

Digital Forensics
Lecture Week 9b

Disk Acquisition

Disk Partitions

File Systems

Nelson Chapters 3 and 5
Readings

Objectives

- To understand data acquisition principles
- To become familiar with disk acquisition tools
- To understand Partition details
- To review File Systems
- To be able to find hidden data on disks.

Forensic Acquisition (week 2)

- Working on a disk may require minor alterations to its contents.
- You need to prove these alterations are minor.
- Best to work on a **copy** of the disk.
- The copy can be to another, similar disk
- Or the copy can be to an **image file**
- Note that disk tools in a VM cannot see the host disk structure.

Disk Acquisition Options

- 1) We can physically extract the disk and copy it to another disk
- 2) We can boot from a USB or CD and copy the disk
 - We can make a disk to disk copy
 - or we can make a disk to file copy
- Either way we need to mount the evidence disk as read only and block writes from the OS

Forensics Tool Testing

- The US National Institute of Standards and Technology (NIST) test available forensic tools
- Their approval helps get a tool results accepted in court.
- <https://www.nist.gov/itl/ssd/software-quality-group/computer-forensics-tool-testing-program-cftt>

CFTT Technical Information

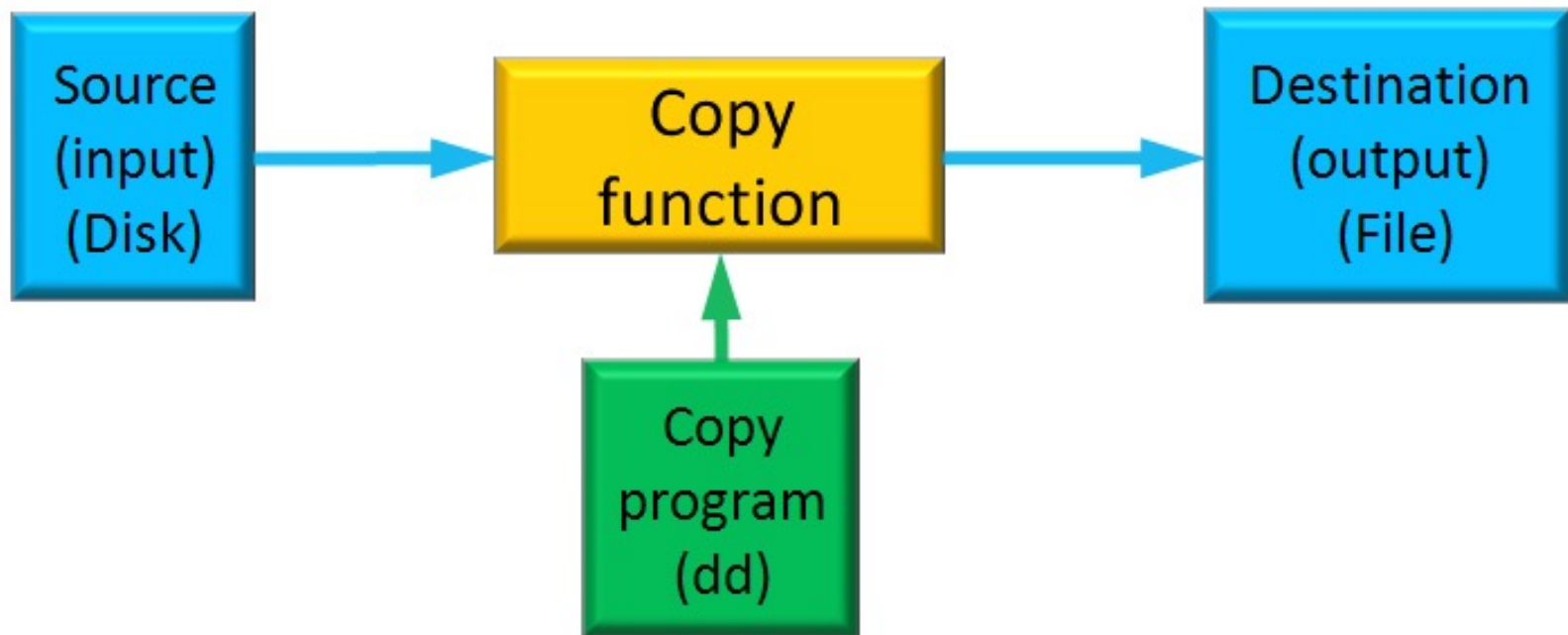
- [Disk Imaging](#)
- [Forensic Media Preparation](#)
- [Write Block \(Software\)](#)
- [Write Block \(Hardware\)](#)
- [Deleted File Recovery](#)
- [Mobile Devices](#)
- [Forensic File Carving](#)
- [String Search](#)
- [MS Windows Registry Tools](#)

Forensic Investigation Options

- 1) We can boot from the disk copy.
- 2) From our forensic laptop OS we can mount the disk.
- 3) We can open a VM and load the disk image file
 - <http://liveview.sourceforge.net/>

Disk Acquisition

- “collect now and examine later”



Objectives

- To understand data acquisition principles
- To become familiar with disk acquisition tools
- To understand how to image a disk
- To review File Systems
- To be able to find hidden data on disks.

Disk Acquisition tools

- We need a tool to capture a disk into a file.
- Forensics tools such as ProDiscover and Autopsy for Windows have built in disk acquisition.
- You can also use specialist tools
 - Access Data FTK Imager
 - <https://accessdata.com/product-download/ftk-imager-version-4-2-1>
- You can use the Linux data dump tool (dd)
 - This will not work on a VM such as Windows WSL.

Disk Acquisition

- We can capture the whole suspect's disk
- Need the same space free on the investigator's disk
- We can capture a partition (Volume)
- Can be much less time and space required
- Once we have the image we can examine it
- We use The Sleuth Kit (TSK) tools
 - Built into Linux

Data Dump - dd

- The default tool for **disk to file** copy
- Uses a raw, sector by sector, copy
- Copies text, binaries, hex files all with ease
- Will pick up hidden data
- **dd --help** will list the command options
- [http://en.wikipedia.org/wiki/Dd \(Unix\)](http://en.wikipedia.org/wiki/Dd_(Unix))
- <http://www.sleuthkit.org/informer/sleuthkit-informer-11.html>

dd variants

- dd - the original dd, part of GNU Core Utilities
- ddrescue - additional support for bad sectors
- dc3dd - support for forensics
 - includes on the fly hashing
- dcfldd – enhanced for govt use
- <http://www.linuxjournal.com/article/1320>

Objectives

- To understand data acquisition principles
- To become familiar with disk acquisition tools
- **To understand Partition details**
- To understand File Systems
- To be able to find hidden data on disks.

Partitions








- GUID UEFI Partitions
 - Most current Laptops
 - Use a GPT viewer such as GDisk
- BIOS MBR Partitions
 - Most USB Drives
 - Extract the first 512 byte block and use a hex viewer

Viewing the Partition table








- There may be hidden partitions so we look at the table
- We can see the live partition table
 - Windows Disk Manager
 - Or HXD Editor
- We can see the acquired image partition table
 - Extract the first two blocks with `dd` and view with `xxd`
 - Or View the first two blocks with the `HXD` Editor

Viewing the Live Partition Tables

- Windows Disk Manager

Volume	Layout	Type	File System	Status
 (Disk 0 partition 1)	Simple	Basic		Healthy (EFI System Partition)
 (Disk 0 partition 6)	Simple	Basic	NTFS	Healthy (Recovery Partition)
 Data (G:)	Simple	Basic	NTFS	Healthy (Basic Data Partition)
 Forensics (E:)	Simple	Basic	NTFS	Healthy (Primary Partition)
 Photos (H:)	Simple	Basic	NTFS	Healthy (Basic Data Partition)
 PYTHON (F:)	Simple	Basic	FAT32	Healthy (Primary Partition)
 Windows (C:)	Simple	Basic	NTFS	Healthy (Boot, Page File,

- HXD Editor

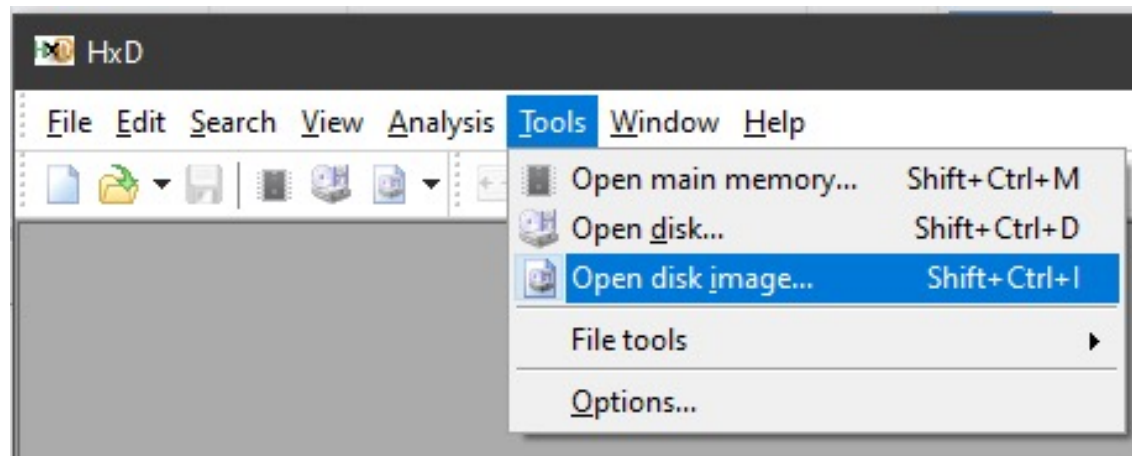
Inserted disks:			
Name ^	Type	Size	Hardware
▼ Logical disks			
 Windows (C:)	Hard disk	151.00 GiB	Hard disk 1
 Forensics (E:)	Removeable disk	20.00 MiB	Removeable disk 1
 PYTHON (F:)	Removeable disk	1E03 MiB	Removeable disk 1
 Data (G:)	Hard disk	51.50 GiB	Hard disk 1
 Photos (H:)	Hard disk	34.20 GiB	Hard disk 1
▼ Physical disks			
 Hard disk 1	Hard disk	238.00 GiB	HFS256G39TND-N210A
 Removeable disk 1	Removeable disk	7.20 GiB	TDK LoR Platinum 3.0

Viewing the Acquired Image USB BIOS Partition Table

- dd and xxd

```
group11/mnt/c/Forensics_Graham$ dd if=USB1.001 count=1  
of=USBSector3.dd  
1+0 records in  
1+0 records out  
512 bytes copied, 0.0024833 s, 206 kB/s  
|  
group11/mnt/c/Forensics_Graham$ xxd USBSector3.dd
```

- HxD



The MBR data structure

Structure of a master boot record

Address			Description		Size in bytes
Hex	Oct	Dec			
0000	0000	0	code area		440 (max. 446)
01B8	0670	440	disk signature (optional)		4
01BC	0674	444	Usually nulls; 0x0000		2
01BE	0676	446	Table of primary partitions (Four 16-byte entries, IBM partition table scheme)		64
01FE	0776	510	55h	MBR signature; 0xAA55	2
01FF	0777	511	AAh		
MBR, total size: 446 + 64 + 2 =					512

MBR Partition Table data structure

Fixed size. 16 bytes= 10 hex

Address	00	01	04	05	08	0C
Data	Boot status	CHS Start Address	Partition Type	CHS End Address	LBA Start address	Sector count

00 = not bootable
80=bootable

Cylinder, Head, Sector format

Logical Block Addressing

4 Bytes
=32 bits

Partition Types

- 07 NTFS
- 0B FAT32 CHS
- 0C FAT32 LBA
- 0E FAT16 LBA
- 0F Extended
- -----
- 82 Linux swap
- 83 Linux native partition (EXT4)

Sample MBR

Note the four **partitions** starting at 0x01BE(i.e., 446)

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000																	
00000010																	
00000020																	
00000120	32	E4	8A	56	00	CD	13	EB	D6	61	F9	C3	49	6E	76	61	Inva
00000130	6C	69	64	20	70	61	72	74	69	74	69	6F	6E	20	74	61	lid partition ta
00000140	62	6C	65	00	45	72	72	6F	72	20	6C	6F	61	64	69	6E	ble.Error loadin
00000150	67	20	6F	70	65	72	61	74	69	6E	67	20	73	79	73	74	g operating syst
00000160	65	6D	00	4D	69	73	73	69	6E	67	20	6F	70	65	72	61	em.Missing opera
00000170	74	69	6E	67	20	73	79	73	74	65	6D	00	00	00	00	00	ting system.....
00000180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001B0	00	00	00	00	00	2C	44	63	B2	B3	B2	B3	00	00	80	01,Dc ²³²³ ..€.
000001C0	01	00	07	FE	FF	FF	3F	00	00	00	B2	8C	7F	02	00	00	...pÿÿ?... ² €....
000001D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000001F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AAU ²

Error msgs start at 012C
 Partition type at offset 02
 Boot status at 0E
 MBR signature

Viewing a Partition (Logical Disk)

- Drive Letters in Windows (C: D: etc)
- Live Partition - use fsutil

```
C:\Users\graha>fsutil fsinfo volumeinfo e:  
Volume Name : Forensics  
Volume Serial Number : 0xf0039552  
Max Component Length : 255  
File System Name : NTFS  
Is ReadWrite
```

- Image Partition – use fsstat

```
$ fsstat USBFolder.001  
FILE SYSTEM INFORMATION  
-----  
File System Type: NTFS  
Volume Serial Number: F8F003D1F0039552  
OEM Name: NTFS  
Volume Name: Forensics  
Version: Windows XP
```

Objectives

- To understand data acquisition principles
- To become familiar with disk acquisition tools
- To understand Partition details
- To review File Systems
- To be able to find hidden data on disks.

File Systems

- Able to store, retrieve and update data
- Manage storage allocation
- Allocate and track file and directory names
- File and directory meta data
 - file size
 - file time stamps
 - file permissions
 - file owner
- Utilities to create, delete and rename files
- https://en.wikipedia.org/wiki/File_system

File Systems #2

- FAT32 32bit File Allocation Table
 - simple, used on USB, phones, cameras
 - 4GB File size limit
- NTFS from Windows New Technology File System
 - ACL permission control
 - encryption using EFS (Encrypting File System)
 - compression
 - quotas
 - Linux mount points

FAT32 Layout

- Sector 0 – Boot Sector
- Sector 1 – File System Information
- Sector 6 – Backup of Sector 0
- FAT Tables (2 for redundancy)
 - contain file pointers to data sectors
 - variable length, can be many sectors
- Data Region – all the rest of the partition

Contents	Boot Sector	FS Information Sector (FAT32 only)	More reserved sectors (optional)	File Allocation Table #1	File Allocation Table #2	Data Region (for files and directories) ... (To end of partition or disk)
Size in sectors	(number of reserved sectors)			(number of FATs)*(sectors per FAT)		NumberOfClusters*SectorsPerCluster

FAT sizes

- USB partitions may use FAT32
 - Minimum disk is 320MB
 - Maximum disk is 32GB
- Small partitions may use FAT16
 - Minimum disk is 16MB
 - Maximum Disk 2GB

NTFS

- New Technology File System
- Only Windows can format as ntfs (licence)
- Linux systems can read and write to ntfs
- Version 1 - 1993
- <http://en.wikipedia.org/wiki/NTFS>
- A good reference at <http://ntfs.com>


NTFS

- Large file systems, currently 256 TB files
- Reliability – aimed at file server use
- Recoverability – journaling file system
- Multiple copies of the Master File Tables (MFT) on disk
- Hot fixing – if a bad sector is detected, the data is silently moved and the sector marked as bad

NTFS features

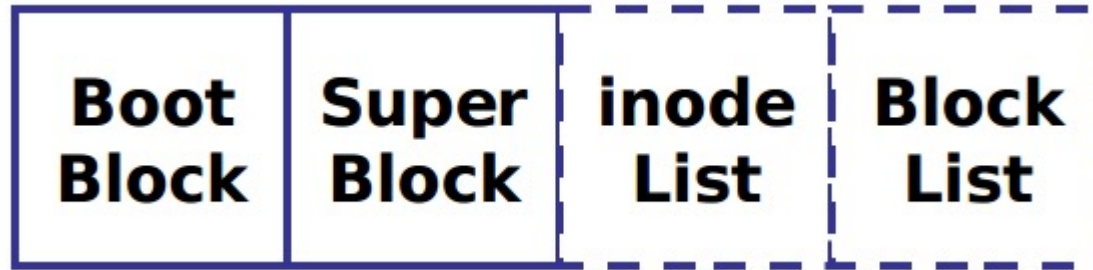
- Update Sequence Number (USN) Journal
 - records changes to files
- ADS – Alternate Data Streams
 - for MacOS fork support
 - used by malware to hide code
 - do not copy to FAT32 partitions
- VSS – Volume Shadow Copy
 - allows a backup of a locked file
- MFT – Metadata File Tables
- File name starts with a \$. See next slide

NTFS Metafiles

Segment Number 	File Name
0	\$MFT
1	\$MFTMirr
2	\$LogFile
3	\$Volume
4	\$AttrDef
5	.
6	\$Bitmap
7	\$Boot
8	\$BadClus
9	\$Secure
10	\$UpCase

- Define and Organise the file system
- Hidden from the user
- \$MFT – master list of all file names
- \$LogFile- NTFS Log – a transactional system to allow rollback of meta data changes
- \$Volume – Volume description
- \$Boot – Volume boot record
- \$BadClus – a list of known bad clusters
- \$Secure – the ACL database
- \$UpCase – Uppercase name version

Linux File System on disk



- Boot block contains the bootstrap code as in Windows
- Superblock contains the disk metadata
- Inode contains a pointer to a data block
- Data block of 512 byte sectors

fdisk on Linux - not wsl

```
root@kali:~# fdisk -l
```

```
Disk /dev/sda: 21.5 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders, total 41943040 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
Disk identifier: 0x000dff5c
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	2048	40136703	20067328	83	Linux
/dev/sda2		40138750	41940991	901121	5	Extended
/dev/sda5		40138752	41940991	901120	82	Linux swap


Default Linux file system - Ext4

- Kernel 2.6.28 was released with
 - 25 Dec 2008 the ext4 filesystem
- Used on Android from version 2.3
- Easy upgrade from Ext3 and Ext2
- Can handle very large files, 16 TB
- Faster as joins contiguous blocks as an Extent
- Journal checksums for reliability
- Good for SSD

file on Linux – no sL on wsl

```
root@kali:~# file -sL /dev/sda1
```

```
/dev/sda1: sticky Linux rev 1.0 ext4 filesystem data,  
UUID=8a833949-3596-4c15-932b-0573f630307c  
(needs journal recovery) (extents) (large files) (huge files)
```



- In a VM, some file system features are simulated

Objectives

- To understand data acquisition principles
- To become familiar with disk acquisition tools
- To understand Partition details
- To review File Systems
- To be able to find hidden data on disks.

Viewing FAT32 image file detail with fls

- Run fls against the volume image.
- fls USBFolderx

r/r = file

d/d = folder

* = deleted file

```
$ fls USBFolder2.001
r/r 3:  PYTHON      (Volume Label Entry)
d/d 6:  System Volume Information
r/r * 9:           Trade_Secrets.txt
r/r 11: Sample.pdf
...
r/r 17: MS Office Meta Data.jpg
r/r * 18:          _s.exe
r/r * 19:          _s2.exe
r/r * 20:          _ogo.gif
r/r 21: test
r/r 22: test1
...
r/r 28: Flowers.txt
v/v 32636931:  $MBR
v/v 32636932:  $FAT1
```

Viewing NTFS image file detail with fls

- Run fls against the volume.
- fls USBFolderx

r/r = file

d/d = folder

-/r = deleted file

```
group11/mnt/c/Forensics_Graham$ fls USBFolder.001
r/r 4-128-1:    $AttrDef
r/r 8-128-2:    $BadClus
...
r/r 50-128-1:   strings.exe
d/d 36-144-1:   System Volume Information
r/- * 0:        Trade_Secrets.txt
-/r * 39-128-3: Trade_Secrets.txt
-/r * 45-128-3: logo.gif
-/r * 46-128-1: ls2.exe
d/d 256:        $OrphanFiles
|
```

Recovering deleted files

- Run **fls** against the Volume folder.
- Note the **inode** number for the deleted file.
- Run **icat** against the inode.

```
group11/mnt/c/Forensics_Graham$ icat USBFolder.001 39-128-3  
A trade secret is a formula, practice, process, design, instrument,  
pattern, or compilation of information which is not generally known  
or reasonably ascertainable, by which a business
```

FIN