Introduction to The Sleuth kit(TSK)

By Vinay Gurram

December 28, 2016

Outline

Introduction--- 2
Installation steps - commands -- 3

Sleuth kit - commands for computer forensics -- 4
Image File Tools-- 4
Volume System Tools-- 5
File System Layer Tools-- 5
File Name Layer Tools-- 6
Meta Data Layer Tools-- 6
Data Unit Layer Tools --- 6
MMLS - Media Management Tools-- 6

In detailed with image disk --- 8
Details of the File System--- 9
fls - File/Dir Listings-- 10
icat - Display a File--- 11

Incidence response - Recovering Deleted Files with the Sleuth Kit--------------------- 12
Steps for any incidece in Sleuth kit tool-- 13
Case study--- 14-18

References--- 19

Introduction: The Sleuth Kit (TSK) is a collection of Unix-based command line tools that allow you to investigate a computer. The current focus of the tools is the file and volume systems and TSK supports many filesystems.
Autopsy is a frontend for TSK which allows browser-based access to the TSK tools.

Download link for File System Analysis using Sleuthkit tool
http://www.sleuthkit.org/

Sleuthkit installation process:

1. >>>./configure
2. >>>make
3. >>>sudo make install

Or

>>>**sudo apt-get update install sleuthkit**

Required packages for sleuth kit tool:
>>>**sudo apt-get update install build-essential zlibig-dev libss1-dev**

The Sleuth Kit (TSK) - Layers

File System Layer
Content / Data Layer
Meta Data Layer / inode Layer
Human Interface / File Layer

Description : The fls program lists file and directory names. This tool will display the names of deleted files as well. The ffind program will identify the name of the file that has allocated a given metadata structure. With some file systems, deleted files will be identified.

SleuthKit Commands for computer forensics

Below is a list of various Sleuth Kit commands used in computer forensics. The majority of these commands are executed against an image file, which in many cases would be a forensic image of a device (e.g. floppy disk, USB key, memory card, hard drive, etc.). Although there are various commercial and open source tools used for creating forensic images, on Linux you can use the native "dd" command to do so. At its simplest level, the command to acquire an image of device /dev/sda (which could be a USB key, or a SATA or SCSI hard drive)

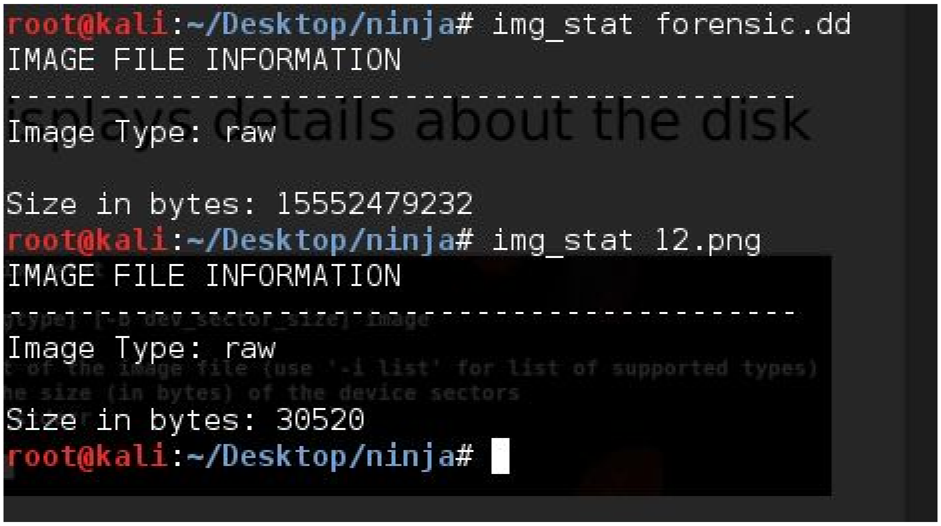
Image File Tools

This layer contains tools for the image file format. For example, if the image format is a split image or a compressed image.

img_stat: tool will show the details of the image format

img_cat: This tool will show the raw contents of an image file.

sample output

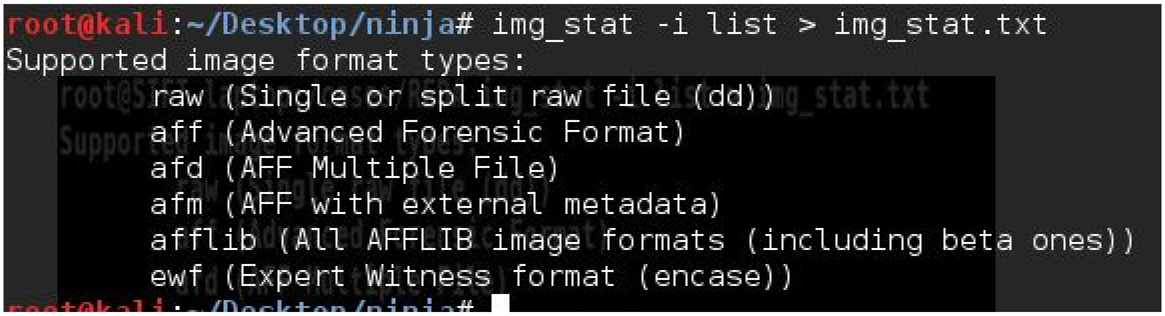


```
root@kali:~/Desktop/ninja# img_stat forensic.dd
IMAGE FILE INFORMATION
--------------------------------------------
Image Type: raw

Size in bytes: 15552479232
root@kali:~/Desktop/ninja# img_stat 12.png
IMAGE FILE INFORMATION
--------------------------------------------
Image Type: raw

Size in bytes: 30520
root@kali:~/Desktop/ninja# 
```

Image formats



```
root@kali:~/Desktop/ninja# img_stat -i list > img_stat.txt
Supported image format types:
        raw (Single or split raw file (dd))
        aff (Advanced Forensic Format)
        afd (AFF Multiple File)
        afm (AFF with external metadata)
        afflib (All AFFLIB image formats (including beta ones))
        ewf (Expert Witness format (encase))
root@kali:~/Desktop/ninja#
```

Volume System Tools

These tools take a disk (or other media) image as input and analyze its partition structures. Examples include DOS artitions, BSD disk labels, and the Sun Volume Table of Contents (VTOC). These can be used find hidden data between partitions and to identify the file system offset for The Sleuth Kit tools. The media management tools support DOS partitions, BSD disk labels, Sun VTOC, and Mac partitions.

mmls: Displays the layout of a disk, including the unallocated spaces.

mmstat: Display details about a volume system (typically only the type).

mmcat: Extracts the contents of a specific volume to STDOUT.

File System Layer Tools

These file system tools process general file system data, such as the layout, allocation structures, and boot blocks

fsstat: Shows file system details and statistics including layout, sizes, and labels.

File Name Layer Tools

These file system tools process the file name structures, which are typically located in the parent directory.

ffind: Finds allocated and unallocated file names that point to a given meta data structure.

fls: Lists allocated and deleted file names in a directory.

Meta Data Layer Tools

These file system tools process the meta data structures, which store the details about a file. Examples of this structure include directory entries in FAT, MFT entries in NTFS, and inodes in ExtX and UFS.

icat: Extracts the data units of a file, which is specified by its meta data address (instead of the file name).

ifind: Finds the meta data structure that has a given file name pointing to it or the meta data structure that points to a given data unit.

Data Unit Layer Tools

These file system tools process the data_units where file content is stored. Examples of this layer include clusters in FAT and NTFS and blocks and fragments in ExtX and UFS.

blkcat: Extracts the contents of a given data unit.

blkls: Lists the details about data units and can extract the unallocated space of the file system.

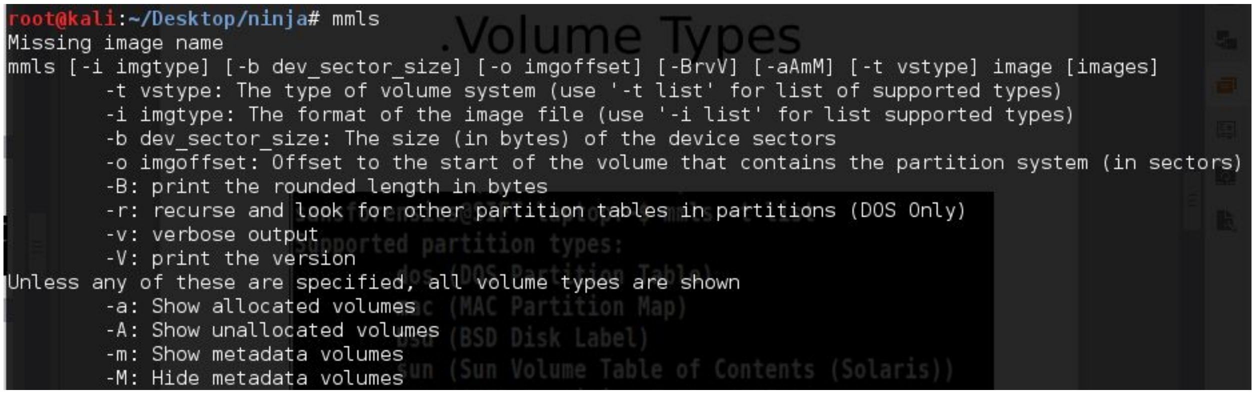
blkstat: Displays the statistics about a given data unit in an easy to read format.

blkcalc: Calculates where data in the unallocated space image (from blkls) exists in the original image. This is used when evidence is found in unallocated space.

MMLS - Media Management Tools

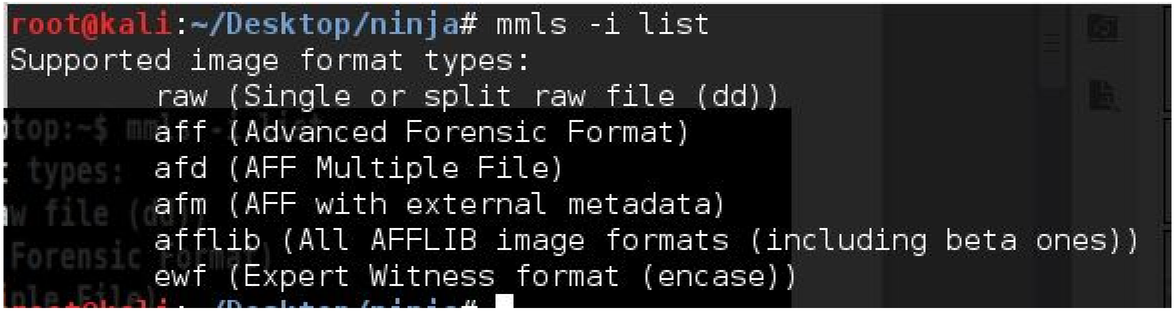
mmls – displays the layout of the disk

Locates the various partitions



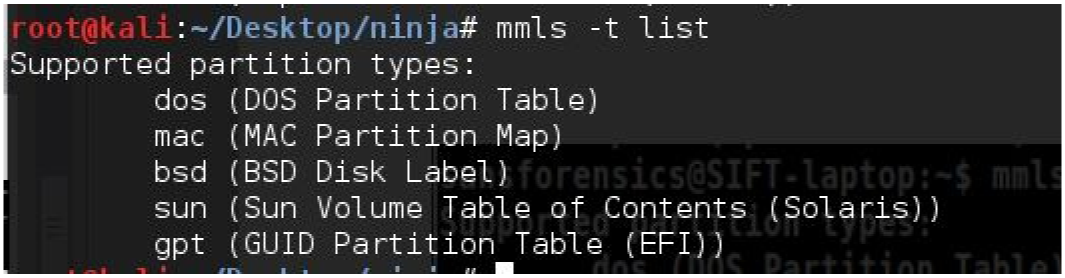
```
root@kali:~/Desktop/ninja# mmls
Missing image name
mmls [-i imgtype] [-b dev_sector_size] [-o imgoffset] [-BrvV] [-aAmM] [-t vstype] image [images]
        -t vstype: The type of volume system (use '-t list' for list of supported types)
        -i imgtype: The format of the image file (use '-i list' for list supported types)
        -b dev_sector_size: The size (in bytes) of the device sectors
        -o imgoffset: Offset to the start of the volume that contains the partition system (in sectors)
        -B: print the rounded length in bytes
        -r: recurse and look for other partition tables in partitions (DOS Only)
        -v: verbose output
        -V: print the version
Unless any of these are specified, all volume types are shown
        -a: Show allocated volumes
        -A: Show unallocated volumes
        -m: Show metadata volumes
        -M: Hide metadata volumes
```

Image types in **mmls:**



```
root@kali:~/Desktop/ninja# mmls -i list
Supported image format types:
        raw (Single or split raw file (dd))
        aff (Advanced Forensic Format)
        afd (AFF Multiple File)
        afm (AFF with external metadata)
        afflib (All AFFLIB image formats (including beta ones))
        ewf (Expert Witness format (encase))
```

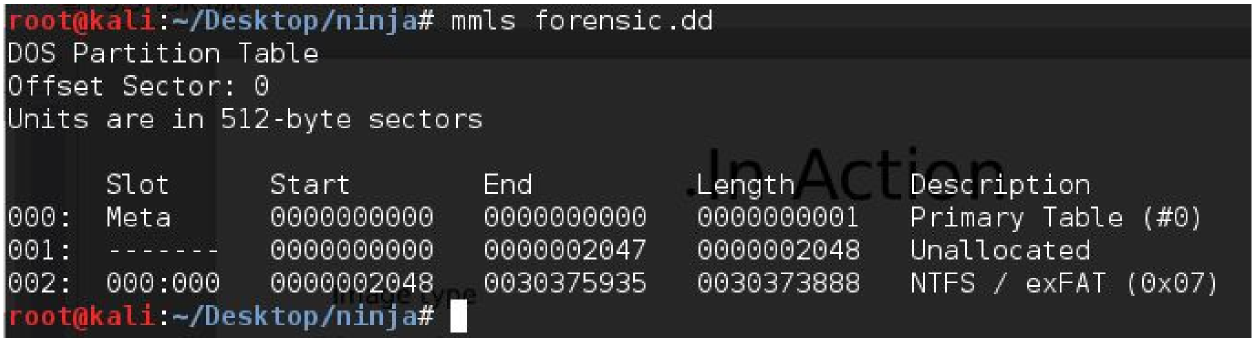
Volume types in **mmls:**



```
root@kali:~/Desktop/ninja# mmls -t list
Supported partition types:
        dos (DOS Partition Table)
        mac (MAC Partition Map)
        bsd (BSD Disk Label)
        sun (Sun Volume Table of Contents (Solaris))
        gpt (GUID Partition Table (EFI))
```

<u>In detailed with image disk</u>

- ➔ Image type
- ➔ Sector size
- ➔ Partition tables
  - ◆ Partition start, end, length, and type
- ➔ Shows unallocated space as separate entries
- ➔ Slot for multiple partition tables as in extended partitions



```
root@kali:~/Desktop/ninja# mmls forensic.dd
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

      Slot      Start        End          Length       Description
000:  Meta      0000000000   0000000000   0000000001   Primary Table (#0)
001:  -------   0000000000   0000002047   0000002048   Unallocated
002:  000:000   0000002048   0030375935   0030373888   NTFS / exFAT (0x07)
root@kali:~/Desktop/ninja#
```

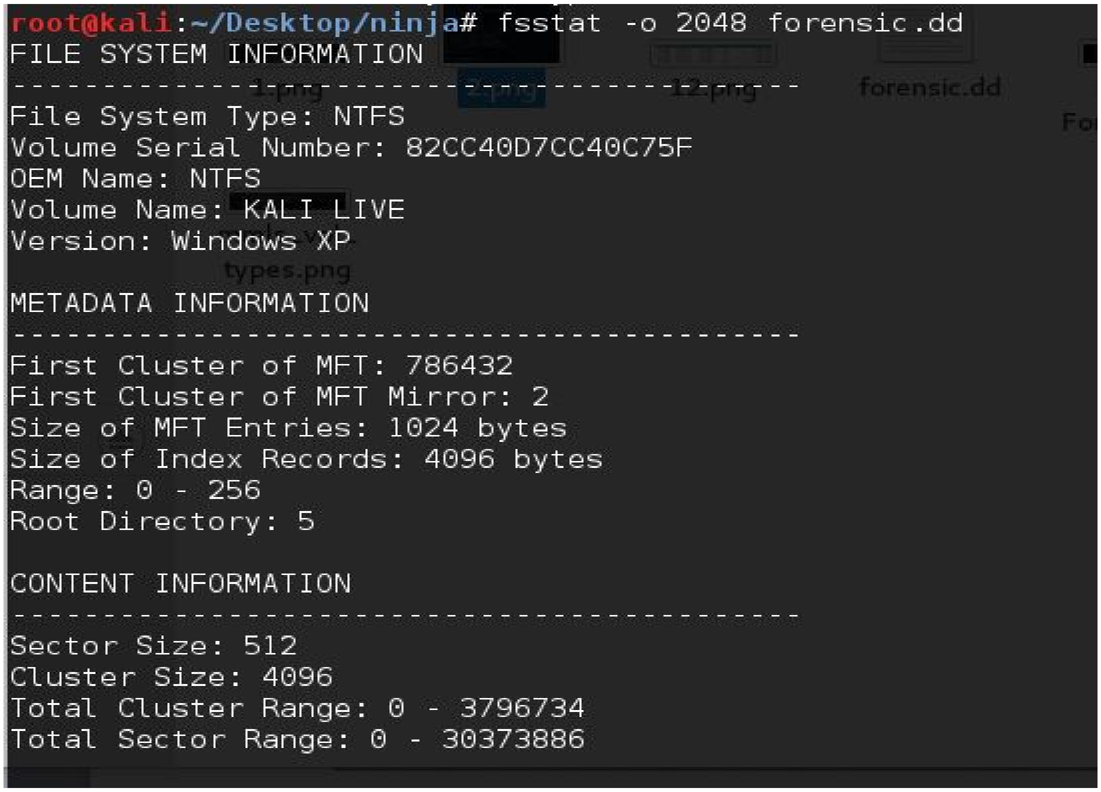
#2048 is the offset here - It will change from disk to disk

Details of the File System

Using fsstat command - we can extract the image of partition

Here Offset - 2048 and disk image - forensic.dd

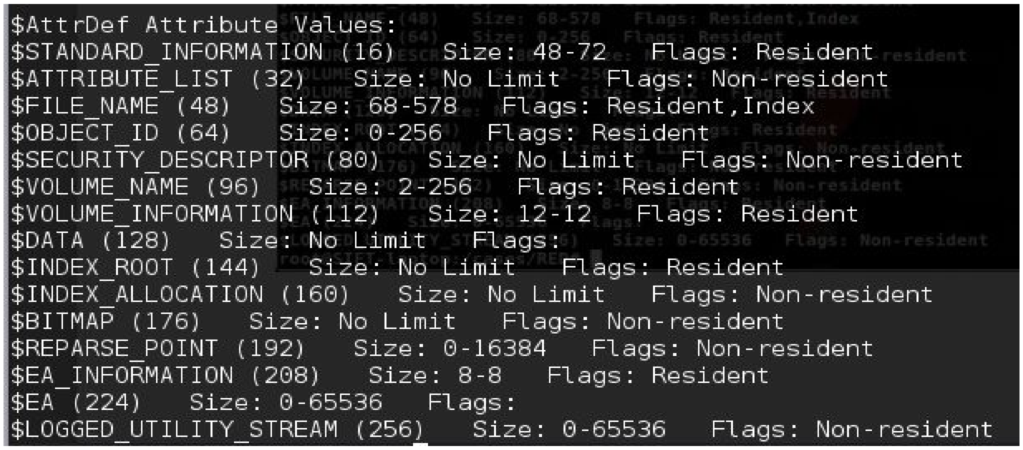
Below image describes about File System, Meta Data, Content Information



```
root@kali:~/Desktop/ninja# fsstat -o 2048 forensic.dd
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: NTFS
Volume Serial Number: 82CC40D7CC40C75F
OEM Name: NTFS
Volume Name: KALI LIVE
Version: Windows XP

METADATA INFORMATION
--------------------------------------------
First Cluster of MFT: 786432
First Cluster of MFT Mirror: 2
Size of MFT Entries: 1024 bytes
Size of Index Records: 4096 bytes
Range: 0 - 256
Root Directory: 5

CONTENT INFORMATION
--------------------------------------------
Sector Size: 512
Cluster Size: 4096
Total Cluster Range: 0 - 3796734
Total Sector Range: 0 - 30373886
```

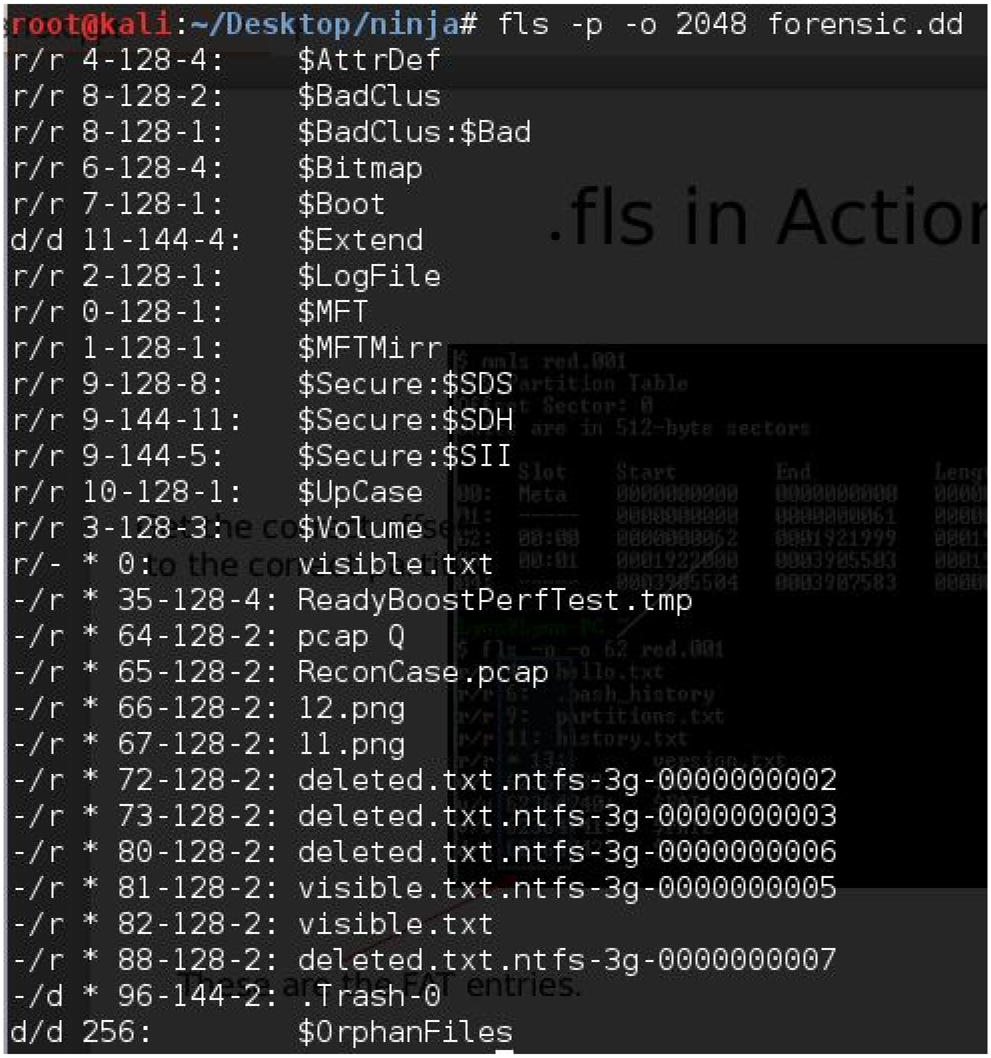


```
$AttrDef Attribute Values:
$STANDARD_INFORMATION (16)   Size: 48-72   Flags: Resident
$ATTRIBUTE_LIST (32)   Size: No Limit   Flags: Non-resident
$FILE_NAME (48)   Size: 68-578   Flags: Resident,Index
$OBJECT_ID (64)   Size: 0-256   Flags: Resident
$SECURITY_DESCRIPTOR (80)   Size: No Limit   Flags: Non-resident
$VOLUME_NAME (96)   Size: 2-256   Flags: Resident
$VOLUME_INFORMATION (112)   Size: 12-12   Flags: Resident
$DATA (128)   Size: No Limit   Flags:
$INDEX_ROOT (144)   Size: No Limit   Flags: Resident
$INDEX_ALLOCATION (160)   Size: No Limit   Flags: Non-resident
$BITMAP (176)   Size: No Limit   Flags: Non-resident
$REPARSE_POINT (192)   Size: 0-16384   Flags: Non-resident
$EA_INFORMATION (208)   Size: 8-8   Flags: Resident
$EA (224)   Size: 0-65536   Flags:
$LOGGED_UTILITY_STREAM (256)   Size: 0-65536   Flags: Non-resident
```

`fls` – File/Dir Listings

- ➔ List all directories and files in an image
  - ◆ Inodes or MFT entries, etc.
  - ◆ Full path
- ➔ List file types
- ➔ List MAC dtg's
- ➔ Lists deleted or undeleted files only

Sample output:

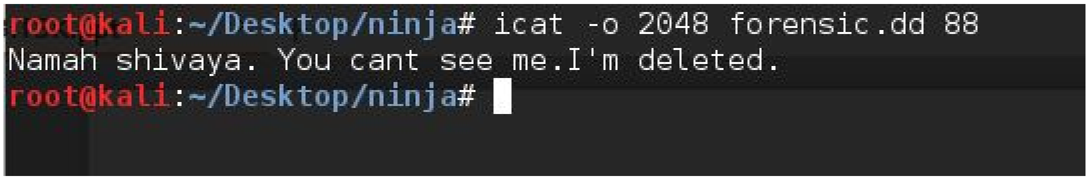


```
root@kali:~/Desktop/ninja# fls -p -o 2048 forensic.dd
r/r 4-128-4:    $AttrDef
r/r 8-128-2:    $BadClus
r/r 8-128-1:    $BadClus:$Bad
r/r 6-128-4:    $Bitmap
r/r 7-128-1:    $Boot
d/d 11-144-4:   $Extend
r/r 2-128-1:    $LogFile
r/r 0-128-1:    $MFT
r/r 1-128-1:    $MFTMirr
r/r 9-128-8:    $Secure:$SDS
r/r 9-144-11:   $Secure:$SDH
r/r 9-144-5:    $Secure:$SII
r/r 10-128-1:   $UpCase
r/r 3-128-3:    $Volume
r/- * 0:        visible.txt
-/r * 35-128-4: ReadyBoostPerfTest.tmp
-/r * 64-128-2: pcap Q
-/r * 65-128-2: ReconCase.pcap
-/r * 66-128-2: 12.png
-/r * 67-128-2: 11.png
-/r * 72-128-2: deleted.txt.ntfs-3g-0000000002
-/r * 73-128-2: deleted.txt.ntfs-3g-0000000003
-/r * 80-128-2: deleted.txt.ntfs-3g-0000000006
-/r * 81-128-2: visible.txt.ntfs-3g-0000000005
-/r * 82-128-2: visible.txt
-/r * 88-128-2: deleted.txt.ntfs-3g-0000000007
-/d * 96-144-2: .Trash-0
d/d 256:        $OrphanFiles
```

icat – Display a File

- ➔ Output the contents of a file based on its inode number
- ➔ Usual calling parameters
    - ◆ r: recover deleted file
    - ◆ s: displays slack space at end of file

Sample output: Shows the deleted file and what is the information



```
root@kali:~/Desktop/ninja# icat -o 2048 forensic.dd 88
Namah shivaya. You cant see me.I'm deleted.
root@kali:~/Desktop/ninja# 
```

#here 88 represnts i-node number of the file

Incidence response - Recovering Deleted Files with the Sleuth Kit

----

Scenario - I have pendrive with two files with some confidential data.Mistakenly I have deleted one file and other file still remaining.I just want to see the information in the file and reovery back using sleuth kit tool.

The steps followed for any deleted files using sleuth kit tool to see and recover the deleted files.

Step 1: How to check the disks/drives/pendrives in the system(ubuntu)
 a)fdisk -l b)dmesg

Step 2: mmls - Display the partition layout of a volume (here pendrive)

Step 3: fsstat - displays the general details of a file system

Step 4: fls - List file and directory names in a disk image

Step 5: icat - Output the contents of a file based on its inode number.

Step 6: tsk_recover - Export files from an image into a local directory

Experiment: Following with the above steps

Case study : looking for deleted files & Recovery of the deleted files

Step 1:

Using fdisk -l

```
root@kali:~/Desktop/forensics# fdisk -l
Disk /dev/sda: 298.1 GiB, 320072933376 bytes, 625142448 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x000447ff

Device          Boot    Start    End   Sectors   Size Id Type
/dev/sda1  *            2048    718847          716800   350M  7 HPFS/NTFS/exFAT
/dev/sda2       718848 205522943 204804096  97.7G  7 HPFS/NTFS/exFAT
/dev/sda3       205522944 415238143 209715200   100G  7 HPFS/NTFS/exFAT
/dev/sda4       415240190 625141759 209901570 100.1G  5 Extended
/dev/sda5       415240192 423239679   7999488   3.8G 82 Linux swap / Solaris
/dev/sda6       423241728 625141759 201900032  96.3G 83 Linux



Disk /dev/sdb: 14.5 GiB, 15552479232 bytes, 30375936 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x000d6d04

Device          Boot Start       End  Sectors  Size Id Type
/dev/sdb1  *    2048 30375935 30373888 14.5G  7 HPFS/NTFS/exFAT
```

Alternative : dmesg

Output : In the terminal - at the end - you can see details about pendrive like below

```
[ 5836.496178] usb 3-2: USB disconnect, device number 2
[ 5838.145609] usb 3-2: new high-speed USB device number 3 using xhci_hcd
[ 5838.275000] usb 3-2: New USB device found, idVendor=0781, idProduct=5581
[ 5838.275009] usb 3-2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[ 5838.275014] usb 3-2: Product: Ultra
[ 5838.275018] usb 3-2: Manufacturer: SanDisk
[ 5838.275023] usb 3-2: SerialNumber: 4C531123451110123324
[ 5838.275738] usb-storage 3-2:1.0: USB Mass Storage device detected
[ 5839.301372]  sdb: sdb1
[ 5839.302927] sd 7:0:0:0: [sdb] Attached SCSI removable disk
```

Step 2:mmls

```
root@kali:~# mmls /dev/sdb
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors
```

```
root@kali:~/Desktop/forensics# mmls /dev/sdb
```

Output - Shows which kind of partition and offset of the disk etc.

```
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

      Slot        Start          End           Length       Description
000:  Meta      0000000000   0000000000   0000000001   Primary Table (#0)
001:  -------   0000000000   0000002047   0000002048   Unallocated
002:  000:000   0000002048   0030375935   0030373888   NTFS / exFAT (0x07)
```

# offset of the disk- 2048

Step 3: fsstat : Display detailed file system and metadata information of drive

----

```
root@kali:~# fsstat -o 2048 /dev/sdb
```

Ouput:

```
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: NTFS
Volume Serial Number: 82CC40D7CC40C75F
OEM Name: NTFS
Volume Name: KALI LIVE
Version: Windows XP

METADATA INFORMATION
--------------------------------------------
First Cluster of MFT: 786432
First Cluster of MFT Mirror: 2
Size of MFT Entries: 1024 bytes
Size of Index Records: 4096 bytes
Range: 0 - 256
Root Directory: 5

CONTENT INFORMATION
--------------------------------------------
Sector Size: 512
Cluster Size: 4096
Total Cluster Range: 0 - 3796734
Total Sector Range: 0 - 30373886

$AttrDef Attribute Values:
$STANDARD_INFORMATION (16)   Size: 48-72   Flags: Resident
$ATTRIBUTE_LIST (32)   Size: No Limit   Flags: Non-resident
$FILE_NAME (48)   Size: 68-578   Flags: Resident,Index
$OBJECT_ID (64)   Size: 0-256   Flags: Resident
$SECURITY_DESCRIPTOR (80)   Size: No Limit   Flags: Non-resident
$VOLUME_NAME (96)   Size: 2-256   Flags: Resident
$VOLUME_INFORMATION (112)   Size: 12-12   Flags: Resident
$DATA (128)   Size: No Limit   Flags:
$INDEX_ROOT (144)   Size: No Limit   Flags: Resident
$INDEX_ALLOCATION (160)   Size: No Limit   Flags: Non-resident
$BITMAP (176)   Size: No Limit   Flags: Non-resident
$REPARSE_POINT (192)   Size: 0-16384   Flags: Non-resident
```

```
        $EA_INFORMATION (208)   Size: 8-8   Flags: Resident
        $EA (224)   Size: 0-65536   Flags:
        $LOGGED_UTILITY_STREAM (256)   Size: 0-65536   Flags: Non-resident
```

Step 4: fls : You can see all list of files

```
root@kali:/tmp# fls -o 2048 /dev/sdb
```

Output : List of files on the File system

```
        r/r 4-128-4:    $AttrDef
        r/r 8-128-2:    $BadClus
        r/r 8-128-1:    $BadClus:$Bad
        r/r 6-128-4:    $Bitmap
        r/r 7-128-1:    $Boot
        d/d 11-144-4:    $Extend
        r/r 2-128-1:    $LogFile
        r/r 0-128-1:    $MFT
        r/r 1-128-1:    $MFTMirr
        r/r 9-128-8:    $Secure:$SDS
        r/r 9-144-11:    $Secure:$SDH
        r/r 9-144-5:    $Secure:$SII
        r/r 10-128-1:    $UpCase
        r/r 3-128-3:    $Volume
        r/r 64-128-2:    pcap Q
        r/r 65-128-2:    ReconCase.pcap
        r/r 82-128-2:    visible.txt
        -/r * 35-128-4:    ReadyBoostPerfTest.tmp
        -/r * 72-128-2:    deleted.txt.ntfs-3g-0000000002
        -/r * 73-128-2:    deleted.txt.ntfs-3g-0000000003
        -/r * 80-128-2:    deleted.txt.ntfs-3g-0000000006
        -/r * 81-128-2:    visible.txt.ntfs-3g-0000000005
        -/r * 88-128-2:    deleted.txt.ntfs-3g-0000000007
        -/d * 96-144-2:    .Trash-0
        d/d 256:    $OrphanFiles
```

Step 5: icat

Case 1:

In this pendrive(NTFS)- I have kept two files, 1)visible.txt and 2)delete.txt
And I have deleted the delete.txt and you can see both files as follows.

In visible.txt , you can see what was in the text file as below.

```
root@kali:/tmp# icat -o 2048 /dev/sdb 81        #comment - 81 represent i-node number
Namah shivaya. I should be visible
```

In delete.txt, you can see what was in the text file as below.

```
root@kali:/tmp# icat -o 2048 /dev/sdb 88        #comment - 88 represent i-node number
Namah shivaya. You cant see me.I'm deleted.    #comment - it is the deleted file content
```

Step 6: tsk_recover

You can recover deleted files using below comand

```
root@kali:/tmp# tsk_recover /dev/sdb1 /tmp/             # comment - /dev/sdb1 is pendrive
Files Recovered: 9

root@kali:cd /tmp/
root@kali:/tmp# ls                                  # comment - bolded names - deleted files
deleted.txt.ntfs-3g-0000000002
deleted.txt.ntfs-3g-0000000003
Deleted.txt.ntfs-3g-0000000006
deleted.txt.ntfs-3g-0000000007
$OrphanFiles
ReadyBoostPerfTest.tmp
ssh-xirmv6aauHKp
systemd-private-f7f047aa2e114fe9b183a81602aa9559-colord.service-WcLDRj
systemd-private-f7f047aa2e114fe9b183a81602aa9559-rtkit-daemon.service-jO34cO
tracker-extract-files.0
visible.txt.ntfs-3g-0000000005
VMwareDnD
```

```
-------------------------------------------------------------------------------------------------------------------
Output  : Recovered files
-------------------------------------------------------------------------------------------------------------------
root@kali:/tmp# cat deleted.txt.ntfs-3g-0000000002
Namah shivaya. You cant see me.I'm deleted. #comment - inside file data

root@kali:/tmp# cat visible.txt.ntfs-3g-0000000005
Namah shivaya. I should be visible                    #comment - inside file data
```

References

http://www.sleuthkit.org/

https://wiki.sleuthkit.org/index.php?title=TSK_Tool_Overview

https://en.wikipedia.org/wiki/The_Sleuth_Kit