Global Information Assurance Certification Paper

Hidden Data Is Evidence Too

GCFA Practical
(GIAC Certified Forensic Analyst)

Bob Pelcher

SANS Conference Denver, CO 2003
Practical Version 1.4

28 Jan 2004

Table of Contents

Abstract:

Part 1: Analyze a binary provided by SANS. Identify everything you can about this binary.

Part 2 Option 2: Test a tool that could be used for computer forensics.

Part 3: Provide legal guidance on four different topics.

Part 1 – Analyze an Unknown Binary

A. An employee, John Price has been suspended from his place of employment when an audit discovered that he was using the organizations computing resources to illegally distribute copyrighted material. Unfortunately Mr. Price was able to wipe the hard disk of his office PC before investigators could be deployed. However, a single 3.5 inch floppy disk (the floppy disk image that you must use for this assignment can be downloaded here) was found in the drive of the PC. Although Mr. Price has subsequently denied that the floppy belonged to him, it was seized and entered into evidence:

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

The floppy disk contains a number of files, including an unknown binary named 'prog'. Your primary task is to analyze this binary to establish its purpose, and how it might have been used by Mr. Price in the course of his alleged illegal activities. You should also examine the disk for any other evidence relating to this case. It is suspected that Mr. Price may have had access to other computers in the workplace.

B. General Information: The following information is provided for the system I will be using to analyze this binary, henceforth called the forensics laptop:

- Sony Vaio Laptop, Model PCG-F490.
  - PIII 645mhz, 256MB RAM
  - 20gig Hard Drive
  - Red Hat 9.0 Kernel 2.4.20-19.9
  - VMWare 4.0.5 build-6030, Windows 2000 (Client)

1. Binary Details

True Program name is bmap1.0.20
File MAC Times:

```
Modified time: Mon Jul 14 08:24:00 2003
Access time: Wed Jul 16 00:12:45 2003
Creation time: Wed Jul 16 00:05:33 2003
```

File Owner User ID: 502 / Group ID 502
File Size: 487476

MD5 Hash: `7b80d9aff486c6aa6aa3efa63cc56880`
Key words from strings:

```
1.0.20 (07/15/03)
newt
bmap_get_slack_block
bmap_get_block_count
bmap_get_block_size
bmap_map_block
bmap_raw_open
bmap_raw_close
use block-list knowledge to perform special operations on files
```

The unknown binary was downloaded from the SANS website located at the following URL: http://www.giac.org/gcfa/binary_v1_4.zip. The binary was placed into the /forensics/SANS directory on the forensics laptop. Since the unknown binary was in a zip format the program *zipinfo* was used to retrieve the pertinent metadata information. Zipinfo is a small utility included in Linux which lists technical information about files in a ZIP archive. The command used was *zipinfo -z binary_v1_4.zip.* The "-z" flag shows comment information that may be contained within the zip archive. In this case the output showed that the zip archive contained the comment "GCFA binary analysis", and the following files:

```
# zipinfo -z binary_v1_4.zip
Archive:  binary_v1_4.zip   459502 bytes   3 files
GCFA binary analysis
-r--------  2.3 unx   474162 bx defN 15-Jul-03 23:03 fl-160703-jp1.dd.gz
-rw-r--r--  2.3 unx       54 tx stor 16-Jul-03 00:14 fl-160703-jp1.dd.gz.md5
-rw-r--r--  2.3 unx       39 tx stor 16-Jul-03 00:14 prog.md5
3 files, 474255 bytes uncompressed, 459030 bytes compressed:  3.2%
```

It appears that two of the three files are md5 hashes and the third file is the gzipped dd image. The unknown binary was then uncompressed using the *unzip* utility. However before the archive was uncompressed *`unzip -l binary_v1_4.zip`* was executed to verify that the results were the same as from *zipinfo*. The *-l* flag in *unzip* is used to list the files in a zip archive. The results showed the following:

```
# unzip -l binary_v1_4.zip
Archive:  binary_v1_4.zip
GCFA binary analysis
  Length     Date   Time    Name
 --------  ----   ----    ----
   474162  07-15-03 23:03   fl-160703-jp1.dd.gz
       54  07-16-03 00:14   fl-160703-jp1.dd.gz.md5
       39  07-16-03 00:14   prog.md5
 --------                   -------
   474255                   3 files
```

This was the exact same result as from *zipinfo*. The archive was then extracted using the following command *`unzip binary_v1_4.zip.`* The following shows the directory listing of the extracted files as well as the results of the provided md5sum hashes. The extracted files included a file "fl-160703-jp1.dd.gz.md5"

which was the *md5* hash of the file "fl-160703-jp1.dd.gz". The next logical step was to compare the hashes to verify that they matched. So the command `md5sum -c fl-160703-jp1.dd.gz.md5` was run which checks the md5 sum of the file "fl-160703-jp1.dd.gz" and verifies if the given hash value matches with the computed value. The result of the command was: `fl-160703-jp1.dd.gz: OK` which means the two md5 hashes were matched. The following screen capture shows the directory listing and the results of the actual md5 hash.

```
# ls -al
total 2384
drwxr-xr-x    2 root     root         4096 Jan 25  2004 .
drwxrwxrwx    6 root     root         4096 Jan 22 10:22 ..
-rwxr--r--    1 root     root       459502 Jan 25  2004 binary_v1_4.zip
-rwxr--r--    1 root     root       474162 Jul 16  2003 fl-160703-jp1.dd.gz
-rwxr--r--    1 root     root           54 Jul 16  2003 fl-160703-jp1.dd.gz.md5
-rwxr--r--    1 root     root           39 Jul 16  2003 prog.md5

# md5sum fl-160703-jp1.dd.gz
4b680767a2aed974cec5fbcbf84cc97a  fl-160703-jp1.dd.gz
# more fl-160703-jp1.dd.gz.md5
4b680767a2aed974cec5fbcbf84cc97a  fl-160703-jp1.dd.gz
```

The two md5sum values matched, this means the two files are the same. The results of an md5 or Message Digest 5 hash are a mathematical algorithm that is basically a fingerprint of a file. The chances of two different files having the same hash are 2 to the 128 power. A forensically sound practice is to make your dd image, then run an *md5sum* against the device just imaged. Later when the two *md5sum* hash values are compared, if they are the same then you have an exact mirror of the device. If they do not match, a new image should be taken, if possible, to ensure that the data has not been altered. Since the two hash values match, the next step would be to analyze the `fl-160703-jp1.dd.gz` image.

The first step in the analysis would be to unzip the file. The file was in a gzipped archive so the easiest way to extract it is with gunzip. So `gunzip fl-160703-jp1.dd.gz` was executed which resulted in a single file called `fl-160703-jp1.dd`. Since the file is a dd image as evidenced by the command:

```
# file fl-160703-jp1.dd
fl-160703-jp1.dd: Linux rev 1.0 ext2 filesystem data
```

The *file* command tests a file in an attempt to classify it. There are three sets of tests, performed in this order: file system tests, magic number tests, and language tests. The first test that succeeds causes the file type to be printed. In this case the dd image was of an ext2 file system which is the default file system for Linux. Since this is a dd image I can now do one of two things; use dd to copy the image onto a floppy disk or mount the dd image on a loopback device. For its shear ease the latter was chosen.

The command for this is followed by the breakdown:

```
mount –o ro,loop,noexec,noatime fl-160703-jp1.dd /mnt/loop
```

mount	– Mount or access a file system.
-o ro	– -o is the options switch, ro is read only.
loop	– To open a dd image
noexec	– Do not execute any program
noatime	– Do not update the access time on a file
fl-160703-jp1.dd	– File to be accessed
/mnt/loop	– Mount point for the accessed file system

Everything in Linux is a file, so once this dd image is mounted it is only a matter of changing to the mount point directory to access the contents of the image.

Next I ran the *ls –Ral* command. This provided me with the following information.

```
# ls -Ral
.:
total 560
drwxr-xr-x    6 root     root         1024 Jul 16  2003 .
drwxr-xr-x    7 root     root         4096 Jan 25 00:19 ..
-rw-r--r--    1 root     root         2592 Jul 14  2003 .~5456g.tmp
drwxr-xr-x    2 502      502          1024 Jul 14  2003 Docs
drwxr-xr-x    2 502      502          1024 Feb  3  2003 John
drwx------    2 root     root        12288 Jul 14  2003 lost+found
drwxr-xr-x    2 502      502          1024 May  3  2003 May03
-rwxr-xr-x    1 502      502         56950 Jul 14  2003 nc-1.10-
16.i386.rpm..rpm
-rwxr-xr-x    1 502      502        487476 Jul 14  2003 prog

./Docs:
total 171
drwxr-xr-x    2 502      502          1024 Jul 14  2003 .
drwxr-xr-x    6 root     root         1024 Jul 16  2003 ..
-rwxr-xr-x    1 502      502         29184 May 21  2003 DVD-Playing-HOWTO-
html.tar
-rwxr-xr-x    1 502      502         27430 May 21  2003 Kernel-HOWTO-
html.tar.gz
-rw-------    1 502      502         29696 Jun 11  2003 Letter.doc
-rw-------    1 502      502         19456 Jul 14  2003 Mikemsg.doc
-rwxr-xr-x    1 502      502         32661 May 21  2003 MP3-HOWTO-html.tar.gz
-rwxr-xr-x    1 502      502         26843 Jul 14  2003 Sound-HOWTO-html.tar.gz

./John:
total 44
drwxr-xr-x    2 502      502          1024 Feb  3  2003 .
drwxr-xr-x    6 root     root         1024 Jul 16  2003 ..
-rwxr-xr-x    1 502      502         19088 Jan 28  2003 sect-num.gif
-rwxr-xr-x    1 502      502         20680 Jan 28  2003 sectors.gif

./lost+found:
total 13
drwx------    2 root     root        12288 Jul 14  2003 .
drwxr-xr-x    6 root     root         1024 Jul 16  2003 ..

./May03:
```

```
total 17
drwxr-xr-x    2 502      502          1024 May  3  2003 .
drwxr-xr-x    6 root     root         1024 Jul 16  2003 ..
-rwxr-xr-x    1 502      502         13487 Jul 14  2003 ebay300.jpg
```

This is an overview of all of the logical files identified within the dd image. Notice one of the files, **.~5456g.tmp**, is hidden. This is evident by the period (.) in the first position of the file name. Also located in the root directory is the program to be analyzed, **prog**. The first step is to ensure no data has changed since this exercise was created, so I ran an *md5sum* against **prog** and compared this output to the *md5sum* hash provided by SANS. Below you will see that they matched.

```
# md5sum prog
7b80d9aff486c6aa6aa3efa63cc56880  prog
# more /forensics/SANS/prog.md5
7b80d9aff486c6aa6aa3efa63cc56880  prog
```

The first thing is to try to discover what I can about this binary. I ran *file* against the binary to see what information it would provide.

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

The *file* program, as explained earlier, determined that the prog binary is an executable program and is statically linked. When a program is described as being statically linked it simply means that it has the required system libraries compiled into it. This, from a forensic stand point, means that the program itself will leave little to no footprint on the file system although it may leave other bits of information behind. Another point of interest is that the binary has been stripped. This means that the object file symbols have been discarded in order to make the binary smaller in size. This also makes understanding the binary that much harder.

The next task is to obtain the MAC times for this binary. File time and date stamps are stored as three separate indexes. These three indices are as follows: the creation time, the last write/modification time; and the last access time.

Creation time is usually the time the file was created. Specifically, it is the time when the file was first created, first written to disk, or copied from another source.

Last write/modification time is the time when a program last made any changes to the file.

Last access time is the last time some action was taken on the file which can include the last time the file was: copied; viewed, opened, or printed.

In order to best determine the MAC times for this binary the inode number where the binary resides within the dd image is required. Inodes are a basic part of the Linux file system. Every file on a Linux system is represented by an inode number which contains the description of the file, file type, access rights, owner, timestamps, size, and pointers to the data blocks which hold the data. There are a finite number of inodes on any given Linux file system.

In order to view the inode number of the prog file the ls command was used with the –i flag. The output shows a number, which is the inode, and the name of the file occupying that inode. In this case the prog file has an inode number of 18.

```
# ls -i prog
     18 prog
```

The *debugfs* command was then used. The *debugfs* command is an interactive file system debugger used to retrieve the contents of an inode structure. The command switches are *-R*, which means run a single command request and *"stat <18>"* which means display the contents of the identified inode, which in this case was inode 18.

```
# debugfs -R "stat <18>" fl-160703-jp1.dd
debugfs 1.19, 13-Jul-2000 for EXT2 FS 0.5b, 95/08/09
Inode: 18   Type: regular    Mode:  0755   Flags: 0x0   Generation: 414131
User:   502   Group:   502   Size: 487476
File ACL: 0    Directory ACL: 0
Links: 1   Blockcount: 960
Fragment:  Address: 0    Number: 0    Size: 0
ctime: 0x3f14eb2d -- Wed Jul 16 00:05:33 2003
atime: 0x3f14ecdd -- Wed Jul 16 00:12:45 2003
mtime: 0x3f12bd00 -- Mon Jul 14 08:24:00 2003
BLOCKS:
278 279 280 281 282 283 284 285 286 287 288 289
Total 480
```

As can be seen from the output the prog file has three unique timestamps. The prog file was created on Wednesday July, 16 2003 at 00:05 or 12:05 am and was last accessed on the same day at 12:12 am. However the prog file was modified on July 14 2003 at 8:24 am. How can this be? It is not possible for a file to be modified before it was created. It is possible for the file to have been created previously (before July 14 8:24 am) modified and then copied to another file system such as a floppy disk. This would explain the discrepancies in the timestamps and is a likely scenario considering the prog file was found on a floppy disk image.

Another point of interest is that the user ID and group ID are the same. Both are set to 502. Default user and group accounts generally start at 500 on most Linux systems so it is safe to assume that the id fields belong to the 3rd user that was added to that Linux system. Since the only findings we have are from a floppy disk there is no way to tell who owns the user or group id 502.

The next step was to run the *strings* command on this binary. This command will pull out all the human readable or printable characters. By default strings prints out the printable character sequences that are at least 4 characters long. This allows an investigator to see inside the binary before it is run. The output of the command *`strings prog`* generated 84 pages of text that can be found as Attachment 1. The following is what is believed to be the important text found.

IDENTIFYING PROGRAM REMARKS:
1.0.20 (07/15/03)
newt
bmap_get_slack_block
bmap_get_block_count
bmap_get_block_size
bmap_map_block
bmap_raw_open
bmap_raw_close
keld@dkuug.dk
Keld Simonsen
ISO/IEC 14652 i18n FDCC-set
C/o Keld Simonsen, Skt. Jorgens Alle 8, DK-1615 Kobenhavn V
ISO/IEC JTC1/SC22/WG20 - internationalization

Investigative leads:

The programmers name “newt” or “Keld Simonsen”.
The version number is possibly 1.0.20
The true name of the program is probably bmap.

PROGRAM ACTIONS:
mft_getopt
mft_log_init
MFT_LOG_THRESH
mft_log_shutdown
test for fragmentation (returns 0 if file is fragmented)
checkfrag
display fragmentation information for the file
frag
wipe the file from the raw device
try '--help' for help.
wipe the file from the raw device
print number of bytes available
extract a copy from the raw device
list sector numbers
use block-list knowledge to perform special operations on files
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

Investigative leads:

This binary seems to access files at a very low level, possibly at the block address. This is suggested by the several references to “raw device” and “mapping block.” Also the “use block-list knowledge to perform special operations on files.” Also this binary accesses the Master File Table (MFT), possibly to identify all blocks associated with a file.

HTML FOUND

<table bgcolor=%s><tr><td>%s: %s</td></tr></table>
<table bgcolor=%s><tr><td>%s</td></tr></table>
<table bgcolor=%s><tr><td></td></tr></table>

This is an HTML code fragment, which may be part of a graphical user interface.

FILES & DIRECTORIES IDENTIFIED

devices in /dev/
/dev/console
/dev/log
/etc/suid-debug
/proc/sys/kernel/osrelease
/usr/lib/gconv
/usr/lib/gconv/gconv-modules.cache
/usr/share/locale
/locale.alias
/etc/localtime (saw 'Universal')
/usr/share/zoneinfo
/proc/self/cwd
/proc
/etc/mtab
/etc/fstab
/cpuinfo
/lib/
/usr/lib/
/etc/ld.so.cache
/proc/self/exe
/usr/lib/locale
.profile

Investigative leads:

Two of the files identified, /etc/mtab and /etc/fstab, are used by the file system to identify different devices involved with accessing partitions & drives.

2. Program Description

At this point it is assumed that the program is in fact bmap version 1.0.20 based upon the *strings* search. The bmap tool is used to store and wipe information from the slack space of a file. In the most simple of terms it is a data hiding tool. To better understand how hiding data into slack space works you first need to understand what slack space is. The Linux file system uses blocks to store

information. All of these blocks are the same size, typically 1,2 or 4 KB in size. If a file is smaller than the block size, the remaining space is not used and is called slack space. Thus on a typical Linux file system with a block size of 4K one could hide data up to that 4KB in each block. One of the interesting side features of slack space is that the data hidden in the slack space will not appear in disk usage, will be invisible from the file system, and be undetectable by file integrity checkers.

In order to truly identify what this program is and what it does, it needs to be run in a safe manner. So, before the binary is executed a monitoring tool named *apptrace* is called by the system to monitor the actions the binary takes when it is executed. *apptrace* will also provide several different data output files so the actions the binary completes can be later viewed. The first step is to move the **prog** binary to a new directory called "test." When the *apptrace* command is run two things will happen. First, a link between **prog** and the *apptrace* program will be created and the original program will be renamed to **"prog.orig"**. Second, *apptrace* creates a directory in the root directory called apptrace which is where the monitoring program will load all of its findings. Review of this directory shows two files of interest - "prog-parameters" and "prog.####.trace." The first collects and records all parameters from each time the binary is run. The second shows all of the functions and calls generated by the binary, and are numbered according to process ID number for each binary execution.

```
# apptrace prog
# ls -al
total 492
drwxr-xr-x    2 root     root         4096 Jan 25 08:01 .
drwxr-xr-x    3 root     root         4096 Jan 25 07:41 ..
lrwxrwxrwx    1 root     root           17 Jan 25 08:01 prog ->
/usr/bin/apptrace
-rwxr-xr-x    1 root     root       487476 Jan 25 07:42 prog.orig

# ./prog
no filename. try '--help' for help.

# ls -al /root/apptrace/
total 16
drwxr-xr-x    2 root     root         4096 Jan 25 08:03 .
drwxr-x---   27 root     root         4096 Jan 25 07:56 ..
-rw-r--r--    1 root     root          742 Jan 25 08:03 prog.3683.trace
-rw-r--r--    1 root     root            0 Jan 25 08:03 prog-last-run
-rw-r--r--    1 root     root           38 Jan 25 08:03 prog-parameters
```

The first time that *apptrace* ran the binary **prog** was without any switches or parameters. Below is the "prog-parameters" and "prog.3683.trace" files created by this first test.

```
# cat prog-parameters
Sun Jan 25 08:03:57 MST 2004 - ./prog

# cat prog.3683.trace
3693  execve("./prog.orig", ["./prog.orig"], [/* 32 vars */]) = 0
3693  fcntl64(0, F_GETFD)               = 0
```

```
3693  fcntl64(1, F_GETFD)               = 0
3693  fcntl64(2, F_GETFD)               = 0
3693  uname({sys="Linux", node="balder", ...}) = 0
3693  geteuid32()                       = 0
3693  getuid32()                        = 0
3693  getegid32()                       = 0
3693  getgid32()                        = 0
3693  brk(0)                            = 0x80bedec
3693  brk(0x80bee0c)                    = 0x80bee0c
3693  brk(0x80bf000)                    = 0x80bf000
3693  brk(0x80c0000)                    = 0x80c0000
3693  write(2, "no filename. try \'--help\' for he"..., 36) = 36
3693  _exit(2)                          = ?
```

Notice the “try \’--help\’ recorded in the “prog.3683.trace” file. The next test will be with the “--help” switch.

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

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

The --help switch provided a great deal of information. First, it is possible to get the version and possibly a man page. Next we see different switches that deal with data - “display data” and “place data.” There are also a couple of options that deal with some sort of “wipe” action. The below screen shots are from both the “prog-parameters” and “prog.3704.trace.” They are provided to show what *apptrace* captured and reported.

```
# cat prog-parameters
Sun Jan 25 08:03:57 MST 2004 - ./prog
Sun Jan 25 08:08:46 MST 2004 - ./prog --help
```

```
# cat prog.3704.trace | more
3714  execve("./prog.orig", ["./prog.orig", "--help"], [/* 32 vars */]) = 0
3714  fcntl64(0, F_GETFD)               = 0
3714  fcntl64(1, F_GETFD)               = 0
3714  fcntl64(2, F_GETFD)               = 0
3714  uname({sys="Linux", node="balder", ...}) = 0
3714  geteuid32()                       = 0
3714  getuid32()                        = 0
3714  getegid32()                       = 0
3714  getgid32()                        = 0
3714  brk(0)                            = 0x80bedec
3714  brk(0x80bee0c)                    = 0x80bee0c
3714  brk(0x80bf000)                    = 0x80bf000
3714  brk(0x80c0000)                    = 0x80c0000
3714  fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
3714  old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x40000000
3714  write(1, "prog:1.0.20 (07/15/03) newt\n", 28) = 28
3714  write(1, "Usage: prog [OPTION]... [<target"..., 44) = 44
3714  write(1, "use block-list knowledge to perf"..., 65) = 65
3714  write(1, "--doc VALUE\n", 12)     = 12
3714  write(1, "  where VALUE is one of:\n", 25) = 25
3714  write(1, "  version  display version and e"..., 36) = 36
3714  write(1, "  help  display options and exit"..., 33) = 33
3714  write(1, "  man  generate man page and exi"..., 34) = 34
3714  write(1, "  sgml  generate SGML invocation"..., 38) = 38
3714  write(1, "--mode VALUE\n", 13)    = 13
3714  write(1, "  where VALUE is one of:\n", 25) = 25
3714  write(1, "  m  list sector numbers\n", 25) = 25
3714  write(1, "  c  extract a copy from the raw"..., 40) = 40
3714  write(1, "  s  display data\n", 18) = 18
3714  write(1, "  p  place data\n", 16) = 16
3714  write(1, "  w  wipe\n", 10)       = 10
3714  write(1, "  chk  test (returns 0 if exist)"..., 33) = 33
3714  write(1, "  sb  print number of bytes avai"..., 38) = 38
3714  write(1, "  wipe  wipe the file from the r"..., 42) = 42
3714  write(1, "  frag  display fragmentation in"..., 55) = 55
3714  write(1, "  checkfrag  test for fragmentat"..., 70) = 70
3714  write(1, "--outfile <filename> write outpu"..., 41) = 41
3714  write(1, "--label\tuseless bogus option\n", 29) = 29
3714  write(1, "--name\tuseless bogus option\n", 28) = 28
3714  write(1, "--verbose\tbe verbose\n", 21) = 21
3714  write(1, "--log-thresh <none | fatal | err"..., 97) = 97
3714  write(1, "--target <filename> operate on ."..., 35) = 35
3714  munmap(0x40000000, 4096)          = 0
3714  _exit(0)                          = ?
```

Since it seems that this binary targets a user specified file, it is not believed that it will attack the hard drive or file system. At this point I'm going to create a small test file. It will contain the phrase "This is a test file used to test the "prog" program." The md5 hash and inode information for the test file is below.

```
# md5sum testfile.txt > testfile.txt.md5
# more testfile.txt.md5
c0cdc7d2051629dac6a236c41af753eb  testfile.txt

# ls -lai testfile.txt
1205743 -rw-r--r--    1 root     root           55 Jan 25 08:16 testfile.txt
```

The first test using the test file will be a display test. I'm going to use option "m – list sector numbers." This will just read the file information and not write or alter any data.

```
# ./prog -mode m testfile.txt
19447008
19447009
19447010
19447011
19447012
19447013
19447014
19447015
#
```

The above is the output with a list of sectors allocated for the "testfile.txt."

```
# cat prog.3800.trace | more
3810  execve("./prog.orig", ["./prog.orig", "-mode", "m", "testfile.txt"], [/*
32 vars */]) = 0
3810  fcntl64(0, F_GETFD)               = 0
3810  fcntl64(1, F_GETFD)               = 0
3810  fcntl64(2, F_GETFD)               = 0
3810  uname({sys="Linux", node="balder", ...}) = 0
3810  geteuid32()                       = 0
3810  getuid32()                        = 0
3810  getegid32()                       = 0
3810  getgid32()                        = 0
3810  brk(0)                            = 0x80bedec
3810  brk(0x80bee0c)                    = 0x80bee0c
3810  brk(0x80bf000)                    = 0x80bf000
3810  brk(0x80c0000)                    = 0x80c0000
3810  lstat64("testfile.txt", {st_mode=S_IFREG|0644, st_size=55, ...}) = 0
3810  open("testfile.txt", O_RDONLY|O_LARGEFILE) = 3
3810  ioctl(3, FIGETBSZ, 0xbffff544)    = 0
3810  ioctl(3, FIGETBSZ, 0xbffff4b4)    = 0
3810  ioctl(3, FIBMAP, 0xbffff544)      = 0
3810  fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
3810  old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x40000000
3810  _llseek(1, 0, 0xbffff2a0, SEEK_CUR) = -1 ESPIPE (Illegal seek)
3810  write(1, "19447008\n", 9)         = 9
3810  munmap(0x40000000, 4096)          = 0
3810  fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
3810  old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x40000000
3810  _llseek(1, 0, 0xbffff2a0, SEEK_CUR) = -1 ESPIPE (Illegal seek)
3810  write(1, "19447009\n", 9)         = 9
3810  munmap(0x40000000, 4096)          = 0
3810  fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
3810  old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x40000000
3810  _llseek(1, 0, 0xbffff2a0, SEEK_CUR) = -1 ESPIPE (Illegal seek)
3810  write(1, "19447010\n", 9)         = 9
3810  munmap(0x40000000, 4096)          = 0
3810  fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
3810  old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x40000000
3810  _llseek(1, 0, 0xbffff2a0, SEEK_CUR) = -1 ESPIPE (Illegal seek)
3810  write(1, "19447011\n", 9)         = 9
3810  munmap(0x40000000, 4096)          = 0
```

```
3810  fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
3810  old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x40000000
3810  _llseek(1, 0, 0xbffff2a0, SEEK_CUR) = -1 ESPIPE (Illegal seek)
3810  write(1, "19447012\n", 9)          = 9
3810  munmap(0x40000000, 4096)          = 0
3810  fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
3810  old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x40000000
3810  _llseek(1, 0, 0xbffff2a0, SEEK_CUR) = -1 ESPIPE (Illegal seek)
3810  write(1, "19447013\n", 9)          = 9
3810  munmap(0x40000000, 4096)          = 0
3810  fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
3810  old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x40000000
3810  _llseek(1, 0, 0xbffff2a0, SEEK_CUR) = -1 ESPIPE (Illegal seek)
3810  write(1, "19447014\n", 9)          = 9
3810  munmap(0x40000000, 4096)          = 0
3810  fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
3810  old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x40000000
3810  _llseek(1, 0, 0xbffff2a0, SEEK_CUR) = -1 ESPIPE (Illegal seek)
3810  write(1, "19447015\n", 9)          = 9
3810  munmap(0x40000000, 4096)          = 0
3810  close(3)                          = 0
3810  close(0)                          = 0
3810  _exit(0)                          = ?
[root@balder apptrace]#
```

The first thing that is apparent is the parameters that were passed to the binary. Next, the binary locates the passed file, finds its size and maps what drive space is allocated for this file. After the binary identifies each sector they are displayed on the screen. The program seems to be looking for space to place data. Next the "p – place data" switch will be tested.

```
# ./prog -mode p testfile.txt
stuffing block 2430876
file size was: 55
slack size: 4041
block size: 4096
this is a test to place hidden data in this text file.

# ./prog -mode s testfile.txt
getting from block 2430876
file size was: 55
slack size: 4041
block size: 4096
this is a test to place hidden data in this text file.

# md5sum testfile.txt >> testfile.txt.md5

# more testfile.txt.md5
c0cdc7d2051629dac6a236c41af753eb  testfile.txt
c0cdc7d2051629dac6a236c41af753eb  testfile.txt
```

After running the "p" switch, the cursor never came back to the screen. The only option was to either break out of the screen with a CTRL-C or try to type something in. The most logical choice was to type in some text so I typed in a phrase, then ran the binary again with the "s – display data" switch. The phrase

was “this is a test to place hidden data in this text file.” it was hidden within the “testfile.txt” file. Next *md5sum* was run to see if the hash had changed. It did not and the MAC times remained the same as well. Opening the file with a hex editor also failed to show the phrase.

```
# cat prog.14859.trace | more
14869 execve("./prog.orig", ["./prog.orig", "-mode", "p", "testfile.txt"], [/*
32 vars */]) = 0
14869 fcntl64(0, F_GETFD)               = 0
14869 fcntl64(1, F_GETFD)               = 0
14869 fcntl64(2, F_GETFD)               = 0
14869 uname({sys="Linux", node="balder", ...}) = 0
14869 geteuid32()                       = 0
14869 getuid32()                        = 0
14869 getegid32()                       = 0
14869 getgid32()                        = 0
14869 brk(0)                            = 0x80bedec
14869 brk(0x80bee0c)                    = 0x80bee0c
14869 brk(0x80bf000)                    = 0x80bf000
14869 brk(0x80c0000)                    = 0x80c0000
14869 lstat64("testfile.txt", {st_mode=S_IFREG|0644, st_size=55, ...}) = 0
14869 open("testfile.txt", O_RDONLY|O_LARGEFILE) = 3
14869 ioctl(3, FIGETBSZ, 0xbfffe144)    = 0
14869 lstat64("testfile.txt", {st_mode=S_IFREG|0644, st_size=55, ...}) = 0
14869 lstat64("/dev/hda3", {st_mode=S_IFBLK|0660, st_rdev=makedev(3, 3), ...})
= 0
14869 open("/dev/hda3", O_WRONLY|O_LARGEFILE) = 4
14869 ioctl(3, FIGETBSZ, 0xbfffe0b4)    = 0
14869 brk(0x80c2000)                    = 0x80c2000
14869 ioctl(3, FIBMAP, 0xbfffe144)      = 0
14869 write(2, "stuffing block 2430876\n", 23) = 23
14869 write(2, "file size was: 55\n", 18) = 18
14869 write(2, "slack size: 4041\n", 17) = 17
14869 write(2, "block size: 4096\n", 17) = 17
14869 _llseek(4, 9956868151, [9956868151], SEEK_SET) = 0
14869 read(0, "this is a test to place hidden d"..., 4041) = 57
14869 write(4, "this is a test to place hidden d"..., 57) = 57
14869 close(3)                          = 0
14869 close(4)                          = 0
14869 _exit(0)                          = ?
```

This is the information again collected by *apptrace* on the file **prog**. Again it identifies the provided file and calculates the slack size. Thus, you can see the text being read into the binary and written to the file slack space.

```
# ./prog -mode p testfile.txt
stuffing block 2430876
file size was: 55
slack size: 4041
block size: 4096
this is test 2 for hidden data

# ./prog -mode s testfile.txt
getting from block 2430876
file size was: 55
slack size: 4041
block size: 4096
this is test 2 for hidden data
data in this text file.
```

Since we have placed data into a file's slack space that was empty, what will happen if data is placed into a file's slack space that is not empty? The above test was run against the test file again, but without using any of the wipe options. This time the second string of data was written over part of the first string. So while the program will place data into file slack space, it will not alter what data is already there. It will only use as much space as required.

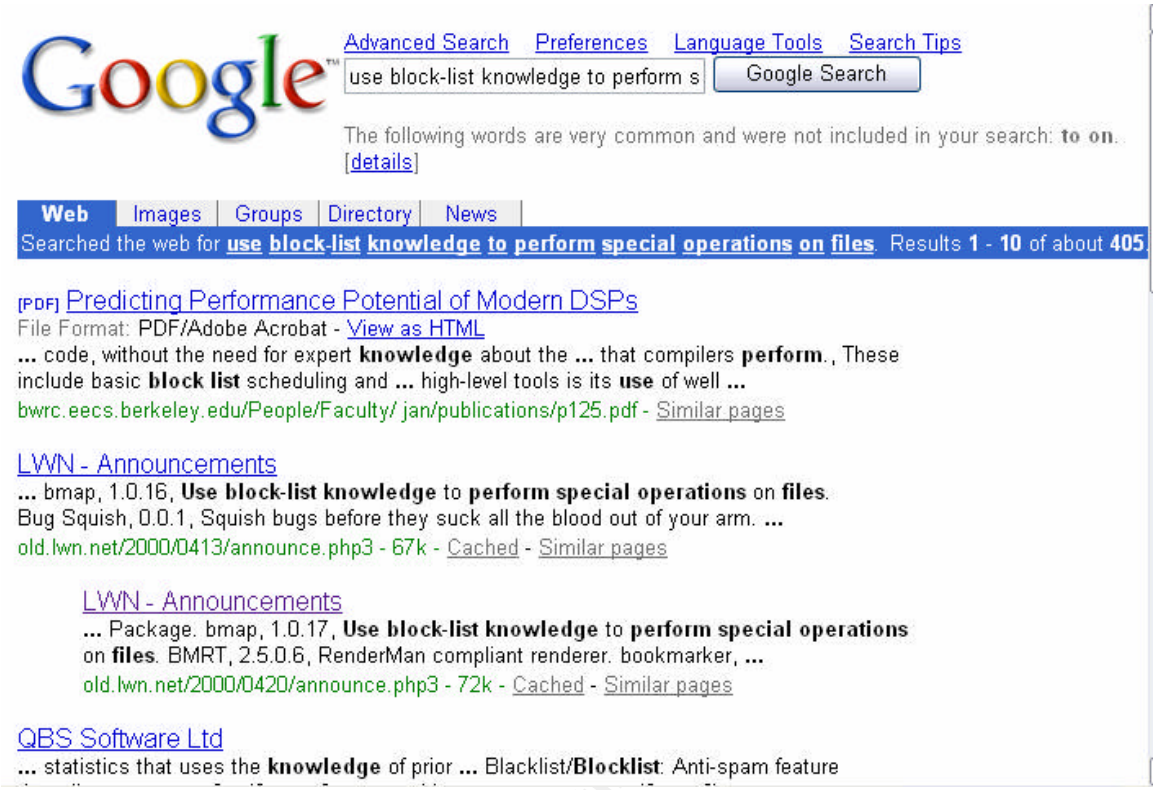
To sum up this section, the binary locates the user provided file, calculates the file size and how much slack is available. Next, the user has different options; he can place data within a file's slack space, display this data or wipe this data.

3. Forensic Details

As can be seen from the previous section the test file did not change in either its md5 hash or the MAC time stamps. This would greatly hinder a forensic investigation because two of the main investigators tools are known hash values and file system timeline. A hex editor could not find the hidden text, so the only sure way to find this type of hidden data is with the program itself. Scanning files with the binary would take a great deal of time and would require an individual to review each file output for pertinent information. A review of all actions monitored and reported by *apptrace* did not reveal any system calls that would have altered any system files. As such, this binary leaves no footprint on a file system and, short of running it against each file, will be undetectable.

4. Program Identification

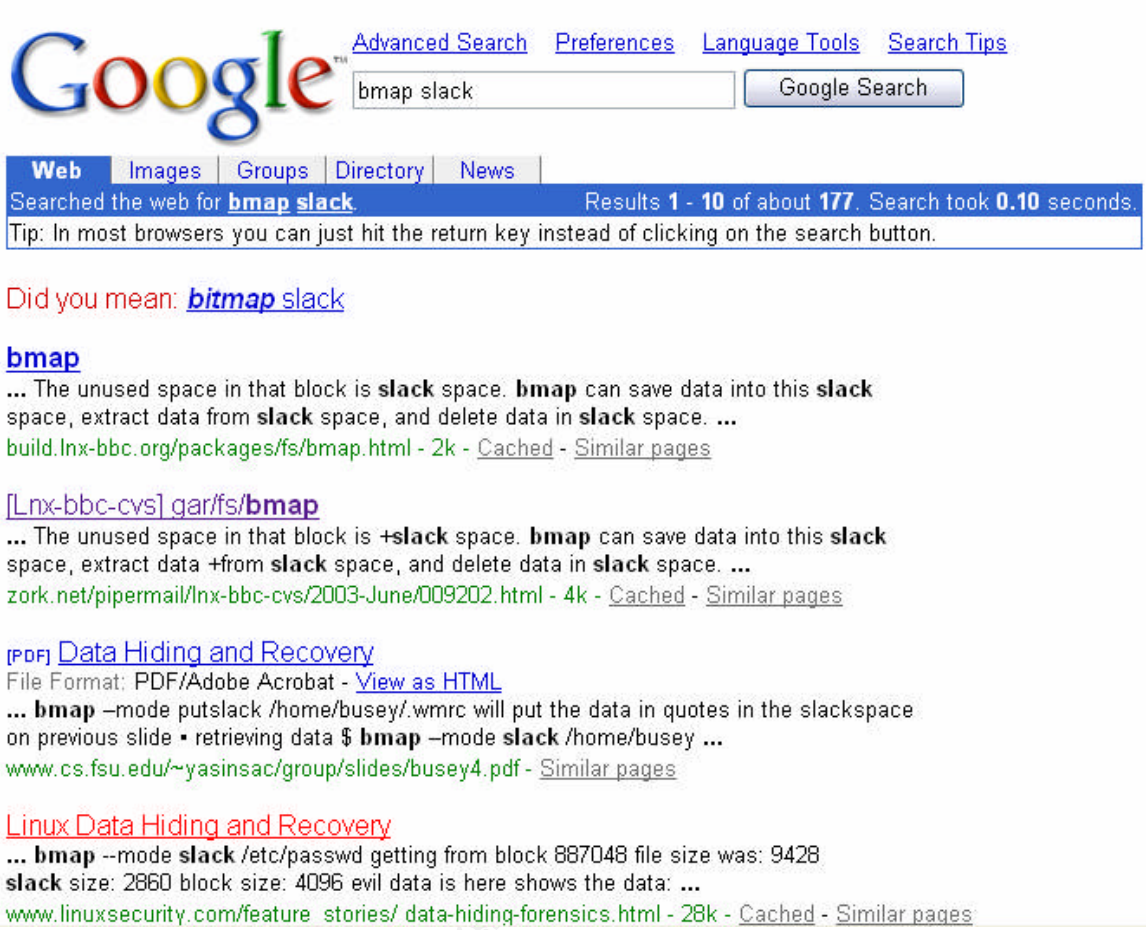
To start the search on the Internet, I go to the best tool a forensic examiner has, Google. The first search will be using the key term "use block-list knowledge to perform special operations on files." The reason for this is simple: this is such a large term the number of hits should be limited to only pertinent web sites.



The third selection, "LWN – Announcements," is a web site that has a link for a binary called bmap, with a description that matches my search term.

Blender	1.74a	Extremely fast and versatile 3D Rendering Package
bmap	1.0.17	Use block-list knowledge to perform special operations on files.
BMRT	2.5.0.6	RenderMan compliant renderer

The link for this goes to http://freshmeat.net/news/2000/04/16/955924691.html, however this page did not have bmap loaded.
For the next search I'm going to use bmap and slack. Bmap because strings provided me with that term and so did the above search. Slack because the binary works with slack space.



This provided me with a Linux Security website http://www.linuxsecurity.com/feature_stories/data-hiding-forensics.html and an article written by Anton Chuvakin, Ph.D on 3/10/2002. This article describes in great detail how a program called **bmap** can be used to hide data within a files slack space. Also on this page is a link to an ftp site, ftp://ftp.scyld.com/pub/forensic_computing/bmap/ that contains different versions of the bmap binary. Strings showed a 1.0.20 that could be a version number and this ftp site has a corresponding "bmap-1.0.20.tar.gz" file. This will be downloaded and tested to see if this is the binary.

The first step is to untar the new binary and review the files using `tar xvzf bmap-1.0.20.tar.gz`. A directory listing of the contents are shown below:

```
# ls
bclump.c  bmap.sgml.m4  COPYING   include     libbmap.c  Makefile  mft
slacker.c  bmap.c    bmap.spec     dev_builder.c  index.html  LICENSE    man
README  slacker-modules.c
```

Review of the README file provides some great information. See Attachment 2 for full content. If the information in the README file is to be believed, this

binary may have been used during an investigation. See the following which was found in the README file - “Written 1998 by Daniel Ridge in support of: Computer Crime Division, Office of Inspector General, National Aeronautics and Space Administration.”

Also the question of who is “newt” has been somewhat answered. The README file contained a large number of version changes. Each posting had a date and version number and the email address of the poster. All of the posts except one were from someone called “newt” – either from an email address of “newt@hq.nasa.gov” or “newt@scyld.com.” The one odd email address was “jakers@hq.nasa.gov.” This email address appears early in “newt’s” postings, possibly revealing that he posted his name in error.

“Newt” worked at “hq.nasa.gov” and now works for “scyld.com”, and seems to be the maintainer of this binary. The version changes provide a great way to learn everything this program is capable of.

First thing accomplished was to compile the binary to see what files were created.

```
# ls -al
total 1280
drwxr-xr-x    5 root     root         4096 Jan 25 09:20 .
drwxr-xr-x    3 root     root         4096 Jan 25 09:15 ..
-rwxr-xr-x    1 root     root        74581 Jan 25 09:20 bclump
-rw-r--r--    1 root     root        10364 May 29  2000 bclump.c
-rw-r--r--    1 root     root          506 Jan 25 09:20 bclump-invoke.sgml
-rw-r--r--    1 root     root        27604 Jan 25 09:20 bclump.o
-rwxr-xr-x    1 root     root       222101 Jan 25 09:20 bmap
-rw-r--r--    1 root     root        13030 May 15  2000 bmap.c
-rw-r--r--    1 root     root         1337 Jan 25 09:20 bmap-invoke.sgml
-rw-r--r--    1 root     root        35860 Jan 25 09:20 bmap.o
-rw-r--r--    1 root     root        15603 Jan 25 09:20 bmap.sgml
-rw-r--r--    1 root     root        12811 May 29  2000 bmap.sgml.m4
-rw-r--r--    1 root     root          824 May 15  2000 bmap.spec
-rw-r--r--    1 root     root        17159 Jan 25 09:20 bmap.tex
-rw-r--r--    1 root     root          266 Jan 25 09:20 config.h
-rw-r--r--    1 root     root        18008 Mar 24  2000 COPYING
-rwxr-xr-x    1 root     root        64495 Jan 25 09:20 dev_builder
-rw-r--r--    1 root     root         1728 Feb 24  2000 dev_builder.c
-rw-r--r--    1 root     root       128539 Jan 25 09:20 dev_entries.c
-rw-r--r--    1 root     root       175968 Jan 25 09:20 dev_entries.o
drwxr-xr-x    2 root     root         4096 Jan 25 09:15 include
-rw-r--r--    1 root     root          913 Feb 14  2000 index.html
-rw-r--r--    1 root     root         8546 Apr 11  2000 libbmap.c
-rw-r--r--    1 root     root        34484 Jan 25 09:20 libbmap.o
-rw-r--r--    1 root     root         1322 Apr 14  2000 LICENSE
-rw-r--r--    1 root     root         2384 May 29  2000 Makefile
drwxr-xr-x    3 root     root         4096 Jan 25 09:15 man
drwxr-xr-x    3 root     root         4096 Jan 25 09:20 mft
-rw-r--r--    1 root     root         6639 May 15  2000 README
-rwxr-xr-x    1 root     root       233827 Jan 25 09:20 slacker
-rw-r--r--    1 root     root         8905 Apr 27  2000 slacker.c
-rw-r--r--    1 root     root         1029 Jan 25 09:20 slacker-invoke.sgml
-rw-r--r--    1 root     root         5517 Mar  8  2000 slacker-modules.c
-rw-r--r--    1 root     root        30280 Jan 25 09:20 slacker-modules.o
```

```
-rw-r--r--    1 root     root        33400 Jan 25 09:20 slacker.o
```

After compiling the binary, the *file* command was run to see what properties the binary had. This showed that the program had not been stripped, so the *strip* command was run on the bmap binary. Also, the bmap binary was dynamically linked rather than statically linked. This translates into the fact that if this downloaded binary, and the unknown prog binary are the same, then the **"bmap.c"** code and **"Makefile"** were altered. Not only were the "bmap" references changed to "prog" but also, the code was statically compiled and stripped.

```
# 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

# strip bmap

# file bmap
bmap: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux
2.2.5, dynamically linked (uses shared libs), stripped
```

Before too much time is spent trying to compile this binary and get the md5 hash values to match, this binary needs to be compared with the prog binary. The *strings* command is run against the **bmap** binary and the results are compared with the output from **prog**. Not only did the two have several matches from the already identified important text fragments, but there were also several matches in blocks of code, for example:

prog	bmap
bmap_get_slack_block NULL value for slack_block Unable to stat fd Unable to determine blocksize 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 write error	bmap_get_slack_block NULL value for slack_block Unable to stat fd Unable to determine blocksize 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 write error

These two binaries seem to be one and the same; and testing will confirm this. The first step is to link this binary to *apptrace* so it will monitor all of the binaries actions. Also, since *apptrace* provides a monitoring report, comparing the two binaries actions will be much easier. The **bmap** binary is executed without any switches, and if it is like **prog** it should tell me to try again with "--help."

```
# ./bmap
no filename. try '--help' for help.
```

As can be seen, it did. The next step is to run it with "--help" and compare this output with the output for **prog.**

```
# ./bmap -help
bmap:1.0.20 (01/25/04) newt@scyld.com
Usage: bmap [OPTION]... [<target-filename>]
use block-list knowledge to perform special operations on files

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

This output looks remarkably like the "--help" output from **prog.** One of the big differences is this one spells out fully some of the switches, while **prog** uses one letter, for example:

prog	**bmap**
m list sector numbers	map list sector numbers
c extract a copy from the raw device	carve extract a copy from the raw device
s display data	slack display data in slack space
p place data	putslack place data into slack
w wipe	wipeslack wipe slack

This could be for speed of use by the user or just something to confuse the investigator. A review of the *apptrace* monitor report for **bmap** reveals similar outputs as the ones created for the **prog** binary.

```
[root@balder apptrace]# cat bmap.15314.trace | more
15324 execve("./bmap.orig", ["./bmap.orig", "-help"], [/* 32 vars */]) = 0
15324 uname({sys="Linux", node="balder", ...}) = 0
15324 brk(0)                              = 0x806df80
15324 old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x40016000
15324 open("/etc/ld.so.preload", O_RDONLY) = -1 ENOENT (No such file or
directory)
15324 open("/etc/ld.so.cache", O_RDONLY) = 3
15324 fstat64(3, {st_mode=S_IFREG|0644, st_size=81041, ...}) = 0
15324 old_mmap(NULL, 81041, PROT_READ, MAP_PRIVATE, 3, 0) = 0x40017000
15324 close(3)                            = 0
15324 open("/lib/tls/libc.so.6", O_RDONLY) = 3
15324 read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0\220W\1"..., 512)
= 512
15324 fstat64(3, {st_mode=S_IFREG|0755, st_size=1536292, ...}) = 0
15324 old_mmap(0x42000000, 1261416, PROT_READ|PROT_EXEC, MAP_PRIVATE, 3, 0) =
0x42000000
15324 old_mmap(0x4212f000, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED,
3, 0x12f000) = 0x4212f000
15324 old_mmap(0x42132000, 8040, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x42132000
15324 close(3)                            = 0
15324 old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
0) = 0x4002b000
15324 set_thread_area({entry_number:-1 -> 6, base_addr:0x4002b280,
limit:1048575, seg_32bit:1, contents:0, read_exec_only:0, limit_in_pages:1,
seg_not_present:0, useable:1}) = 0
15324 munmap(0x40017000, 81041)           = 0
15324 fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
15324 mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0)
= 0x40017000
15324 write(1, "bmap:1.0.20 (01/25/04) newt@scyl"..., 38) = 38
15324 write(1, "Usage: bmap [OPTION]... [<target"..., 44) = 44
15324 write(1, "use block-list knowledge to perf"..., 65) = 65
15324 write(1, "--doc VALUE\n", 12)       = 12
15324 write(1, "  where VALUE is one of:\n", 25) = 25
```

All the evidence is pointing to these two binaries being one in the same; the next test should confirm this. The test file created in the previous step will be used again, this time using **bmap** to read what the **prog** binary hid, then the test will be reversed.

```
# ./prog -mode s testfile.txt
getting from block 2430876
file size was: 55
slack size: 4041
block size: 4096
this is test 2 for hidden data
data in this text file.

# ./bmap -mode slack ../../testfile.txt
getting from block 2430876
file size was: 55
slack size: 4041
block size: 4096
```

```
this is test 2 for hidden data
data in this text file.
```

Now **prog** will be used to read what the **bmap** binary hid. This worked flawlessly; each can read the others hidden data.

```
# ./bmap -mode wipeslack ../../testfile.txt
stuffing block 2430876
file size was: 55
slack size: 4041
block size: 4096
write error
write error
write error

# ./bmap -mode slack ../../testfile.txt
getting from block 2430876
file size was: 55
slack size: 4041
block size: 4096

# ./bmap -mode putslack ../../testfile.txt
stuffing block 2430876
file size was: 55
slack size: 4041
block size: 4096
this is test3 with bmap to prog

# ./bmap -mode slack ../../testfile.txt
getting from block 2430876
file size was: 55
slack size: 4041
block size: 4096
this is test3 with bmap to prog

# ./prog -mode s testfile.txt
getting from block 2430876
file size was: 55
slack size: 4041
block size: 4096
this is test3 with bmap to prog
```

The **bmap** binary was then used to wipe the slack space and verify it had been wiped. It was then used to place a new hidden text message, which **prog** was able to read. This worked perfectly.

When a file has an md5 hash created for it, the md5 is reading all of the contents of that file, even the spaces. Review of the *strings* output from **prog** reveals some but not all of the **bmap** references were changed to **prog.** Since some of the references were changed, that would indicate that this was done manually. If the hex editing program had done the changes, it would have most likely changed all references.

If the individual changing these references had left a space or even added one without knowing it, that space would have affected the md5 hash. The chance of

editing the **bmap** code then compiling it to match the **prog** binary would be almost impossible. Below are three md5 hash values:

```
ff96c32eac425f0514db61e9276fa20a  bmap    Compiled
a8a10ce3f9bab84d5159ed3d928f4f0e  bmap    Compiled and Stripped
e62ebe512354ab4fe243e8e9c0403617  prog    Compiled, Stripped & Edited
```

Even something as simple as using the *strip* command changes the hash greatly. That is because *strip* also changes the contents of the binary it is run against.

After reviewing the article by Daniel Ridge, as well as analyzing the output from both *strings* and *apptrace*, it appears that these two binaries are the same. Furthermore, it appears that someone simply downloaded **bmap**, edited both the "Makefile" and "bmap.c" code and renamed it to **prog.**

5. Legal Implications

From the information provided in the scenario and researching the binary, the binary was executed. The program was compiled on "15 Jul 03." The "Sound-HOWTO-html.tar.gz" file had hidden data located within the file. The date stamp on the file was "16 Jul 03", so a version of **bmap** or **prog** had to be ran against this file. The file contained a hidden data stream of web site URL's. These web sites traffic in the downloading of MP3 files. The two main violations of law and policy are:

Offenses indicated by the evidence:

> Policy violations of this companies acceptable use policy, because the equipment used for the distribution of the illegal files is company owned.
>
> Copyright infringement: Trading of MP3s, these are copyrighted music, movies (DVD rips of copyrighted movies etc) under Law: 17 U.S.C. Chapter 5, Copyright Infringement and Remedies. http://www4.law.cornell.edu/uscode/18/2319A.html

Penalties for these violations can be, but not limited to the following:

> Acceptable use policy violation: If the policy states so, the subject's employment and the employment of all involved parties could be terminated; any damages done (i.e. lawsuits against the company because of the copyright infringement) could be attributed to the subject.
>
> Copyright infringement, under Law: 17 U.S.C. Chapter 5, Copyright Infringement and Remedies: Sections 502-506 describe the remedies. 17 U.S.C. Chapter 5, Section 504 (c) establishes that the subject or if working with others, can be held liable for damages from $750 to $30,000.

6. Interview Questions

The first thing I would do is talk to John and try to engage him in small talk. This will hopefully do two things; get him talking and put him at ease. Ask him about his favorite sport, or about the weather; anything to start to work my way into the following questions:

1. Who else has access to your office? (This would be used to try and establish that the floppy is his)

2. Do you use a screen saver password and if so, does anyone else know your screen saver password? (Again, this is to establish if the floppy is his)

3. If no one has your password and the floppy is not yours. Then how did a document called, "Mikemsg.doc," record metadata stating the user John Price was the last user to save that file? (The key here is how the first two questions go. Either way, this would be good information to use during the interview.)

4. What type of music do you listen to? Do you have an mp3 player in your vehicle or office? (This would be used to see if he readily uses mp3's or has access to players)

5. How many different networks and systems do you have access to? (This is a test question. I already know the answer to what he is authorized to have access to, but I want to see if he may have access to more.)

6. How many different operating systems are you comfortable using? (This was used on a Linux file system. This is a two fold question; establish floppy ownership and whether or not he knows Linux.)

7. Do you know what the program "netcat" does? (This is a setup question and also measures the depth of his knowledge on Linux.)

8. A review of our past Internal IDS logs revealed unusual network traffic originating from your system to another. What task were you performing? (This is the setup. He may or may not know if you have an Internal IDS. If he tries to bluff his way out by saying that it was authorized traffic, follow up by asking what program required this. If he says it could not have been him, push the issue and make him think you do have logs.)

9. Tell us what is going on; because all we want to do is be clear of this network anomaly we picked up. I mean, it is not like you were hacking or breaking any laws. (This is providing him with a way out. You are condoning his activity by saying he did not break any laws. Plus, you are not looking to hammer him, only to clear some paperwork.)

10. We have identified other individuals and are preparing to talk to them next. Is there anything you would like to say? (This is just another way out for him, let his story be told first before anyone else.)

There are several different questions you can ask, but all of them depend on how the interview is going. If he his talkative, then keep going with follow up questions to clarify a point. If he is being difficult, then just let him talk as much as you can before you start getting confrontational.

7. Case Information

The system administrator(s) needs to review all user accounts and access control lists (ACL) to ensure nothing is out of the ordinary and to ensure John Price only had access to the authorized systems. Also, all remote access accounts and any telecommuter accounts need to be checked.

Next, depending on how large this company this is, having everybody change passwords would not be a bad idea. Then all Linux and UNIX systems would need to be scanned with **bmap** to verify that no other data was hidden. This will be a very time intensive process, but the piece of mind that will come from knowing that all hidden data has been found will be worth it.

Next, all servers / systems that had an NTFS file system need to be searched for Alternate Data Stream (ADS) files. For all intents and purposes ADS is similar to **bmap** only for NTFS, so checking all NTFS servers / systems may not be a bad idea. Again, this would be very time consuming, but the piece of mind this would provide would be worth it.

This next part is confusing; this is taken from the section requirement: "What, if anything, did you find that would lead you to believe that John Price was using the organizations computing resources to distribute copyrighted material?" Now this part is taken form Part 1 scenario: "…audit discovered that he was using the organizations computing resources to illegally distribute copyrighted material." So if I say, you told me he was, that would be wrong?

The file on the floppy containing the hidden data may have been the way John was communicating with other people who were his downloading partners. The data is hidden in slack space, which is only present where the file is. Once the file is moved, the hidden data is lost. The hidden data had to be placed on the floppy, thus his computer had to put it there. Since the computer was used to help in the commission of a crime – which in this case was the downloading of MP3s – then John is in violation.

First I will check each file with **bmap** to see if any files have hidden data. If they do, that means that the data was hidden while the file was on the floppy. When a file is moved or copied, the slack space does not move.

The command line for this scan is:

```
# /forensics/SANS/test/prog –s(file on floppy loop to scan)
```

Only one file returned any unusual value. See the word “downloads.” That is very unusual to find a readable word in the middle on a large amount of unreadable characters.

```
# /forensics/SANS/test/prog -s Docs/Sound-HOWTO-html.tar.gz
getting from block 190
file size was: 26843
slack size: 805
block size: 1024
h? ? downloadsM???  Eâ”¬???Iâ–’âŽ»âŽ½4Æ¤????  ?BR  P??â””?\Ï¹'???
??/??Ï€???\â”,?
? Â“â‰¤Ó¹[âŽ¼âŽ°âŽ°â”œ@â ‰â–’â”Œâ  â ŠâŽ¼ â”ŒâŽ°âŽ°âŽ»]# ?P?W?â  â‰ Ý¥#????3â”,
?â ‰ ?Z/?3 ??H?A?M?$3â”œBâ ‹?â”¤]7N ?M3??â Š ?â Š??
```

After resetting my prompt, the above command is run again, this time redirecting the output to a file.

```
# /forensics/SANS/test/prog -s Docs/Sound-HOWTO-html.tar.gz >
/forensics/SANS/test/Sound-HOWTO-html.tar.gz.out
getting from block 190
file size was: 26843
slack size: 805
block size: 1024
```

This file is reviewed with the *file* command, which returns the following information.

```
# file -m /usr/share/magic Sound-HOWTO-html.tar.gz.out
Sound-HOWTO-html.tar.gz.out: gzip compressed data, was "downloads", from Unix
```

Since the file is a *gzip* file, a quick search of the Red Hat man page reveals a command called *zcat*. This command can read the contents of a *gzip* file. Below is the result.

```
# zcat Sound-HOWTO-html.tar.gz.out
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***
```

Other items of interest that were found in the floppy image:

.~5456g.tmp This is a programs temporary file, a hex editor review and *strings* command analysis, revealed to information. This file was most likely left behind by a program that could not or did not close cleanly. Since this file is on a floppy

disk, it is most likely the floppy was removed from the drive before the transaction was complete.

nc-1.10-16.i386.rpm..rpm NETCAT is a data transfer program for reading and writing data across a network. The program works on two parts, a server and a client. One listens and one transmits the data. Both halves can work together if they are on the same port.

DVD-Playing-HOWTO-html.tar This is a document explaining how to get DVD movie playback in Linux.

Kernel-HOWTO-html.tar.gz This a guide on how to configure the Linux kernel. This would be a very helpful reference to someone who either does not have a great deal of kernel knowledge, or is looking to tweak the kernel.

09 Letter.doc This seems to be a Company Letterhead template.

// letter template:
"Company Name Here
DATE
[Click here and type recipient's address]
Dear Sir or Madam:
*Type your letter here. For more details on modifying this letter template, double-click *. To return to this letter, use the Window menu.*
Sincerely,
[Click here and type your name]
[Click here and type job title]"

Mikemsg.doc This document contained the following:

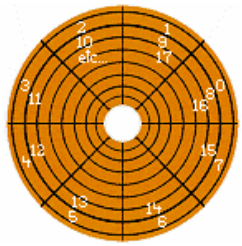
> ***Hey Mike,***
> ***I received the latest batch of files last night and I'm ready to rock-n-roll (ha-ha). I have some advance orders for the next run. Call me soon.***
>
> ***JP***

This could possibly be a reference to MP3's or music in general. If this is a coworker, all of his accounts need to be reviewed. Maybe his system can be looked at late at night to verify if he is involved. A review of the file properties in Windows show the author as "John Price." Also, the last saved by is "John Price" and it was last saved on "13 Jul 2003." This information can be used during questioning to prove ownership of the floppy.

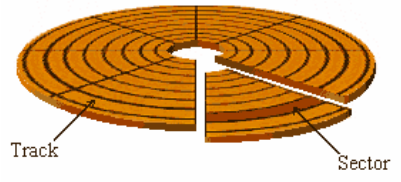
MP3-HOWTO-html.tar.gz This document is an MP3 guide for how to play, mix and stream MP3's.

Sound-HOWTO-html.tar.gz This document is a Linux sound support guide. **This is a file that also contains the hidden message with URLs to MP3 sites.**

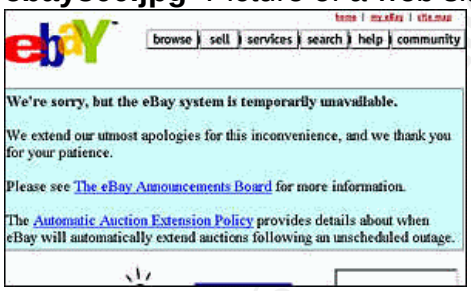
sect-num.gif Here is a picture of a disk drive platter. Notice how sectors are numbered, this looks almost like some sort of training aid to explain how data is stored on a drive platter.



sectors. gif Here is another picture of a disk drive platter. This is the same platter, now one step higher, going from sectors to tracks and clusters.



ebay300.jpg Picture of a web site, www.ebay.com.



To summarize, John has way too much time on his hands. It looks like he is doing a large amount of research on how to work with MP3 files. Maybe he and Mike are going to open a web site with MP3s for downloads.

8. Additional Information

Some different URL's for copyright material and binary source.

http://www4.law.cornell.edu/uscode/18/2319A.html
http://cyber.law.harvard.edu/mp3/
http://www.law.cornell.edu/copyright/courses/downloads/copyright_act.pdf
http://www.linuxsecurity.com/feature_stories/data-hiding-forensics.html
ftp://ftp.scyld.com/pub/forensic_computing/bmap/

Part 2 – OPTION 2: Perform Forensic Tool Validation

Choose a tool that is or could be used to help obtain forensic information from a system. This tool could be a tool that you have already been introduced to or a tool that you think may make a good forensic tool but has not had any testing performed on it. Choose a tool that has not had any testing accomplished on it. Your goal is to analyze the tool so you can show that the evidence it obtains is verifiable and repeatable. Your tests should include enough data to easily show if the tools output could possibly be supported or refuted if you are called to testify to the tools use in court.

1. Scope

The tool that will be tested is called "Metadata Assistant" (MA). This tool can be used in any Windows forensic environment to discover and display metadata hidden from within a file. This tool works predominantly with the files produced by three main programs: Microsoft Word, Microsoft Excel and Microsoft Power Point. This tool was designed to analyze a file extracted from a forensic image or a logical file structure. The file is then analyzed by MA and any hidden metadata can be displayed and reported.

This validation will test if the tool makes any changes to the selected file as well as determining how well the tool extracts metadata. The tools marketing information states that MA will "display its findings then offer the ability to clean the document", thus allowing an individual to see, erase or alter the metadata in a file. Since this tool can work on Word, Excel and Power Point documents, files produced by each program will be analyzed with this tool.

The major point of this validation is the fact that this is a utility to assist the forensic investigator. Because of this, care must be taken on any files analyzed by this tool. Simple metadata such as MAC times will be changed by moving or copying a file. However, other embedded metadata like date last printed, will not be changed. Knowledge of computer forensics is assumed for the use of this tool as one must manipulate a file in order to examine it.

2. Tool Description

Metadata Assistant (MA) was created and is supported by the Payne Consulting Group, Inc., Seattle, WA. The organization also has other products, "Forms Assistant", "Number Assistant" and "BATES Label Maker." The version of MA being test is v. 1.61.0161 and can be found at http://www.payneconsulting.com/public/products/ProductDetail.asp?nProductID=7.

This product is not free; however for the information it provides the $79.00 price tag is well worth it. This tool has the capability to retrieve large amounts of hidden data from within a file.

Each and every day Intellectual Property cases are becoming more prevalent. Having a quick and easy method to retrieve this hidden data will greatly decrease case processing time. This tool currently only works in a Windows environment and requires the above indicated programs to work. This program is not a forensic tool, but rather a tool to assist the forensic investigator. The following types of metadata are identified and displayed by MA:

- built-in document properties
- document statistics
- custom document properties
- last 10 author information
- template
- routing slip information
- document versions
- tracked changes
- fast saves
- hidden text
- document comments
- embedded graphics
- hyperlinks
- document variables
- Smart Tags (Word 2002)
- Include Fields
- Font Size 1
- White Font

Again, this tool is designed for assisting the forensic examiner in extracting metadata content. As such, it was deemed not important to determine what system files were utilized during the execution of this tool. This tool was installed and tested on a forensic analysis machine, which contained many other forensic programs. It was not tested from a CD-ROM as it would never be used in an Incident Response capacity. It was also felt that this tool would only be used to assist in a forensic exam.

3. Test Apparatus

The testing scenario for this tool was comprised of two primary environments. The first environment was in a computer forensic laboratory and tested on an Intel based computer. This computer was comprised of the following:

- AMD Athlon 2500+ XP Processor
- 1 GB of memory.
- 40 GB 7200 RPM Western Digital Hard Drive
- Two Removable Drive Bays (Not Utilized)
- Sony 4X +_DVD RW
- Windows XP Professional, service pack 1 last updated with all of the updates from Microsoft Update on Dec 04, 2003.

- Microsoft Office XP Professional, Service Pack 2, last updated with all of the updates from Windows Office update on Dec 04, 2003.
- Many various forensics software tools (Encase, FTK, NetAnalysis, etc)

The second setting was in a normal production environment on my work laptop. This was a Sony PCG-R505GCP laptop and contained the following:

- Mobile Intel Pentium III 1200 MHz Processor
- 512 MB of memory.
- 30 GB 7200 RPM IBM Hard Drive
- Sony 8X +_CDRW
- Windows XP Professional, service pack 1 last updated with all of the updates from Microsoft Update on Jan 23, 2004
- Microsoft Office XP Professional, Service Pack 2, last updated with all of the updates from Windows Office update on Jan 23, 2004.

The primary software testing environment will be in Microsoft Windows and will utilize Microsoft Office – specifically, Word 2002, Excel 2002, and PowerPoint 2002. Since the MA tool can analyze files on both a local system and from a network share the test will only analyze the activity from the local machine. This is simply because Word, Excel, and PowerPoint do not exist on the network but rather reside on the local computer.

4. Environmental Conditions

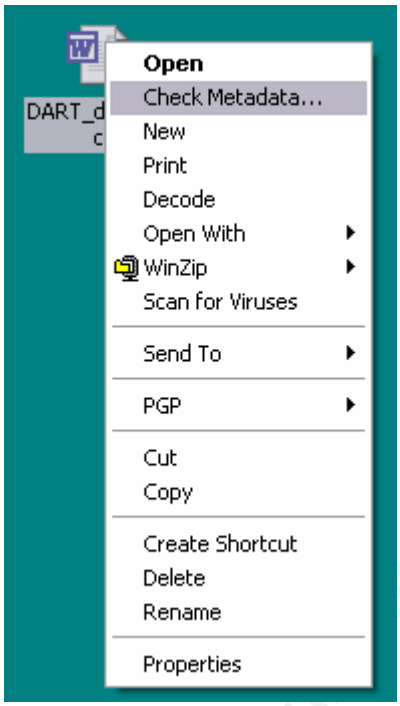
All of the analysis for these tests will be conducted on the systems described above. The two environments are drastically different. The laboratory systems stand completely on their own – no Internet or other network connectivity exists. The system also contains many tools utilized in computer forensic analysis, such as Encase, FTK, etc. The production environment consists of my work laptop which is used on a daily basis and does have an Internet connection. However, the purpose of the tool is clear – to analyze metadata. Therefore the environment is not a critical factor assuming one understands how files are volatile and that their MAC times can very easily be changed. This is, however, not extremely critical since the tool deals with many types of metadata that will not be easily changed by copying and/or moving files.

5. Description of Procedures

The first step in analyzing and validating this tool was to install it on the two individual test systems. The Payne Consulting Group (PCG), Inc. web site was checked and the latest version of MA was downloaded. The file which was downloaded was called `metadataassistant.exe`. This file was subsequently installed by double clicking on it. A standard install was conducted with all of the defaults chosen.

When MA installs, several files are created and/or extracted in to the local machines C:\Program Files\Payne Consulting Group\Metadata Assistant directory. Once MA has finished the installation process there will be three different program options in the programs menu; “Metadata Assistant”, “Metadata Assistant Excel” and “Metadata Assistant PP.” This allows the user to manually select which program they want to run.

When MA is installed, several registry keys are also created for the program. Once installed, MA can be selected from any one of several Windows pop up menus. When a single file is selected and the right mouse clicked, you have the option to “Check Metadata…” When this is selected MA is launched and the file is already mapped to be analyzed.



During the course of this test, several files with the .doc, .xls and .ppt file extensions will be identified and analyzed by MA. These are the files most commonly associated with Word, Excel and Power Point programs.
The first step in analyzing a file will be to right click on a file and review its properties. Windows only provides some of the metadata that MA can discover.

The tests were conducted a total of 3 times for each file. MA was run against a known Microsoft Word file 3 times, with each test comprising a different scenario. The same tests were also conducted on a known Power Point file and a known Excel file in the same fashion as the Microsoft Word tests. Each test comprised of the following methods after locating one of each type of file (.doc, .ppt, .xls):

Test 1 – (Retrieve all metadata):

1. A baseline needed to be set so a MD5 hash was created for the file by running the MD5 utility by Dan Mares - www.dmares.com/maresware/.
2. The primary Windows metadata which includes modified, access, and created (MAC) times, author, title and summary information is retrieved by right clicking on file and viewing the file properties.
3. MA is executed by right clicking on the file and selecting “Check Metadata”. The metadata is saved out to a separate file for later viewing.
4. Run an MD5 hash on the file to see if the file contents have changed.
5. Right click on the file and retrieve file properties to see any changes.

One of the problems with metadata tests is that every time a file is copied/moved from device to device the creation and access times change and each time the file is copied/move between directories the access time changes. This demonstrates the value of the MA program as it shows so much more information.

After a file has been identified, an md5 hash will be created for each file before and after MA has processed the files. This is done to ensure that while the file may be read, no data has been changed.

Test 2 - (Discover changes to document statistics and hidden text):

This test will determine how text can be hidden within a document and still be found by the MA program as well as how specific metadata like last print time can be shown.

1. The baseline from test 1 will be used.
2. The primary Windows metadata from test 1 will be used.
3. The file will be manipulated:
    a. The file will be opened for editing
    b. Text will be hidden by inserting it as the same color as the background.
    c. The file will be saved to test the saved time metadata.
    d. The file will be printed to test the print time metadata.
    e. The file will be closed.
4. The program MA will then be executed by right clicking on the file and selecting “Check Metadata”. The metadata is saved out to a separate file for later viewing.
5. Run an MD5 hash on the file to see if it the file contents have changed.
6. Right click on the file and retrieve file properties to see any changes.

Test 3 – (Test the clean metadata ability):

This test will determine how metadata within a document can be purged.

1. The baseline from test 1 will be used.

2. The primary Windows metadata from test 1 will be used.
3. The program MA will be executed by right clicking on the file and selecting "Check Metadata".
    a. The metadata will be analyzed to determine what exists.
    b. The metadata will be cleaned using the program.
4. Run an MD5 hash on the file to see if it the file contents have changed.
5. Right click on the file and retrieve file properties to see any changes.
6. The program MA will be executed again by right clicking on the file and selecting "Check Metadata".
    a. The metadata will be analyzed to determine if it has been cleaned.

The tests were first run on the forensics laboratory computer system by extracting the required files (doc, ppt, xls) from a forensics image using Encase 4.17. The 3 files were copied to the forensics machine desktop.

The same 3 tests from above were then conducted on the production laptop using 3 files contained on the laptop itself. In addition to the testing of files on the laptop an additional test was conducting utilizing the MA programs network functionality. This test consisted of conducting tests 1 and 2 across a network connection on a known Microsoft Word file.

6. Criteria for Approval

The most important thing to remember is that this tool does manipulate the files, but not the contents, that are being analyzed. However, this is not terribly important considering its purpose is to garner the metadata which will not be changed by this programs use.

Since this tool will most likely be run against a file exported or copied from within a disk image, the original file will still be preserved. In addition the ability to make copies or preserve the original files exists. The MA tool is going to access data contained within the selected file. With this tool being able to run on any system and the data it is extracting held within the file, any system files that are touched or changed are not important for the approval of this tool.

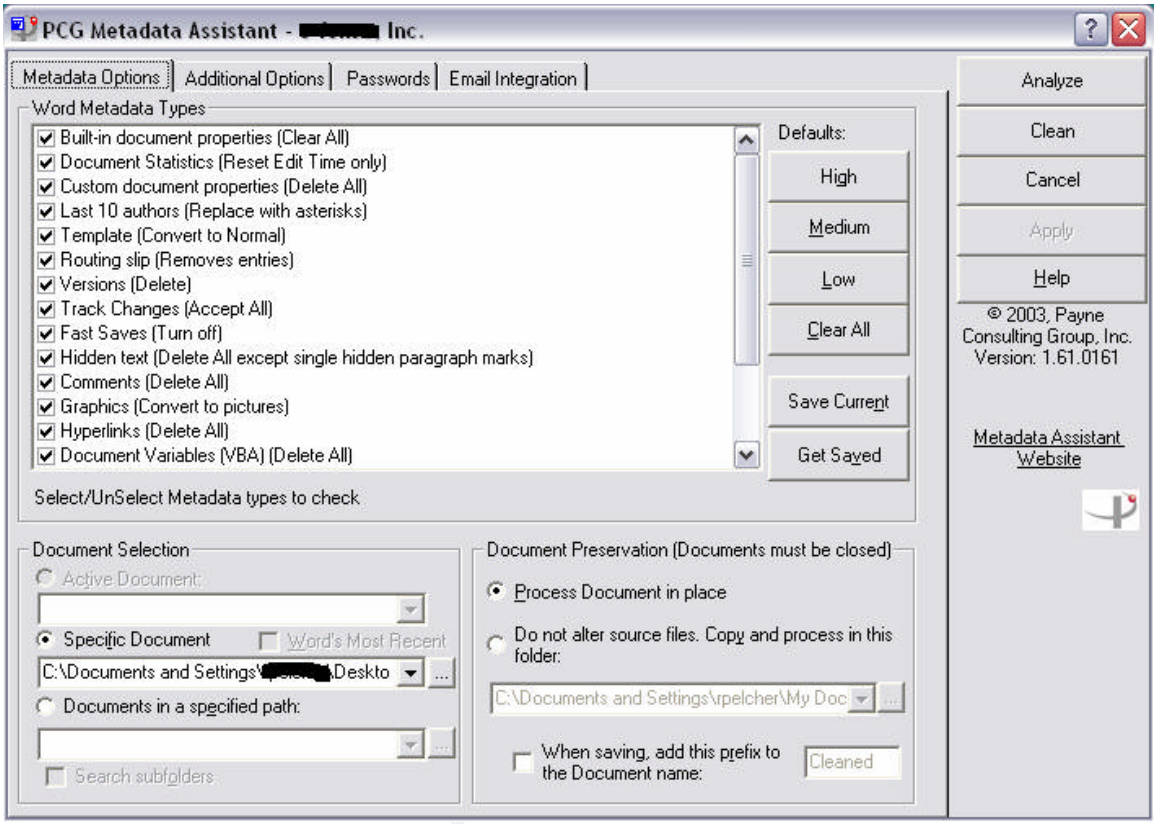
The best results would be for this program to read the metadata from a file and not alter any date and time stamps or change the md5 hash of the file being analyzed. But, the best may not be possible. The tests would be considered a success if the tool reports all of the found metadata it reads and does not change that same metadata. In addition the tool would be considered successful if it can detect metadata changes.

7. Data and Results

The screenshots from this section are from the test conducted on the Microsoft Word document. Adding screenshots for the other two tested files would have

been too repetitive as the screenshots were identical. The only differences were in the output files that were included to show the results.

The main screen of MA for Word is below:



This provides an overview of all of the options available to select from. First is to identify the file you wish to analyze. This can be done in one of two methods. Either right click on any Office file of the type Word, Excel, or Power Point and select “Check Metadata” or use the document browser key located in the lower left corner. This shows the full path for the file that was selected.

After the file has been selected you can select from group defaults or any of 19 individual selections concerning which data is to be retrieved. If a group default, “High”, “Medium” or “Low”, is selected then a certain number individual items will be selected. If “Low” is selected, then 3 items are selected, 13 for “Medium” and all for “High.” This provides a great deal of flexibility in selecting what types of data, or how much data is retrieved. For this test, the “High” option will be selected.

One of the options is to provide a password, if one is required to open the document. This can be found on the “Passwords” tab. One great security feature for this tool is that any password entered, does not carry over from session to session.

Another interesting option is on the “Additional Options” tag that is the amount of detail you require in your report. You can choose from three different options:

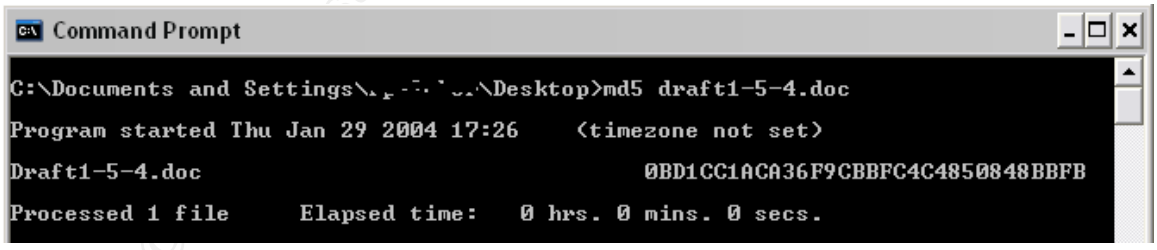
Level 1 – Summary Only
Level 2 – Standard Details
Level 3 – Full Details

Each level provides a greater detail of what metadata was identified.

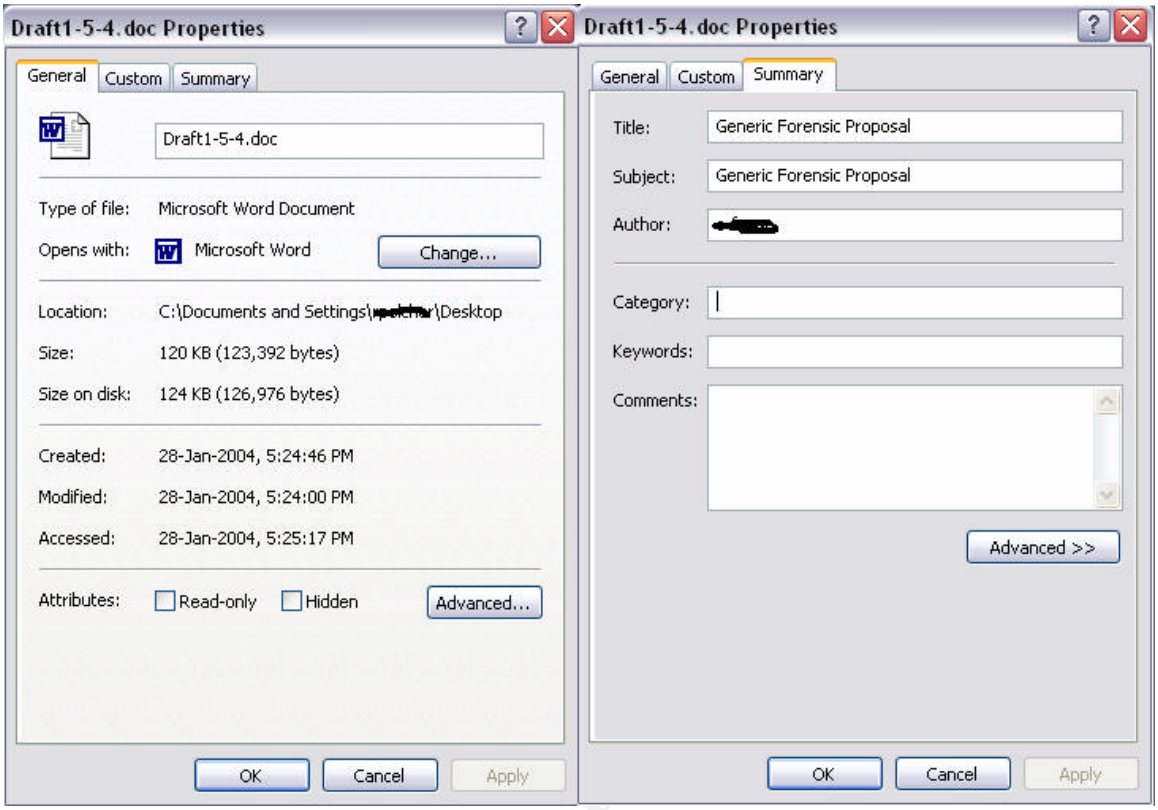
The last main tab - “Email Integration” - allows a user to select different options to help clean outbound email traffic. This is designed to work with “Outlook 2000” or higher and “GroupWise 6.0.1” and higher. This provides the user with the options of being prompted concerning an outbound email from one of the recognized file extensions monitored by this tool and selecting what actions they would like taken. If this program was located on a Subjects system, knowing if this option is enabled could be important. In this version of MA 1.61.0161, this feature is enabled by default.

Results for TEST 1:
The first test involved a Word file located within an Encase image file on the forensics analysis system. This Word file was chosen at random from the image file and extracted from Encase by simply right clicking on the file and selecting copy/unerase. The file was copied to the desktop for easy analysis.

The first step to accomplish according to the testing procedures, was to create an MD5 hash of the file just copied to the desktop. Remember the original file is located within an Encase image file so the original is still forensically sound if we need it. The purpose of the MD5 hash is to let us know if the contents of the file changes. The MD5 tool from Dan Mares was executed on the file and provided the following output:



Next I obtained the windows properties for the file by right clicking on it and selecting properties. As can be seen by the following two screenshots Windows contains basic information such as the created, accessed, modified time stamps as well as size, name, and location. Additionally the summary tab may contain the author and title of the document.



As can be seen there is very little metadata that can be identified. So the next step is to run the file through the MA program. The first screen to appear is the one previously identified as the main MA Word screen. Leaving all of the defaults set to "High" and selecting "Analyze", several things take place at once.

A small pop up window message box appears with a title of "Metadata Assistant Status Box" and a message of "Starting Word." Once Word starts the title bar changes to "Analyzing: \File Name" and the full path of the file being analyzed is displayed. Within the status box a real time running process list is displayed, showing each of the individually checked switch options from the main page.

Once the analysis is complete, the report screen pops up with the results. This contains all of the metadata identified by MA and it provides you with two different data save options and one review identified data option. For this test, the data will be saved with the "Save Results as RTF" option.

Another option, "Switch to Detailed View", takes the identified data and overlays this against the selection switchboard from the main screen. This allows you to quickly identify which metadata could be cleaned.

The output from the tool is a "Level 2 - Standard Details" report generated in a Rich Text Format for the above selected file.

Analyzing C:\Documents and Settings\xxxxxxx\Desktop\Draft1-5-4.doc

Document Name: Draft1-5-4.doc
Path: C:\Documents and Settings\xxxxxxx\Desktop
Document Format: Word Document

Built-in document properties:
Built-in Properties Containing Metadata: 4
Title: Draft Test
Subject: Generic Forensic Proposal
Author: John Doe
Company: xxxxxxx

Document Statistics:
Document Statistics Containing Metadata: 6
Creation Date: 1/5/2004 1:19 PM
Last Save Time: 1/20/2004 4:33 PM
Time Last Printed: [Blank]
Last Saved By: John T. Doe
Revision Number: 5
Total Edit Time (Minutes): 12 Minutes

Custom document properties:
No Custom Document Properties

Last 10 authors:
Has Last 10 Data
xxxxxxxxx

Attached Template (Convert to Normal):
Attached to Normal

Routing slip:
Has Routing Slip: 0 Recipient(s)

Versions:
No Versions

Track Changes:
No Tracked Changes

Fast Saves:
Fast Saves is Off

Hidden text:
No Hidden Text

Comments:
No Comments

Graphics:
Embedded Objects: 1
Object 1 Word.Picture.8

Hyperlinks:
Hyperlinks: 1
Text: jdoe@yahoo.com

Hyperlink: mailto: jdoe@yahoo.com

Document Variables (VBA):
No Document Variables

Smart Tags:
Smart Tags: 0

Remove Personal Information:
Remove Personal Information: Off Store Random Number is On

Include Fields:
Does not contain any Include Fields

Font size 1:
No Font 1 Text

White font:
No White Font Text

A review of this report can provide an investigator with a ton of helpful information. The following are just a few that would be important to anyone performing a computer forensic analysis.

Document Statistics: Internal document times

Last Save Time: 1/20/2004 4:33 PM
Time Last Printed: [Blank]
Last Saved By: John T. Doe

This information may be invaluable when trying to prove when a document was last printed or even who saved it.

Last 10 authors:
xxxxxxxxx

Again, when dealing with most internal Intellectual Property investigations, knowing something as simple as where the document was last saved may be vital to your investigation.

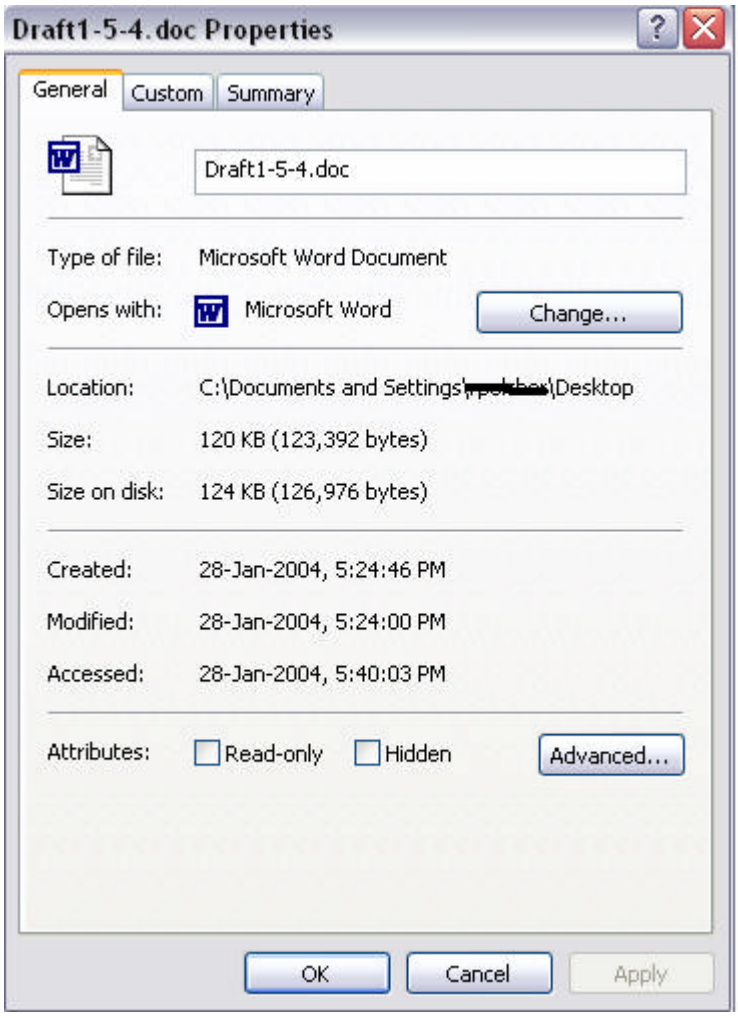
Comments:
No Comments

This is a unique feature. This will list any embedded comments in the document. Again, having all of this data pulled out and placed in and organized easy to read format can save countless hours.

Fast Saves:
Fast Saves is Off

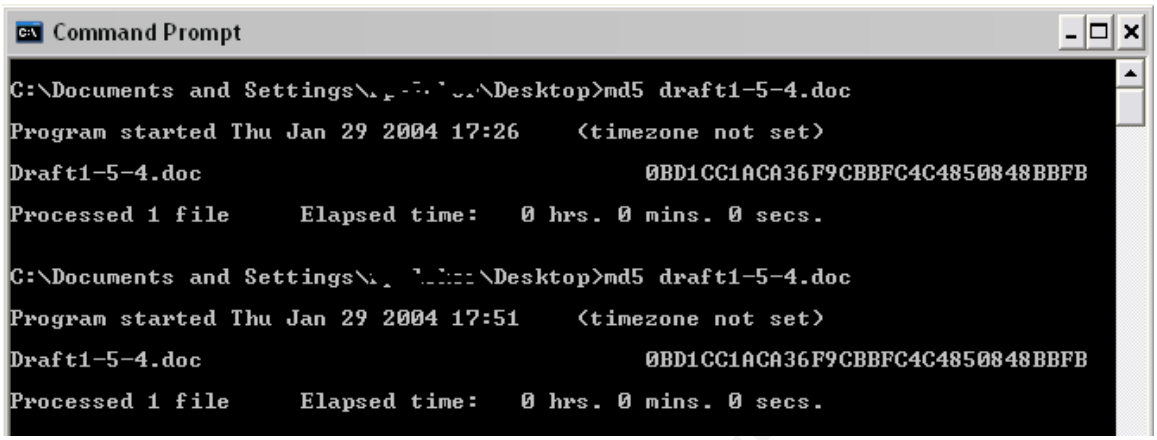
This is another unique feature that an investigator will find helpful to his investigation. Fast Save reduces the amount of time taken to save changes to a document. The side effects are quite unique, the file size will continue to grow vastly greater then the input changes. The reason is because Word does not save the changes, it saves the whole document, plus a record of all of the changes. The second problem is as you continue to make changes, the original version and the new version will be stored. This could be a problem if you send one of the documents to someone, they may be able to see the other document.

After finishing the report review, I again right clicked on the file and selected properties. Below is the file property value after MA has analyzed the selected document.



A quick review of the file's property showed only one date and time stamp changed that was the Accessed. This date and time stamp represents the last time the file had its contents read. No changes took place in the file or the modified date and time stamp would have changed. Also the md5 hash will verify if the contents have changed.

MD5 was run again against the file upon completion of MA. Here are the two md5 hashes. They match; no changes to the file's content have taken place.



Since the md5s match and only the Access date and time stamp were changed, the file itself is still intact.

This test was repeated on an Excel file and a Power Point file with the exact same results. The only change made was to the access time stamp. However, all the metadata from each file was extracted and provided in a report exactly like the one given above.

This test was also conducted on the production laptop with the exact same results, leading me to believe that this tool could be used in all types of environments.

TEST 2:
For the second test, I edited the same Word document used in test 1 and changed three lines of text from black to white. This is one of those myths that has been around forever, hiding data by changing the text to white to match the background. The original metadata can be seen below:

Analyzing C:\Documents and Settings\xxxxxxx\Desktop\Draft1-5-4.doc

Document Name: Draft1-5-4.doc
Path: C:\Documents and Settings\xxxxxxx\Desktop
Document Format: Word Document

Built-in document properties:
Built-in Properties Containing Metadata: 4
Title: Draft Test
Subject: Generic Forensic Proposal
Author: John Doe
Company: xxxxxx

Document Statistics:

Document Statistics Containing Metadata: 6
Creation Date: 1/5/2004 1:19 PM
Last Save Time: 1/20/2004 4:33 PM
Time Last Printed: [Blank]
Last Saved By: John T. Doe
Revision Number: 5
Total Edit Time (Minutes): 12 Minutes

Custom document properties:
No Custom Document Properties

Last 10 authors:
Has Last 10 Data
xxxxxxxxx

Attached Template (Convert to Normal):
Attached to Normal

Routing slip:
Has Routing Slip: 0 Recipient(s)

Versions:
No Versions

Track Changes:
No Tracked Changes

Fast Saves:
Fast Saves is Off

Hidden text:
No Hidden Text

Comments:
No Comments

Graphics:
Embedded Objects: 1
Object 1 Word.Picture.8

Hyperlinks:
Hyperlinks: 1
Text: jdoe@yahoo.com
Hyperlink: mailto: jdoe@yahoo.com

Document Variables (VBA):
No Document Variables

Smart Tags:
Smart Tags: 0

Remove Personal Information:
Remove Personal Information: Off Store Random Number is On

Include Fields:
Does not contain any Include Fields

Font size 1:
No Font 1 Text

White font:
No White Font Text

As stated I added three lines of text to the document but made the text color white to match the background. The file was then saved and printed. Upon re-examining the file with the MA tool it was noticed that the metadata had indeed been changed. The program found the hidden text in seconds and not only counted how many lines but also what was in the masked lines. It also found other changes in the metadata, specifically the dates when the file was saved and printed, as seen below.

Analyzing C:\Documents and Settings\xxxxxxx\Desktop\Draft1-5-4.doc

Document Name: Draft1-5-4.doc
Path: C:\Documents and Settings\xxxxxxx\Desktop
Document Format: Word Document

Built-in document properties:
Built-in Properties Containing Metadata: 4
Title: Draft Test
Subject: Generic Forensic Proposal
Author: John Doe
Company: xxxxxx

Document Statistics:
Document Statistics Containing Metadata: 6
Creation Date: 1/5/2004 1:19 PM
Last Save Time: 1/27/2004 2:13 PM
Time Last Printed: 1/27/2004 2:13 PM
Last Saved By: Robert Pelcher
Revision Number: 6
Total Edit Time (Minutes): 13 Minutes

Custom document properties:
No Custom Document Properties

Last 10 authors:
Has Last 10 Data
xxxxxxxxx
xxxxxxxxx

Attached Template (Convert to Normal):
Attached to Normal

Routing slip:
Has Routing Slip: 0 Recipient(s)

Versions:
No Versions

Track Changes:
No Tracked Changes

Fast Saves:
Fast Saves is Off

Hidden text:
No Hidden Text

Comments:
No Comments

Graphics:
Embedded Objects: 1
Object 1 Word.Picture.8

Hyperlinks:
Hyperlinks: 1
Text: jdoe@yahoo.com
Hyperlink: mailto: jdoe@yahoo.com

Document Variables (VBA):
No Document Variables

Smart Tags:
Smart Tags: 0

Remove Personal Information:
Remove Personal Information: Off Store Random Number is On

Include Fields:
Does not contain any Include Fields

Font size 1:
No Font 1 Text

White font:
Blocks of White Font Text: 3
White Font: 1
Text: this is test 1

White Font: 2
Text: this is test 2

White Font: 3
Text: this is test 3

As can be clearly seen from the output quite a few pieces of metadata had been changed. As a note, the "White Font" is only applicable to Word documents. Any white font in Excel or Power Point slides is not detected by this program at this time.

Obviously it can be seen that automating the process of searching each page for hidden text, would greatly speed the process time for an investigation. This frees up the investigator for other duties.

It was decided not to run the MD5 hash on the files as the file had obviously been altered, which would cause the MD5 hash not to match. The properties had changed as well since I had made alterations to the file itself. This test was also conducted on the Excel spreadsheet and the Power Point file with almost identical results. The only difference in results was that in Excel and Power Point the "White Font" is called Font Matching Background. It was also tested on the production laptop with identical results.

TEST 3:
The final test was run to see if the clean feature of MA left any evidence behind. As with many tools appearing on the Internet, some are used to hide nefarious activity. If this tool is to be used to hide illegal activity, hopefully it will leave a trace or footprint for the investigator to find.

The files used for this test are again the same from the previous tests. Here is the report output after the "Clean" option was selected. Notice how it identifies each entry and what its status is, "Changed To", "Unchanged", "Cleaned" or "Removed."

Cleaning C:\Documents and Settings\xxxxxxx\Desktop\Draft1-5-4.doc

Document Name: Draft1-5-4.doc
Path: C:\Documents and Settings\xxxxxxx\Desktop
Document Format: Word Document

Built-in document properties:
Built-in Properties Containing Metadata: 4. CLEANED: 4
Title: Draft Test CHANGED TO [Blank]
Subject: Generic Forensic Proposal CHANGED TO [Blank]
Author: John Doe CHANGED TO [Blank]
Company: e-fense CHANGED TO [Blank]

Document Statistics:
Document Statistics Containing Metadata: 6. CLEANED: 1
Creation Date: 1/5/2004 1:19 PM UNCHANGED
Last Save Time: 1/20/2004 4:33 PM UNCHANGED
Time Last Printed: [Blank] UNCHANGED
Last Saved By: John T. Doe UNCHANGED
Revision Number: 5 UNCHANGED
Total Edit Time (Minutes): 12 Minutes CHANGED TO 0

Custom document properties:
No Custom Document Properties

Last 10 authors:
No Last 10 Author Data

???

Attached Template (Convert to Normal):
Attached to Normal

Routing slip:
Has Routing Slip: 0 Recipient(s). REMOVED

Versions:
No Versions

Track Changes:
No Tracked Changes

Fast Saves:
Fast Saves is Off

Hidden text:
No Hidden Text

Comments:
No Comments

Graphics:
Embedded Objects: 1 CONVERTED: 1
Object 1 Word.Picture.8 CONVERTED

Hyperlinks:
Hyperlinks: 1. DELETED: 1
Text: jdoe@yahoo.com
Hyperlink: mailto: jdoe@yahoo.com REMOVED

Document Variables (VBA):
No Document Variables

Smart Tags:
Smart Tags: 0. DELETED: 0

Remove Personal Information:
Remove Personal Information: Off Store Random Number is On REMOVE PERSONAL INFORMATION TURNED ON

Include Fields:
Does not contain any Include Fields

Font size 1:
No Font 1 Text

White font:
No White Font Text

Analyzing this report after running the “Clean” option shows several fields were changed to blank. Other fields had all of the stored data completely removed. This could allow an individual to make linkage to a document harder, if not impossible.

After shutting down MA and reviewing this file with MA again showed only the current values, no indication of past values. A review of the program's home directory revealed no logs or program reports indicating that a file's metadata had been cleaned. Without the above report, showing which fields had been changed, no footprint of the changes would be found.

The only detectable change to the file was the md5 hash. Since MA opened the file to edit the metadata, it changed the contents of the file. If an investigator was able to get an md5 hash of a file before it was cleaned, it would confirm that the data had changed, but it still would not be able to display the old removed data.

Final Test:

Network-based files were selected for the last test. These files are located on a file server, available to all employees. MA was able to locate all selected files when they were right clicked and "Check Metadata" was selected. One surprise was the fact that there seemed to be no speed lost running this tool against a network-stored file. Considering the main programs open on the local system, and read the required information over a network.

An interesting side note. When I attempted to clean this document on the network file server, it returned "Not Processed" for almost all of the data areas. When the document was moved to my local machine and run MA with the "Clean" option again, the above output was produced.

8. Analysis

The data output from this tool is very straight forward, as shown in the last section. With its easy to read layout and clear header format, this output could be read and understood by anybody. But, the most important value is in the speed in which it can pull information from a file, group it all together and present it for review. This allows an investigator to quickly review volumes of files faster then using a hex editor or right clicking and reading each files data available through windows properties. In addition, some of this metadata is not easily accessible.

The output, while easy to understand, is broken down as the following for Microsoft Word:

Metadata Type	Details
Built-In Document Properties	Information contained in the **Summary** tab of the Document Properties dialog box. This information travels with the document, can reveal potentially sensitive and/or confidential information, and can be easily exposed.

Document Statistics (Primary Info Only)	Information contained in the **Statistics** tab of the Document Properties dialog box. Includes Last Saved By, Total Editing Time and so on.
Custom Document Properties	Information contained in the **Custom** tab of the Document Properties dialog box. Custom properties are stored with the document and are often added to provide more specific information about the document. This information might be sensitive or confidential in nature.
Last Ten Authors	The names of the last ten authors who have worked on a document are stored with the document and can be easily exposed using a basic text editor. Author information is pulled from the **Name** field in the User Information tabin the Options dialog box.
Template	Every document created in Word is attachedto an underlying template. The attached template can provide potentially revealing information as to the true origin or location of the document.
Routing Slip	Document Routing is a feature in Word that allows you to route a document to one or more users for review/editing. The names of the routing recipient(s) are stored in a routing slip and can be easily exposed.
Versions	Words **Versions** feature create versions of a document within the same document. Each version is actually a part of the original file as opposed to a separate, unique file. The contents of each version can be easily exposed using a text editor. It is highly recommended that this feature not be used for this and other reasons, including dramatically increased file size.
Track Changes	Track Changes is a feature in Word used to highlight changes made to a document. Depending on the number of authors making changes to a given document, and the filter set to display those changes, it can often be difficult to detect track changes. If not fully accepted/rejected, track changes can reveal potentially sensitive information about the document and/or the author.
Fast Saves	Fast Saves is a feature in Word that appends changes made to a document at the end of the file rather than overwriting the entire file. Appended text can be easily exposed using a basic text editor, and can also increase file size dramatically
Hidden Text	Text within Word documents can be formattedas hidden. Although hidden text does not print (unless that option is turned onit can be difficult to detect online if Show/Hide is not activated, and as a result can be unknowingly passed along. Hidden text can be easily exposed using a basic text editor, or simply by turning on Show/Hide.
Comments	Comments often contain confidential/sensitive information about the contents of a given document. Comment also include the name of the person inserting the comment as well as the time the comment was inserted. In some cases, (particularly when working in Normal view in Word 2002), comments can be difficult to detect.

Graphics	Some Word documents contain embedded objects - such as Excel worksheets. Embedded objects can be opened within the document so that confidential information including formulas, hidden rows and other sensitive information might be revealed.
Hyperlinks	Hyperlinks can point to file servers, intranet locations, web sites and other potentially confidential areas. The underlying URL associated with each hyperlink can be easily exposed to provide revealing or sensitive information about the document, the author, or related information.
Document Variables	Custom macros, add-ins and other third party programs often rely heavily on the use of document variables within Word documents. Document variables can reveal among other things, potentially sensitive and/or confidential information about the document, and/or the author of the document.
Smart Tags (Word 2002 or Higher)	Smart tags are a feature in Word 2002 that recognize and label, among other things, names, addresses and places within a document. The information associated with Smart tags can provide potentially sensitive and/or confidential information about the document author, the document itself, or related information.
Include Fields	A security 'hole' that affects all versions of Word but is most problematic in Word 97. Using the **IncludeText** or **Include Picture** field, a person could embed information into a file that would allow them to steal files from a target computer without the target user's knowledge.
White Font	Document text may be intentionally formatted with a font color of white, making it 'invisible' in the document but easily discoverable.
Font Size 1 pt.	Document text may be intentionally formatted with a font size of 1 pt., making it difficult to detect in a long document.
Remove Personal Information (Word 2002 or Higher)	This option is available in Word 2002 and higher and is located on the **Security** tab of the Options dialog boxWhen turned on, this option removes the following personal information from the document: • Document Properties • Names associated with Comments or Tracked Changes. • Routing slip • The e-mail message header that's generated with the E-mail button is removed. • Versioning

The following is the output explanation for Microsoft Excel:

Metadata Type	Details

Built-In File Properties	Information contained in the **Summary** tab of the Book Properties dialog box. This information travels with the document, can reveal potentially sensitive and/or confidential information, and can be easily exposed.
Custom Workbook Properties	Information contained in the **Custom** tab of the Book Propertiesdialog box. Custom properties are stored with the file and are often added to provide more specific information about the file. This information might be sensitive or confidential in nature.
Blank Cell Contents	Due to formatting or other Excel features, a cell that contains values may actually display as blank.
Routing Slip	Workbook Routing is a feature in Excel that allows you to 'route' a workbook to one or more users for review/editing. The names of the routing recipient(s) are stored in a routing slip and can be easily exposed.
Shared Workbook and Track Changes	Track Changes are used to highlight changes made to a worksheet. Depending on the number of authors making changes to a given worksheet, and the filter set to display those changes, it can often be difficult to detect track changes. If not fully accepted/rejected, track changes can reveal potentially sensitive information about the worksheet and/or the author.
Comments	Comments often contain confidential/sensitive information about the contents of a given file. Comments also include the name of the person inserting the comment as well as the time the comment was inserted.
Graphics	Some Excel files contain embedded objects. Embedded objects can be 'opened' within the worksheet so that confidential information might be revealed.
Hyperlinks	Hyperlinks can point to file servers, intranet locations, web sites and other potentially confidential areas. The underlying URL associated with each hyperlink can be easily exposed to provide revealing or sensitive information about the file, the author, or related information.
Defined Names	Defined names can reveal potentially sensitive information you may not want others to see.
Smart Tags (Excel 2002 or Higher)	Smart tags are a feature in Excel 2002that recognize and label recent Outlook e-mail recipients. This potentially sensitive information can be recovered under certain circumstances.
Remove Personal Information (Excel 2002 or Higher)	This option is available in Excel 2002 and higher and is located on the Security tab of the Options dialog box. When turned on, this option limits the information that is given on the Summary tab of the Propertiesdialog box.
Scenarios	Scenarios are sets of data used to analyze models and create potential outcomes based on specific data.
Font Matching Background	Cell text may be intentionally formatted with a font color similar to the background color, making it 'invisible' in the worksheet, but easily discoverable.
Small Font	Cell text may be intentionally formatted with small font size., making it difficult to detect in a worksheet.

Hidden Rows	Rows in a worksheet can be hidden and contain potentially confidential data.
Hidden Columns	Columns in a worksheet can be hidden and contain potentially confidential data.
Hidden Worksheets	Worksheets within a Workbook can be hidden.
Pivot Table Cache	Information regarding Pivot Table Cache could potentially include metadata about the source data used to populate the Pivot Table Cache.
Hidden Objects	This includes text boxes, shapes, or other objects that could be inadvertently hidden or left in a worksheet.
Header and Footer Content	In Excel, Header/Footer content is only visible in the printed file, or in Print Preview - making it easy to overlook.
Remove Custom Styles from Workbook	Styles that may include firm or organization name or initials or other potentially revealing information.
Check for External File Links	Excel files can have links to other Excel files.
Check for Custom Views	Using custom views, Excel files can be setup to display filtered ranges of data, by in effect, 'hiding' other data within the worksheet.
AutoFilter	This feature allows a user to view selective sets of data while potentially 'hiding' other data within the worksheet.

The following is the output explanation for Microsoft PowerPoint:

Metadata Type	**Details**
Built-In File Properties	Information contained in the **Summary** tab of the File Propertiesdialog box. This information travels with the document, can reveal potentially sensitive and/or confidential information, and can be easily exposed.
Custom File Properties	Information contained in the **Custom** tab of the File Propertiesdialog box. Custom properties are stored with the file and are often added to provide more specific information about the file. This information might be sensitive or confidential in nature.
Graphics	Some PowerPoint presentations contain embedded objects - such as Excel worksheets and other PowerPoint presentations. Embedded objects can be opened within the presentation so that confidential information including formulas, hidden data and other sensitive information might be revealed.
Hidden Objects	This includes text boxes, shapes, or other objects that are programmatically hidden in a presentation.
Fast Saves	Fast Saves is a feature in PowerPointthat appends changes made to a presentation at the end of the file rather than overwriting the entire file. Appended text can be easily exposed using a basic text editor, and can also increase file size dramatically

Remove Personal Information (PowerPoint 2002 or higher)	This option is available in PowerPoint 2002 and higher and is located on the Security tab of the Options dialog boxWhen turned on, this option removes the following personal information from the presentation: • Presentation Properties • Names Associated with Comments or Tracked Changes • Routing Slip • The e-mail message header that's generated with the E-mail button is removed.
Header and Footer Content	In PowerPoint, Header/Footer content is visible in the slides. In the Presentation Notes it is visible in the printed file, or in Print Preview - making it easy to overlook.
Hidden Slides	Slides within a presentation can be can be hidden.
Routing Slip	Presentation Routing is a feature in PowerPointthat allows you to 'route' a presentation to one or more users for review/editing. The names of the routing recipient(s) are stored in a routing slip and can be easily exposed.
Comments	Comments often contain confidential/sensitive information about the contents of a given presentation. Comments also include the name of the person inserting the comment as well as the time the comment was inserted. In some cases (particularly when viewing slides in a Slide Sorter, Slide Show or Notes Page), comments can be difficult to detect.
Hyperlinks	Hyperlinks can point to file servers, intranet locations, web sites and other potentially confidential areas. The underlying URL associated with each hyperlink can be easily exposed to provide revealing or sensitive information about the file, the author, or related information.
Font Matching Background	Slides may be intentionally formatted with a font color similar to the background color, making the text 'invisible' in the presentation but easily discoverable.
Small Font (size 3 or smaller)	Presentation text may be intentionally formatted with a font size of 3 pt., making it difficult to detect in a slide.

9. Presentation

This tool offers a flexible means of storing the retrieved data. With the "Save Results as RTF" or "Open Results as an XML Document" buttons, the investigator can select the output that's most favorable. Since this tool is windows-based, a windows file browser window pops up, so you are allowed to save your report in any location.

The output of this tool is in an easily-readable format. Each section header describes the grouped information, further broken down into each reportable area. An example is provided below:

Document Statistics:
Document Statistics Containing Metadata: 6.
Creation Date: 12/4/2002 8:49 AM
Last Save Time: 12/4/2002 11:03 AM
Time Last Printed: 12/4/2002 10:37 AM
Last Saved By: Robert Pelcher

As you can see this information could be easily understood by a lay person. This gives the investigator a final report as soon as it is generated.

10. Conclusion

This tool would not have any use in an incident response situation. It is a valuable forensic investigation assistance tool only. Many investigations are driven by the allegations "he said, she said", or "who knew what, when", this tool finally gives the investigator a quick and easy to use tool that can answer these questions.

While it is possible to retrieve metadata solely by looking at the file itself or by using a hex editor, it is apparent that this tool does the job but makes it easier for the investigator. As was shown the file could have been copied/moved to any location but the main metadata evidence of the file still existed and was able to be retrieved. This is an invaluable resource for forensic investigators.

That said, I believe this tool test was a success and the program "Metadata Assistant" has a place in the computer forensic investigator's tool bag.

Part 3 - Legal Issues of Incident Handling

Question A: *Based upon the type of material John Price was distributing, what if any, laws have been broken based upon the distribution?*

17 U.S.C. Chapter 5, Copyright Infringement and Remedies
Sec 501 Infringement of Copyrights, (a) States:

"Anyone who violates any of the exclusive rights of the copyright owner as provided by sections 106 through 121 or of the author as provided in section 106A(a), or who imports copies or phonorecords into the United States in violation of section 602, is an infringer of the copyright or right of the author, as the case may be. For purposes of this chapter (other than section 506), any reference to copyright shall be deemed to include the rights conferred by section 106A(a). As used in this subsection, the term "anyone" includes any State, any instrumentality of a State, and any officer or employee of a State or instrumentality of a State acting in his or her official capacity. Any State, and any such instrumentality, officer, or employee, shall be subject to the provisions ofthis title in the same manner and to the same extent as any nongovernmental entity."

http://www4.law.cornell.edu/uscode/17/501.html

The artists and record labels that created the music are in fact the owners, no matter what form the music takes. Whether written in a song book, or even when the music is converted into an MP3 format they are still the owner. When John downloads songs and passes them along, the true owners of the music are not receiving their fair compensation. Record labels alone are responsible for hundreds of employees. Since these individuals rely on music sales for there income, John is depriving them of their fair wage. Because of the information found, he is in violation of this statue.

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

The system owner or administrator needs to notify upper management of this discovery immediately. Depending on your company's policy, maybe even general counsel will need to be notified. Once this has been done, collection of evidence must take place until management has made a decision. In the corporate world, a computer security incident is really a business decision. The decision will be whether to proceed with the investigation or simply apply the patch and forget it. No matter which decision is made, all incidents need to be treated the same. Every step must be documented with the utmost care, including chain of custody for any evidence. If your Incident Response policy calls for monitoring internal networks or monitoring authorized users then go ahead and start monitoring.

Traffic monitoring is allowed under the Provider Exception of the 18 U.S.C. part I Ch. 119 Sec. 2511 (2)(a)(i) - Interception and Disclosure of Wire, Oral, or Electronic Communications Prohibited. Under this circumstance, to protect the rights of the company's network infrastructure and servers, monitoring of traffic is authorized. This plan of action needs to be approved by everybody prior to any incident occurring. Investigative plans of action that provide a clear and concise course of action are best when they are created before they are needed, not during the incident.

"(2)(a)(i) It shall not be unlawful under this chapter for an operator of a switchboard, or an officer, employee, or agent of a provider of wire or electronic communication service, whose facilities are used in the transmission of a wire or electronic communication, to intercept, disclose, or use that communication in the normal course of his employment while engaged in any activity which is a necessary incident to the rendition of his service or to the protection of the rights or property of the provider of that service, except that a provider of wire communication service to the public shall not utilize service observing or random monitoring except for mechanical or service quality control checks."

The best scenario would be to have banners in place prior to the incident, and a specific course of action predetermined for responding to the incident. General Counsel, upper management & key personnel should all be involved in creating

the banner as well as the steps that will be taken once an incident is identified. Then when an incident does occur, minimum data lose will occur because everyone will know their role and the incident will become a training exercise instead.

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

This is becoming a very common occurrence, corporations making a business decision to not pursue the investigations. That's why each and every incident needs to be treated like it will go to trial. Once everything is documented and all electronic media is transferred to a permanent storage media, an after actions report needs to be created. This has two purposes; first it summarizes all of your evidence and its current status. Next, it allows for easy research and review of the incident if such actions are required. Finally, all of the investigators notes, worksheets, checklists, and the archived media need to be stored in an appropriate location – one with controlled limited access and suitable to storage of electronic media. One last note, the after actions report can also be used to refine the incident handling process. As incidents change, so should the way we handle them.

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

If child pornography is found on any system, management and law enforcement are to be notified immediately. Child pornography is considered contraband and needs to be dealt with immediately. Child Porn falls under Title 18 USC hap 110, Sec. 2252A - Certain activities relating to material involving the sexual exploitation of minors. The following is part taken from this site: http://www.washingtonwatchdog.org/documents/usc/ttl18/ptI/ch110/sec2252.html

(a) Any person who -
(1) knowingly transports or ships in interstate or foreign commerce by any means including by computer or mails, any visual depiction, if -
(A) the producing of such visual depiction involves the use of a minor engaging in sexually explicit conduct; and
(B) such visual depiction is of such conduct;
(2) knowingly receives, or distributes, any visual depiction that has been mailed, or has been shipped or transported in interstate or foreign commerce, or which contains materials which have been mailed or so shipped or transported, by any means including by computer, or knowingly reproduces any visual depiction for distribution in interstate or foreign commerce or through the mails, if -
(A) the producing of such visual depiction involves the use of a minor engaging in sexually explicit conduct; and
(B) such visual depiction is of such conduct;

The bottom line is this: child porn is contraband and as such there is a very high probability that law enforcement will seize your systems. That is why having a process in place to minimize the number of systems that come in contact with this will also limit the number of systems seized. Limit which systems have child porn on them by not taking any steps to backup or image the suspect computers until after law enforcement has given the approval to go ahead.

Attachment 1
Strings from “prog”

PTRh	/dev/sdbp9
QVhx	/dev/sdbp8
h@=	/dev/sdbp7
hK=	/dev/sdbp6
h@=	/dev/sdbp5
0hT=	/dev/sdbp4
h@=	/dev/sdbp3
0he=	/dev/sdbp2
8-tx	/dev/sdbp15
h@=	/dev/sdbp14
h@=	/dev/sdbp13
h!>	/dev/sdbp12
h=>	/dev/sdbp11
hP>	/dev/sdbp10
0hm>	/dev/sdbp1
h@=	/dev/sdbp
h$?	/dev/sdbo9
h-?	/dev/sdbo8
h7?	/dev/sdbo7
h7?	/dev/sdbo6
hH?	/dev/sdbo5
hH?	/dev/sdbo4
PhI?	/dev/sdbo3
hI?	/dev/sdbo2
hU?	/dev/sdbo15
h[?	/dev/sdbo14
h`?	/dev/sdbo13
hg?	/dev/sdbo12
hn?	/dev/sdbo11
hs?	/dev/sdbo10
hy?	/dev/sdbo1
Ph @	/dev/sdbo
hg@	/dev/sdbn9
0h A	/dev/sdbn8
0h@A	/dev/sdbn7
0h B	/dev/sdbn6
0h B	/dev/sdbn5
0h&C	/dev/sdbn4
0h5C	/dev/sdbn3
0hDC	/dev/sdbn2
0hXC	/dev/sdbn15
0h_C	/dev/sdbn14
0hbC	/dev/sdbn13
hhC	/dev/sdbn12
0hnC	/dev/sdbn11
hzC	/dev/sdbn10
h,D	/dev/sdbn1
0h7D	/dev/sdbn
0h`D	/dev/sdbm9
0h_C	/dev/sdbm8
0hbC	/dev/sdbm7
hzC	/dev/sdbm6
h,D	/dev/sdbm5

0h-E	/dev/sdbm4
h@I	/dev/sdbm3
hdI	/dev/sdbm2
hxI	/dev/sdbm15
h)J	/dev/sdbm14
hCJ	/dev/sdbm13
h`J	/dev/sdbm12
h K	/dev/sdbm11
PhAK	/dev/sdbm10
PhHK	/dev/sdbm1
hLK	/dev/sdbm
hbK	/dev/sdbl9
huK	/dev/sdbl8
hbK	/dev/sdbl7
huK	/dev/sdbl6
hbK	/dev/sdbl5
huK	/dev/sdbl4
PhHK	/dev/sdbl3
hHK	/dev/sdbl2
h L	/dev/sdbl15
h M	/dev/sdbl14
h:M	/dev/sdbl13
hKM	/dev/sdbl12
h`M	/dev/sdbl11
RPSQ	/dev/sdbl10
RPSQ	/dev/sdbl1
[^_]	/dev/sdbl
h@N	/dev/sdbk9
I,RPSQ	/dev/sdbk8
h_N	/dev/sdbk7
p8hxN	/dev/sdbk6
h@N	/dev/sdbk5
h`M	/dev/sdbk4
h@O	/dev/sdbk3
heO	/dev/sdbk2
H^_]	/dev/sdbk15
X^_]	/dev/sdbk14
[^_]	/dev/sdbk13
[^_]	/dev/sdbk12
tY9u	/dev/sdbk11
}.;]	/dev/sdbk10
L[^_]	/dev/sdbk1
< v61	/dev/sdbk
ug;]	/dev/sdbj9
s=;]	/dev/sdbj8
PSh|	/dev/sdbj7
[^_]	/dev/sdbj6
XZSV	/dev/sdbj5
VPVS	/dev/sdbj4
[^_]	/dev/sdbj3
[^_]	/dev/sdbj2
[^_]	/dev/sdbj15
<bt!<b	/dev/sdbj14
RPSW	/dev/sdbj13
/FBH~	/dev/sdbj12
[^_]	/dev/sdbj11
;C tU	/dev/sdbj10

t$QVPS	/dev/sdbj1
PVRS	/dev/sdbj
;S t<	/dev/sdbi9
[^_]	/dev/sdbi8
G +G	/dev/sdbi7
G +G	/dev/sdbi6
t';]	/dev/sdbi5
V +V	/dev/sdbi4
FDHP	/dev/sdbi3
[^_]	/dev/sdbi2
[^_]	/dev/sdbi15
t PS	/dev/sdbi14
VPRQ	/dev/sdbi13
[^_]	/dev/sdbi12
VPRQ	/dev/sdbi11
~"PSV	/dev/sdbi10
CDHP	/dev/sdbi1
[^_]	/dev/sdbi
F0+V	/dev/sdbh9
[^_]	/dev/sdbh8
V$PQ)	/dev/sdbh7
~VQSV	/dev/sdbh6
[^_]	/dev/sdbh5
FAJy	/dev/sdbh4
~>PS	/dev/sdbh3
[^_]	/dev/sdbh2
AFJy	/dev/sdbh15
[^_]	/dev/sdbh14
[^_]	/dev/sdbh13
[^_]	/dev/sdbh12
[^_]	/dev/sdbh11
C t	/dev/sdbh10
XZSh	/dev/sdbh1
[^_]	/dev/sdbh
[^_]	/dev/sdbg9
F\xX	/dev/sdbg8
[^_]	/dev/sdbg7
P0+H	/dev/sdbg6
[^_]	/dev/sdbg5
L dQ	/dev/sdbg4
C +C	/dev/sdbg3
RPWV	/dev/sdbg2
2PVS	/dev/sdbg15
[^_]	/dev/sdbg14
[^_]	/dev/sdbg13
[^_]	/dev/sdbg12
[^_]	/dev/sdbg11
@t:	/dev/sdbg10
[^_]	/dev/sdbg1
Ph"@	/dev/sdbg
[^_]	/dev/sdbf9
[^_]	/dev/sdbf8
[^_]	/dev/sdbf7
w&;=	/dev/sdbf6
[^_]	/dev/sdbf5
[^_]	/dev/sdbf4
[^_]	/dev/sdbf3

PWVS	/dev/sdbf2
[^_]	/dev/sdbf15
tB;u	/dev/sdbf14
$v?V	/dev/sdbf13
[^_]	/dev/sdbf12
[^_]	/dev/sdbf11
[^_]	/dev/sdbf10
[^_]	/dev/sdbf1
[^_]	/dev/sdbf
@t-	/dev/sdbe9
[^_]	/dev/sdbe8
[^_]	/dev/sdbe7
AELD	/dev/sdbe6
;AELD	/dev/sdbe5
[^_]	/dev/sdbe4
[^_]	/dev/sdbe3
@t5=	/dev/sdbe2
[^_]	/dev/sdbe15
[^_]	/dev/sdbe14
[^_]	/dev/sdbe13
[^_]	/dev/sdbe12
[^_]	/dev/sdbe11
[^_]	/dev/sdbe10
RSVP	/dev/sdbe1
[^_]	/dev/sdbe
[^_]	/dev/sdbd9
SPWQ	/dev/sdbd8
[^_]	/dev/sdbd7
[^_]	/dev/sdbd6
[^_]	/dev/sdbd5
Gu~1	/dev/sdbd4
GuL1	/dev/sdbd3
[^_]	/dev/sdbd2
[^_]	/dev/sdbd15
\[^_]	/dev/sdbd14
CX9C	/dev/sdbd13
\[^_]	/dev/sdbd12
CX9C	/dev/sdbd11
<[^_]	/dev/sdbd10
[^_]	/dev/sdbd1
gfff	/dev/sdbd
[^_]	/dev/sdbc9
t/Qj	/dev/sdbc8
[^_]	/dev/sdbc7
WVSQ	/dev/sdbc6
0< v	/dev/sdbc5
Z[^_]	/dev/sdbc4
C< w+	/dev/sdbc3
0< v	/dev/sdbc2
[^_]	/dev/sdbc15
Rj@WS	/dev/sdbc14
< w1	/dev/sdbc13
0< v	/dev/sdbc12
tHRV	/dev/sdbc11
[^_]	/dev/sdbc10
[^_]	/dev/sdbc1
[^_]	/dev/sdbc

[^_]	/dev/sdbb9
Ph|b	/dev/sdbb8
[^_]	/dev/sdbb7
|,;u	/dev/sdbb6
t(;u	/dev/sdbb5
t`Qh`b	/dev/sdbb4
tLRh`b	/dev/sdbb3
[^_]	/dev/sdbb2
[^_]	/dev/sdbb15
Qh`b	/dev/sdbb14
Rh`b	/dev/sdbb13
[^_]	/dev/sdbb12
[^_]	/dev/sdbb11
[^_]	/dev/sdbb10
t B<:u	/dev/sdbb1
AF<:te1	/dev/sdbb
t B<:u	/dev/sdba9
VQRP	/dev/sdba8
Vh`b	/dev/sdba7
SQRP	/dev/sdba6
[^_]	/dev/sdba5
[^_]	/dev/sdba4
[^_]	/dev/sdba3
[^_]	/dev/sdba2
SRVW	/dev/sdba15
[^_]	/dev/sdba14
[^_]	/dev/sdba13
[^_]	/dev/sdba12
@F;E	/dev/sdba11
[^_]	/dev/sdba10
@F;E	/dev/sdba1
@G;E	/dev/sdba
[^_]	/dev/sdb9
@G;E	/dev/sdb8
[^_]	/dev/sdb7
FB;u	/dev/sdb6
FB;u	/dev/sdb5
BG;U	/dev/sdb4
tS;}	/dev/sdb3
AC;M	/dev/sdb2
[^_]	/dev/sdb15
AC;M	/dev/sdb14
@F;E	/dev/sdb13
[^_]	/dev/sdb12
@F;E	/dev/sdb11
AC;M	/dev/sdb10
[^_]	/dev/sdb1
AC;M	/dev/sdb
@F;E	/dev/sdaz9
[^_]	/dev/sdaz8
@F;E	/dev/sdaz7
[^_]	/dev/sdaz6
[^_]	/dev/sdaz5
[^_]	/dev/sdaz4
[^_]	/dev/sdaz3
[^_]	/dev/sdaz2
[^_]	/dev/sdaz15

?/tt	/dev/sdaz14
[^_]	/dev/sdaz13
[^_]	/dev/sdaz12
[^_]	/dev/sdaz11
[^_]	/dev/sdaz10
WQRV	/dev/sdaz1
B</t	/dev/sdaz
[^_]	/dev/sday9
[^_]	/dev/sday8
[^_]	/dev/sday7
_Xhs	/dev/sday6
[^_]	/dev/sday5
u @P	/dev/sday4
u$@P	/dev/sday3
u,@P	/dev/sday2
u4@P	/dev/sday15
u8@P	/dev/sday14
[^_]	/dev/sday13
AC;]	/dev/sday12
_tQOt	/dev/sday11
[^_]	/dev/sday10
t=Ky	/dev/sday1
t=Ky	/dev/sday
t=Ky	/dev/sdax9
t=Ky	/dev/sdax8
t=Ky	/dev/sdax7
t=Ky	/dev/sdax6
t=Ky	/dev/sdax5
0< w	/dev/sdax4
0< v	/dev/sdax3
9=u>A	/dev/sdax2
< w1j	/dev/sdax15
[^_]	/dev/sdax14
[^_]	/dev/sdax13
[^_]	/dev/sdax12
[^_]	/dev/sdax11
[^_]	/dev/sdax10
[^_]	/dev/sdax1
[^_]	/dev/sdax
t,RVWP	/dev/sdaw9
[^_]	/dev/sdaw8
tY9u	/dev/sdaw7
}.;]	/dev/sdaw6
L[^_]	/dev/sdaw5
< v61	/dev/sdaw4
ug;]	/dev/sdaw3
s=;]	/dev/sdaw2
< w[	/dev/sdaw15
[^_]	/dev/sdaw14
[^_]	/dev/sdaw13
tkWQ	/dev/sdaw12
WQj0	/dev/sdaw11
WQj0	/dev/sdaw10
F(Pj	/dev/sdaw1
;$t,	/dev/sdaw
9$t.	/dev/sdav9
[^_]	/dev/sdav8

[^_]	/dev/sdav7
SWh|	/dev/sdav6
[^_]	/dev/sdav5
[^_]	/dev/sdav4
QWVS	/dev/sdav3
[^_]	/dev/sdav2
t SVj	/dev/sdav15
QVj	/dev/sdav14
t*QVj	/dev/sdav13
RVj	/dev/sdav12
t);E	/dev/sdav11
gfff	/dev/sdav10
@PSR	/dev/sdav1
RWVS	/dev/sdav
PWVS	/dev/sdau9
v A)	/dev/sdau8
[^_]	/dev/sdau7
QWVS	/dev/sdau6
[^_]	/dev/sdau5
t PVj	/dev/sdau4
WVj	/dev/sdau3
t&QVj	/dev/sdau2
RVj	/dev/sdau15
WVj0	/dev/sdau14
SVj0	/dev/sdau13
RWVS	/dev/sdau12
QVh|	/dev/sdau11
t";5	/dev/sdau10
VSh|	/dev/sdau1
C,+C$)	/dev/sdau
[^_]	/dev/sdat9
[^_]	/dev/sdat8
QVWP	/dev/sdat7
[^_]	/dev/sdat6
[^_]	/dev/sdat5
QSh|	/dev/sdat4
[^_]	/dev/sdat3
tQRS	/dev/sdat2
[^_]	/dev/sdat15
~pRSV	/dev/sdat14
[^_]	/dev/sdat13
[^_]	/dev/sdat12
x~QS	/dev/sdat11
[^_]	/dev/sdat10
[^_]	/dev/sdat1
[^_]	/dev/sdat
[^_]	/dev/sdas9
~ERS	/dev/sdas8
[^_]	/dev/sdas7
tANt:	/dev/sdas6
[^_]	/dev/sdas5
[^_]	/dev/sdas4
~!Q+B	/dev/sdas3
[^_]	/dev/sdas2
[^_]	/dev/sdas15
@t5	/dev/sdas14
G +G	/dev/sdas13

G +G	/dev/sdas12
t-;]	/dev/sdas11
vwWSQ	/dev/sdas10
[^_]	/dev/sdas1
[^_]	/dev/sdas
[^_]	/dev/sdar9
CTPV	/dev/sdar8
[^_]	/dev/sdar7
CTPV	/dev/sdar6
[^_]	/dev/sdar5
C(PV	/dev/sdar4
[^_]	/dev/sdar3
PSVj	/dev/sdar2
[^_]	/dev/sdar15
[^_]	/dev/sdar14
t0@t	/dev/sdar13
GuP1	/dev/sdar12
[^_]	/dev/sdar11
t(Nu	/dev/sdar10
[^_]	/dev/sdar1
Gu[1	/dev/sdar
Gu#1	/dev/sdaq9
t0@Nt	/dev/sdaq8
t(@Nt	/dev/sdaq7
[^_]	/dev/sdaq6
[^_]	/dev/sdaq5
[^_]	/dev/sdaq4
[^_]	/dev/sdaq3
[^_]	/dev/sdaq2
RSj	/dev/sdaq15
QSj0	/dev/sdaq14
0t'QSj	/dev/sdaq13
PSj0	/dev/sdaq12
RSj	/dev/sdaq11
QSj0	/dev/sdaq10
RSj	/dev/sdaq1
QSj0	/dev/sdaq
RSj	/dev/sdap9
QSj0	/dev/sdap8
QSj	/dev/sdap7
PSj0	/dev/sdap6
RSj	/dev/sdap5
QSj0	/dev/sdap4
0tsQSj	/dev/sdap3
PSj0	/dev/sdap2
RSj	/dev/sdap15
QSj0	/dev/sdap14
RSj	/dev/sdap13
QSj0	/dev/sdap12
RSj	/dev/sdap11
QSj0	/dev/sdap10
<GtZ<gt.	/dev/sdap1
RSj	/dev/sdap
QSj0	/dev/sdao9
0t'RSj	/dev/sdao8
QSj0	/dev/sdao7
gfff	/dev/sdao6

0t+PSj	/dev/sdao5
RSj0	/dev/sdao4
0tsRVj	/dev/sdao3
QVj0	/dev/sdao2
0t(RVj	/dev/sdao15
QVj0	/dev/sdao14
[^_]	/dev/sdao13
[^_]	/dev/sdao12
Wj@j	/dev/sdao11
VQSP	/dev/sdao10
</t$	/dev/sdao1
;:t7G	/dev/sdao
;:toG	/dev/sdan9
t 9P	/dev/sdan8
$[^_]	/dev/sdan7
gfff	/dev/sdan6
tCVS	/dev/sdan5
C$+E	/dev/sdan4
[^_]	/dev/sdan3
[^_]	/dev/sdan2
WQRV	/dev/sdan15
[^_]	/dev/sdan14
tnF;5	/dev/sdan13
[^_]	/dev/sdan12
[^_]	/dev/sdan11
\[^_]	/dev/sdan10
CX9C	/dev/sdan1
[^_]	/dev/sdan
WVS1	/dev/sdam9
[^_]	/dev/sdam8
[^_]	/dev/sdam7
[^_]	/dev/sdam6
[^_]	/dev/sdam5
[^_]	/dev/sdam4
tLPj	/dev/sdam3
[^_]	/dev/sdam2
tJPVj	/dev/sdam15
[^_]	/dev/sdam14
[^_]	/dev/sdam13
Qj h_	/dev/sdam12
_Xhs	/dev/sdam11
Qj hh	/dev/sdam10
Wj hr	/dev/sdam1
ZYhs	/dev/sdam
[^_]	/dev/sdal9
0j$P	/dev/sdal8
[^_]	/dev/sdal7
t8<:u4	/dev/sdal6
<$t0	/dev/sdal5
[^_]	/dev/sdal4
t8<:u4	/dev/sdal3
RSVP	/dev/sdal2
[^_]	/dev/sdal15
t0QV	/dev/sdal14
[^_]	/dev/sdal13
[^_]	/dev/sdal12
F<:t	/dev/sdal11

VQSP	/dev/sdal10
[^_]	/dev/sdal1
[^_]	/dev/sdal
@bQs	/dev/sdak9
[^_]	/dev/sdak8
[^_]	/dev/sdak7
v)WRj	/dev/sdak6
[^_]	/dev/sdak5
[^_]	/dev/sdak4
WVSQ	/dev/sdak3
0< v	/dev/sdak2
Z[^_]	/dev/sdak15
C< w+	/dev/sdak14
0< v	/dev/sdak13
[^_]	/dev/sdak12
C09U	/dev/sdak11
9T00w	/dev/sdak10
tcB9	/dev/sdak1
[^_]	/dev/sdak
w%;]	/dev/sdaj9
[^_]	/dev/sdaj8
[^_]	/dev/sdaj7
[^_]	/dev/sdaj6
[^_]	/dev/sdaj5
t ;}	/dev/sdaj4
w!;]	/dev/sdaj3
[^_]	/dev/sdaj2
t ;}	/dev/sdaj15
w%;u	/dev/sdaj14
[^_]	/dev/sdaj13
[^_]	/dev/sdaj12
t'WS	/dev/sdaj11
QPSV	/dev/sdaj10
[^_]	/dev/sdaj1
[^_]	/dev/sdaj
st C	/dev/sdai9
[^_]	/dev/sdai8
<0/t	/dev/sdai7
t$PS	/dev/sdai6
tj/P	/dev/sdai5
VQSP	/dev/sdai4
@j/P	/dev/sdai3
[^_]	/dev/sdai2
VRSP	/dev/sdai15
WRSP	/dev/sdai14
4$tk	/dev/sdai13
[^_]	/dev/sdai12
P ;U	/dev/sdai11
[^_]	/dev/sdai10
[^_]	/dev/sdai1
Ph4n	/dev/sdai
[^_]	/dev/sdah9
Wh4n	/dev/sdah8
Sj:P	/dev/sdah7
[^_]	/dev/sdah6
tQ9u	/dev/sdah5
=S@P	/dev/sdah4

=S@P	/dev/sdah3
[^_]	/dev/sdah2
SQRP	/dev/sdah15
/GBH~	/dev/sdah14
/GBH~	/dev/sdah13
[^_]	/dev/sdah12
[^_]	/dev/sdah11
/SYS	/dev/sdah10
[^_]	/dev/sdah1
|M9u	/dev/sdah
<[^_]	/dev/sdag9
[^_]	/dev/sdag8
[^_]	/dev/sdag7
w_;M	/dev/sdag6
[^_]	/dev/sdag5
wA;U	/dev/sdag4
Jt.P	/dev/sdag3
X[^_	/dev/sdag2
Bt(P	/dev/sdag15
X[^_	/dev/sdag14
Jt}1	/dev/sdag13
[^_]	/dev/sdag12
WVUS	/dev/sdag11
[]^_	/dev/sdag10
JtTG	/dev/sdag1
[^_]	/dev/sdag
JtPG	/dev/sdaf9
[^_]	/dev/sdaf8
SWVV	/dev/sdaf7
RSWV	/dev/sdaf6
RSWV	/dev/sdaf5
Jte1	/dev/sdaf4
[^_]	/dev/sdaf3
Jtl1	/dev/sdaf2
Jtd1	/dev/sdaf15
[^_]	/dev/sdaf14
Jtd1	/dev/sdaf13
[^_]	/dev/sdaf12
R Iu	/dev/sdaf11
WVUS	/dev/sdaf10
[]^_	/dev/sdaf1
L[^_]	/dev/sdaf
\[^_]	/dev/sdae9
[^_]	/dev/sdae8
[^_]	/dev/sdae7
tC;E	/dev/sdae6
[^_]	/dev/sdae5
[^_]	/dev/sdae4
[^_]	/dev/sdae3
[^_]	/dev/sdae2
[^_]	/dev/sdae15
t&PS	/dev/sdae14
RSVP	/dev/sdae13
[^_]	/dev/sdae12
[^_]	/dev/sdae11
[^_]	/dev/sdae10
H%T=	/dev/sdae1

<\t~<\tn	/dev/sdae
[^_]	/dev/sdad9
[^_]	/dev/sdad8
< tZ< tB<\t2	/dev/sdad7
[^_]	/dev/sdad6
/j hs	/dev/sdad5
[^_]	/dev/sdad4
@t0R	/dev/sdad3
XZh@	/dev/sdad2
[^_]	/dev/sdad15
[^_]	/dev/sdad14
[^_]	/dev/sdad13
PSRW	/dev/sdad12
QPRW	/dev/sdad11
[^_]	/dev/sdad10
R Iu	/dev/sdad1
WVUS	/dev/sdad
[]^_	/dev/sdac9
[^_]	/dev/sdac8
<*tm<'ti<Ite	/dev/sdac7
<*t><*	/dev/sdac6
*<'t	/dev/sdac5
PVh|	/dev/sdac4
[^_]	/dev/sdac3
QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ	/dev/sdac2
QQQQQQQQQQQQQQQQQQQj	/dev/sdac15
t4Qj	/dev/sdac14
t>Qj	/dev/sdac13
x Rj	/dev/sdac12
[^_]	/dev/sdac11
RPh`	/dev/sdac10
VQSW	/dev/sdac1
RPh`	/dev/sdac
QSVW	/dev/sdab9
@bQs	/dev/sdab8
[^_]	/dev/sdab7
[^_]	/dev/sdab6
[^_]	/dev/sdab5
[^_]	/dev/sdab4
[^_]	/dev/sdab3
tmPh	/dev/sdab2
tiPh	/dev/sdab15
[^_]	/dev/sdab14
[^_]	/dev/sdab13
}+;M	/dev/sdab12
[^_]	/dev/sdab11
t(;U	/dev/sdab10
< v;1	/dev/sdab1
uc;u	/dev/sdab
s=;u	/dev/sdaa9
}+;M	/dev/sdaa8
[^_]	/dev/sdaa7
t(;U	/dev/sdaa6
< v;1	/dev/sdaa5
uc;u	/dev/sdaa4
s=;u	/dev/sdaa3
[^_]	/dev/sdaa2

t%Pj	/dev/sdaa15
0< v	/dev/sdaa14
0< w	/dev/sdaa13
< v$	/dev/sdaa12
VUUU	/dev/sdaa11
~3SVRR	/dev/sdaa10
t0Wj	/dev/sdaa1
0< v	/dev/sdaa
t N;	/dev/sda9
[^_]	/dev/sda8
t%Pj	/dev/sda7
0< v	/dev/sda6
0< w	/dev/sda5
< v$	/dev/sda4
VUUU	/dev/sda3
t0Wj	/dev/sda2
0< v	/dev/sda15
0< v	/dev/sda14
t N;	/dev/sda13
[^_]	/dev/sda12
t%Pj	/dev/sda11
0< v	/dev/sda10
0< w	/dev/sda1
< v$	/dev/sda
VUUU	/dev/scd7
t0Wj	/dev/scd6
0< v	/dev/scd5
0< v	/dev/scd4
t N;	/dev/scd3
[^_]	/dev/scd2
(^_]	/dev/scd1
u^9u	/dev/scd0
8^_]	/dev/sbpcd9
mft_getopt	/dev/sbpcd8
no index	/dev/sbpcd7
invalid index %d	/dev/sbpcd6
argv[%d] is NULL	/dev/sbpcd5
argv[%d] (%s) is not an option	/dev/sbpcd4
examining a filename or url!	/dev/sbpcd3
%s is a well-formed argument	/dev/sbpcd2
checking against %s	/dev/sbpcd15
flag-	/dev/sbpcd14
flagized option invokation	/dev/sbpcd13
examining an enum!	/dev/sbpcd12
matched against an enum val	/dev/sbpcd11
examining a venum!	/dev/sbpcd10
matched against an venum val	/dev/sbpcd1
arg matches against %s	/dev/sbpcd0
process_match	/dev/ram9
true	/dev/ram8
matches against %s	/dev/ram7
invalid value for enum	/dev/ram6
mft_log_init	/dev/ram5
nbd-server	/dev/ram4
MFT_LOG_THRESH	/dev/ram3
none	/dev/ram2
fatal	/dev/ram19

error	/dev/ram18
info	/dev/ram17
branch	/dev/ram16
progress	/dev/ram15
entryexit	/dev/ram14
mft_log_shutdown	/dev/ram13
unspecified	/dev/ram12
enter	/dev/ram11
exit	/dev/ram10
%s: %s	/dev/ram1
violet	/dev/ram0
blue	/dev/pf3
green	/dev/pf2
yellow	/dev/pf1
orange	/dev/pf0
white	/dev/pdd9
%s: %s	/dev/pdd8
<table bgcolor=%s><tr><td>%s:	/dev/pdd7
%s</td></tr></table> 	/dev/pdd6
<table	/dev/pdd5
bgcolor=%s><tr><td>%s</td></tr></table>	/dev/pdd4
 	/dev/pdd3
<table	/dev/pdd2
bgcolor=%s><tr><td></td></tr></table><b	/dev/pdd15
r>	/dev/pdd14
Brazil	/dev/pdd13
.TH %s "%d" "%s" "%s" "%s"	/dev/pdd12
.SH NAME	/dev/pdd11
%s \- %s	/dev/pdd10
.SH SYNOPSIS	/dev/pdd1
.B %s	/dev/pdd
[\fIOPTION\fR]...	/dev/pdc9
.SH DESCRIPTION	/dev/pdc8
\fB\-\-%s\fR %s	/dev/pdc7
\fB\-\-%s\fR \fIARG\fR %s	/dev/pdc6
\fB\-\-%s\fR \fIINT\fR %s	/dev/pdc5
\fB\-\-%s\fR \fIFILENAME\fR %s	/dev/pdc4
\fB\-\-%s\fR \fIVALUE\fR %s	/dev/pdc3
\fIVALUE\fR can be one of:	/dev/pdc2
\fB%s\fR	/dev/pdc15
| \fB%s\fR	/dev/pdc14
\fBSHORTHAND INVOKATION:\fR	/dev/pdc13
Any of the valid values for \fB--%s\fR	/dev/pdc12
can be supplied directly as options.	/dev/pdc11
For instance, \fB--%s\fR can be used in	/dev/pdc10
place of \fB--%s=%s\fR.	/dev/pdc1
\fB%s\fR %s	/dev/pdc
--%s %s	/dev/pdb9
.SH REPORTING BUGS	/dev/pdb8
Report bugs to %s.	/dev/pdb7
Usage: %s [OPTION]...	/dev/pdb6
[<%s-filename>]	/dev/pdb5
--%s %s	/dev/pdb4
--%s <arg> %s	/dev/pdb3
--%s <int> %s	/dev/pdb2
--%s <filename> %s	/dev/pdb15
--%s <	/dev/pdb14

| %s	/dev/pdb13
> %s	/dev/pdb12
--%s VALUE	/dev/pdb11
where VALUE is one of:	/dev/pdb10
%s %s	/dev/pdb1
<tt>%s</tt> invocation	/dev/pdb
<tt>%s [<OPTIONS>]	/dev/pda9
[<%s-filename>]	/dev/pda8
</tt>	/dev/pda7
Where <bf>OPTIONS</bf> may include any	/dev/pda6
of:	/dev/pda5
<descrip>	/dev/pda4
<tag>--%s</tag> %s	/dev/pda3
<tag>--%s <arg></tag> %s	/dev/pda2
<tag>--%s <int></tag> %s	/dev/pda15
<tag>--%s <filename></tag> %s	/dev/pda14
<tag>--%s <	/dev/pda13
></tag> %s	/dev/pda12
<tag>--%s VALUE</tag>	/dev/pda11
<tag>%s</tag> %s	/dev/pda10
</descrip>	/dev/pda1
<tag>--%s</tag> %s	/dev/pda
%s:%s %s	/dev/pcd3
operate on ...	/dev/pcd2
target	/dev/pcd1
entryexit	/dev/pcd0
progress	/dev/optcd
branch	/dev/nb9
info	/dev/nb8
error	/dev/nb7
fatal	/dev/nb6
none	/dev/nb5
logging threshold ...	/dev/nb4
log-thresh	/dev/nb31
be verbose	/dev/nb30
verbose	/dev/nb3
name	/dev/nb29
useless bogus option	/dev/nb28
label	/dev/nb27
write output to ...	/dev/nb26
outfile	/dev/nb25
test for fragmentation (returns 0 if	/dev/nb24
file is fragmented)	/dev/nb23
checkfrag	/dev/nb22
display fragmentation information for	/dev/nb21
the file	/dev/nb20
frag	/dev/nb2
wipe the file from the raw device	/dev/nb19
print number of bytes available	/dev/nb18
test (returns 0 if exist)	/dev/nb17
wipe	/dev/nb16
place data	/dev/nb15
display data	/dev/nb14
extract a copy from the raw device	/dev/nb13
list sector numbers	/dev/nb12
operation to perform on files	/dev/nb11
mode	/dev/nb10

generate SGML invocation info	/dev/nb1
sgml	/dev/nb0
generate man page and exit	/dev/md31
display options and exit	/dev/md30
help	/dev/md29
display version and exit	/dev/md28
version	/dev/md27
autogenerate document ...	/dev/md26
1.0.20 (07/15/03)	/dev/md25
newt	/dev/md24
use block-list knowledge to perform	/dev/md23
special operations on files	/dev/md22
prog	/dev/md21
main	/dev/md20
off_t too small!	/dev/md19
07/15/03	/dev/md18
invalid option: %s	/dev/md17
try '--help' for help.	/dev/md16
how did we get here?	/dev/mcdx
no filename. try '--help' for help.	/dev/mcd
target filename: %s	/dev/loop9
Unable to stat file: %s	/dev/loop8
%s is not a regular file.	/dev/loop7
%s has multiple links.	/dev/loop6
Unable to open file: %s	/dev/loop5
Unable to determine blocksize	/dev/loop4
target file block size: %d	/dev/loop3
unable to raw open %s	/dev/loop2
Unable to determine count	/dev/loop15
Unable to allocate buffer	/dev/loop14
%s has holes in excess of %ld bytes...	/dev/loop13
error mapping block %d (%s)	/dev/loop12
nul block while mapping block %d.	/dev/loop11
seek failure	/dev/loop10
read error	/dev/loop1
write error	/dev/loop0
%s fragmented between %d and %d	/dev/jsfd
%d %s	/dev/initrd
getting from block %d	/dev/hitcd
file size was: %ld	/dev/hdt9
slack size: %d	/dev/hdt8
block size: %d	/dev/hdt7
seek error	/dev/hdt6
# File: %s Location: %Ld size: %d	/dev/hdt5
stuffing block %d	/dev/hdt4
%s has slack	/dev/hdt32
%s does not have slack	/dev/hdt31
%s has fragmentation	/dev/hdt30
%s does not have fragmentation	/dev/hdt3
bmap_get_slack_block	/dev/hdt29
NULL value for slack_block	/dev/hdt28
Unable to stat fd	/dev/hdt27
Unable to determine blocksize	/dev/hdt26
error getting block count	/dev/hdt25
fd has no blocks	/dev/hdt24
mapping block %lu	/dev/hdt23
error mapping block %d. ioctl failed	/dev/hdt22

with %s	/dev/hdt21
error mapping block %d. block returned	/dev/hdt20
0	/dev/hdt2
bmap_get_block_count	/dev/hdt19
unable to stat fd	/dev/hdt18
unable to determine filesystem	/dev/hdt17
blocksize	/dev/hdt16
filesystem reports 0 blocksize	/dev/hdt15
computed block count: %d	/dev/hdt14
stat reports %d blocks: %d	/dev/hdt13
bmap_get_block_size	/dev/hdt12
bmap_map_block	/dev/hdt11
nul block while mapping block %d.	/dev/hdt10
bmap_raw_open	/dev/hdt1
NULL filename supplied	/dev/hdt
Unable to stat file: %s	/dev/hds9
%s is not a regular file.	/dev/hds8
unable to determine raw device of %s	/dev/hds7
unable to stat raw device %s	/dev/hds6
device mismatch 0x%x != 0x%x	/dev/hds5
unable to open raw device %s	/dev/hds4
raw fd is %d	/dev/hds32
bmap_raw_close	/dev/hds31
/.../image	/dev/hds30
bogowipe	/dev/hds3
write error	/dev/hds29
/dev/xdb9	/dev/hds28
/dev/xdb8	/dev/hds27
/dev/xdb7	/dev/hds26
/dev/xdb63	/dev/hds25
/dev/xdb62	/dev/hds24
/dev/xdb61	/dev/hds23
/dev/xdb60	/dev/hds22
/dev/xdb6	/dev/hds21
/dev/xdb59	/dev/hds20
/dev/xdb58	/dev/hds2
/dev/xdb57	/dev/hds19
/dev/xdb56	/dev/hds18
/dev/xdb55	/dev/hds17
/dev/xdb54	/dev/hds16
/dev/xdb53	/dev/hds15
/dev/xdb52	/dev/hds14
/dev/xdb51	/dev/hds13
/dev/xdb50	/dev/hds12
/dev/xdb49	/dev/hds11
/dev/xdb48	/dev/hds10
/dev/xdb47	/dev/hds1
/dev/xdb46	/dev/hds
/dev/xdb45	/dev/hdr9
/dev/xdb44	/dev/hdr8
/dev/xdb43	/dev/hdr7
/dev/xdb42	/dev/hdr6
/dev/xdb41	/dev/hdr5
/dev/xdb40	/dev/hdr4
/dev/xdb5	/dev/hdr32
/dev/xdb4	/dev/hdr31
/dev/xdb39	/dev/hdr30

/dev/xdb38	/dev/hdr3
/dev/xdb37	/dev/hdr29
/dev/xdb36	/dev/hdr28
/dev/xdb35	/dev/hdr27
/dev/xdb34	/dev/hdr26
/dev/xdb33	/dev/hdr25
/dev/xdb32	/dev/hdr24
/dev/xdb31	/dev/hdr23
/dev/xdb30	/dev/hdr22
/dev/xdb3	/dev/hdr21
/dev/xdb29	/dev/hdr20
/dev/xdb28	/dev/hdr2
/dev/xdb27	/dev/hdr19
/dev/xdb26	/dev/hdr18
/dev/xdb25	/dev/hdr17
/dev/xdb24	/dev/hdr16
/dev/xdb23	/dev/hdr15
/dev/xdb22	/dev/hdr14
/dev/xdb21	/dev/hdr13
/dev/xdb20	/dev/hdr12
/dev/xdb2	/dev/hdr11
/dev/xdb19	/dev/hdr10
/dev/xdb18	/dev/hdr1
/dev/xdb17	/dev/hdr
/dev/xdb16	/dev/hdq9
/dev/xdb15	/dev/hdq8
/dev/xdb14	/dev/hdq7
/dev/xdb13	/dev/hdq6
/dev/xdb12	/dev/hdq5
/dev/xdb11	/dev/hdq4
/dev/xdb10	/dev/hdq32
/dev/xdb1	/dev/hdq31
/dev/xdb	/dev/hdq30
/dev/xda9	/dev/hdq3
/dev/xda8	/dev/hdq29
/dev/xda7	/dev/hdq28
/dev/xda63	/dev/hdq27
/dev/xda62	/dev/hdq26
/dev/xda61	/dev/hdq25
/dev/xda60	/dev/hdq24
/dev/xda6	/dev/hdq23
/dev/xda59	/dev/hdq22
/dev/xda58	/dev/hdq21
/dev/xda57	/dev/hdq20
/dev/xda56	/dev/hdq2
/dev/xda55	/dev/hdq19
/dev/xda54	/dev/hdq18
/dev/xda53	/dev/hdq17
/dev/xda52	/dev/hdq16
/dev/xda51	/dev/hdq15
/dev/xda50	/dev/hdq14
/dev/xda5	/dev/hdq13
/dev/xda49	/dev/hdq12
/dev/xda48	/dev/hdq11
/dev/xda47	/dev/hdq10
/dev/xda46	/dev/hdq1
/dev/xda45	/dev/hdq

/dev/xda44	/dev/hdp9
/dev/xda43	/dev/hdp8
/dev/xda42	/dev/hdp7
/dev/xda41	/dev/hdp6
/dev/xda40	/dev/hdp5
/dev/xda4	/dev/hdp4
/dev/xda39	/dev/hdp32
/dev/xda38	/dev/hdp31
/dev/xda37	/dev/hdp30
/dev/xda36	/dev/hdp3
/dev/xda35	/dev/hdp29
/dev/xda34	/dev/hdp28
/dev/xda33	/dev/hdp27
/dev/xda32	/dev/hdp26
/dev/xda31	/dev/hdp25
/dev/xda30	/dev/hdp24
/dev/xda3	/dev/hdp23
/dev/xda29	/dev/hdp22
/dev/xda28	/dev/hdp21
/dev/xda27	/dev/hdp20
/dev/xda26	/dev/hdp2
/dev/xda25	/dev/hdp19
/dev/xda24	/dev/hdp18
/dev/xda23	/dev/hdp17
/dev/xda22	/dev/hdp16
/dev/xda21	/dev/hdp15
/dev/xda20	/dev/hdp14
/dev/xda2	/dev/hdp13
/dev/xda19	/dev/hdp12
/dev/xda18	/dev/hdp11
/dev/xda17	/dev/hdp10
/dev/xda16	/dev/hdp1
/dev/xda15	/dev/hdp
/dev/xda14	/dev/hdo9
/dev/xda13	/dev/hdo8
/dev/xda12	/dev/hdo7
/dev/xda11	/dev/hdo6
/dev/xda10	/dev/hdo5
/dev/xda1	/dev/hdo4
/dev/xda	/dev/hdo32
/dev/sonycd	/dev/hdo31
/dev/sjcd	/dev/hdo30
/dev/sdz9	/dev/hdo3
/dev/sdz8	/dev/hdo29
/dev/sdz7	/dev/hdo28
/dev/sdz6	/dev/hdo27
/dev/sdz5	/dev/hdo26
/dev/sdz4	/dev/hdo25
/dev/sdz3	/dev/hdo24
/dev/sdz2	/dev/hdo23
/dev/sdz15	/dev/hdo22
/dev/sdz14	/dev/hdo21
/dev/sdz13	/dev/hdo20
/dev/sdz12	/dev/hdo2
/dev/sdz11	/dev/hdo19
/dev/sdz10	/dev/hdo18
/dev/sdz1	/dev/hdo17

/dev/sdz	/dev/hdo16
/dev/sdy9	/dev/hdo15
/dev/sdy8	/dev/hdo14
/dev/sdy7	/dev/hdo13
/dev/sdy6	/dev/hdo12
/dev/sdy5	/dev/hdo11
/dev/sdy4	/dev/hdo10
/dev/sdy3	/dev/hdo1
/dev/sdy2	/dev/hdo
/dev/sdy15	/dev/hdn9
/dev/sdy14	/dev/hdn8
/dev/sdy13	/dev/hdn7
/dev/sdy12	/dev/hdn6
/dev/sdy11	/dev/hdn5
/dev/sdy10	/dev/hdn4
/dev/sdy1	/dev/hdn32
/dev/sdy	/dev/hdn31
/dev/sdx9	/dev/hdn30
/dev/sdx8	/dev/hdn3
/dev/sdx7	/dev/hdn29
/dev/sdx6	/dev/hdn28
/dev/sdx5	/dev/hdn27
/dev/sdx4	/dev/hdn26
/dev/sdx3	/dev/hdn25
/dev/sdx2	/dev/hdn24
/dev/sdx15	/dev/hdn23
/dev/sdx14	/dev/hdn22
/dev/sdx13	/dev/hdn21
/dev/sdx12	/dev/hdn20
/dev/sdx11	/dev/hdn2
/dev/sdx10	/dev/hdn19
/dev/sdx1	/dev/hdn18
/dev/sdx	/dev/hdn17
/dev/sdw9	/dev/hdn16
/dev/sdw8	/dev/hdn15
/dev/sdw7	/dev/hdn14
/dev/sdw6	/dev/hdn13
/dev/sdw5	/dev/hdn12
/dev/sdw4	/dev/hdn11
/dev/sdw3	/dev/hdn10
/dev/sdw2	/dev/hdn1
/dev/sdw15	/dev/hdn
/dev/sdw14	/dev/hdm9
/dev/sdw13	/dev/hdm8
/dev/sdw12	/dev/hdm7
/dev/sdw11	/dev/hdm6
/dev/sdw10	/dev/hdm5
/dev/sdw1	/dev/hdm4
/dev/sdw	/dev/hdm32
/dev/sdv9	/dev/hdm31
/dev/sdv8	/dev/hdm30
/dev/sdv7	/dev/hdm3
/dev/sdv6	/dev/hdm29
/dev/sdv5	/dev/hdm28
/dev/sdv4	/dev/hdm27
/dev/sdv3	/dev/hdm26
/dev/sdv2	/dev/hdm25

/dev/sdv15	/dev/hdm24
/dev/sdv14	/dev/hdm23
/dev/sdv13	/dev/hdm22
/dev/sdv12	/dev/hdm21
/dev/sdv11	/dev/hdm20
/dev/sdv10	/dev/hdm2
/dev/sdv1	/dev/hdm19
/dev/sdv	/dev/hdm18
/dev/sdu9	/dev/hdm17
/dev/sdu8	/dev/hdm16
/dev/sdu7	/dev/hdm15
/dev/sdu6	/dev/hdm14
/dev/sdu5	/dev/hdm13
/dev/sdu4	/dev/hdm12
/dev/sdu3	/dev/hdm11
/dev/sdu2	/dev/hdm10
/dev/sdu15	/dev/hdm1
/dev/sdu14	/dev/hdm
/dev/sdu13	/dev/hdl9
/dev/sdu12	/dev/hdl8
/dev/sdu11	/dev/hdl7
/dev/sdu10	/dev/hdl6
/dev/sdu1	/dev/hdl5
/dev/sdu	/dev/hdl4
/dev/sdt9	/dev/hdl32
/dev/sdt8	/dev/hdl31
/dev/sdt7	/dev/hdl30
/dev/sdt6	/dev/hdl3
/dev/sdt5	/dev/hdl29
/dev/sdt4	/dev/hdl28
/dev/sdt3	/dev/hdl27
/dev/sdt2	/dev/hdl26
/dev/sdt15	/dev/hdl25
/dev/sdt14	/dev/hdl24
/dev/sdt13	/dev/hdl23
/dev/sdt12	/dev/hdl22
/dev/sdt11	/dev/hdl21
/dev/sdt10	/dev/hdl20
/dev/sdt1	/dev/hdl2
/dev/sdt	/dev/hdl19
/dev/sds9	/dev/hdl18
/dev/sds8	/dev/hdl17
/dev/sds7	/dev/hdl16
/dev/sds6	/dev/hdl15
/dev/sds5	/dev/hdl14
/dev/sds4	/dev/hdl13
/dev/sds3	/dev/hdl12
/dev/sds2	/dev/hdl11
/dev/sds15	/dev/hdl10
/dev/sds14	/dev/hdl1
/dev/sds13	/dev/hdl
/dev/sds12	/dev/hdk9
/dev/sds11	/dev/hdk8
/dev/sds10	/dev/hdk7
/dev/sds1	/dev/hdk6
/dev/sds	/dev/hdk5
/dev/sdr9	/dev/hdk4

/dev/sdr8	/dev/hdk32
/dev/sdr7	/dev/hdk31
/dev/sdr6	/dev/hdk30
/dev/sdr5	/dev/hdk3
/dev/sdr4	/dev/hdk29
/dev/sdr3	/dev/hdk28
/dev/sdr2	/dev/hdk27
/dev/sdr15	/dev/hdk26
/dev/sdr14	/dev/hdk25
/dev/sdr13	/dev/hdk24
/dev/sdr12	/dev/hdk23
/dev/sdr11	/dev/hdk22
/dev/sdr10	/dev/hdk21
/dev/sdr1	/dev/hdk20
/dev/sdr	/dev/hdk2
/dev/sdq9	/dev/hdk19
/dev/sdq8	/dev/hdk18
/dev/sdq7	/dev/hdk17
/dev/sdq6	/dev/hdk16
/dev/sdq5	/dev/hdk15
/dev/sdq4	/dev/hdk14
/dev/sdq3	/dev/hdk13
/dev/sdq2	/dev/hdk12
/dev/sdq15	/dev/hdk11
/dev/sdq14	/dev/hdk10
/dev/sdq13	/dev/hdk1
/dev/sdq12	/dev/hdk
/dev/sdq11	/dev/hdj9
/dev/sdq10	/dev/hdj8
/dev/sdq1	/dev/hdj7
/dev/sdq	/dev/hdj6
/dev/sdp9	/dev/hdj5
/dev/sdp8	/dev/hdj4
/dev/sdp7	/dev/hdj32
/dev/sdp6	/dev/hdj31
/dev/sdp5	/dev/hdj30
/dev/sdp4	/dev/hdj3
/dev/sdp3	/dev/hdj29
/dev/sdp2	/dev/hdj28
/dev/sdp15	/dev/hdj27
/dev/sdp14	/dev/hdj26
/dev/sdp13	/dev/hdj25
/dev/sdp12	/dev/hdj24
/dev/sdp11	/dev/hdj23
/dev/sdp10	/dev/hdj22
/dev/sdp1	/dev/hdj21
/dev/sdp	/dev/hdj20
/dev/sdo9	/dev/hdj2
/dev/sdo8	/dev/hdj19
/dev/sdo7	/dev/hdj18
/dev/sdo6	/dev/hdj17
/dev/sdo5	/dev/hdj16
/dev/sdo4	/dev/hdj15
/dev/sdo3	/dev/hdj14
/dev/sdo2	/dev/hdj13
/dev/sdo15	/dev/hdj12
/dev/sdo14	/dev/hdj11

/dev/sdo13	/dev/hdj10
/dev/sdo12	/dev/hdj1
/dev/sdo11	/dev/hdj
/dev/sdo10	/dev/hdi9
/dev/sdo1	/dev/hdi8
/dev/sdo	/dev/hdi7
/dev/sdn9	/dev/hdi6
/dev/sdn8	/dev/hdi5
/dev/sdn7	/dev/hdi4
/dev/sdn6	/dev/hdi32
/dev/sdn5	/dev/hdi31
/dev/sdn4	/dev/hdi30
/dev/sdn3	/dev/hdi3
/dev/sdn2	/dev/hdi29
/dev/sdn15	/dev/hdi28
/dev/sdn14	/dev/hdi27
/dev/sdn13	/dev/hdi26
/dev/sdn12	/dev/hdi25
/dev/sdn11	/dev/hdi24
/dev/sdn10	/dev/hdi23
/dev/sdn1	/dev/hdi22
/dev/sdn	/dev/hdi21
/dev/sdm9	/dev/hdi20
/dev/sdm8	/dev/hdi2
/dev/sdm7	/dev/hdi19
/dev/sdm6	/dev/hdi18
/dev/sdm5	/dev/hdi17
/dev/sdm4	/dev/hdi16
/dev/sdm3	/dev/hdi15
/dev/sdm2	/dev/hdi14
/dev/sdm15	/dev/hdi13
/dev/sdm14	/dev/hdi12
/dev/sdm13	/dev/hdi11
/dev/sdm12	/dev/hdi10
/dev/sdm11	/dev/hdi1
/dev/sdm10	/dev/hdi
/dev/sdm1	/dev/hdh9
/dev/sdm	/dev/hdh8
/dev/sdl9	/dev/hdh7
/dev/sdl8	/dev/hdh6
/dev/sdl7	/dev/hdh5
/dev/sdl6	/dev/hdh4
/dev/sdl5	/dev/hdh32
/dev/sdl4	/dev/hdh31
/dev/sdl3	/dev/hdh30
/dev/sdl2	/dev/hdh3
/dev/sdl15	/dev/hdh29
/dev/sdl14	/dev/hdh28
/dev/sdl13	/dev/hdh27
/dev/sdl12	/dev/hdh26
/dev/sdl11	/dev/hdh25
/dev/sdl10	/dev/hdh24
/dev/sdl1	/dev/hdh23
/dev/sdl	/dev/hdh22
/dev/sdk9	/dev/hdh21
/dev/sdk8	/dev/hdh20
/dev/sdk7	/dev/hdh2

/dev/sdk6	/dev/hdh19
/dev/sdk5	/dev/hdh18
/dev/sdk4	/dev/hdh17
/dev/sdk3	/dev/hdh16
/dev/sdk2	/dev/hdh15
/dev/sdk15	/dev/hdh14
/dev/sdk14	/dev/hdh13
/dev/sdk13	/dev/hdh12
/dev/sdk12	/dev/hdh11
/dev/sdk11	/dev/hdh10
/dev/sdk10	/dev/hdh1
/dev/sdk1	/dev/hdh
/dev/sdk	/dev/hdg9
/dev/sdj9	/dev/hdg8
/dev/sdj8	/dev/hdg7
/dev/sdj7	/dev/hdg6
/dev/sdj6	/dev/hdg5
/dev/sdj5	/dev/hdg4
/dev/sdj4	/dev/hdg32
/dev/sdj3	/dev/hdg31
/dev/sdj2	/dev/hdg30
/dev/sdj15	/dev/hdg3
/dev/sdj14	/dev/hdg29
/dev/sdj13	/dev/hdg28
/dev/sdj12	/dev/hdg27
/dev/sdj11	/dev/hdg26
/dev/sdj10	/dev/hdg25
/dev/sdj1	/dev/hdg24
/dev/sdj	/dev/hdg23
/dev/sdi9	/dev/hdg22
/dev/sdi8	/dev/hdg21
/dev/sdi7	/dev/hdg20
/dev/sdi6	/dev/hdg2
/dev/sdi5	/dev/hdg19
/dev/sdi4	/dev/hdg18
/dev/sdi3	/dev/hdg17
/dev/sdi2	/dev/hdg16
/dev/sdi15	/dev/hdg15
/dev/sdi14	/dev/hdg14
/dev/sdi13	/dev/hdg13
/dev/sdi12	/dev/hdg12
/dev/sdi11	/dev/hdg11
/dev/sdi10	/dev/hdg10
/dev/sdi1	/dev/hdg1
/dev/sdi	/dev/hdg
/dev/sdh9	/dev/hdf9
/dev/sdh8	/dev/hdf8
/dev/sdh7	/dev/hdf7
/dev/sdh6	/dev/hdf6
/dev/sdh5	/dev/hdf5
/dev/sdh4	/dev/hdf4
/dev/sdh3	/dev/hdf32
/dev/sdh2	/dev/hdf31
/dev/sdh15	/dev/hdf30
/dev/sdh14	/dev/hdf3
/dev/sdh13	/dev/hdf29
/dev/sdh12	/dev/hdf28

/dev/sdh11	/dev/hdf27
/dev/sdh10	/dev/hdf26
/dev/sdh1	/dev/hdf25
/dev/sdh	/dev/hdf24
/dev/sdg9	/dev/hdf23
/dev/sdg8	/dev/hdf22
/dev/sdg7	/dev/hdf21
/dev/sdg6	/dev/hdf20
/dev/sdg5	/dev/hdf2
/dev/sdg4	/dev/hdf19
/dev/sdg3	/dev/hdf18
/dev/sdg2	/dev/hdf17
/dev/sdg15	/dev/hdf16
/dev/sdg14	/dev/hdf15
/dev/sdg13	/dev/hdf14
/dev/sdg12	/dev/hdf13
/dev/sdg11	/dev/hdf12
/dev/sdg10	/dev/hdf11
/dev/sdg1	/dev/hdf10
/dev/sdg	/dev/hdf1
/dev/sdf9	/dev/hdf
/dev/sdf8	/dev/hde9
/dev/sdf7	/dev/hde8
/dev/sdf6	/dev/hde7
/dev/sdf5	/dev/hde6
/dev/sdf4	/dev/hde5
/dev/sdf3	/dev/hde4
/dev/sdf2	/dev/hde32
/dev/sdf15	/dev/hde31
/dev/sdf14	/dev/hde30
/dev/sdf13	/dev/hde3
/dev/sdf12	/dev/hde29
/dev/sdf11	/dev/hde28
/dev/sdf10	/dev/hde27
/dev/sdf1	/dev/hde26
/dev/sdf	/dev/hde25
/dev/sde9	/dev/hde24
/dev/sde8	/dev/hde23
/dev/sde7	/dev/hde22
/dev/sde6	/dev/hde21
/dev/sde5	/dev/hde20
/dev/sde4	/dev/hde2
/dev/sde3	/dev/hde19
/dev/sde2	/dev/hde18
/dev/sde15	/dev/hde17
/dev/sde14	/dev/hde16
/dev/sde13	/dev/hde15
/dev/sde12	/dev/hde14
/dev/sde11	/dev/hde13
/dev/sde10	/dev/hde12
/dev/sde1	/dev/hde11
/dev/sde	/dev/hde10
/dev/sddx9	/dev/hde1
/dev/sddx8	/dev/hde
/dev/sddx7	/dev/hdd9
/dev/sddx6	/dev/hdd8
/dev/sddx5	/dev/hdd7

/dev/sddx4	/dev/hdd6
/dev/sddx3	/dev/hdd5
/dev/sddx2	/dev/hdd4
/dev/sddx15	/dev/hdd32
/dev/sddx14	/dev/hdd31
/dev/sddx13	/dev/hdd30
/dev/sddx12	/dev/hdd3
/dev/sddx11	/dev/hdd29
/dev/sddx10	/dev/hdd28
/dev/sddx1	/dev/hdd27
/dev/sddx	/dev/hdd26
/dev/sddw9	/dev/hdd25
/dev/sddw8	/dev/hdd24
/dev/sddw7	/dev/hdd23
/dev/sddw6	/dev/hdd22
/dev/sddw5	/dev/hdd21
/dev/sddw4	/dev/hdd20
/dev/sddw3	/dev/hdd2
/dev/sddw2	/dev/hdd19
/dev/sddw15	/dev/hdd18
/dev/sddw14	/dev/hdd17
/dev/sddw13	/dev/hdd16
/dev/sddw12	/dev/hdd15
/dev/sddw11	/dev/hdd14
/dev/sddw10	/dev/hdd13
/dev/sddw1	/dev/hdd12
/dev/sddw	/dev/hdd11
/dev/sddv9	/dev/hdd10
/dev/sddv8	/dev/hdd1
/dev/sddv7	/dev/hdd
/dev/sddv6	/dev/hdc9
/dev/sddv5	/dev/hdc8
/dev/sddv4	/dev/hdc7
/dev/sddv3	/dev/hdc6
/dev/sddv2	/dev/hdc5
/dev/sddv15	/dev/hdc4
/dev/sddv14	/dev/hdc32
/dev/sddv13	/dev/hdc31
/dev/sddv12	/dev/hdc30
/dev/sddv11	/dev/hdc3
/dev/sddv10	/dev/hdc29
/dev/sddv1	/dev/hdc28
/dev/sddv	/dev/hdc27
/dev/sddu9	/dev/hdc26
/dev/sddu8	/dev/hdc25
/dev/sddu7	/dev/hdc24
/dev/sddu6	/dev/hdc23
/dev/sddu5	/dev/hdc22
/dev/sddu4	/dev/hdc21
/dev/sddu3	/dev/hdc20
/dev/sddu2	/dev/hdc2
/dev/sddu15	/dev/hdc19
/dev/sddu14	/dev/hdc18
/dev/sddu13	/dev/hdc17
/dev/sddu12	/dev/hdc16
/dev/sddu11	/dev/hdc15
/dev/sddu10	/dev/hdc14

/dev/sddu1	/dev/hdc13
/dev/sddu	/dev/hdc12
/dev/sddt9	/dev/hdc11
/dev/sddt8	/dev/hdc10
/dev/sddt7	/dev/hdc1
/dev/sddt6	/dev/hdc
/dev/sddt5	/dev/hdb9
/dev/sddt4	/dev/hdb8
/dev/sddt3	/dev/hdb7
/dev/sddt2	/dev/hdb6
/dev/sddt15	/dev/hdb5
/dev/sddt14	/dev/hdb4
/dev/sddt13	/dev/hdb32
/dev/sddt12	/dev/hdb31
/dev/sddt11	/dev/hdb30
/dev/sddt10	/dev/hdb3
/dev/sddt1	/dev/hdb29
/dev/sddt	/dev/hdb28
/dev/sdds9	/dev/hdb27
/dev/sdds8	/dev/hdb26
/dev/sdds7	/dev/hdb25
/dev/sdds6	/dev/hdb24
/dev/sdds5	/dev/hdb23
/dev/sdds4	/dev/hdb22
/dev/sdds3	/dev/hdb21
/dev/sdds2	/dev/hdb20
/dev/sdds15	/dev/hdb2
/dev/sdds14	/dev/hdb19
/dev/sdds13	/dev/hdb18
/dev/sdds12	/dev/hdb17
/dev/sdds11	/dev/hdb16
/dev/sdds10	/dev/hdb15
/dev/sdds1	/dev/hdb14
/dev/sdds	/dev/hdb13
/dev/sddr9	/dev/hdb12
/dev/sddr8	/dev/hdb11
/dev/sddr7	/dev/hdb10
/dev/sddr6	/dev/hdb1
/dev/sddr5	/dev/hdb
/dev/sddr4	/dev/hda9
/dev/sddr3	/dev/hda8
/dev/sddr2	/dev/hda7
/dev/sddr15	/dev/hda6
/dev/sddr14	/dev/hda5
/dev/sddr13	/dev/hda4
/dev/sddr12	/dev/hda32
/dev/sddr11	/dev/hda31
/dev/sddr10	/dev/hda30
/dev/sddr1	/dev/hda3
/dev/sddr	/dev/hda29
/dev/sddq9	/dev/hda28
/dev/sddq8	/dev/hda27
/dev/sddq7	/dev/hda26
/dev/sddq6	/dev/hda25
/dev/sddq5	/dev/hda24
/dev/sddq4	/dev/hda23
/dev/sddq3	/dev/hda22

/dev/sddq2	/dev/hda21
/dev/sddq15	/dev/hda20
/dev/sddq14	/dev/hda2
/dev/sddq13	/dev/hda19
/dev/sddq12	/dev/hda18
/dev/sddq11	/dev/hda17
/dev/sddq10	/dev/hda16
/dev/sddq1	/dev/hda15
/dev/sddq	/dev/hda14
/dev/sddp9	/dev/hda13
/dev/sddp8	/dev/hda12
/dev/sddp7	/dev/hda11
/dev/sddp6	/dev/hda10
/dev/sddp5	/dev/hda1
/dev/sddp4	/dev/hda
/dev/sddp3	/dev/gscd
/dev/sddp2	/dev/fd7u830
/dev/sddp15	/dev/fd7u820
/dev/sddp14	/dev/fd7u800
/dev/sddp13	/dev/fd7u720
/dev/sddp12	/dev/fd7u3840
/dev/sddp11	/dev/fd7u360
/dev/sddp10	/dev/fd7u3520
/dev/sddp1	/dev/fd7u3200
/dev/sddp	/dev/fd7u2880
/dev/sddo9	/dev/fd7u1920
/dev/sddo8	/dev/fd7u1840
/dev/sddo7	/dev/fd7u1760
/dev/sddo6	/dev/fd7u1743
/dev/sddo5	/dev/fd7u1722
/dev/sddo4	/dev/fd7u1680
/dev/sddo3	/dev/fd7u1660
/dev/sddo2	/dev/fd7u1440
/dev/sddo15	/dev/fd7u1120
/dev/sddo14	/dev/fd7u1040
/dev/sddo13	/dev/fd7h880
/dev/sddo12	/dev/fd7h720
/dev/sddo11	/dev/fd7h420
/dev/sddo10	/dev/fd7h410
/dev/sddo1	/dev/fd7h360
/dev/sddo	/dev/fd7h1660
/dev/sddn9	/dev/fd7h1494
/dev/sddn8	/dev/fd7h1476
/dev/sddn7	/dev/fd7h1440
/dev/sddn6	/dev/fd7h1200
/dev/sddn5	/dev/fd7d360
/dev/sddn4	/dev/fd7CompaQ
/dev/sddn3	/dev/fd7
/dev/sddn2	/dev/fd6u830
/dev/sddn15	/dev/fd6u820
/dev/sddn14	/dev/fd6u800
/dev/sddn13	/dev/fd6u720
/dev/sddn12	/dev/fd6u3840
/dev/sddn11	/dev/fd6u360
/dev/sddn10	/dev/fd6u3520
/dev/sddn1	/dev/fd6u3200
/dev/sddn	/dev/fd6u2880

/dev/sddm9	/dev/fd6u1920
/dev/sddm8	/dev/fd6u1840
/dev/sddm7	/dev/fd6u1760
/dev/sddm6	/dev/fd6u1743
/dev/sddm5	/dev/fd6u1722
/dev/sddm4	/dev/fd6u1680
/dev/sddm3	/dev/fd6u1660
/dev/sddm2	/dev/fd6u1440
/dev/sddm15	/dev/fd6u1120
/dev/sddm14	/dev/fd6u1040
/dev/sddm13	/dev/fd6h880
/dev/sddm12	/dev/fd6h720
/dev/sddm11	/dev/fd6h420
/dev/sddm10	/dev/fd6h410
/dev/sddm1	/dev/fd6h360
/dev/sddm	/dev/fd6h1660
/dev/sddl9	/dev/fd6h1494
/dev/sddl8	/dev/fd6h1476
/dev/sddl7	/dev/fd6h1440
/dev/sddl6	/dev/fd6h1200
/dev/sddl5	/dev/fd6d360
/dev/sddl4	/dev/fd6CompaQ
/dev/sddl3	/dev/fd6
/dev/sddl2	/dev/fd5u830
/dev/sddl15	/dev/fd5u820
/dev/sddl14	/dev/fd5u800
/dev/sddl13	/dev/fd5u720
/dev/sddl12	/dev/fd5u3840
/dev/sddl11	/dev/fd5u360
/dev/sddl10	/dev/fd5u3520
/dev/sddl1	/dev/fd5u3200
/dev/sddl	/dev/fd5u2880
/dev/sddk9	/dev/fd5u1920
/dev/sddk8	/dev/fd5u1840
/dev/sddk7	/dev/fd5u1760
/dev/sddk6	/dev/fd5u1743
/dev/sddk5	/dev/fd5u1722
/dev/sddk4	/dev/fd5u1680
/dev/sddk3	/dev/fd5u1660
/dev/sddk2	/dev/fd5u1440
/dev/sddk15	/dev/fd5u1120
/dev/sddk14	/dev/fd5u1040
/dev/sddk13	/dev/fd5h880
/dev/sddk12	/dev/fd5h720
/dev/sddk11	/dev/fd5h420
/dev/sddk10	/dev/fd5h410
/dev/sddk1	/dev/fd5h360
/dev/sddk	/dev/fd5h1660
/dev/sddj9	/dev/fd5h1494
/dev/sddj8	/dev/fd5h1476
/dev/sddj7	/dev/fd5h1440
/dev/sddj6	/dev/fd5h1200
/dev/sddj5	/dev/fd5d360
/dev/sddj4	/dev/fd5CompaQ
/dev/sddj3	/dev/fd5
/dev/sddj2	/dev/fd4u830
/dev/sddj15	/dev/fd4u820

/dev/sddj14	/dev/fd4u800
/dev/sddj13	/dev/fd4u720
/dev/sddj12	/dev/fd4u3840
/dev/sddj11	/dev/fd4u360
/dev/sddj10	/dev/fd4u3520
/dev/sddj1	/dev/fd4u3200
/dev/sddj	/dev/fd4u2880
/dev/sddi9	/dev/fd4u1920
/dev/sddi8	/dev/fd4u1840
/dev/sddi7	/dev/fd4u1760
/dev/sddi6	/dev/fd4u1743
/dev/sddi5	/dev/fd4u1722
/dev/sddi4	/dev/fd4u1680
/dev/sddi3	/dev/fd4u1660
/dev/sddi2	/dev/fd4u1440
/dev/sddi15	/dev/fd4u1120
/dev/sddi14	/dev/fd4u1040
/dev/sddi13	/dev/fd4h880
/dev/sddi12	/dev/fd4h720
/dev/sddi11	/dev/fd4h420
/dev/sddi10	/dev/fd4h410
/dev/sddi1	/dev/fd4h360
/dev/sddi	/dev/fd4h1660
/dev/sddh9	/dev/fd4h1494
/dev/sddh8	/dev/fd4h1476
/dev/sddh7	/dev/fd4h1440
/dev/sddh6	/dev/fd4h1200
/dev/sddh5	/dev/fd4d360
/dev/sddh4	/dev/fd4CompaQ
/dev/sddh3	/dev/fd4
/dev/sddh2	/dev/fd3u830
/dev/sddh15	/dev/fd3u820
/dev/sddh14	/dev/fd3u800
/dev/sddh13	/dev/fd3u720
/dev/sddh12	/dev/fd3u3840
/dev/sddh11	/dev/fd3u360
/dev/sddh10	/dev/fd3u3520
/dev/sddh1	/dev/fd3u3200
/dev/sddh	/dev/fd3u2880
/dev/sddg9	/dev/fd3u1920
/dev/sddg8	/dev/fd3u1840
/dev/sddg7	/dev/fd3u1760
/dev/sddg6	/dev/fd3u1743
/dev/sddg5	/dev/fd3u1722
/dev/sddg4	/dev/fd3u1680
/dev/sddg3	/dev/fd3u1660
/dev/sddg2	/dev/fd3u1440
/dev/sddg15	/dev/fd3u1120
/dev/sddg14	/dev/fd3u1040
/dev/sddg13	/dev/fd3h880
/dev/sddg12	/dev/fd3h720
/dev/sddg11	/dev/fd3h420
/dev/sddg10	/dev/fd3h410
/dev/sddg1	/dev/fd3h360
/dev/sddg	/dev/fd3h1660
/dev/sddf9	/dev/fd3h1494
/dev/sddf8	/dev/fd3h1476

/dev/sddf7	/dev/fd3h1440
/dev/sddf6	/dev/fd3h1200
/dev/sddf5	/dev/fd3d360
/dev/sddf4	/dev/fd3H720
/dev/sddf3	/dev/fd3H360
/dev/sddf2	/dev/fd3H1440
/dev/sddf15	/dev/fd3D720
/dev/sddf14	/dev/fd3D360
/dev/sddf13	/dev/fd3CompaQ
/dev/sddf12	/dev/fd3
/dev/sddf11	/dev/fd2u830
/dev/sddf10	/dev/fd2u820
/dev/sddf1	/dev/fd2u800
/dev/sddf	/dev/fd2u720
/dev/sdde9	/dev/fd2u3840
/dev/sdde8	/dev/fd2u360
/dev/sdde7	/dev/fd2u3520
/dev/sdde6	/dev/fd2u3200
/dev/sdde5	/dev/fd2u2880
/dev/sdde4	/dev/fd2u1920
/dev/sdde3	/dev/fd2u1840
/dev/sdde2	/dev/fd2u1760
/dev/sdde15	/dev/fd2u1743
/dev/sdde14	/dev/fd2u1722
/dev/sdde13	/dev/fd2u1680
/dev/sdde12	/dev/fd2u1660
/dev/sdde11	/dev/fd2u1440
/dev/sdde10	/dev/fd2u1120
/dev/sdde1	/dev/fd2u1040
/dev/sdde	/dev/fd2h880
/dev/sddd9	/dev/fd2h720
/dev/sddd8	/dev/fd2h420
/dev/sddd7	/dev/fd2h410
/dev/sddd6	/dev/fd2h360
/dev/sddd5	/dev/fd2h1660
/dev/sddd4	/dev/fd2h1494
/dev/sddd3	/dev/fd2h1476
/dev/sddd2	/dev/fd2h1440
/dev/sddd15	/dev/fd2h1200
/dev/sddd14	/dev/fd2d360
/dev/sddd13	/dev/fd2H720
/dev/sddd12	/dev/fd2H360
/dev/sddd11	/dev/fd2H1440
/dev/sddd10	/dev/fd2D720
/dev/sddd1	/dev/fd2D360
/dev/sddd	/dev/fd2CompaQ
/dev/sddc9	/dev/fd2
/dev/sddc8	/dev/fd1u830
/dev/sddc7	/dev/fd1u820
/dev/sddc6	/dev/fd1u800
/dev/sddc5	/dev/fd1u720
/dev/sddc4	/dev/fd1u3840
/dev/sddc3	/dev/fd1u360
/dev/sddc2	/dev/fd1u3520
/dev/sddc15	/dev/fd1u3200
/dev/sddc14	/dev/fd1u2880
/dev/sddc13	/dev/fd1u1920

/dev/sddc12	/dev/fd1u1840
/dev/sddc11	/dev/fd1u1760
/dev/sddc10	/dev/fd1u1743
/dev/sddc1	/dev/fd1u1722
/dev/sddc	/dev/fd1u1680
/dev/sddb9	/dev/fd1u1660
/dev/sddb8	/dev/fd1u1440
/dev/sddb7	/dev/fd1u1120
/dev/sddb6	/dev/fd1u1040
/dev/sddb5	/dev/fd1h880
/dev/sddb4	/dev/fd1h720
/dev/sddb3	/dev/fd1h420
/dev/sddb2	/dev/fd1h410
/dev/sddb15	/dev/fd1h360
/dev/sddb14	/dev/fd1h1660
/dev/sddb13	/dev/fd1h1494
/dev/sddb12	/dev/fd1h1476
/dev/sddb11	/dev/fd1h1440
/dev/sddb10	/dev/fd1h1200
/dev/sddb1	/dev/fd1d360
/dev/sddb	/dev/fd1H720
/dev/sdda9	/dev/fd1H360
/dev/sdda8	/dev/fd1H1440
/dev/sdda7	/dev/fd1D720
/dev/sdda6	/dev/fd1D360
/dev/sdda5	/dev/fd1CompaQ
/dev/sdda4	/dev/fd1
/dev/sdda3	/dev/fd0u830
/dev/sdda2	/dev/fd0u820
/dev/sdda15	/dev/fd0u800
/dev/sdda14	/dev/fd0u720
/dev/sdda13	/dev/fd0u3840
/dev/sdda12	/dev/fd0u360
/dev/sdda11	/dev/fd0u3520
/dev/sdda10	/dev/fd0u3200
/dev/sdda1	/dev/fd0u2880
/dev/sdda	/dev/fd0u1920
/dev/sdd9	/dev/fd0u1840
/dev/sdd8	/dev/fd0u1760
/dev/sdd7	/dev/fd0u1743
/dev/sdd6	/dev/fd0u1722
/dev/sdd5	/dev/fd0u1680
/dev/sdd4	/dev/fd0u1660
/dev/sdd3	/dev/fd0u1440
/dev/sdd2	/dev/fd0u1120
/dev/sdd15	/dev/fd0u1040
/dev/sdd14	/dev/fd0h880
/dev/sdd13	/dev/fd0h720
/dev/sdd12	/dev/fd0h420
/dev/sdd11	/dev/fd0h410
/dev/sdd10	/dev/fd0h360
/dev/sdd1	/dev/fd0h1660
/dev/sdd	/dev/fd0h1494
/dev/sdcz9	/dev/fd0h1476
/dev/sdcz8	/dev/fd0h1440
/dev/sdcz7	/dev/fd0d360
/dev/sdcz6	/dev/fd0H720

/dev/sdcz5	/dev/fd0H360
/dev/sdcz4	/dev/fd0h1200
/dev/sdcz3	/dev/fd0D720
/dev/sdcz2	/dev/fd0D360
/dev/sdcz15	/dev/fd0H1440
/dev/sdcz14	/dev/fd0
/dev/sdcz13	/dev/fd0CompaQ
/dev/sdcz12	/dev/cm206cd
/dev/sdcz11	/dev/cm205cd
/dev/sdcz10	/dev/cdu535
/dev/sdcz1	/dev/cdu31a
/dev/sdcz	/dev/bpcd
/dev/sdcy9	/dev/aztcd
/dev/sdcy8	/dev/md15
/dev/sdcy7	/dev/md14
/dev/sdcy6	/dev/md13
/dev/sdcy5	/dev/md12
/dev/sdcy4	/dev/md11
/dev/sdcy3	/dev/md9
/dev/sdcy2	/dev/md8
/dev/sdcy15	/dev/md7
/dev/sdcy14	/dev/md6
/dev/sdcy13	/dev/md5
/dev/sdcy12	/dev/md4
/dev/sdcy11	/dev/md3
/dev/sdcy10	/dev/md2
/dev/sdcy1	/dev/md1
/dev/sdcy	/dev/md0
/dev/sdcx9	/dev/md10
/dev/sdcx8	/dev/null
/dev/sdcx7	Wrong medium type
/dev/sdcx6	No medium found
/dev/sdcx5	Disk quota exceeded
/dev/sdcx4	Remote I/O error
/dev/sdcx3	Is a named type file
/dev/sdcx2	No XENIX semaphores available
/dev/sdcx15	Not a XENIX named type file
/dev/sdcx14	Structure needs cleaning
/dev/sdcx13	Stale NFS file handle
/dev/sdcx12	Operation now in progress
/dev/sdcx11	Operation already in progress
/dev/sdcx10	No route to host
/dev/sdcx1	Host is down
/dev/sdcx	Connection refused
/dev/sdcw9	Connection timed out
/dev/sdcw8	No buffer space available
/dev/sdcw7	Connection reset by peer
/dev/sdcw6	Network is unreachable
/dev/sdcw5	Network is down
/dev/sdcw4	Address already in use
/dev/sdcw3	Protocol family not supported
/dev/sdcw2	Operation not supported
/dev/sdcw15	Socket type not supported
/dev/sdcw14	Protocol not supported
/dev/sdcw13	Protocol not available
/dev/sdcw12	Message too long
/dev/sdcw11	Destination address required

/dev/sdcw10	Too many users
/dev/sdcw1	Streams pipe error
/dev/sdcw	Remote address changed
/dev/sdcv9	File descriptor in bad state
/dev/sdcv8	Name not unique on network
/dev/sdcv7	Bad message
/dev/sdcv6	RFS specific error
/dev/sdcv5	Multihop attempted
/dev/sdcv4	Protocol error
/dev/sdcv3	Communication error on send
/dev/sdcv2	Srmount error
/dev/sdcv15	Advertise error
/dev/sdcv14	Link has been severed
/dev/sdcv13	Object is remote
/dev/sdcv12	Package not installed
/dev/sdcv11	Machine is not on the network
/dev/sdcv10	Out of streams resources
/dev/sdcv1	Timer expired
/dev/sdcv	No data available
/dev/sdcu9	Device not a stream
/dev/sdcu8	Bad font file format
/dev/sdcu7	Invalid slot
/dev/sdcu6	Invalid request code
/dev/sdcu5	No anode
/dev/sdcu4	Exchange full
/dev/sdcu3	Invalid request descriptor
/dev/sdcu2	Invalid exchange
/dev/sdcu15	Level 2 halted
/dev/sdcu14	No CSI structure available
/dev/sdcu13	Protocol driver not attached
/dev/sdcu12	Link number out of range
/dev/sdcu11	Level 3 reset
/dev/sdcu10	Level 3 halted
/dev/sdcu1	Level 2 not synchronized
/dev/sdcu	Channel number out of range
/dev/sdct9	Identifier removed
/dev/sdct8	No message of desired type
/dev/sdct7	Directory not empty
/dev/sdct6	Function not implemented
/dev/sdct5	No locks available
/dev/sdct4	File name too long
/dev/sdct3	Resource deadlock avoided
/dev/sdct2	Numerical result out of range
/dev/sdct15	Broken pipe
/dev/sdct14	Too many links
/dev/sdct13	Read-only file system
/dev/sdct12	Illegal seek
/dev/sdct11	No space left on device
/dev/sdct10	File too large
/dev/sdct1	Text file busy
/dev/sdct	Too many open files
/dev/sdcs9	Too many open files in system
/dev/sdcs8	Invalid argument
/dev/sdcs7	Is a directory
/dev/sdcs6	Not a directory
/dev/sdcs5	No such device
/dev/sdcs4	Invalid cross-device link

/dev/sdcs3	File exists
/dev/sdcs2	Device or resource busy
/dev/sdcs15	Block device required
/dev/sdcs14	Bad address
/dev/sdcs13	Permission denied
/dev/sdcs12	Cannot allocate memory
/dev/sdcs11	No child processes
/dev/sdcs10	Bad file descriptor
/dev/sdcs1	Exec format error
/dev/sdcs	Argument list too long
/dev/sdcr9	No such device or address
/dev/sdcr8	Input/output error
/dev/sdcr7	Interrupted system call
/dev/sdcr6	No such process
/dev/sdcr5	No such file or directory
/dev/sdcr4	Operation not permitted
/dev/sdcr3	Success
/dev/sdcr2	Too many references: cannot splice
/dev/sdcr15	Cannot send after transport endpoint
/dev/sdcr14	shutdown
/dev/sdcr13	Transport endpoint is not connected
/dev/sdcr12	Transport endpoint is already
/dev/sdcr11	connected
/dev/sdcr10	Software caused connection abort
/dev/sdcr1	Network dropped connection on reset
/dev/sdcr	Cannot assign requested address
/dev/sdcq9	Address family not supported by
/dev/sdcq8	protocol
/dev/sdcq7	Protocol wrong type for socket
/dev/sdcq6	Socket operation on non-socket
/dev/sdcq5	Interrupted system call should be
/dev/sdcq4	restarted
/dev/sdcq3	Invalid or incomplete multibyte or
/dev/sdcq2	wide character
/dev/sdcq15	Cannot exec a shared library directly
/dev/sdcq14	Attempting to link in too many shared
/dev/sdcq13	libraries
/dev/sdcq12	.lib section in a.out corrupted
/dev/sdcq11	Accessing a corrupted shared library
/dev/sdcq10	Can not access a needed shared library
/dev/sdcq1	Value too large for defined data type
/dev/sdcq	Too many levels of symbolic links
/dev/sdcp9	Numerical argument out of domain
/dev/sdcp8	Inappropriate ioctl for device
/dev/sdcp7	Resource temporarily unavailable
/dev/sdcp6	,ccs=
/dev/sdcp5	TOP_PAD_
/dev/sdcp4	MMAP_MAX_
/dev/sdcp3	TRIM_THRESHOLD_
/dev/sdcp2	MMAP_THRESHOLD_
/dev/sdcp15	Arena %d:
/dev/sdcp14	system bytes = %10u
/dev/sdcp13	in use bytes = %10u
/dev/sdcp12	Total (incl. mmap):
/dev/sdcp11	max mmap regions = %10u
/dev/sdcp10	max mmap bytes = %10lu
/dev/sdcp1	malloc: top chunk is corrupt

/dev/sdcp	free(): invalid pointer %p!
/dev/sdco9	malloc: using debugging hooks
/dev/sdco8	realloc(): invalid pointer %p!
/dev/sdco7	Unknown error
/dev/sdco6	ANSI_X3.4-1968//TRANSLIT
/dev/sdco5	syslog: unknown facility/priority: %x
/dev/sdco4	out of memory [
/dev/sdco3	<%d>
/dev/sdco2	%h %e %T
/dev/sdco15	[%d]
/dev/sdco14	/dev/console
/dev/sdco13	/dev/log
/dev/sdco12	apic
/dev/sdco11	mtrr
/dev/sdco10	cmov
/dev/sdco1	pse36
/dev/sdco	clflush
/dev/sdcn9	acpi
/dev/sdcn8	fxsr
/dev/sdcn7	sse2
/dev/sdcn6	ia64
/dev/sdcn5	amd3d
/dev/sdcn4	i386
/dev/sdcn3	i486
/dev/sdcn2	i586
/dev/sdcn15	i686
/dev/sdcn14	LD_AOUT_LIBRARY_PATH
/dev/sdcn13	LD_AOUT_PRELOAD
/dev/sdcn12	LD_PRELOAD
/dev/sdcn11	LD_LIBRARY_PATH
/dev/sdcn10	LD_ORIGIN_PATH
/dev/sdcn1	LD_DEBUG_OUTPUT
/dev/sdcn	LD_PROFILE
/dev/sdcm9	GCONV_PATH
/dev/sdcm8	HOSTALIASES
/dev/sdcm7	LOCALDOMAIN
/dev/sdcm6	LOCPATH
/dev/sdcm5	MALLOC_TRACE
/dev/sdcm4	NLSPATH
/dev/sdcm3	RESOLV_HOST_CONF
/dev/sdcm2	RES_OPTIONS
/dev/sdcm15	TMPDIR
/dev/sdcm14	TZDIR
/dev/sdcm13	LD_WARN
/dev/sdcm12	LD_LIBRARY_PATH
/dev/sdcm11	LD_BIND_NOW
/dev/sdcm10	LD_BIND_NOT
/dev/sdcm1	LD_DYNAMIC_WEAK
/dev/sdcm	/etc/suid-debug
/dev/sdcl9	MALLOC_CHECK_
/dev/sdcl8	/proc/sys/kernel/osrelease
/dev/sdcl7	FATAL: kernel too old
/dev/sdcl6	FATAL: cannot determine library
/dev/sdcl5	version
/dev/sdcl4	/usr/lib/gconv
/dev/sdcl3	gconv-modules
/dev/sdcl2	=INTERNAL->ucs2reverse

/dev/sdcl15	=ucs2reverse->INTERNAL
/dev/sdcl14	=INTERNAL->ascii
/dev/sdcl13	=ascii->INTERNAL
/dev/sdcl12	=INTERNAL->ucs2
/dev/sdcl11	=ucs2->INTERNAL
/dev/sdcl10	=utf8->INTERNAL
/dev/sdcl1	=INTERNAL->utf8
/dev/sdcl	=ucs4le->INTERNAL
/dev/sdck9	=INTERNAL->ucs4le
/dev/sdck8	UCS-4LE//
/dev/sdck7	=ucs4->INTERNAL
/dev/sdck6	=INTERNAL->ucs4
/dev/sdck5	UCS-2BE// UNICODEBIG//
/dev/sdck4	UCS-2LE// ISO-10646/UCS2/
/dev/sdck3	CSASCII// ANSI_X3.4-1968//
/dev/sdck2	CP367// ANSI_X3.4-1968//
/dev/sdck15	IBM367// ANSI_X3.4-1968//
/dev/sdck14	US-ASCII// ANSI_X3.4-1968//
/dev/sdck13	ISO646-US// ANSI_X3.4-1968//
/dev/sdck12	ISO-IR-6// ANSI_X3.4-1968//
/dev/sdck11	ANSI_X3.4// ANSI_X3.4-1968//
/dev/sdck10	OSF00010102// ISO-10646/UCS2/
/dev/sdck1	OSF00010101// ISO-10646/UCS2/
/dev/sdck	OSF00010100// ISO-10646/UCS2/
/dev/sdcj9	UCS-2// ISO-10646/UCS2/
/dev/sdcj8	UCS2// ISO-10646/UCS2/
/dev/sdcj7	OSF05010001// ISO-10646/UTF8/
/dev/sdcj6	ISO-IR-193// ISO-10646/UTF8/
/dev/sdcj5	UTF-8// ISO-10646/UTF8/
/dev/sdcj4	UTF8// ISO-10646/UTF8/
/dev/sdcj3	WCHAR_T// INTERNAL
/dev/sdcj2	OSF00010106// ISO-10646/UCS4/
/dev/sdcj15	OSF00010105// ISO-10646/UCS4/
/dev/sdcj14	OSF00010104// ISO-10646/UCS4/
/dev/sdcj13	ISO-10646// ISO-10646/UCS4/
/dev/sdcj12	CSUCS4// ISO-10646/UCS4/
/dev/sdcj11	UCS-4BE// ISO-10646/UCS4/
/dev/sdcj10	UCS-4// ISO-10646/UCS4/
/dev/sdcj1	alias
/dev/sdcj	module
/dev/sdci9	UNICODELITTLE// ISO-10646/UCS2/
/dev/sdci8	OSF00010020// ANSI_X3.4-1968//
/dev/sdci7	ISO_646.IRV:1991// ANSI_X3.4-1968//
/dev/sdci6	ANSI_X3.4-1986// ANSI_X3.4-1968//
/dev/sdci5	ISO-10646/UTF-8/ ISO-10646/UTF8/
/dev/sdci4	10646-1:1993/UCS4/ ISO-10646/UCS4/
/dev/sdci3	10646-1:1993// ISO-10646/UCS4/
/dev/sdci2	GCONV_PATH
/dev/sdci15	/usr/lib/gconv/gconv-modules.cache
/dev/sdci14	gconv
/dev/sdci13	gconv_init
/dev/sdci12	gconv_end
/dev/sdci11	toupper
/dev/sdci10	tolower
/dev/sdci1	upper
/dev/sdci	lower
/dev/sdch9	alpha

/dev/sdch8	digit
/dev/sdch7	xdigit
/dev/sdch6	space
/dev/sdch5	print
/dev/sdch4	graph
/dev/sdch3	blank
/dev/sdch2	cntrl
/dev/sdch15	punct
/dev/sdch14	alnum
/dev/sdch13	libc
/dev/sdch12	POSIX
/dev/sdch11	ANSI_X3.4-1968
/dev/sdch10	messages
/dev/sdch1	/usr/share/locale
/dev/sdch	POSIX
/dev/sdcg9	LC_COLLATE
/dev/sdcg8	LC_CTYPE
/dev/sdcg7	LC_MONETARY
/dev/sdcg6	LC_NUMERIC
/dev/sdcg5	LC_TIME
/dev/sdcg4	LC_MESSAGES
/dev/sdcg3	LC_ALL
/dev/sdcg2	LC_XXX
/dev/sdcg15	LANGUAGE
/dev/sdcg14	charset=
/dev/sdcg13	OUTPUT_CHARSET
/dev/sdcg12	/usr/share/locale
/dev/sdcg11	/locale.alias
/dev/sdcg10	parse error
/dev/sdcg1	parser stack overflow
/dev/sdcg	plural=
/dev/sdcf9	nplurals=
/dev/sdcf8	0123456789abcdefghijklmnopqrstuvwxyz
/dev/sdcf7	(null)
/dev/sdcf6	(nil)
/dev/sdcf5	0000000000000000
/dev/sdcf4	%m/%d/%y
/dev/sdcf3	%Y-%m-%d
/dev/sdcf2	%H:%M
/dev/sdcf15	%I:%M:%S %p
/dev/sdcf14	%H:%M:%S
/dev/sdcf13	/etc/localtime
/dev/sdcf12	Universal
/dev/sdcf11	%[^0-9,+-]
/dev/sdcf10	%hu:%hu:%hu
/dev/sdcf1	M%hu.%hu.%hu%n
/dev/sdcf	/usr/share/zoneinfo
/dev/sdce9	TZDIR
/dev/sdce8	posixrules
/dev/sdce7	/proc/self/cwd
/dev/sdce6	/proc
/dev/sdce5	/etc/mtab
/dev/sdce4	/etc/fstab
/dev/sdce3	proc
/dev/sdce2	/cpuinfo
/dev/sdce15	processor
/dev/sdce14	/meminfo

/dev/sdce13	MemTotal: %ld kB
/dev/sdce12	MemFree: %ld kB
/dev/sdce11	/lib/
/dev/sdce10	/usr/lib/
/dev/sdce1	ORIGIN
/dev/sdce	PLATFORM
/dev/sdcd9	cannot allocate name record
/dev/sdcd8	system search path
/dev/sdcd7	cannot stat shared object
/dev/sdcd6	cannot read file data
/dev/sdcd5	cannot map zero-fill pages
/dev/sdcd4	cannot create searchlist
/dev/sdcd3	search path=
/dev/sdcd2	(%s from file %s)
/dev/sdcd15	(%s)
/dev/sdcd14	file too short
/dev/sdcd13	invalid ELF header
/dev/sdcd12	ELF file OS ABI invalid
/dev/sdcd11	ELF file ABI version invalid
/dev/sdcd10	internal error
/dev/sdcd1	trying file=%s
/dev/sdcd	file=%s; needed by %s
/dev/sdcc9	find library=%s; searching
/dev/sdcc8	RPATH
/dev/sdcc7	RUNPATH
/dev/sdcc6	cannot create cache for search path
/dev/sdcc5	cannot create RUNPATH/RPATH copy
/dev/sdcc4	cannot create search path array
/dev/sdcc3	file=%s; generating link map
/dev/sdcc2	cannot create shared object descriptor
/dev/sdcc15	ELF load command alignment not page-aligned
/dev/sdcc14	
/dev/sdcc13	ELF load command address/offset not properly aligned
/dev/sdcc12	
/dev/sdcc11	failed to map segment from shared object
/dev/sdcc10	
/dev/sdcc1	cannot dynamically load executable
/dev/sdcc	cannot change memory protections
/dev/sdcb9	cannot allocate memory for program header
/dev/sdcb8	
/dev/sdcb7	object file has no dynamic section
/dev/sdcb6	dynamic: 0x%0*lx base: 0x%0*lx
/dev/sdcb5	size: 0x%0*Zx
/dev/sdcb4	entry: 0x%0*lx phdr: 0x%0*lx
/dev/sdcb3	phnum: %*u
/dev/sdcb2	shared object cannot be dlopen()ed
/dev/sdcb15	ELF file data encoding not big-endian
/dev/sdcb14	ELF file data encoding not little-endian
/dev/sdcb13	
/dev/sdcb12	ELF file version ident does not match current one
/dev/sdcb11	
/dev/sdcb10	ELF file version does not match current one
/dev/sdcb1	
/dev/sdcb	ELF file's phentsize not the expected size
/dev/sdca9	
/dev/sdca8	only ET_DYN and ET_EXEC can be loaded
/dev/sdca7	cannot open shared object file

/dev/sdca6	AT_HWCAP:
/dev/sdca5	/etc/ld.so.cache
/dev/sdca4	search cache=%s
/dev/sdca3	ld.so-1.7.0
/dev/sdca2	glibc-ld.so.cache1.1
/dev/sdca15	undefined symbol:
/dev/sdca14	symbol=%s; lookup in file=%s
/dev/sdca13	file=%s; needed by %s (relocation
/dev/sdca12	dependency)
/dev/sdca11	binding file %s to %s: %s symbol `%s'
/dev/sdca10	relocation error
/dev/sdca1	<main program>
/dev/sdca	symbol
/dev/sdc9	, version
/dev/sdc8	not defined in file
/dev/sdc7	with link time reference
/dev/sdc6	(no version symbols)
/dev/sdc5	protected
/dev/sdc4	normal
/dev/sdc3	[%s]
/dev/sdc2	out of memory
/dev/sdc15	DYNAMIC LINKER BUG!!!
/dev/sdc14	<program name unknown>
/dev/sdc13	%s: %s: %s%s%s%s%s
/dev/sdc12	error while loading shared libraries
/dev/sdc11	/proc/self/exe
/dev/sdc10	IGNORE
/dev/sdc1	gconv_trans_context
/dev/sdc	gconv_trans
/dev/sdbz9	gconv_trans_init
/dev/sdbz8	gconv_trans_end
/dev/sdbz7	LC_IDENTIFICATION
/dev/sdbz6	LC_MEASUREMENT
/dev/sdbz5	LC_TELEPHONE
/dev/sdbz4	LC_ADDRESS
/dev/sdbz3	LC_NAME
/dev/sdbz2	LC_PAPER
/dev/sdbz15	LOCPATH
/dev/sdbz14	/usr/lib/locale
/dev/sdbz13	LANG
/dev/sdbz12	/SYS_
/dev/sdbz11	^[nN]
/dev/sdbz10	^[yY]
/dev/sdbz1	%a %b %e %H:%M:%S %Z %Y
/dev/sdbz	%a %b %e %H:%M:%S %Y
/dev/sdby9	December
/dev/sdby8	November
/dev/sdby7	October
/dev/sdby6	September
/dev/sdby5	August
/dev/sdby4	July
/dev/sdby3	June
/dev/sdby2	April
/dev/sdby15	March
/dev/sdby14	February
/dev/sdby13	January
/dev/sdby12	Saturday

/dev/sdby11	Friday
/dev/sdby10	Thursday
/dev/sdby1	Wednesday
/dev/sdby	Tuesday
/dev/sdbx9	Monday
/dev/sdbx8	Sunday
/dev/sdbx7	%p%t%g%t%m%t%f
/dev/sdbx6	%a%N%f%N%d%N%b%N%s %h %e %r%N%C-%z
/dev/sdbx5	%T%N%c%N
/dev/sdbx4	+%c %a %l
/dev/sdbx3	i18n:1999
/dev/sdbx2	i18n:1999
/dev/sdbx15	i18n:1999
/dev/sdbx14	i18n:1999
/dev/sdbx13	i18n:1999
/dev/sdbx12	i18n:1999
/dev/sdbx11	i18n:1999
/dev/sdbx10	i18n:1999
/dev/sdbx1	i18n:1999
/dev/sdbx	i18n:1999
/dev/sdbw9	i18n:1999
/dev/sdbw8	i18n:1999
/dev/sdbw7	i18n:1999
/dev/sdbw6	i18n:1999
/dev/sdbw5	i18n:1999
/dev/sdbw4	i18n:1999
/dev/sdbw3	1997-12-20
/dev/sdbw2	+45 3325-6543
/dev/sdbw15	+45 3122-6543
/dev/sdbw14	keld@dkuug.dk
/dev/sdbw13	Keld Simonsen
/dev/sdbw12	ISO/IEC 14652 i18n FDCC-set
/dev/sdbw11	C/o Keld Simonsen, Skt. Jorgens Alle
/dev/sdbw10	8, DK-1615 Kobenhavn V
/dev/sdbw1	ISO/IEC JTC1/SC22/WG20 -
/dev/sdbw	internationalization
/dev/sdbv9	!"#$%&'()*+,-
/dev/sdbv8	./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRS
/dev/sdbv7	TUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxy
/dev/sdbv6	z{|}~
/dev/sdbv5	[Am-
/dev/sdbv4	kpnJ
/dev/sdbv3	uD;s
/dev/sdbv2	)r+[
/dev/sdbv15	[!|n
/dev/sdbv14	uYD?e
/dev/sdbv13	I9C-
/dev/sdbv12	I!G.
/dev/sdbv11	U^h6LU3
/dev/sdbv10	U.y`
/dev/sdbv1	3?Cy
/dev/sdbv	'_Djz
/dev/sdbu9	$po?b
/dev/sdbu8	w};u
/dev/sdbu7	=t%j
/dev/sdbu6	MP0!
/dev/sdbu5	t0tv

/dev/sdbu4	=u8Q)+
/dev/sdbu3	*~xx
/dev/sdbu2	~j2=
/dev/sdbu15	|;#o
/dev/sdbu14	Ac+;
/dev/sdbu13	^2XX%
/dev/sdbu12	!{>;b
/dev/sdbu11	dI@B
/dev/sdbu10	2I%%
/dev/sdbu1	{fG5
/dev/sdbu	0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ
/dev/sdbt9	%d %d
/dev/sdbt8	%s %s %s %s %d %d
/dev/sdbt7	gmon
/dev/sdbt6	seconds
/dev/sdbt5	.profile
/dev/sdbt4	%s: cannot open file: %s
/dev/sdbt3	%s: cannot stat file: %s
/dev/sdbt2	%s: cannot create file: %s
/dev/sdbt15	%s: cannot map file: %s
/dev/sdbt14	%s: file is no correct profile data
/dev/sdbt13	file for `%s'
/dev/sdbt12	Out of memory while initializing
/dev/sdbt11	profiler
/dev/sdbt10	cannot extend global scope
/dev/sdbt1	dlopen
/dev/sdbt	cannot create scope list
/dev/sdbs9	invalid mode for dlopen()
/dev/sdbs8	DST not allowed in SUID/SGID programs
/dev/sdbs7	empty dynamic string token
/dev/sdbs6	substitution
/dev/sdbs5	opening file=%s; opencount == %u
/dev/sdbs4	shared object not open
/dev/sdbs3	calling fini: %s
/dev/sdbs2	closing file=%s; opencount == %u
/dev/sdbs15	(lazy)
/dev/sdbs14	relocation processing: %s%s
/dev/sdbs13	cannot make segment writable for
/dev/sdbs12	relocation
/dev/sdbs11	%s: Symbol `%s' has different size in
/dev/sdbs10	shared object, consider re-linking
/dev/sdbs1	%s: profiler found no PLTREL in object
/dev/sdbs	%s
/dev/sdbr9	%s: profiler out of memory shadowing
/dev/sdbr8	PLTREL of %s
/dev/sdbr7	cannot restore segment prot after
/dev/sdbr6	reloc
/dev/sdbr5	unexpected reloc type 0x
/dev/sdbr4	unexpected PLT reloc type 0x
/dev/sdbr3	empty dynamics string token
/dev/sdbr2	substitution
/dev/sdbr15	cannot load auxiliary `%s' because of
/dev/sdbr14	empty dynamic string token
/dev/sdbr13	substitution
/dev/sdbr12	load auxiliary object=%s requested by
/dev/sdbr11	file=%s
/dev/sdbr10	load filtered object=%s requested by

/dev/sdbr1 /dev/sdbr /dev/sdbq9 /dev/sdbq8 /dev/sdbq7 /dev/sdbq6 /dev/sdbq5 /dev/sdbq4 /dev/sdbq3 /dev/sdbq2 /dev/sdbq15 /dev/sdbq14 /dev/sdbq13 /dev/sdbq12 /dev/sdbq11 /dev/sdbq10 /dev/sdbq1 /dev/sdbq	file=%s cannot allocate dependency list cannot allocate symbol search list Filters not supported with LD_TRACE_PRELINKING calling init: %s calling preinit: %s checking for version `%s' in file %s required by file %s no version information available (required by cannot allocate version reference table unsupported version of Verdef record weak version ` ' not found (required by of Verneed record inity

Attachment 2
Readme from “bmap-1.0.20.tar.gz”

bmap: A filesystem-independant file blockmap utility for Linux

Maintained 2000 by Daniel Ridge in support of:
Scyld Computing Corporation.

The maintainer may be reached as newt@scyld.com or C/O
Scyld Computing Corporation
10227 Wincopin Circle, Suite 212
Columbia, MD 21044

Written 1998 by Daniel Ridge in support of:
Computer Crime Division, Office of Inspector General,
National Aeronautics and Space Administration.

The author may no longer be reached at NASA.
Please direct all inquiries to the maintainer.

This code is licensed to you under the terms of the GPL.
See the file COPYING in this distribution for the terms.

WARNING: This may spank your hard drive.

VERSION CHANGES

1.0.20: (5/15/2000) newt@scyld.com
* added bclump program for combining and sorting bmap output
(contributed by Robert J. Hergert)

1.0.19: (4/30/2000) newt@scyld.com
* documentation target now builds just bmap.ps.gz in place
of bmap.dvi and bmap.ps. Spec file updated to reflect this
change.

1.0.18: (4/15/2000) newt@scyld.com
* updated to reflect tweaked mft_log_init()
* updated spec file to build better RPMs

1.0.17: (4/14/2000) newt@scyld.com
* removed archaic index.html

* removed mft as an included component. The scyld packager
now auto-includes this when you ask for bmap. mft will
now be maintained and versioned seperately.
* BUGFIX: casting error created problems on files located above 2gb.
to fix this in older copies, look for assignments to 'offset' and
cast the first argument as 'long long'

1.0.16: (4/11/2000) newt@scyld.com
* maintenance release. No useful changes.

1.0.15: (4/03/2000) newt@scyld.com
* improved SGML documentation

1.0.14: (3/25/2000) newt@scyld.com
* cleanup patchlevel. Removed stale patches from CVS to reflect
some new organization.

1.0.13: (3/24/2000) newt@scyld.com
* released courtesy copy to FBI CART.

1.0.12: (3/22/2000) newt@scyld.com
* released courtesy copy to DCFL with interim documentation
improvements.

1.0.11: (3/15/2000) newt@scyld.com
* released courtesy copy to State Department.

1.0.10: (3/6/2000) newt@scyld.com
* man pages are now auto-generated. This is made possible
now through additions to the support libraries. 'bmap' and
'slacker' will generate man pages when run with '--man'.
* added a new option flag -- 'MOF_HIDDEN' that allows an option
to exist without being displayed in help screens or man pages.
* added a new mode option to bmap '--mode=checkfrag'. This
checks for fragmentation and returns 0 if the file is fragmented.
* moved bmap option '--mode=fragment' to '--mode=frag'
* spun the support library functions into the 'mft' directory.
(These are the common library routines that various forensic
tools share with mcgruff).

1.0.9: (3/5/2000) newt@scyld.com
* integrated latest option processing code from mcgruff.
try 'bmap --help' to see the difference that the new
'verbose enum' has made to the readability of the
built-in documentation.

1.0.8: (3/4/2000) newt@scyld.com
* Updated man pages. Built on PowerPC linux.

1.0.7: (3/3/2000) newt@scyld.com
* 'slacker' added as a companion utility for bmap. This utility operates on the collective slack of a directory tree and performs many of the same slack operations as bmap.

1.0.6: (2/28/2000) newt@scyld.com
* BUGFIX: bmap modes 'wipe','wipeslack','putslack','slackbytes' failed to correctly operate on zero-length files. In certian cases, this tool may attempt to write to block 0 of a raw device. This is very bad.
* BUGFIX: stat sometimes lies about the number of blocks in the file. bmap no longer trusts these counts.

1.0.5: (2/24/2000) newt@scyld.com
* improved logging.

1.0.4: (2/24/2000) newt@scyld.com
* added support for 'raw device' operations in Rick Niles' userspace filesystem shell.

1.0.3: (2/23/2000) newt@scyld.com
* modified logging code to try to get initial log thresh from environment variable (MCGRUFF_LOG_THRESH).
* modified option processing code to allow options like '--carve' to be interpreted as '--mode=carve' to supply backwards compatibility

1.0.2: (2/22/2000) newt@scyld.com
* rearranged invokation slightly. '--carve','--wipe', etc now are invoked as '--mode=carve','--mode=wipe', etc.
* added 'putslack' mode to write into slack.
* added 'checkslack' mode to quietly test for slack.

1.0.1: (2/15/2000) newt@scyld.com
* now maintained by Scyld Computing Corporation
* added option and logging code from mcgruff

1.0.0: (12/28/1999) newt@hq.nasa.gov
* promoted version to 1.0.

0.1.10: (12/22/1999) jakers@hq.nasa.gov
* added 'label' option to print the physical sector information on slack space

* added 'fragfile' option to print fragmented file info to the filename specified. Enables a file to be sector mapped and highlighted if it is fragmented on same pass.
* added 'name' option to print the name of the current file being bmapped to stdout
* added 'verbose' option to print status info on execution

0.1.9: (12/2/1999) newt@hq.nasa.gov
* minor tweaks.

0.1.8: (11/12/1999) newt@hq.nasa.gov
* bmap can now automatically find the device to slack and carve from.

0.1.7: (11/12/1999) newt@hq.nasa.gov
* Trailing whitespace in Makefile caused 'make install' to fail.
* man pages were not being installed. (after all the trouble I went to to write them!)

0.1.6: (9/2/1999) newt@hq.nasa.gov
* added LICENSE file with copyright, warranty, and license information.

0.1.5: (1/7/1999) newt@hq.nasa.gov
* added cheesy b2s byte->sector converter because bash only performs shell arithmetic on longs.

0.1.4: (1/7/1999) newt@hq.nasa.gov
* altered bmap to output sector numbers instead of block numbers

0.1.3: (1/4/1999) newt@hq.nasa.gov
* built for AlphaLinux
* built for SparcLinux
* added '--carve' to bmap to carve out blocks associated with a file.
* added '--slack' to bmap to carve out trailing data in the terminal block of a file.
* added '--raw' to bmap to specify name of raw device to read for '--carve' and '--slack'

0.1.2: (1/1/1999) newt@hq.nasa.gov
* added skeleton for bogoseek() for seeking on large files.
* modified bmap to use lstat() for statting filenames. This allows us to easily collate the results of several runs without having to uniq blocks double-counted through dereferencing symlinks
* corrected block count calculation to match observed behavior of ext2fs on the author's machine.
* added crude hole detection. high-quality hole detection will be

difficult.

0.1.1: (12/31/1998) newt@hq.nasa.gov
* initial release.