Name of the file	Descripton or purpose
/root/quarantaene/sbin/	init-infected	Init file of the rootkit
/root/quarantaene/usr/include/linux/modules/.lib.so/	.sniffer	A network sniffer or its output ?
/root/quarantaene/usr/include/linux/modules/.lib.so/	.syslogs	A system log file?
/root/quarantaene/usr/include/linux/modules/.lib.so/	syslogs	A system log file?

Table 4-9 saved files of the rootkit

Let's have a look inside these files using Autopsy and the *strings* command. But first check what kind of file I am dealing with using the *file* command.

```
[root@localhost reports]# file images-c0d0p3.dd-root.quarantaene.sbin.init-infected
images-c0d0p3.dd-root.quarantaene.sbin.init-infected: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), statically linked, stripped
[root@localhost reports]#
```

Table 4-10 *file* command on the init-infected

Now the strings command is applied to the file. As shown in table 4-11 there are a lot of wording hints marked in yellow which indicate that this modified init might be some kind of rootkit, trojan, backdoor or all together because you can find entries like “backdoor”, “hiding process id´s”, “redirecting the history to /dev/null”. Additionally you can find something that looks like the version information “1.3b” which corresponds to the reported output of the chkrootkit tool which claimed that the suckit rootkit version1.3b was installed.

```
The unreadable parts were deleted

 [^_]
[^_]
[^_]
[^_]
PATH=/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin:./bin:/usr/include/linux/modules/.lib.so/:/usr/include/linux/modules/.lib.so//bin
HOME=/usr/include/linux/modules/.lib.so/
PS1=\[\033[1;30m\][\[\033[0;32m\]\u\[\033[1;32m\]@\[\033[0;32m\]\h \[\033[1;37m\]\W\[\033[1;30m\]]\[\033[0m\]#
Can't open a tty, all in use ?
Can't fork subshell, there is no way...
/usr/include/linux/modules/.lib.so/
BD_Init: Starting backdoor daemon...
FUCK: Can't allocate raw socket (%d)
/usr/include/linux/modules/.lib.so//.rc
HISTFILE=/dev/null
SHELL=/bin/bash
TERM=linux
pqrstuvwxyzabcde
0123456789abcdef
/dev/ptmx
/dev/pty
/dev/tty
/dev/null
/bin/sh
Can't execve shell!
FUCK: Can't fork child (%d)
Done, pid=%d
use:
%s <uivfp> [args]
u       - uninstall
i       - make pid invisible
v       - make pid visible
f [0/1] - toggle file hiding
p [0/1] - toggle pid hiding
FUCK: Failed to uninstall (%d)
Suckit uninstalled sucesfully!
FUCK: Failed to hide pid %d (%d)
FUCK: Failed to unhide pid %d (%d)
Failed to change %s hiding (%d)!
Detected version: %s
Pid %d is hidden now!
Pid %d is visible now!
file
%s hiding is now %s!
__kmalloc
/dev/kmem
RK_Init: idt=0x%08x,
sct[]=0x%08x,
FUCK: Can't find kmalloc()!
kmalloc()=0x%08x, gfp=0x%x
FUCK: Out of kernel memory!
Done, %d bytes, base=0x%08x
FUCK: Can't open %s for read/write (%d)
FUCK: IDT table read failed (offset 0x%08x)
FUCK: Can't find sys_call_table[]
FUCK: Can't read syscall %d addr
Z_Init: Allocating kernel-code memory...
core
/sbin/init.syslogs
FUCK: Got signal %d while manipulating kernel!
0123456789abcdefghijklmnopqrstuvwxyz
0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ
<NULL>
/dev/null
1.3b
.syslogs
/usr/include/linux/modules/.lib.so//.syslogs
/proc/
/proc/net/
socket:[
/sbin/init
/sbin/init.syslogs
login
telnet
rlogin
rexec
passwd
```

```
adduser
mysql
ssword:
 [^_]
[^_]
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.1 2.96-98)
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.1 2.96-98)
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.1 2.96-98)
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.1 2.96-98)
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.1 2.96-98)
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.1 2.96-98)
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.1 2.96-98)
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.1 2.96-98)
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.1 2.96-98)
GCC: (GNU) 2.96 20000731 (Red Hat Linux 7.1 2.96-98)
01.01
01.01
01.01
01.01
01.01
01.01
01.01
01.01
01.01
01.01
.symtab
.strtab
.shstrtab
.text
.rodata
.data
.sbss
.bss
.comment
.note
```

Table 4-11 *strings* output of the init-infected file

Now let's check the timeline for the first appearance of this file. Therefore a search in the timeline is done using the search pattern "init-infected".

```
Tue Jan 06 2004 16:57:50     4096 m.c d/drwxr-xr-x root     root     164761   /usr/include/linux/modules
                                1 m.. -/-rw--w--w- root     root     164763
/root/quarantaene/usr/include/linux/modules/.lib.so/.sniffer
                            12288 m.c d/drwxr-xr-x root     root     17043    /usr/include/linux
                             4096 m.c -/drwxr-xr-x root     root     164761   /opt/phpMyAdmin-2.5.1/.htaccess.swp (deleted-realloc)
                                1 m.. -/-rw--w--w- root     root     164763   /usr/include/linux/modules/.lib.so/.sniffer (deleted-
realloc)
Tue Jan 06 2004 16:57:57    28344 m.. -/-rwxr-xr-x root     root     101472   /sbin/init-infected (deleted-realloc)
                            26636 m.. -/-rwxr-xr-x root     root     101180   /sbin/init.syslogs (deleted-realloc)
                            26636 m.. -/-rwxr-xr-x root     root     101180   /sbin/init
                            28344 m.. -/-rwxr-xr-x root     root     164765
/root/quarantaene/usr/include/linux/modules/.lib.so/syslogs
                            28344 m.. -/-rwxr-xr-x root     root     164765   /usr/include/linux/modules/.lib.so/syslogs (deleted-
realloc)
                            28344 m.. -/-rwxr-xr-x root     root     101472   /root/quarantaene/sbin/init-infected
```

Table 4-12 first appearance of the "init-infected" file in the timeline

As you can see the file first appears on the system on Tuesday the 6th January of 2004 at 16:57:57. In combination we see the other files which were under investigation ".sniffer" and "syslogs". So some when around this time it seems that the suckit rootkit was firstly installed or activated.

Now let's investigate the second file ".sniffer". Because of its file size of 1 byte it is not very likely to find anything interesting in it but let's see what the *file* command says about this file. The output of the *file* command is shown in table 4-12. It says that it is a PCX image data file. Quite an old file format nowadays and not so often used graphic file format any more.
So the strings command does not lead to any valuable information. So let's try to open the file with Adobe Photoshop© which is capable of the PCX file format. The file is opened, but Photoshop seems to have problems to recognize it as a valid pcx graphic file and prompts the user to create a new image. So the last chance to see if there is any information in it is

to use a hex editor but this shows that the file just consists of just one byte in hex “0a”. Therefore the guess of the *file* command must have been false.

The first appearance in the timeline of the “.sniffer” file could be seen in table 3-13. It corresponds to the other investigated files.

```
[root@localhost reports]# file images-c0d0p3.dd-.root.quarantaene.usr.include.linux.modules..lib.so..sniffer
images-c0d0p3.dd-.root.quarantaene.usr.include.linux.modules..lib.so..sniffer: PCX ver. 2.5 image data
[root@localhost reports]#
```

Table 4-13 *file* command on “.sniffer”

So let’s check the next file, “.syslogs”. In table 4-14 you can see a typical part of the content of the “.syslog” file. As you can see there are a few passwords, also a root password for a backup server and the root password for the local machine.

```
/usr/bin/mysqldump -C --host=buderus-web2-bak --user=root --password=ar4isc -e --add-drop-table --add-locks --databases sixcms4_buderus :
scp -f rrd/buderus-web1.cpu.day.png rrd/buderus-web1.cpu.month.png rrd/buderus-web1.cpu.week.png rrd/buderus-web1.cpu.year.png rrd/buderus-web1.cputimes.day.png
rrd/buderus-web1.cputimes.month.png rrd/buderus-web1.cputimes.week.png rrd/buderus-web1.cputimes.year.png rrd/buderus-web1.eth0_bytes.day.png rrd/buderus-
web1.eth0_bytes.month.png rrd/buderus-web1.eth0_bytes.week.png rrd/buderus-web1.eth0_bytes.year.png rrd/buderus-web1.eth0_packets.day.png rrd/buderus-
web1.eth0_packets.month.png rrd/buderus-web1.eth0_packets.week.png rrd/buderus-web1.eth0_packets.year.png rrd/buderus-web1.filesystems.day.png rrd/buderus-
web1.filesystems.month.png rrd/buderus-web1.filesystems.week.png rrd/buderus-web1.filesystems.year.png rrd/buderus-web1.memory.day.png rrd/buderus-
web1.memory.month.png rrd/buderus-web1.memory.week.png rrd/buderus-web1.memory.year.png rrd/buderus-web1.new_procs.day.png rrd/buderus-
web1.new_procs.month.png rrd/buderus-web1.new_procs.week.png rrd/buderus-web1.new_procs.year.png :
/usr/bin/mysqldump -C --host=buderus-web2-bak --user=root --password=ar4isc -e --add-drop-table --add-locks --databases buderus_cms :
/usr/bin/mysqldump -C --host=buderus-web2-bak --user=root --password=ar4isc -e --add-drop-table --add-locks --databases sixcms51_buderus :
/usr/bin/mysqldump -C --host=buderus-web2-bak --user=root --password=ar4isc -e --add-drop-table --add-locks --databases sixcms5_buderus :
scp -f rrd/buderus-web1.cpu.day.png rrd/buderus-web1.cpu.month.png rrd/buderus-web1.cpu.week.png rrd/buderus-web1.cpu.year.png rrd/buderus-web1.cputimes.day.png
rrd/buderus-web1.cputimes.month.png rrd/buderus-web1.cputimes.week.png rrd/buderus-web1.cputimes.year.png rrd/buderus-web1.eth0_bytes.day.png rrd/buderus-
web1.eth0_bytes.month.png rrd/buderus-web1.eth0_bytes.week.png rrd/buderus-web1.eth0_bytes.year.png rrd/buderus-web1.eth0_packets.day.png rrd/buderus-
web1.eth0_packets.month.png rrd/buderus-web1.eth0_packets.week.png rrd/buderus-web1.eth0_packets.year.png rrd/buderus-web1.filesystems.day.png rrd/buderus-
web1.filesystems.month.png rrd/buderus-web1.filesystems.week.png rrd/buderus-web1.filesystems.year.png rrd/buderus-web1.memory.day.png rrd/buderus-
web1.memory.month.png rrd/buderus-web1.memory.week.png rrd/buderus-web1.memory.year.png rrd/buderus-web1.new_procs.day.png rrd/buderus-
web1.new_procs.month.png rrd/buderus-web1.new_procs.week.png rrd/buderus-web1.new_procs.year.png :
/usr/bin/mysql -C --host=localhost --user=root --password=ar4isc :
/usr/bin/mysql -C --host=localhost --user=root --password=ar4isc :
/usr/bin/mysql -C --host=localhost --user=root --password=ar4isc :
rsh -l root buderus-web2-bak /usr/bin/rsync --server --sender -vulHogDtpr --delete . /home/ftp-lsybs :
rsh -l root buderus-web2-bak /usr/bin/rsync --server --sender -vulHogDtpr --delete . /var/log/httpd/internet :
rsh -l root buderus-web2-bak /usr/bin/rsync --server --sender -vulHogDtpr --delete . /home/internet :
```

Table 4-14 example of the content of the .syslog file

Now it is time for the last file “syslogs”. Interestingly it has the same size as the “init-infected” file, exactly 28344 bytes. To check if these files are the same I will do an md5sum comparison. As you can see in table 4-15 the generated md5sums of both files are identically so the files must be identically too.

```
[root@localhost reports]# md5sum images-c0d0p3.dd-root.quarantaene.sbin.init-infected
12d3ecbe25ff0939cc1599288972617a  images-c0d0p3.dd-root.quarantaene.sbin.init-infected
[root@localhost reports]# md5sum images-c0d0p3.dd-.root.quarantaene.usr.include.linux.modules..lib.so.syslogs
12d3ecbe25ff0939cc1599288972617a  images-c0d0p3.dd-.root.quarantaene.usr.include.linux.modules..lib.so.syslogs
[root@localhost reports]#
```

Table 4-15 md5 comparison of the files “init-infected” and “syslogs”

Now sum up what I already found out. The system might have been compromised with a Suckit rootkit version 1.3 b on Tuesday, January the 6th at 16:57:57. The rootkit was started and activated and interesting information like passwords for root users of other systems and databases have been collected in the following 3 months until March the 8th. If these information’s have been transmitted to the hacker is unclear by now. Sorrowfully nothing else in a time frame of about 5 days earlier could be found what might be of interest or looks even suspicious.

4.6.5 Analysis by deleted files

Another sometimes very good source for forensic information's is the analysis of deleted files on a compromised system. For the fact that I am dealing with a very large and quite long running file system the awaited number of found files would be enormous. So the usage of Autopsy for this would be very unhandy because it is only capable of producing the wanted output in html or text format and the deleted files can only be recovered and saved for an analysis by hand and only file for file. There is no possibility of a batch processing. But luckily there is a script available in the GCFA training book 8.2 by GIAC.org which fulfills that need. This script was run on each of the six partition images. For the recovered files for each partition an extra directory was created. The name of directory and its binding to the partition is shown in table 4-16.

Directory name	Contains recovered files of partition:	Mount point
/GCFA/buderus-web1/web1/deleted1	c0d0p3.dd	/
/GCFA/buderus-web1/web1/deleted2	c0d0p1.dd	/boot
/GCFA/buderus-web1/web1/deleted3	c0d2p1.dd	/home
/GCFA/buderus-web1/web1/deleted4	c0d0p6.dd	/tmp
/GCFA/buderus-web1/web1/deleted5	c0d0p7.dd	/var
/GCFA/buderus-web1/web1/deleted6	c0d1p1.dd	/mnt/ftp1

Table 4-16 overview of the directories of the undeleted files

```
[root@localhost web1]# cd /GCFA/buderus-web1/web1
[root@localhost web1]# mkdir deleted1
[root@localhost web1]# mkdir deleted2
[root@localhost web1]# mkdir deleted3
[root@localhost web1]# mkdir deleted4
[root@localhost web1]# mkdir deleted5
[root@localhost web1]# mkdir deleted6
[root@localhost bin]# ./ils -rf linux-ext2 /mnt/imagehd1/c0d0p3.dd | awk -F '|' '($2=="f") {print $1}' | while read i; do /usr/local/sleuthkit/
bin/icat -f linux-ext2 /mnt/imagehd1/c0d0p3.dd $i > /GCFA/buderus-web1/web1/deleted1/$i; done
[root@localhost bin]# ./ils -rf linux-ext2 /mnt/imagehd1/c0d0p1.dd | awk -F '|' '($2=="f") {print $1}' | while read i; do /usr/local/sleuthkit/
bin/icat -f linux-ext2 /mnt/imagehd1/c0d0p1.dd $i > /GCFA/buderus-web1/web1/deleted2/$i; done
[root@localhost bin]# ./ils -rf linux-ext2 /mnt/imagehd2/c0d2p1.dd | awk -F '|' '($2=="f") {print $1}' | while read i; do /usr/local/sleuthkit/
bin/icat -f linux-ext2 /mnt/imagehd2/c0d2p1.dd $i > /GCFA/buderus-web1/web1/deleted3/$i; done
[root@localhost bin]# ./ils -rf linux-ext2 /mnt/imagehd1/c0d0p6.dd | awk -F '|' '($2=="f") {print $1}' | while read i; do /usr/local/sleuthkit/
bin/icat -f linux-ext2 /mnt/imagehd1/c0d0p6.dd $i > /GCFA/buderus-web1/web1/deleted4/$i; done
/usr/local/sleuthkit/bin/icat: Invalid address in indirect list (too large): 1768256101
/usr/local/sleuthkit/bin/icat: Invalid address in indirect list (too large): 1766074156
/usr/local/sleuthkit/bin/icat: Invalid address in indirect list (too large): 1869770797
/usr/local/sleuthkit/bin/icat: Invalid address in indirect list (too large): 2088514638
/usr/local/sleuthkit/bin/icat: Invalid address in indirect list (too large): 740763436
[root@localhost bin]# ./ils -rf linux-ext2 /mnt/imagehd1/c0d0p7.dd | awk -F '|' '($2=="f") {print $1}' | while read i; do /usr/local/sleuthkit/
bin/icat -f linux-ext2 /mnt/imagehd1/c0d0p7.dd $i > /GCFA/buderus-web1/web1/deleted5/$i; done
[root@localhost bin]# ./ils -rf linux-ext2 /mnt/imagehd1/c0d1p1.dd | awk -F '|' '($2=="f") {print $1}' | while read i; do /usr/local/sleuthkit/
bin/icat -f linux-ext2 /mnt/imagehd1/c0d1p1.dd $i > /GCFA/buderus-web1/web1/deleted6/$i; done
```

Table 4-17 the command sequence for recovering deleted files for all partitions

So let's inspect the results. For partition "c0d0p3.dd" 169 file entries were found but only two could be recovered. The first one has a byte size of 24 bytes and is named "100886" because this is the inode number the file begins with. A *strings* command against this file resulted in no output. The second file is named "100890" and has a byte size of 23. Here also a *strings* command resulted in no output. For partition "c0d0p1.dd" one file "4083" was found and recovered. The size is 704 bytes and the strings result can be found in table 4-18. From the output it looks like a deleted configuration file of the "grub" boot loader.

```
[root@localhost deleted2]# strings 4083
# grub.conf generated by anaconda
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/cciss/c0d0p3
#          initrd /initrd-version.img
#boot=/dev/cciss/c0d0
default=0
timeout=10
```

```
splashimage=(hd0,0)/grub/splash.xpm.gz
title Red Hat Linux (2.4.7-10smp)
        root (hd0,0)
        kernel /vmlinuz-2.4.7-10smp ro root=/dev/cciss/c0d0p3
        initrd /initrd-2.4.7-10smp.img
title Red Hat Linux-up (2.4.7-10)
        root (hd0,0)
        kernel /vmlinuz-2.4.7-10 ro root=/dev/cciss/c0d0p3
        initrd /initrd-2.4.7-10.img
[root@localhost deleted2]#
```

Table 4-18 strings against file 4083

For the partition “c0d2p1.dd” 16581 deleted files were found. From these 16581 files only 12 could be recovered. The file size of these files ranges from 40 to 22 bytes. None of these files did deliver any output when using the strings command against it. On the partition “c0d0p6.dd” 134 deleted files were found and 9 files could be recovered. These 9 files are shown in table 4-19. The files 12051 to 12055 are normal HTML files without anything special in it. The other files again did not deliver any output using the *strings* command.

Filename	Size in byte
12051	12288
12052	12288
12053	12288
12054	12288
12055	12288
16073	55
20083	9
30123	9
30122	3

Table 4-19 overview of recovered files on partition c0d0p6.dd

For partition “c0d0p7.dd” 223769 files were found and 73 files could be recovered. The list of files and their sizes are shown in table 4-20. Because of their seize ranging between 40 and 8 bytes the chances of finding valuable information are very low, nevertheless I gave strings and the 73 files a try. And again after processing all 73 files I ended up with no results or better no valuable output. What might the reason for this be? One reason might be the fact that this partition is a very highly utilized one with lot of processes writing and deleting files and by this often eliminating the chance to recover deleted files. The other reason might be the time this partition is in use in combination with its size.
For partition “c0d1p1.dd” the situation gets even worse. There were found 2292 deleted files found but none if them was recoverable.

Filename	Size in byte
1111947	40
2583721	40
1111946	38
261648	36
261637	35
261638	35
261640	35
261641	35
261652	35
261642	34
637744	33
686795	33
261651	32
637748	31
686792	31
1111944	30
1111945	30
16442	30
2583722	30
261647	30
261650	30
263259	30
637743	30
637747	30
686794	30

1111941	29
1798875	29
261636	27
261639	27
261646	27
261649	27
263258	27
1111940	26
1111942	26
16441	26
277988	26
637746	26
686791	26
703142	26
3123329	25
637745	24
686790	24
1667972	22
261643	22
261644	22
261645	22
2812731	22
3237771	22
1667967	17
2812726	17
3286761	17
1111943	12
392452	12
1667970	11
1667973	11
2812729	11
2812732	11
3172384	11
3286822	11
343396	11
1340868	10
1340869	10
1340870	10
1340871	10
1340872	10
1798821	10
1831432	10
294340	9
1160996	8
1160998	8
1160999	8
1161000	8
1161001	8

Table 4-20 overview of the recovered files on partition "c0d0p7.dd"

4.6.6 Analysis of the Snort log file

As the shown in chapter 4.6.7 Snort was installed on the afternoon of March the 8th 2004. As this was not the best option the administrators did choose maybe something interesting might had been logged e.g. the hacker trying to get access to his system again but if he is a cautious one I do not think so. The best thing would have been if the system administrators had Snort installed when the rootkit and its backdoor still was active and not afterwards and even better on another machine. So there would have been a chance that the forensic evidences which they destroyed by installing Snort would be a bit compensated by the potential log of the intruder logging onto the machine.
But this is all theory. Let's get back to the facts. The alert logging file of Snort is found on the partition c0d0p7 in the "/var/log/snort" directory. The file is named "alert" and has a size of 375043 byte. It was recovered using the export option of Autopsy. The logging of Snort did start on Monday the 8th 2004 at 17:36:28 as shown in table 4-21.

```
03/08-17:36:28.984247  [**] [119:4:1] (http_inspect) BARE BYTE UNICODE ENCODING [**] {TCP} 172.21.0.173:1256 -> 193.24.34.212:80
03/08-17:36:29.094250  [**] [119:12:1] (http_inspect) APACHE WHITESPACE (TAB) [**] {TCP} 172.21.0.173:1256 -> 193.24.34.212:80
03/08-17:47:43.935111  [**] [119:13:1] (http_inspect) NON-RFC HTTP DELIMITER [**] {TCP} 210.50.211.234:3094 -> 193.24.34.212:80
```

Table 4-21 start of the alert logging of Snort

So what else did Snort log. For the first view hours nothing of a major interest happened despite a few IIS Web server attacks and a view but on this system normal ftp

connections. Then the next day, the 9th of March at 14:01:35 an OpenSSL worm attack began. The output of Snort is shown in table 4-22.

```
Beginning of the attack:
-------------------------------
03/09-14:01:35.161355  [**] [1:1887:2] MISC OpenSSL Worm traffic [**] [Classification: Web Application Attack] [Priority: 1] {TCP} 210.50.211.234:2553 -> 193.24.34.212:443
03/09-14:01:35.191355  [**] [1:1887:2] MISC OpenSSL Worm traffic [**] [Classification: Web Application Attack] [Priority: 1] {TCP} 210.50.211.234:2554 -> 193.24.34.212:443
03/09-14:01:35.211356  [**] [1:1887:2] MISC OpenSSL Worm traffic [**] [Classification: Web Application Attack] [Priority: 1] {TCP} 210.50.211.234:2555 -> 193.24.34.212:443
03/09-14:01:35.231357  [**] [1:1887:2] MISC OpenSSL Worm traffic [**] [Classification: Web Application Attack] [Priority: 1] {TCP} 210.50.211.234:2556 -> 193.24.34.212:443
03/09-14:01:35.281358  [**] [1:1887:2] MISC OpenSSL Worm traffic [**] [Classification: Web Application Attack] [Priority: 1] {TCP} 210.50.211.234:2557 -> 193.24.34.212:443
End of the attack:
-------------------------
03/09-14:04:51.587400  [**] [1:1887:2] MISC OpenSSL Worm traffic [**] [Classification: Web Application Attack] [Priority: 1] {TCP} 210.50.211.234:2347 -> 193.24.34.212:443
03/09-14:04:51.607401  [**] [1:1887:2] MISC OpenSSL Worm traffic [**] [Classification: Web Application Attack] [Priority: 1] {TCP} 210.50.211.234:2348 -> 193.24.34.212:443
03/09-14:06:44.550877  [**] [1:1229:6] FTP CWD ... [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 172.25.167.102:1034 -> 193.24.34.212:21
03/09-14:09:48.606546  [**] [119:15:1] (http_inspect) OVERSIZE REQUEST-URI DIRECTORY [**] {TCP} 172.19.12.58:1161 -> 193.24.34.212:80
03/09-14:13:59.284276  [**] [119:15:1] (http_inspect) OVERSIZE REQUEST-URI DIRECTORY [**] {TCP} 172.21.1.237:1987 -> 193.24.34.212:80
03/09-14:16:02.878090  [**] [1:1229:6] FTP CWD ... [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 172.25.166.51:1028 -> 193.24.34.212:21
03/09-14:16:37.069145  [**] [1:1229:6] FTP CWD ... [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 172.25.175.53:1133 -> 193.24.34.212:21
03/09-14:19:42.914893  [**] [1:1229:6] FTP CWD ... [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 172.25.153.70:1033 -> 193.24.34.212:21
```

Table 4-22 beginning of Snort alert log, interesting sequence

A short lookup in the Snort signature database what the SID 1887 does mean brought the following information[30]:" This event is generated when a web server infected by the slapper worm attempts to infect a web server running OpenSSL". This attack did last until 14:04:51 or about for 3 minutes. After that moment nothing suspicious concerning this attack did appear in the log files nor did this IP address appear in any other regularly log file. If this was just random or the hacker trying to get back using the old vulnerability he did come in first could not be solved. To find out to whom this IP address is related to I used an online "whois" questionnaire[31]. The output is shown in table 4-23.

```
% [whois.apnic.net node-2]
% Whois data copyright terms    http://www.apnic.net/db/dbcopyright.html

inetnum:      210.50.211.232 - 210.50.211.235
netname:      HKA0142
descr:        Hotkey
descr:        Melbourne
country:      AU
admin-c:      JD29-AP
tech-c:       HA13-AP
mnt-by:       MAINT-PRIMUS-AU
changed:      netops@iprimus.com.au 20020211
status:             ASSIGNED NON-PORTABLE
source:       APNIC
changed:      hm-changed@apnic.net  20020827

person:       Jim Dandy
nic-hdl:      JD29-AP
e-mail:       netops@iprimus.com.au
address:      L3 1 Alfred Circular Quay
address:      Sydney NSW Australia
address:      2000
phone:        +61-2-94232400
fax-no:       +61-2-94232410
country:      AU
changed:      netops@primus.com.au 20030724
mnt-by:       MAINT-PRIMUS-AU
source:       APNIC

person:       Hal Abolooza
nic-hdl:      HA13-AP
e-mail:       netops@primus.com.au
address:      L3 1 Alfred Street
address:      Circular Quay
```

[30] http://www.snort.org/snort-db/sid.html?sid=1887
[31] http://www.rogon.de/tools/whois/

```
address:     Sydney NSW 2000
phone:       +61-2-9423-234
fax-no:      +61-2-94232410
country:     AU
changed:     netops@primus.com.au 20030724
mnt-by:      MAINT-PRIMUS-AU
source:      APNIC
```

Table 4-23 the result of the whois of 210.50.211.234

The questioned IP address is registered to a person called “Jim Dandy” and located in Melbourne or Sydney, Australia. If you follow the mentioned domain “primus.com.au” you will get to the web site of an Australian Internet Provider[32].

The logging itself ended as shown in table 4-24 on the 10th March 2004 at 12:05:28. This should be the time the image of the partition was taken (because I did an online backup).

```
03/10-12:04:16.072364  [**] [1:1229:6] FTP CWD ... [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 172.21.200.18:1416 -> 193.24.34.212:21
03/10-12:05:28.994525  [**] [1:1229:6] FTP CWD ... [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 172.21.0.58:1127 -> 193.24.34.212:21
```

Table 4-24 the end of the Snort logging

4.6.7 Analysis by timeline and log files

What is clear is the fact that the system must have been compromised some when earlier than March the 5th 2004 and this will be the next direction for the investigation. Checking the timeline for interesting information’s, checking log files for errors, warnings etc. to maybe find the way the hacker got into the system.

First let’s check in the timeline again when the system administrator began with their work. The corresponding extract of the timeline is shown in table 4-25. As you can see he somehow transferred the chkrootkit.tar.gz file to a directory “/root”, extracted it and then started the installation. This whole process started on Friday, March the 5th at 13:55:24.

```
Fri Mar 05 2004 13:24:43     6225 m.. -/-rw------- apache   apache   366962
/home/intranet/httpd/html/sixcms_upload/cache/9f/93f411764295ca31c15c6d1d8d541d9f
Fri Mar 05 2004 13:25:18     3027 m.. -/-rw------- apache   apache   360601
/home/intranet/httpd/html/sixcms_upload/cache/ed/9e95c14061348030ba9c1d83b1bbe0ed
Fri Mar 05 2004 13:25:33     3145 m.. -/-rw------- apache   apache   367430
/home/intranet/httpd/html/sixcms_upload/cache/0b/d38ce2530d1bac6e654de179c268bf0b
Fri Mar 05 2004 13:25:38     3143 m.. -/-rw------- apache   apache   367552
/home/intranet/httpd/html/sixcms_upload/cache/fa/f1986bb9c7b187fbdac4ac0154903bfa
Fri Mar 05 2004 13:35:12     2324 m.. -/-rw------- apache   apache   367341
/home/intranet/httpd/html/sixcms_upload/cache/7f/a214c17c82b598f73daab84a93e2f37f
Fri Mar 05 2004 13:55:14    33355 m.. -/-rw------- root     root     101491   /root/chkrootkit.tar.gz
                            33355 m.. -/-rw------- root     root     101491   /etc/xinetd.d/vopied~ (deleted-realloc)
Fri Mar 05 2004 13:55:24    67736 ..c -/-rwxr-xr-x 1000     1000     213742   /root/chkrootkit-0.43/chkrootkit
                             6676 ..c -/-r--r--r-- 1000     wheel    213730   /root/chkrootkit-0.43/chkproc.c
                             1292 .ac -/-r--r--r-- 1000     1000     213733   /root/chkrootkit-0.43/README.chkwtmp
                             1448 ..c -/-r--r--r-- 1000     1000     213735   /root/chkrootkit-0.43/Makefile
                             6676 ..c -/-r--r--r-- 1000     wheel    213730   /opt/xlhtml-0.5/conftest.tail (deleted-realloc)
                             3966 .ac -/-r--r--r-- 1000     1000     213726   /root/chkrootkit-0.43/ACKNOWLEDGMENTS
                            12387 .ac -/-r--r--r-- 1000     1000     213731   /root/chkrootkit-0.43/README
                             1323 .ac -/-r--r--r-- 1000     1000     213743   /root/chkrootkit-0.43/README.chklastlog
                             6781 ..c -/-r--r--r-- 1000     wheel    213739   /root/chkrootkit-0.43/chkdirs.c
                             8771 ..c -/-r--r--r-- 1000     1000     213738   /root/chkrootkit-0.43/ifpromisc.c
                             1945 ..c -/-r--r--r-- 1000     1000     213741   /root/chkrootkit-0.43/chkwtmp.c
                            33355 .a. -/-rw------- root     root     101491   /root/chkrootkit.tar.gz
                             2437 ..c -/-r--r--r-- 1000     1000     213737   /root/chkrootkit-0.43/strings.c
                             3966 .ac -/-r--r--r-- 1000     1000     213726   /opt/xlhtml-0.5/conftest.s1 (deleted-realloc)
                              565 .ac -/-r--r--r-- 1000     1000     213740   /root/chkrootkit-0.43/chkrootkit.lsm
                             7195 ..c -/-r--r--r-- 1000     wheel    213736   /root/chkrootkit-0.43/check_wtmpx.c
```

[32] http://www.primus.com.au/

```
                          7729 ..c -/-r--r--r-- 1000     wheel    213732   /root/chkrootkit-0.43/chklastlog.c
                          6676 ..c -/-r--r--r-- 1000     wheel    213730   /opt/xlhtml-0.5/conftest.s2 (deleted-realloc)
                          3966 .ac -/-r--r--r-- 1000     1000     213726   /etc/sysconfig/networking/devices/ifcfg-eth1~
(deleted-realloc)
                         33355 .a. -/-rw------- root     root     101491   /etc/xinetd.d/vopied~ (deleted-realloc)
                          1343 .ac -/-r--r--r-- 1000     1000     213734   /root/chkrootkit-0.43/COPYRIGHT
```

Table 4-25 the system administrator starting is investigations by downloading chkrootkit

If you go down the timeline you will see the administrator installing the skdetect tool and directly after running it rebooting the machine. This is shown in table 4-26.

```
Fri Mar 05 2004 14:07:15    11015 m.. -/-rw------- root     root     101529   /root/skdetect-0.4b.tar.gz
Fri Mar 05 2004 14:07:22      178 ..c -/-rw-r--r-- 1000     1000     229769   /root/skdetect-0.4b/install.h
                              631 ..c -/-rw-r--r-- 1000     1000     229760   /root/skdetect-0.4b/kmem.c
                              379 ..c -/-rw-r--r-- 1000     1000     229766   /root/skdetect-0.4b/README
                              413 ..c -/-rw-r--r-- 1000     1000     229757   /etc/sysconfig/static-routes~ (deleted-realloc)
                            11015 .a. -/-rw------- root     root     101529   /root/skdetect-0.4b.tar.gz
                             2084 ..c -/-rw-r--r-- 1000     1000     229759   /root/skdetect-0.4b/install.c
                             1387 ..c -/-rw-r--r-- 1000     1000     229761   /root/skdetect-0.4b/main.c
                              413 ..c -/-rw-r--r-- 1000     1000     229757   /root/skdetect-0.4b/Makefile
                              233 ..c -/-rw-r--r-- 1000     1000     229767   /root/skdetect-0.4b/pattern.h
                              908 ..c -/-rw-r--r-- 1000     1000     229765   /root/skdetect-0.4b/Changelog
                              773 ..c -/-rw-r--r-- 1000     1000     229763   /root/skdetect-0.4b/syscall.h
                             2207 ..c -/-rw-r--r-- 1000     1000     229762   /root/skdetect-0.4b/pattern.c
                              217 ..c -/-rw-r--r-- 1000     1000     229758   /root/skdetect-0.4b/idt.h
                              203 ..c -/-rw-r--r-- 1000     1000     229768   /root/skdetect-0.4b/kmem.h
                            17992 ..c -/-rw-r--r-- 1000     1000     229764   /root/skdetect-0.4b/LICENSE
Fri Mar 05 2004 14:07:26     1784 m.. -/-rw-rw-r-- root     root     229771   /root/skdetect-0.4b/pattern.o
                           389072 m.. -/-rwxrwxr-x root     root     229776   /root/skdetect-0.4b/skdetect.static
                             1176 m.. -/-rw-rw-r-- root     root     229770   /root/skdetect-0.4b/kmem.o
                             6660 m.. -/-rwxrwxr-x root     root     229775   /root/skdetect-0.4b/skdetect
                             2264 m.. -/-rw-rw-r-- root     root     229773   /root/skdetect-0.4b/main.o
                             2464 m.. -/-rw-rw-r-- root     root     229772   /root/skdetect-0.4b/install.o
Fri Mar 05 2004 14:11:39     4200 m.. -/-rw------- apache   apache   364391
/home/intranet/httpd/html/sixcms_upload/cache/07/0b9788fe9a068ea1f32e966fe5eaf207
Fri Mar 05 2004 14:22:28    18642 .a. -/-rw-r--r-- root     root     1766033  /var/log/lastlog.gz
Fri Mar 05 2004 14:38:15   356050 .a. -/-rwxr-xr-x apache   apache   4751834
/home/intranet/httpd/html/sixcms_upload/media/20/ups_sendungsverfolgung00.pdf
Fri Mar 05 2004 14:40:12       20 .a. l/lrwxrwxrwx root     root     27       /boot/module-info -> module-info-2.4.7-10
                               16 .a. l/lrwxrwxrwx root     root     25       /boot/vmlinuz -> vmlinuz-2.4.7-10
                               11 .a. l/lrwxrwxrwx root     root     4084     /boot/grub/menu.lst -> ./grub.conf
                                3 .a. l/lrwxrwxrwx root     root     65053    /dev/fb -> fb0
                               13 .a. l/lrwxrwxrwx root     root     65016    /dev/core -> ../proc/kcore
                               15 .a. l/lrwxrwxrwx root     root     65086    /dev/fd -> ../proc/self/fd
                               29 .a. l/lrwxrwxrwx root     root     26       /boot/System.map -> System.map-2.4.7-10enterprise
Fri Mar 05 2004 14:40:18        3 .a. l/lrwxrwxrwx root     root     145150   /dev/inet/rip -> udp
                                4 .a. l/lrwxrwxrwx root     root     65363    /dev/ftape -> qft0
                                6 .a. l/lrwxrwxrwx root     root     66508    /dev/sbpcd -> sbpcd0
                                9 .a. l/lrwxrwxrwx root     root     65821    /dev/js1 -> input/js1
                                9 .a. l/lrwxrwxrwx root     root     65822    /dev/js2 -> input/js2
                                9 .a. l/lrwxrwxrwx root     root     65823    /dev/js3 -> input/js3
                                9 .a. l/lrwxrwxrwx root     root     65725    /dev/ipip -> inet/ipip
                                4 .a. l/lrwxrwxrwx root     root     66464    /dev/ram -> ram1
                                7 .a. l/lrwxrwxrwx root     root     65715    /dev/ip -> inet/ip
                                4 .a. -/lrwxrwxrwx root     root     66464    /usr/src/linux-2.4.7-10/drivers/crypto/bcm/  -> ram1
(deleted-realloc)
                               10 .a. l/lrwxrwxrwx root     root     66488    /dev/rawip -> inet/rawip
                                6 .a. l/lrwxrwxrwx root     root     66459    /dev/radio -> radio0
                                9 .a. l/lrwxrwxrwx root     root     65800    /dev/isdnctrl -> isdnctrl0
                                4 .a. l/lrwxrwxrwx root     root     66485    /dev/ramdisk -> ram0
                                3 .a. l/lrwxrwxrwx root     root     145141   /dev/inet/arp -> udp
                                9 .a. l/lrwxrwxrwx root     root     65820    /dev/js0 -> input/js0
Fri Mar 05 2004 14:40:30        3 .a. l/lrwxrwxrwx root     root     68606    /dev/sgh -> sg7
                                3 .a. l/lrwxrwxrwx root     root     68600    /dev/sgb -> sg1
```

Table 4-26 administrator running skdetect and rebooting the system

The next interesting action by the administrators was done on Monday, March the 8th at 10:36:19. There he moved the files identified by the skdetect tool as parts of the rootkit to a subdirectory called “/root/quarantaene”.

```
Mon Mar 08 2004 10:36:19    28344 .a. -/-rwxr-xr-x root     root     101472   /sbin/init-infected (deleted-realloc)
                            28344 .a. -/-rwxr-xr-x root     root     101472   /root/quarantaene/sbin/init-infected
Mon Mar 08 2004 10:36:26     4263 m.. -/-rw------- apache   apache   363121
/home/intranet/httpd/html/sixcms_upload/cache/db/dd6decffb418f984a35d54d49d280adb
Mon Mar 08 2004 10:37:19     4096 m.c d/drwxrwxr-x root     root     229777   /root/quarantaene/sbin
                            28344 ..c -/-rwxr-xr-x root     root     101472   /root/quarantaene/sbin/init-infected
                             4096 m.c d/drwxr-xr-x root     root     96317    /sbin
                            28344 ..c -/-rwxr-xr-x root     root     101472   /sbin/init-infected (deleted-realloc)
                             4096 m.c -/drwxrwxr-x root     root     229777   /root/skdetect-0.4b/.main.c.swx (deleted-realloc)
Mon Mar 08 2004 10:37:47     4096 m.c d/drwxrwxr-x root     root     229780   /root/quarantaene/usr/include/linux
                             4096 m.c d/drwxrwxr-x root     root     229778   /root/quarantaene/usr
                             4096 m.c d/drwxrwxr-x root     root     229779   /root/quarantaene/usr/include
                             4096 m.. -/d--------- root     root     229774   /root/skdetect-0.4b/.main.c.swp (deleted-realloc)
                             4096 m.. d/d--------- root     root     229774   /root/quarantaene
                             4096 m.c d/drwxrwxr-x root     root     229781   /root/quarantaene/usr/include/linux/modules
Mon Mar 08 2004 10:38:10    28344 ..c -/-rwxr-xr-x root     root     164765
/root/quarantaene/usr/include/linux/modules/.lib.so/syslogs
                            28344 ..c -/-rwxr-xr-x root     root     164765   /usr/include/linux/modules/.lib.so/syslogs (deleted-
realloc)
Mon Mar 08 2004 10:38:18        1 ..c -/-rw--w--w- root     root     164763
/root/quarantaene/usr/include/linux/modules/.lib.so/.sniffer
                        17051250 ..c -/-rw--w--w- root     root     164764
/root/quarantaene/usr/include/linux/modules/.lib.so/.syslogs
                               1 ..c -/-rw--w--w- root     root     164763   /usr/include/linux/modules/.lib.so/.sniffer (deleted-
realloc)
                             4096 m.c d/drwxr-xr-x root     root     164762   /usr/include/linux/modules/.lib.so
                             4096 m.c -/drwxr-xr-x root     root     164762   /opt/phpMyAdmin-2.5.1/.htaccess~ (deleted-realloc)
                         17051250 ..c -/-rw--w--w- root     root     164764   /usr/include/linux/modules/.lib.so/.syslogs (deleted-
realloc)
                             4096 m.c d/drwxrwxr-x root     root     5331     /root/quarantaene/usr/include/linux/modules/.lib.so
Mon Mar 08 2004 10:38:59     4096 ..c d/d--------- root     root     229774   /root/quarantaene
                             4096 ..c -/d--------- root     root     229774   /root/skdetect-0.4b/.main.c.swp (deleted-realloc)
```

Table 4-27 sys administrator moving files identifified by himself as parts of the suckit rootkit

The next interesting and even worse thing what the administrator did is downloading the Snort rpm package and installing it on Monday, March the 8th at 15:59:32 as shown in table 4-28. So far for the actions done by the system administrators. What I left out where a few reboots initiated by the system administrators.

```
Mon Mar 08 2004 15:59:32  2392698 m.. -/-rw-r--r-- root     root     101534   /root/snort-2.1.1-1.i386.rpm
Mon Mar 08 2004 15:59:41      563 ma. -/-r-------- root     root     211661   /etc/gshadow
                              749 m.. -/-rw-r--r-- root     root     213754   /etc/group
                              553 .ac -/-r-------- root     root     213144   /etc/gshadow-
                            24268 .a. -/-rwxr-xr-x root     root     192727   /usr/sbin/groupadd
                             1969 .a. -/-r-------- root     root     213143   /etc/shadow-
                             1180 .a. -/-rw-r--r-- root     root     208819   /etc/login.defs
                             2642 m.. -/-rw-r--r-- root     root     213756   /etc/passwd
                             2592 .a. -/-rw-r--r-- root     root     209359   /etc/passwd-
                              735 .ac -/-rw-r--r-- root     root     209022   /etc/group-
Mon Mar 08 2004 15:59:42      819 ..c -/-rwxr-xr-x root     root     135314   /usr/share/doc/snort-2.1.1/signatures/624.txt.gz
                              581 ..c -/-rwxr-xr-x root     root     134885   /usr/share/doc/snort-2.1.1/signatures/2262.txt.gz
                              980 ..c -/-rwxr-xr-x root     root     133841   /usr/share/doc/snort-2.1.1/signatures/1239.txt.gz
                              617 ..c -/-rwxr-xr-x root     root     134854   /usr/share/doc/snort-2.1.1/signatures/2234.txt.gz
                             1336 ..c -/-rwxr-xr-x root     root     135191   /usr/share/doc/snort-2.1.1/signatures/492.txt.gz
                              811 ..c -/-rwxr-xr-x root     root     134024   /usr/share/doc/snort-2.1.1/signatures/1427.txt.gz
                             7980 ..c -/-rwxr-xr-x root     root     213774   /usr/share/doc/snort-2.1.1/contrib/snort2html.pl
                              511 ..c -/-rwxr-xr-x root     root     135104   /usr/share/doc/snort-2.1.1/signatures/395.txt.gz
                             1090 ..c -/-rwxr-xr-x root     root     135170   /usr/share/doc/snort-2.1.1/signatures/469.txt.gz
                              806 ..c -/-rwxr-xr-x root     root     135048   /usr/share/doc/snort-2.1.1/signatures/333.txt.gz
                              689 ..c -/-rwxr-xr-x root     root     133861   /usr/share/doc/snort-2.1.1/signatures/1261.txt.gz
                              582 ..c -/-rwxr-xr-x root     root     134891   /usr/share/doc/snort-2.1.1/signatures/2268.txt.gz
                              745 ..c -/-rwxr-xr-x root     root     135367   /usr/share/doc/snort-2.1.1/signatures/680.txt.gz

--- rest of the installation is cutted out because of needed space ---
```

Table 4-28 sys admin installing Snort as rpm package

Now let's check what versions of potential weaknesses we have in the system. There is an Apache 1.3.22, PHP 4.1.2, phpMyAdmin 2.5.1, OpenSSL 0.9.6b, SSH 2.9p2 and MySQL 4.0.18-Standard installed. For each of the above versions exploits, hacks or other compromising information is available on the internet e.g. at sites like www.packetstormsecurity.com or others. Therefore a lot of possibilities are given how the hacker could have gained access to the investigated system.

Due to the fact of the limited possibilities I was given to save evidences because of the management decision that I was only allowed to take hard disk images and keeping the

system still online without being allowed to even touch the console and a system administrator who did what he could do to save his system but by this he destroyed a lot of forensic information's there are only a few chances to make a whole picture out of the case.

Maybe a lot of information would have been found in the timeline if the system administrator would not had installed so hard disk space consuming applications like Snort. About 20 % of the timeline are inodes which are not recoverable and this is a huge amount of timeline considering a timeline file size of 155 MB. This is definitely a part for a lessons learned workshop! An example of these lines of the timeline is shown in table 4-29.

```
0 .a. -rw-rw-rw- root    root    1997262  <c0d0p7.dd-dead-1997262>
0 .a. -rw-rw-rw- root    root    1064956  <c0d0p7.dd-dead-1064956>
0 .a. -rw-rw-rw- root    root    182309   <c0d0p7.dd-dead-182309>
0 .a. -rw-rw-rw- root    root    2046196  <c0d0p7.dd-dead-2046196>
```

Table 4-29 dead inodes in the timeline

So I will continue to show a few interesting log entries found which might show some kind of interesting actions which combine with the known time of the first execution or installation of the rootkit or even afterwards.

First I started to check every ".bash_history" file on the system to see if there is anything of value in it already having in mind the line from the strings output of the infected init file where is says that the bash history is redirected to /dev/null. After checking all files I just can state out that the rootkit made a perfect job. There have been no recoverable or valuable information's left in the history files.
Second interesting bit of pieces is the shown in the timeline extract in table 4-30. Someone with UID and GID 1000 starting December the 24th extracted and compiled chkrootkit version 0.43. Here we see two very interesting things, the UID/GID and the date of the event. The 24th is Xmas time and nearly in all western countries all around the world a holiday and holidays are known as high times for hackers.

```
Wed Dec 24 2003 17:37:39     1343 m.. -/-r--r--r-- 1000     1000     213734   /root/chkrootkit-0.43/COPYRIGHT
Wed Dec 24 2003 17:37:43     1323 m.. -/-r--r--r-- 1000     1000     213743   /root/chkrootkit-0.43/README.chklastlog
Wed Dec 24 2003 17:37:44     1292 m.. -/-r--r--r-- 1000     1000     213733   /root/chkrootkit-0.43/README.chkwtmp
Wed Dec 24 2003 17:37:59     1945 m.. -/-r--r--r-- 1000     1000     213741   /root/chkrootkit-0.43/chkwtmp.c
Wed Dec 24 2003 17:38:02     2437 m.. -/-r--r--r-- 1000     1000     213737   /root/chkrootkit-0.43/strings.c
Wed Dec 24 2003 22:00:05     6591 m.. -/-rw-r--r-- apache   apache   10961061
/home/internet/httpd/html/pdf/produktbegleitende_unterlagen/baureihensixcms.csv.25-12-2003~06-02-06
Wed Dec 24 2003 22:09:08  1720011 m.. -/-rw-r--r-- sap_internet apache   6979698  /home/internet/httpd/auth/pass.dat_20031225
Wed Dec 24 2003 22:50:57    12213 m.. -/-rw-r--r-- root     root     2959763  /var/log/httpd/internet/ApacheLog12_2003/24-
error.log.gz

-------------- cutted parts in the timeline --------------


access.log.gz
Fri Dec 26 2003 10:40:45     4096 m.. d/drwxrwx--- apache   apache   15728669

-------------- cutted parts in the timeline --------------


/home/internet/httpd/html51/sixcms_upload/cache/media/1479
Fri Dec 26 2003 18:02:26     3966 m.. -/-r--r--r-- 1000     1000     213726   /root/chkrootkit-0.43/ACKNOWLEDGMENTS
                             3966 m.. -/-r--r--r-- 1000     1000     213726   /opt/xlhtml-0.5/conftest.s1 (deleted-realloc)
                             3966 m.. -/-r--r--r-- 1000     1000     213726   /etc/sysconfig/networking/devices/ifcfg-eth1~
(deleted-realloc)
Fri Dec 26 2003 18:26:09     7195 m.. -/-r--r--r-- 1000     wheel    213736   /root/chkrootkit-0.43/check_wtmpx.c
Fri Dec 26 2003 18:27:10     6781 m.. -/-r--r--r-- 1000     wheel    213739   /root/chkrootkit-0.43/chkdirs.c
Fri Dec 26 2003 18:30:12     7729 m.. -/-r--r--r-- 1000     wheel    213732   /root/chkrootkit-0.43/chklastlog.c
Fri Dec 26 2003 18:35:26     6676 m.. -/-r--r--r-- 1000     wheel    213730   /opt/xlhtml-0.5/conftest.s2 (deleted-realloc)
                             6676 m.. -/-r--r--r-- 1000     wheel    213730   /opt/xlhtml-0.5/conftest.tail (deleted-realloc)
                             6676 m.. -/-r--r--r-- 1000     wheel    213730   /root/chkrootkit-0.43/chkproc.c
Fri Dec 26 2003 21:34:10     1448 m.. -/-r--r--r-- 1000     1000     213735   /root/chkrootkit-0.43/Makefile
Fri Dec 26 2003 22:55:44    24837 m.. -/-rw-r--r-- root     root     2959767  /var/log/httpd/internet/ApacheLog12_2003/26-
```

```
error.log.gz
Fri Dec 26 2003 22:58:28     6691 m.. -/-rw-r--r-- root     root     2207571  /var/log/httpd/intranet/ApacheLog12_2003/26-
access.log.gz
Fri Dec 26 2003 22:59:39  1655884 m.. -/-rw-r--r-- root     root     2959764  /var/log/httpd/internet/ApacheLog12_2003/26-
access.log.gz
Sat Dec 27 2003 00:09:57     8771 m.. -/-r--r--r-- 1000     1000     213738   /root/chkrootkit-0.43/ifpromisc.c
Sat Dec 27 2003 03:26:16    21069 .a. -/-rw-r--r-- root     root     2959765  /var/log/httpd/internet/ApacheLog12_2003/25-
error.log.gz
                         1387284 .a. -/-rw-r--r-- root     root     2959762  /var/log/httpd/internet/ApacheLog12_2003/25-
access.log.gz
Sat Dec 27 2003 03:36:08     6591 .a. -/-rw-r--r-- apache   apache   10961067
/home/internet/httpd/html/pdf/produktbegleitende_unterlagen/baureihensixcms.csv.26-12-2003~06-01-57
Sat Dec 27 2003 03:36:11        0 mac -rw------- apache   apache   16678965 <c0d2p1.dd-dead-16678965>
Sat Dec 27 2003 03:55:11     2486 .a. -/-rw-r--r-- root     root     2207570  /var/log/httpd/intranet/ApacheLog12_2003/25-
access.log.gz
Sat Dec 27 2003 12:35:10      565 m.. -/-r--r--r-- 1000     1000     213740   /root/chkrootkit-0.43/chkrootkit.lsm
Sat Dec 27 2003 12:40:06    12387 m.. -/-r--r--r-- 1000     1000     213731   /root/chkrootkit-0.43/README
Sat Dec 27 2003 17:21:47      170 m.. -/-rw-r--r-- root     root     2207566  /var/log/httpd/intranet/ApacheLog12_2003/27-error.log
Sat Dec 27 2003 18:46:15     4096 m.. d/drwxrwx--- apache   apache   5111860
/home/internet/httpd/html51/sixcms_upload/cache/media/1174
Sat Dec 27 2003 22:46:19    25937 m.. -/-rw-r--r-- root     root     2959769  /var/log/httpd/internet/ApacheLog12_2003/27-
error.log.gz
Sat Dec 27 2003 22:58:29  1629217 m.. -/-rw-r--r-- root     root     2959766  /var/log/httpd/internet/ApacheLog12_2003/27-
access.log.gz
Sat Dec 27 2003 22:58:37     8076 m.. -/-rw-r--r-- root     root     2207573  /var/log/httpd/intranet/ApacheLog12_2003/27-
access.log.gz
Sun Dec 28 2003 03:26:20    24837 .a. -/-rw-r--r-- root     root     2959767  /var/log/httpd/internet/ApacheLog12_2003/26-
error.log.gz
                         1655884 .a. -/-rw-r--r-- root     root     2959764  /var/log/httpd/internet/ApacheLog12_2003/26-
access.log.gz
Sun Dec 28 2003 03:55:13     6691 .a. -/-rw-r--r-- root     root     2207571  /var/log/httpd/intranet/ApacheLog12_2003/26-
access.log.gz
Sun Dec 28 2003 03:55:14      170 .a. -/-rw-r--r-- root     root     2207566  /var/log/httpd/intranet/ApacheLog12_2003/27-error.log
Sun Dec 28 2003 16:48:16    67736 m.. -/-rwxr-xr-x 1000     1000     213742   /root/chkrootkit-0.43/chkrootkit
```

Table 4-30 hacker checking for other rootkits ?

The UID/GID is deleted by now. Additionally it is interesting that this whole process took about 3 days. During the investigation I did not find any hints why it took so long or what exact user had these id's. Afterwards the seen UID disappeared. The GID "wheel" still exists as you can see in the actual output of the group file in table 4-31 (the group is marked in yellow) found on partition "c0d0p3" in path "/etc/" (see also timeline extract in table 4-32). That the group still exists is quite normal because the wheel group is generated by default with every Linux installation. The wheel group has a special purpose. Only members of this group are allowed to *su* to root. So here someone with the UID 1000 was member of the group wheel and was able to *su* to root! Afterwards this UID disappeared. What is unclear is why the ownership of the files changed during the process as shown in the timeline in table 4-30.

```
root:x:0:root
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
tty:x:5:
disk:x:6:root
lp:x:7:daemon,lp
mem:x:8:
kmem:x:9:
wheel:x:10:root
mail:x:12:mail
news:x:13:news
uucp:x:14:uucp
man:x:15:
games:x:20:
gopher:x:30:
dip:x:40:
ftp:x:50:
lock:x:54:
nobody:x:99:
users:x:100:
slocate:x:21:
floppy:x:19:
utmp:x:22:
mailnull:x:47:
rpm:x:37:
xfs:x:43:
ntp:x:38:
rpc:x:32:
gdm:x:42:
rpcuser:x:29:
nfsnobody:x:65534:
```

```
nscd:x:28:
ident:x:98:
radvd:x:75:
apache:x:48:buderusdev,sixsupprt
mysql:x:27:
ar4isc:x:500:
intranet:x:501:
internet:x:502:
u066715:x:600:
u077567:x:601:
opcgrp:x:77:
u056478:x:602:
buderusdev:x:603:
sixsupprt:x:604:
buderus:x:778:
ftpchroot:x:779:
hasi:x:2000:
vppd:x:2001:
snort:x:2002:
```

Table 4-31 the group file

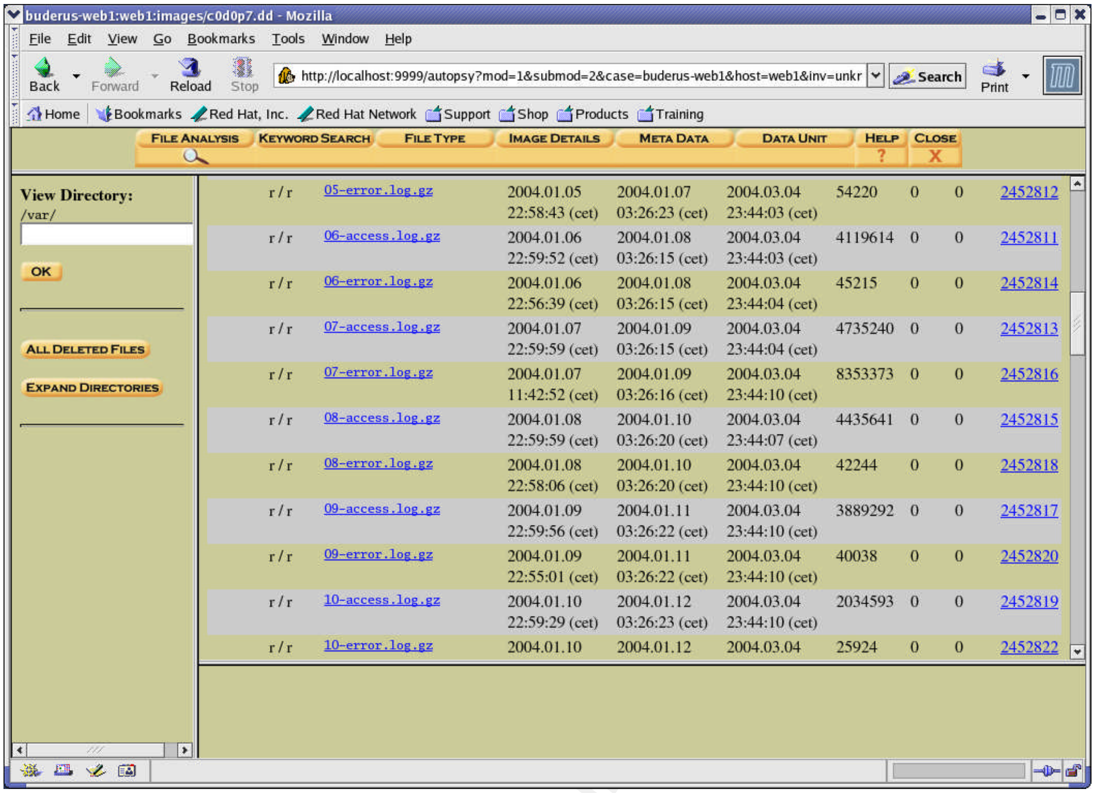
Also it was not possible to track the date of the changes because the last changes to the files "/etc/group" and "/etc/passwd" were done as shown in table 4-32 on the 8th March 2004 at 20:00:08 and in the timeline you can only find the last changes.

```
Mon Mar 08 2004 20:00:08     1997 ..c -/-r-------- root     root     211665   /etc/shadow
                              563 ..c -/-r-------- root     root     211661   /etc/gshadow
                             2642 ..c -/-rw-r--r-- root     root     213756   /etc/passwd
                              749 ..c -/-rw-r--r-- root     root     213754   /etc/group
                             1969 ..c -/-r-------- root     root     213143   /etc/shadow-
                             2592 ..c -/-rw-r--r-- root     root     209359   /etc/passwd-
```

Table 4-32 the sys admins changing the passwd and group file

The question is now who and why did someone install chkrootkit? Maybe it was a hacker who wanted to check if the system he broke into was already rootkitted? Sorrowfully I was not able with the given material to find this out.

The next interesting thing happened on January the 7th 2004, a day after the installed rootkit was activated. The interesting output of Autopsy can be seen in picture 4-3.



Picture 4-3 interesting entry in the log file directory

The information which is very interesting is the size of the different log files. Especially the size comparison between the access log files and the error log files. Normally the access log files are much bigger than the error log files. But on Wednesday the 7th the file "07-error.log.gz" is unusual big in comparison with the other files. So I extracted this file using the corresponding option of Autopsy. Afterwards I unzipped the file using the *gunzip* command and received a log file of 2,1 GB!

So first let's see what the last entries are in the file. Therefore I used the *tail* command. The output is shown in table 4-33.

```
[root@localhost reports]# tail images-c0d0p7.dd-var..log.httpd.internet.ApacheLog01_2004.07-error.log
[Wed Jan  7 12:42:52 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:42:52 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:42:52 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:42:52 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:42:52 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:42:52 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:42:52 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:42:52 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:42:52 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:42:52 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd[root@localhost reports]#
```

Table 4-33 *tail* on the apache error from 7th January

As you can see the log file ended at 12:42:52 on that Wednesday in middle of a line which is very unusual for a log file, despite that the corresponding service crashed or ran into unrecoverable problems. As you can also see the problem is related to a PHP.

Now let's check when this problem started on that day. As doing a *less* and searching for the first PHP warning I come to that point at 12:21:27 shown in table 4-34.

```
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  OpenDir: No such file or directory (errno 2) in /home/internet/httpd/html/buderus-websttest.php on line 30
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
[Wed Jan  7 12:21:27 2004] [error] PHP Warning:  Supplied argument is not a valid Directory resource in /home/internet/httpd/html/buderus-websttest.php on line 31
```

Table 4-34 beginning of the error

The problem starts with one line *"Wed Jan 7 12:21:27 2004] [error] PHP Warning: OpenDir: No such file or directory (errno 2) in /home/internet/httpd/html/buderus-websttest.php on line 30"* and then is repeatedly followed by millions of lines as seen in table 4-34. This logging was continued until 12:42:52 for about 21 minutes and than the logging mechanism or the service or something else crashed. This is quite interesting because in these 21 minutes about 2,09 GB of log file have been written. What happened afterwards is unclear and why. Was this the hacker trying something on this machine? Or was this another hacker trying to get on that machine using a PHP exploit? Did he know that machine was already rootkitted?

I failed again finding an answer to these questions. None of the log files nor a search in the Internet with Google brought up any information which would had helped to continue the investigations in a useful manner.

This leads to a very unsatisfying situation, especially having in mind that this investigation is the investigation documented for the practical for the GCFA assignment.

First of all I was able to show that the system might have been compromised on December the 24th 2003, but not exactly when, how and by whom and this is the sad part. The next thing which is proofed is the existence of a rootkit activated on January the 6th running until March the 5th. Maybe the hacker did try to come back on March the 9th as shown in chapter 4.6.6 using the old used OpenSSL vulnerability. Reasons why I was not able to find more information's or better the total picture including the hacker entering the system and the vulnerability he used are discussed in chapter 4.7. Additionally there you can find a lesson learned paragraph on how to make these things better in the future.

4.7 The conclusion

The conclusion of all this investigation is quite disillusioning. From my personal view and from the view of a forensic investigation it is always much more satisfying if a hunter gets his booty and not returns without anything.

On the other hand this investigation is a very good demonstration for what can happen if no good guidelines in a company exist on how to cope with events like the one of this case. During this investigation I came across a lot of situations and acts of involved persons which destroyed with each step of their action a part of the puzzle which maybe would have a led to a clue what had happened to the system and why.

The most important issues I wrote down for a later "lessons learned" workshop:

1. A company without any policies in place coping with the processes what to do in case of the suspicion of a computer crime has only a small chance for a fast and evident obtaining recovery process and a later necessary prosecution. This is one of the biggest challenges which my company will have to deal with in the future! Building up a computer incident handling team and establishing the fundamental policies and processes.
2. The investigated system is quite a good demonstration for a Linux system which is installed, stable running and never patched! It was installed on September the 6th 2001 and from then on only new software was applied to it if it was necessary for its main purpose, as Intranet and Internet web server. Security updates were not maintained. Why is unclear.
3. Actually it is unclear if the audit of the system administrators just happened "accidentally" or by a routine schedule. Because of the information from the interview it seems that it happened just by accident. Also something which should be changed in the future! Every running system should be audited on a regular time schedule and a fixed set of procedures added by system specific additions. Also mechanisms have to be established which make the integrity check of a system possible e.g. tools like tripwire[33].
4. One of the most important things which should change in the future are the guidelines and policies how to react if someone discovers an attack, an intrusion or just entertains the suspicion that a system might have been compromised. In this case the most important rule must be "keep the hands of the keyboard" and inform the responsible division in your company e.g. an internal CERT[34] team. Any other reaction will destroy important evidences.
   As in this case the system was altered by the two system administrators in any

[33] www.tripwire.com
[34] http://www.us-cert.gov/ for more information to find out what a cert is

thinkable way. They installed Snort, they installed tools like chkrootkit, skdetect, by this deleted and deactivated the rootkit, installed a new version of mysql, rebooted thy system during 4 days a few times and by all these actions destroyed volatile evidences like history files, log files, network connections, memory and many, many other.

5. Another task would be to establish a policy or the creation of the awareness to the management that without giving the investigators the chance to collect the evidences the chances of solving a case will be very low and by this the chance of a prosecution will be reduced or vanished.

So last but not least you can say this time the hacker won. Mostly because of a lack of procedures and rules in place and people who were aware of these guidelines. Hopefully this will change in the future.

5 Part III – Legal issues of Incident Handling

5.1 Legal actions corresponding the assumption that Mr. Price is an employee of my company

The consequences of the usage of the questioned binary and the distribution of copyright protected material are discussed in depth in chapter 3.6.

If this kind of action is discovered at our company usually the following steps of action are taken:

1. The company incident handling team is informed and the supervisor of the employee is informed. Additionally the legal department will be informed too. At this time is the duty of the incident handling team is to seize as many evidences as possible for a later investigation of the case and to give a first judgment if any laws might have been violated.
2. Depending on the situation and the potential crime which has taken place the consequences for the employee range from an immediate suspension to just the collection of the evidences, storing the collected evidences in a safe and redundancy way, cleaning or restoring the questioned PC or hardware and an entry in his employee file.
3. Decision of an internal council depending on the facts if our company is confronted with one of the following scenarios:
   - local or international law was or might have been broken
   - none of the found information's direct to a committed crime, just to a violation of company internal rules and policies
   - were are dealing with a false alarm

   Depending on the evaluation of the internal council what kind of incident the company is dealing with the next steps are taken. In case that there is just any hint that law might have been violated the case is automatically passed on to law enforcement and the next steps depend on them.
   If we are "just" dealing with an internal affair the company itself or the corresponding council can decide what actions have to be taken between the company and the employee. These actions are based on the employment contract between the employee, the violation of internal policies which was committed and the civil law (BGB[35] in Germany).
   In any of the above two scenarios the following additional steps are taken even if the case stays internal to prevent the company from any possible monetary or justice harm. This is done by involving the company lawyers, information of the management about the ongoing investigations, starting the internal investigations

[35] BGB Bürgeliches Gesetzbuch (german book of the civil law)

by the incident handling team and the computer forensics team.
If during the investigation it is discovered that the whole thing was a false alarm all information is documented. Afterwards all seized evidence and the case documentation is stored in a secure cabinet for 5 years for the situation that the case has to be investigated again or new evidences arrive from elsewhere or from law enforcement. The last action which is taken is that the case is closed. Additionally all informed persons were given an all-clear notice.

4. Doing a "lessons learned workshop" with the necessary parties from the company to make sure that in the future most possible actions were taken to prevent this kind of crime to happen again or in case that it was a false-negative to enhance in future the discovery mechanisms.

5.2 Legal consequences if Mr. Price is a company employee and used company facilities to distribute child pornography

In the case that Mr. Price is an employee of the company I work for and it would have been discovered that he is involved in the crime of distributing child pornography the following actions would have been taken:

1. Immediately at the point of the discovery that the investigated case is dealing with child pornography the employee would have been suspended, the management would be informed and law enforcement would be informed and further investigations were passed on to them to do a official investigation. The legal basis will be discussed in detail in chapter 5.2.1.
2. Depending on the company's policies and the employment contract also justice actions were taken between the company and the employee.
3. Further actions were decided by law enforcement.
4. Our legal department will strictly analyze the case in close connection with the law enforcement to prevent the company from any harm.

5.2.1 Legal basis for crimes facing child pornography

The legal basis for prosecution of child pornography is based on the "6. Reform des Strafrechts" (6. StrRG, released 1.4.1998). In § 176a[36], article 2 StGB[37] the protection of children against any kind of sexual abuse is defined. Additionally any kind of gaining profit due to selling pornography material with children involved is prohibited and will be punished with imprisonment not less than two years.

---

[36] http://bundesrecht.juris.de/bundesrecht/stgb/__176a.html
[37] StGB = Strafrechts Gesetzbuch

The distribution of material which is considered as child pornography will be punished with imprisonment from 3 months to 5 years according to § 184[38] StGB for a single person. If the violation is committed in an organized way by a group or organization the minimum imprisonment starts with the 6 months and ends up with 10 years for each person according to § 184 StGB. Distribution also includes any kind of transfer using any kind of computer equipment or the internet (see §11 article 3 "Datenspeicher"). If the distribution was done in any kind of business behavior or to gain profit the punishment will be increased from a minimum of 6 months to 10 years for a single person.

For the distribution of child pornography over the Internet the federal supreme court of Germany had to define the legal basis and definition were the distribution over the Internet begins and by this from which point this crime can be prosecuted. This definition was given by the federal supreme court of Germany in the finding from the 27th June 2001 (1 StR 66/01). In this finding the distribution of child pornography is defined like the following:

The distribution of child pornography over the Internet starts when the questioned file is arrived in the volatile data storage of a computer (e.g. RAM[39]) or the permanent data storage (e.g. hard disk, floppy drive etc.) of a computer. Additionally it is irrelevant if this information did arrive by just activating an Internet link or the data was pushed from a distributor to the questioned device because of a subscription.

According to § 184 StGB just the attempt of getting the property of material which is considered as child pornography will be punished with a fine or imprisonment up to 1 year.

A special situation occurs if the offense is connected to actions abroad. According to the German law actions which violate the above mentioned paragraphs will be prosecuted in Germany for German and foreign citizens and if a German citizen violates these laws abroad he will be also prosecuted in Germany. The basis for this prosecution is defined in §5[40], article 8b StGB and §6[41], article 6 StGB.

[38] http://bundesrecht.juris.de/bundesrecht/stgb/__184.html
[39] RAM = Random Acess Memory; volatile main memory of computer
[40] http://bundesrecht.juris.de/bundesrecht/stgb/__5.html
[41] http://bundesrecht.juris.de/bundesrecht/stgb/__6.html

6 Timeline

Normally here you would find the complete timeline of the system investigated in part 2 of this practical. But as mentioned before in this document the timeline has an enormous size, about 155 MB! Even compressed it has a size of about 16 MB. So as this is far too much for Word to handle I decided to do the following. I deleted every line of the timeline before the 1st December 2003 and included it to the submission mail as a separate and compressed Zip file. If the grader likes to have the rest I can send in the whole timeline file as a few mails in which every mail contains a part of the timeline. Before I can send in the timeline I will need the maximum size for emails on the certify@giac.org account.