Disk Imaging Guide

Tom Ensom

Time-based Media Conservation, Tate

Document Last Updated: January 2021

Document Version: 01.00

Contents

1 Document Background

This document is a guide to workflows and tools for the disk imaging of storage media, intended for conservators of time-based media artworks. Disk imaging is a technique of growing importance in time-based media conservation, which allows conservators to create files which contain bit-for-bit captures of storage media such as computer hard disk drives, flash storage devices and optical media (e.g. CD-ROMs and DVDs). This guide was created as part of Tate's Software-based Art Preservation Project, as an introduction to the topic and as a way of standardising Tate's internal workflows.

This document was intended for internal use at Tate and being shared more widely with several caveats. As it was created primarily for internal use by Tate's Time-based Media

Conservation team, it contains references to other internal documentation which may not be clear to an outside party. The practical aspects of this document assume that the user has access to a Linux Workstation running a recent version of the BitCurator environment or an equivalent installation of Ubuntu 18.04 with forensic imaging related packages installed. It is also important to note that this a snapshot of a living document which resides on Tate's Time-based Media Conservation Wiki, captured on the date noted at the top of this document. The wiki-based document will continue to evolve alongside other internal workflows, policies and procedures.

This document builds on existing work in the fields of time-based media conservation and digital preservation, and a list of useful resources and readings is provided in Section 7.

2 Introduction to Imaging

Disk imaging is the process of creating disk images: files which encapsulate the structure and content of a digital storage medium. Disk images can be created from optical media (e.g. CD-ROMs and DVDs), magnetic media (e.g. hard disk drives), and flash storage (e.g. solid-state drives, flash drives and SD cards). Disk images are created by the Time-based Media Conservation team for various purposes:

- Capturing a bit-for-bit copy of the contents of acquired media for preservation purposes
- Accessing and analysing contained digital materials
- As a base for emulation and virtualisation
- As a base for cloning to new storage media

Imaging optical media and drive media (magnetic disk and solid-state drives) involve different workflows and tools, which are covered in Section 3 and Section 4 respectively. Image quality control, which follows from image acquisition, uses a unified workflow describe in Section 5.

Imaging should be used with care as it can generate very large quantities of data, particularly when generating raw images which may include empty space and deleted content. The most common use case is imaging of computer hard drives, which may have complex structures involving boot sector data and partitioning, the integrity of which is critical for preservation. The value of imaging in other scenarios should be identified on a case-by-case basis by considering the structure and content of the disk and considering whether there is any benefit to maintaining this structure in preservation copies. For example, imaging an installation disc for an operating system is a good use case, as the ISO image created could be used to install the operating system in the future. An SD card intended for a media player, on the other hand, presents a weaker use case for imaging, as imaging offers little added value beyond what would be provided by simply copying the data off the card.

3 Optical Media Imaging

This section discusses the imaging of optical media such as data CD, data DVD and Blu-ray discs. Optical media which *cannot* be imaged include audio CD (CD-DA) and LaserDisc discs.

3.1 Optical Disc Structures

Data is written onto optical discs as a spiral of data from the centre to the outer edge. This spiral is divided into sections called sessions (typically only one), which can contain one or more tracks. Tracks are made up of sectors of data which are typically 2048 bytes in size.

It is essential that prior to imaging the structure of the optical disc is identified, as this dictates the approach you will use. The important things to identify are:

- The number of sessions on the disc
- The number of tracks on the disc
- Whether these sessions and tracks contain data or audio

There are various Linux tools that can be used to probe disc contents, the usage of which is described further in the Structural Analysis section, including:

- disktype
- cd-info
- cdrdao disk-info

3.2 Optical Drives

Several TiBM workstations are equipped with optical drives suitable for imaging:

- Internal ASUS 16x BW-16D1HT BD-RE 16X drive installed on dual-boot Windows 10 / BitCurator workstation (Workstation 5)
- Internal LiteOn IHAS124-14 24x DVD-RW drives installed on the Windows 10 (Workstation 3) and BitCurator (Workstation 4) workstations

External drives are also available:

- External USB Lacie drive
- External USB ASUS 16x BW-16D1HT BD-RE 16X drive
- Apple SuperDrive (see note below)

Apple SuperDrive's will not work with Linux without manually unlocking them:

1. Install SCSI utilities: `sudo apt-get install sg3-utils`
2. Attach the SuperDrive
3. Identify the drive name using `lsblk`
4. Send the unlock byte sequence to the identified drive: `sg_raw /dev/sr# EA 00 00 00 00 00 01`

Write-blockers are not required when imaging optical media, as a very specific sequence of actions is required to write data onto optical discs, where they are writable at all.

3.3 Optical Image Formats

There are two image formats of use in imaging optical media: ISO and BIN/CUE. For discs containing a single data track, an ISO image should be created. An ISO image can only contain

a single data track, so for discs with more than one track or session, a BIN/CUE image should be created. ISO images can be created from the data tracks acquired with the BIN/CUE files.

3.3.1 ISO

The ISO format is used to capture a disk image of a single track of data from an optical disc. ISO is not a file format per se, but rather a naming convention indicative of a optical disc image containing an optical disc filesystem (usually ISO 9660). ISO files are widely recognised by operating systems, as they are a common method of distributing software over the web. Most imaging tools are capable of creating ISO files, although some (e.g. Guymager) may not use the .iso extension in the resulting image file.

3.3.2 BIN/CUE

The BIN/CUE format is used to capture disk images of the entire contents of an optical disc. A BIN/CUE image consists of two files for each session contained on the disc:

- A CUE (or sometimes TOC) file which captures the table of contents for that session, indexing the tracks contained.
- A BIN file which is a raw binary dump of all the data in a session.

BIN/CUE files are less convenient than ISO files for general access purposes (operating systems do not recognise them automatically) but present the only means of capturing the entirety of multi-track or multi-session discs. The tool cdrdao can be used to capture BIN/CUE files (note: cdrdao uses the TOC table of contents format instead of CUE).

3.4 Optical Media Imaging Workflow

Note: this workflow assumes you are using a Linux workstation with an Ubuntu 18.04 version of BitCurator installed (e.g. Workstation 4 or 5).

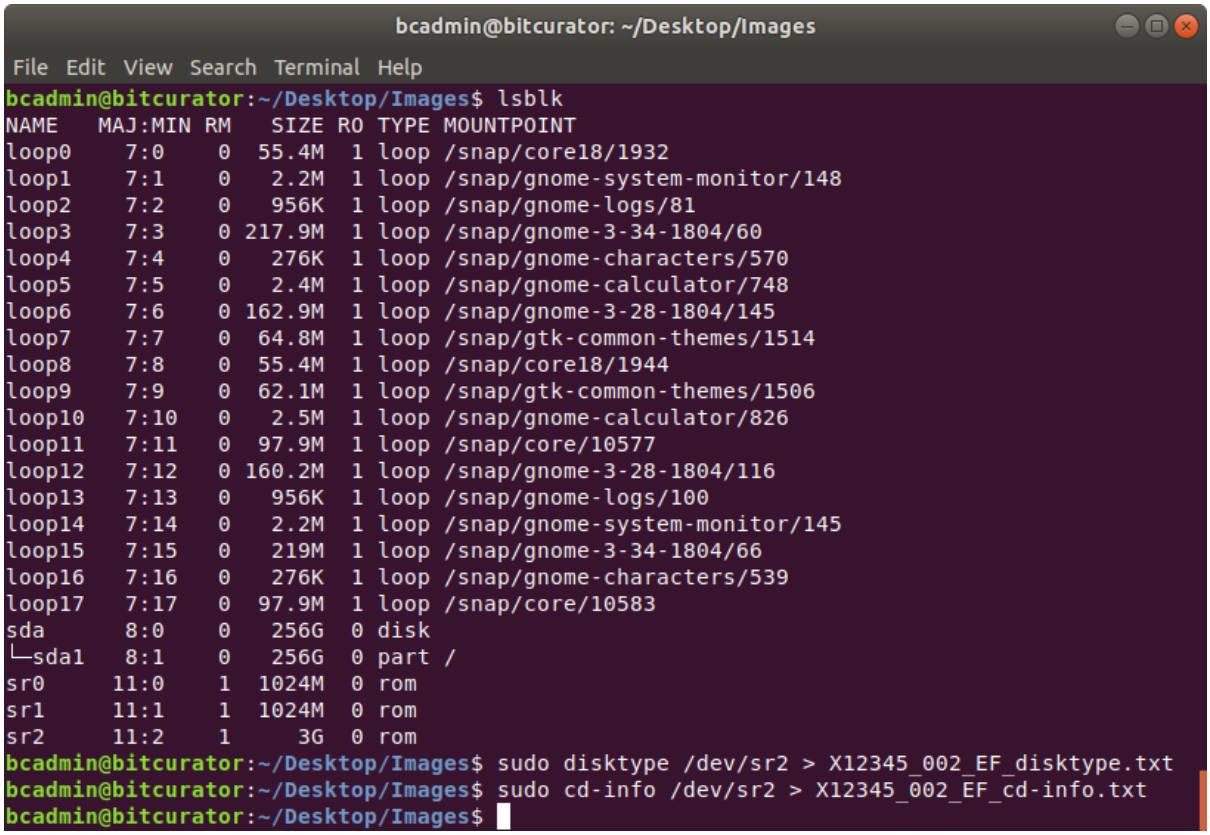
1. Prepare a Disk Imaging Report using the template (T:\Conservation\7 Policy and procedures\7 10 Templates\TiBM\02_MEDIA\DiskImaging_Forms) and complete it as you carry out the following steps.

2. If using an external drive, connect the drive to the host computer. Use of a write-blocker is not necessary when imaging optical media as a very specific sequence of actions is required to write data onto optical discs, where they are writable at all.

3. Insert the disc to be imaged into the drive.

4. Identify the device name given to the optical drive by Linux using `lsblk`. Optical drive device names are in the format /dev/sr#, where # is a number (starting from 0) assigned by Linux.

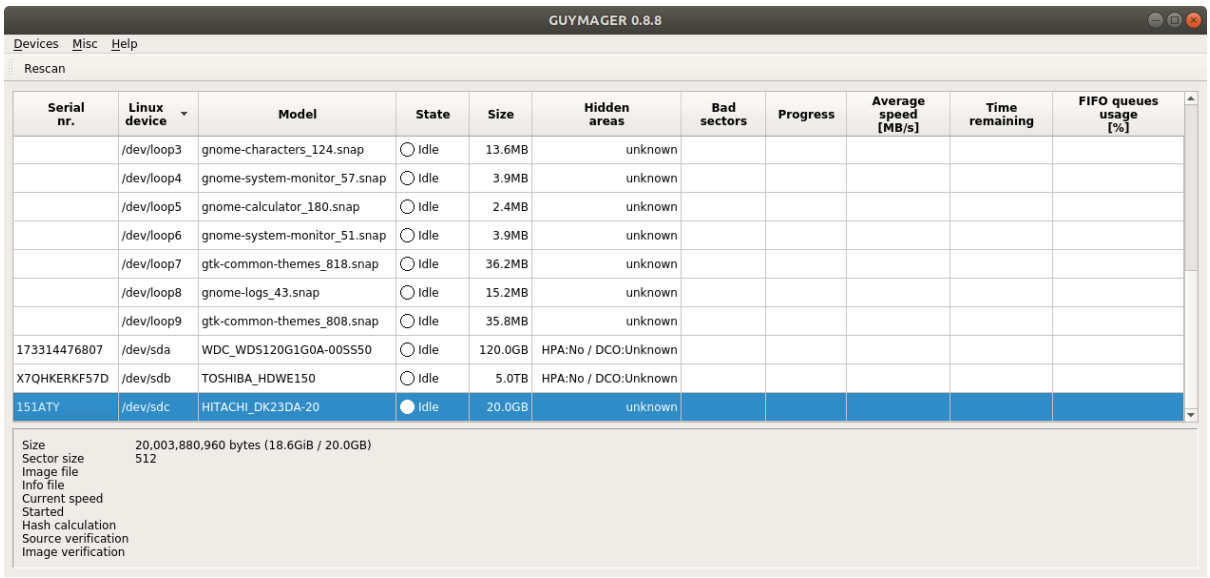
5. Use `disktype` and `cd-info` to identify the structure of the optical disc and save their output to text files in the artwork folder under Structure_and_Examination/Media/Disk_Imaging.



6. Depending on the structure of the disc identified in the previous step, choose from one of the following options:

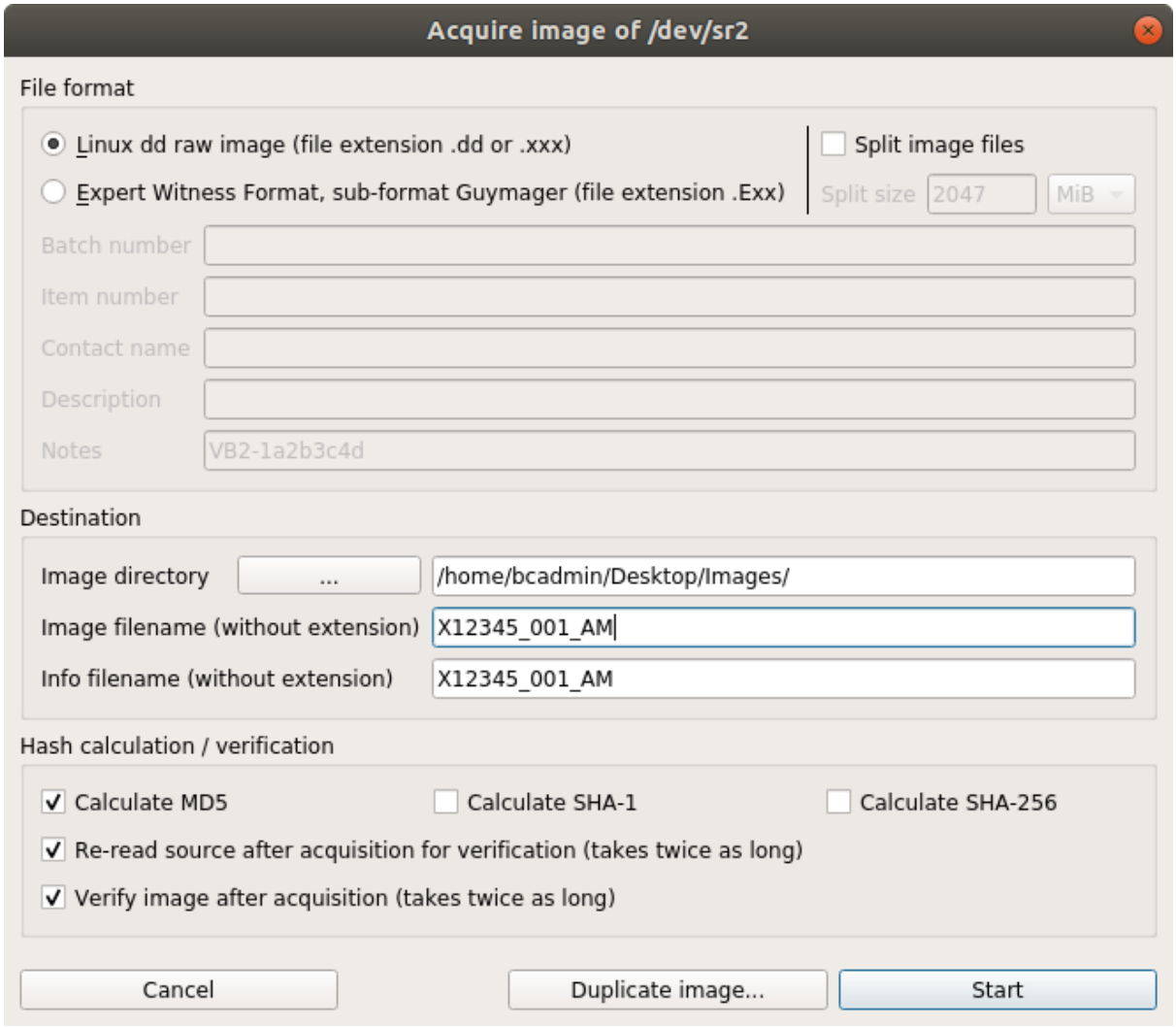
6.1. The disc contains a **single data track**:

6.1.1. Open Guymager and locate the optical drive in the list.

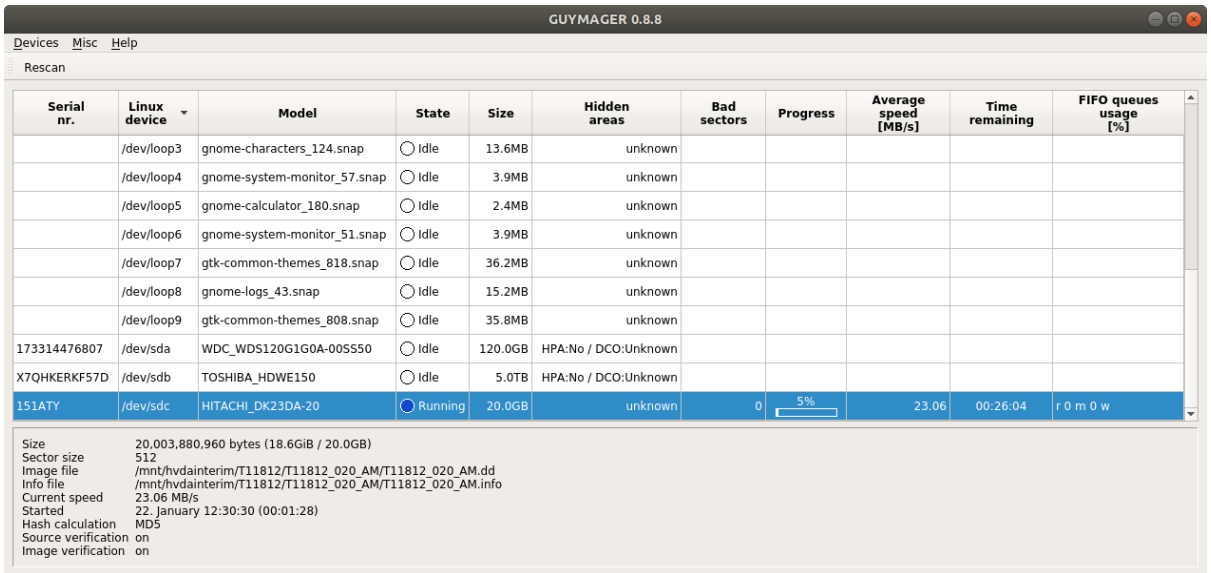


6.1.2. Right click the entry and select 'Acquire image'.

6.1.3. Select the following options and enter suitable filenames, ensuring that the re-read source checkbox is checked.

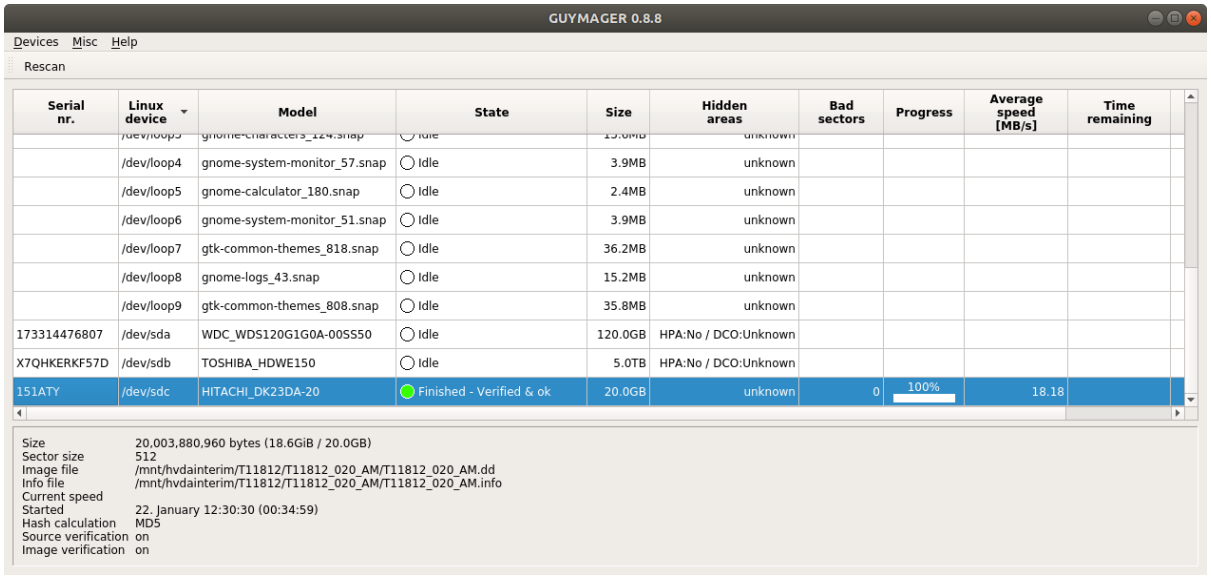


6.1.4. Wait for imaging to complete.



6.1.5. When imaging is complete, ensure that the verification status of the image is OK and no bad sectors have been encountered. If either of these

problems has been encountered, see Optical Media Imaging Troubleshooting. A plain text log file is generated alongside the image by Guymager, which includes the checksum value generated, and can be saved in the artwork folder under Structure_and_Examination/Media/Disk_Imaging.



6.1.6. Before archiving the file, change the extension of the resulting .dd file to .iso to indicate that this is an optical disc image.

6.2. If the disc is a **multi-track or multi-session data disc**, use the commands detailed in the cdrdao section to create a BIN/TOC image and convert data tracks to ISO.

6.3. If the disc is an **audio disc (CD-DA)**, use an appropriate workflow and tools for CD-DA audio ripping (e.g. ExactAudioCopy).

7. Apply image quality control procedures (see Image Quality Control).

3.5 Optical Media Imaging Troubleshooting

3.5.1 Optical Read Errors

When imaging optical media, read errors are common and checksum validation should be applied when imaging to ensure accurate capture. When using Guymager (with source re-verification enabled), the interface will clearly indicate that image verification has failed or when bad sectors have been encountered (which can happen even when verification has been successful). If using another tool, you may have to manually create and validate checksums.

Where bad sectors / read errors are encountered during imaging or checksum validation fails, various approaches can be taken to address this:

- If the disc is dirty, clean it using isopropyl alcohol and a microfiber cloth.
- Try a different model of optical drive and see if imaging gives the same result. This is particularly relevant where read errors are not identified by imaging software, but

source-image checksum verification fails, as the in these cases the optical drive being used may not be identifying or correctly reporting a read error.

- Try using specialised imaging software such as ddrescue (see usage guidance) or dvdisaster. These tools can be used to attempt recovery of bad sectors by repeatedly re-reading the affected region of the disc.
- If the disc is badly scratched, it may be possible to use polishing techniques to smooth off scratches from the bottom layer of plastic. Approaches vary from the use of automated machines to general purpose plastic polishing liquids. However, disc polishing should only be used as a last resort, as it risks further damage to the disc.

Note that with damaged or degraded media, it may be difficult to tell how the outputs of different tools compare in terms of completeness. In order to check this you can compare image files in a hex diffing program (e.g. BeyondCompare is able to handle large files). This will tell you how much data they have retrieved as opposed to blank/zeroed regions. However, presence of data does not guarantee that it is error free or meaningful; recovered images should also be subjected to QC procedures (see Image Quality Control).

3.5.2 Audio Discs

It is not possible to create valid disk images of Compact Disc Digital Audio (CD-DA) audio CDs. Audio discs are very prone to read errors due to the format's preference for error concealment over error correction. Tools such as ExactAudioCopy can employ features of the drive (if available) to help ensure an accurate read of the data, but associated workflows are not discussed further in this document.

4 Hard Drive and Flash Drive Imaging

This section discusses the imaging of drive media such as magnetic hard disk drives, solid state drives and USB sticks.

4.1 Drive Interface Types

Drive storage can come with a variety of interfaces, which may require the use of adaptors for imaging. When disk imaging drives it is important to use write-blocking to prevent data from being written to the attached drive. See the Write-blockers page for more information on the write-blockers available at Tate.

Some commonly encountered drive interfaces are:

- IDE/PATA (44 and 40 pin variants)
- SATA/AHCI
- SAS
- Parallel SCSI
- M.2 / PCI-E (NVME and SATA variants)
- USB (for external drives)
- FireWire 400/800 variants (for external drives)

4.2 Drive Data Structures

Drives, whether they are magnetic disk or flash memory based, are divided into units of storage called sectors. The physical size of the sectors on disk varies but are usually treated as if they are 512 bytes in size regardless.

When analysing the structure of a hard drive, the key information to identify is:

- Is it bootable? Does it use a BIOS or UEFI type boot system?
- What is the partition structure? Is this stored in an MBR, APT or GPT partition table?
- What filesystems are present in the drive's partitions?

This information can be gathered using various software tools. While disktype is a reliable choice in most cases, the other tools listed below present similar information slightly differently and may also be useful:

- disktype: command-line tool which describes the structure of a storage medium.
- GNOME Disks: GUI tool built into BitCurator/Ubuntu (in the OS is labelled *Disks*).
- mmls: command-line tool with similar output to disktype, presented as a table.
- fdisk: command-line tool useful for identifying the sector size of the drive (e.g. when using ddrescue).

Although these tools can be applied to devices or images, for the most accurate information you should apply them prior to imaging, directly on the source drive.

4.3 Drive Image Formats

4.3.1 RAW

A RAW disk image (sometimes called a 'dd' image and given the .dd extension) is a bit-for-bit copy of a storage medium, capturing all data found on the physical disk (including the partition table and any empty space). RAW is not a file format per se as the data captured is neither transformed nor wrapped in any way when a RAW image is created, rather it is a convention for referring to raw binary dumps of storage volumes.

4.3.2 EWF

The Expert Witness Disk Image Format (EWF, also known a EnCase E01) is a disk image format that contains a raw bit-for-bit copy of a physical disk (i.e. a RAW image as described above), but additionally incorporates metadata about the disk image and additional CRC error detection features. This is a proprietary format maintained by the commercial digital forensics company Guidance Software, although there is a reverse-engineered open specification available as part of the libewf project. We currently do not create drive images in the EWF format due to the proprietary nature of the specification, but the format is in use at other museums and archives, so is described here for information.

4.4 Drive Imaging Workflow

Before beginning, make sure you have prepared an appropriate work space. When opening or disassembling a computer to access media it is important to minimise risks to the data, equipment and yourself. Before you begin make sure you have disconnected the computer from any power source and have plenty of space to remove screws and components. To minimise danger from static discharge we have antistatic equipment that may be appropriate to use when working with certain media, particularly internal drives with exposed circuits. This equipment includes:

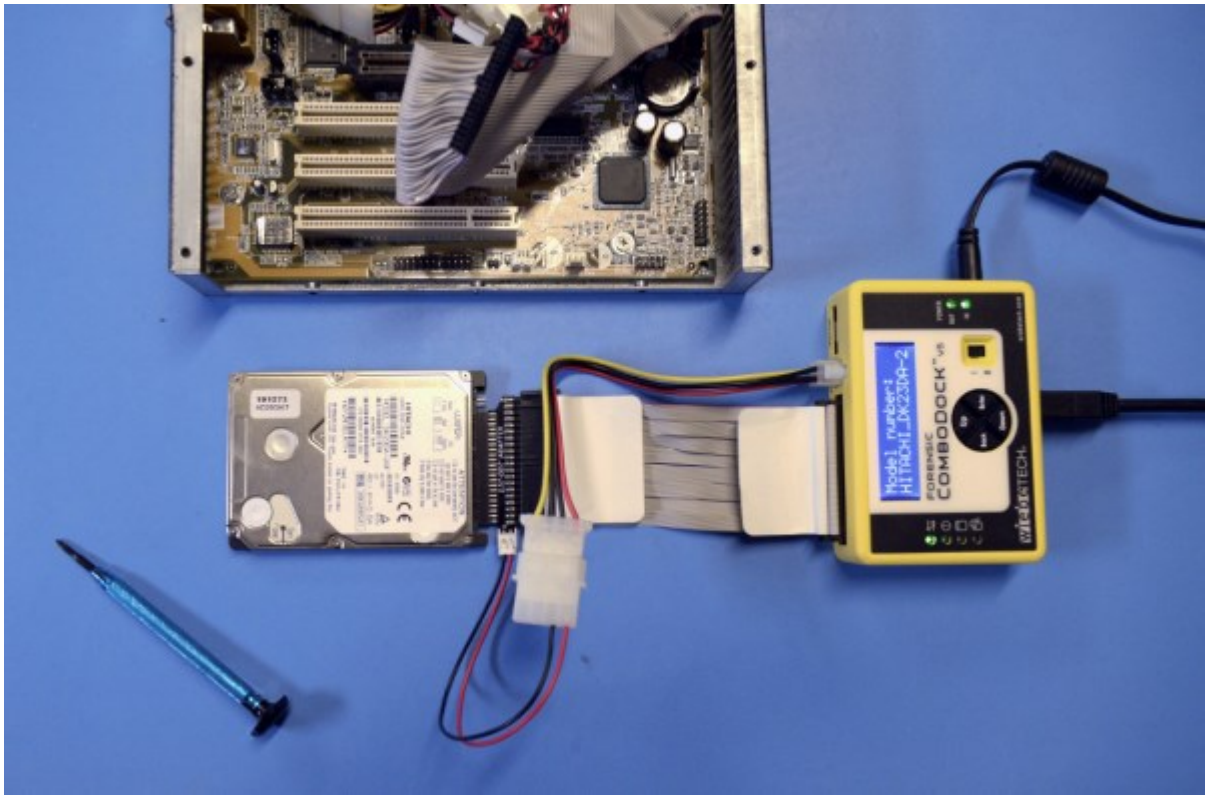
- antistatic work mat
- antistatic wrist band
- antistatic gloves (nitrile gloves can also be used as they are naturally anti-static)

Note: this workflow assumes you are using a Linux workstation with an Ubuntu 18.04 version of BitCurator installed (e.g. Workstation 4 or 5).

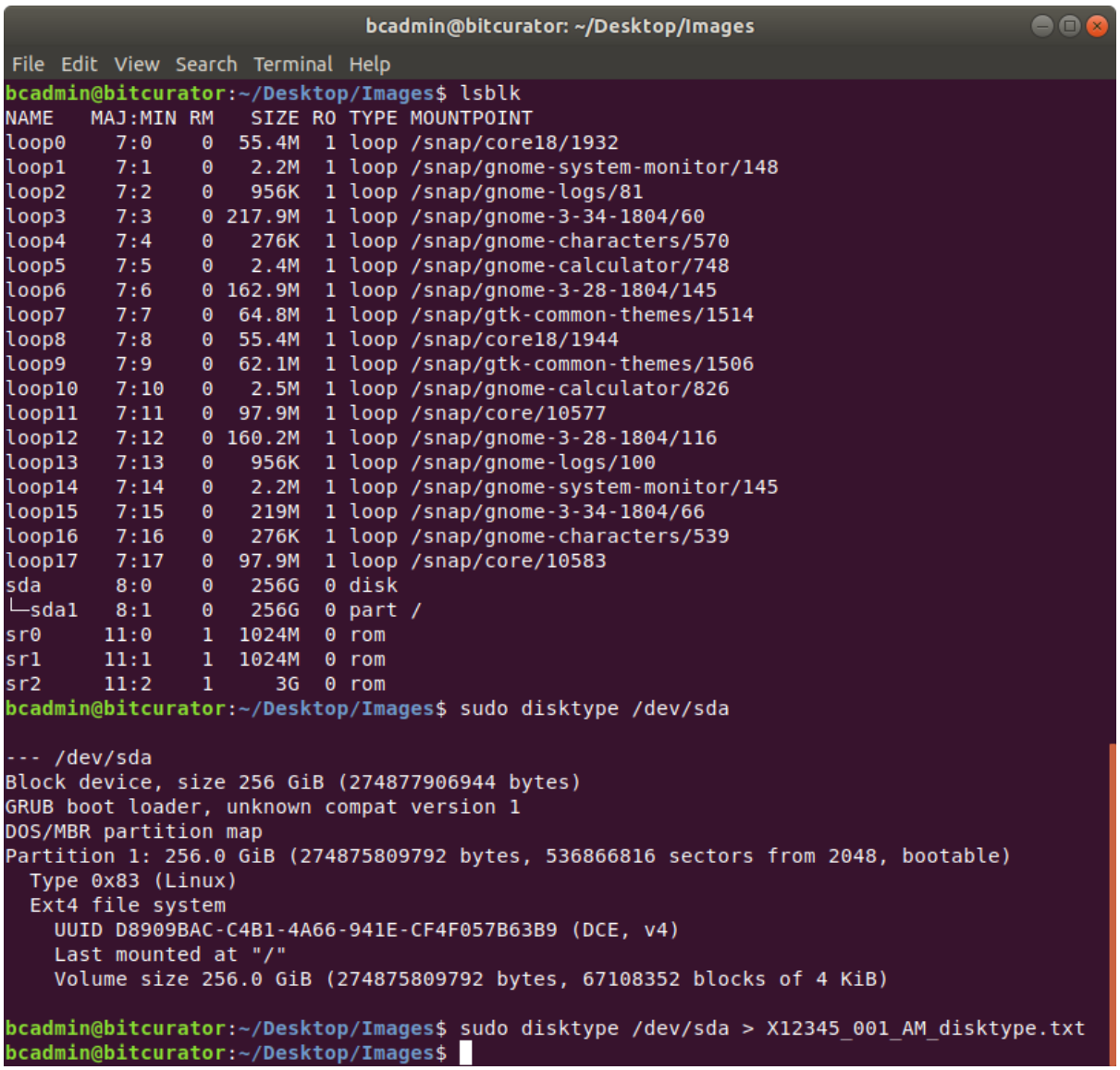
1. Prepare a Disk Imaging Report using the template (T:\Conservation\7 Policy and procedures\7 10 Templates\TiBM\02_MEDIA\DiskImaging_Forms) and complete it as you carry out the following steps.

2. If working with a computer system, identify the number of storage devices within that system and carefully remove them for imaging. You should assign component numbers for the images you will be generating in advance so that you can label them appropriately as you create the files.

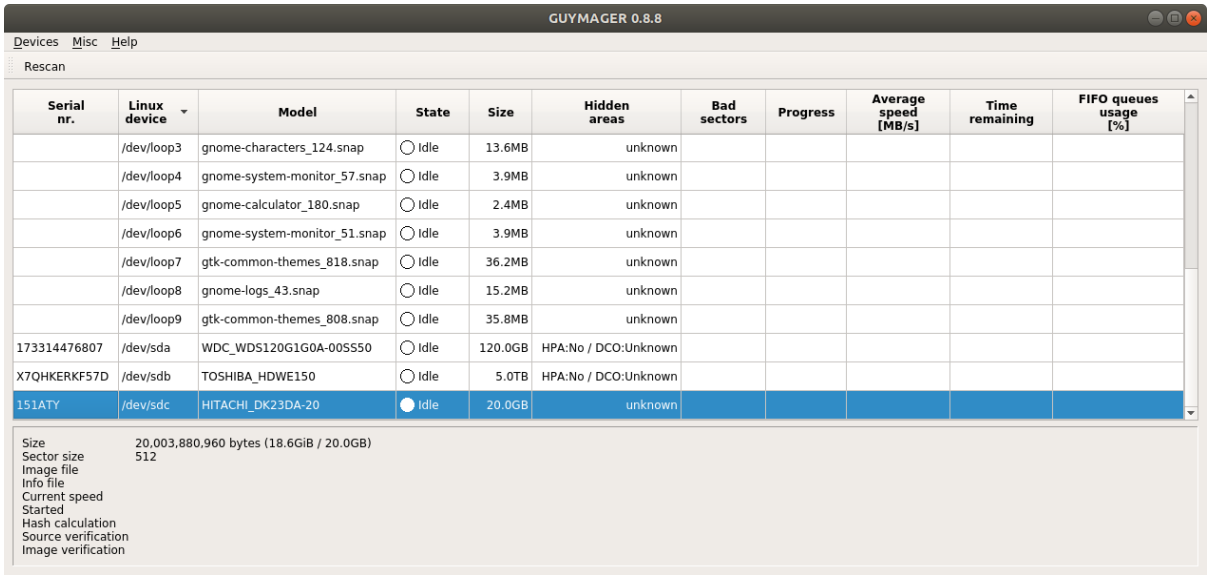
3. Attach the storage device to a suitable write-blocker and connect this to the computer with which you will be imaging. Ensure the drive is receiving power if necessary.



4. Open Terminal and use the command-line program `disktype` to identify and record the structure of the disk. Save the disktype output as a plain text file in Structure_and_Examination/Media/Disk_Imaging in the artwork folder. To use disktype, you will need to identify the Linux device name of the connected storage device. This can be done using the `lsblk` command or using Guymager itself (see next step). Ignore devices with the /loop/ prefix as these are virtual devices.

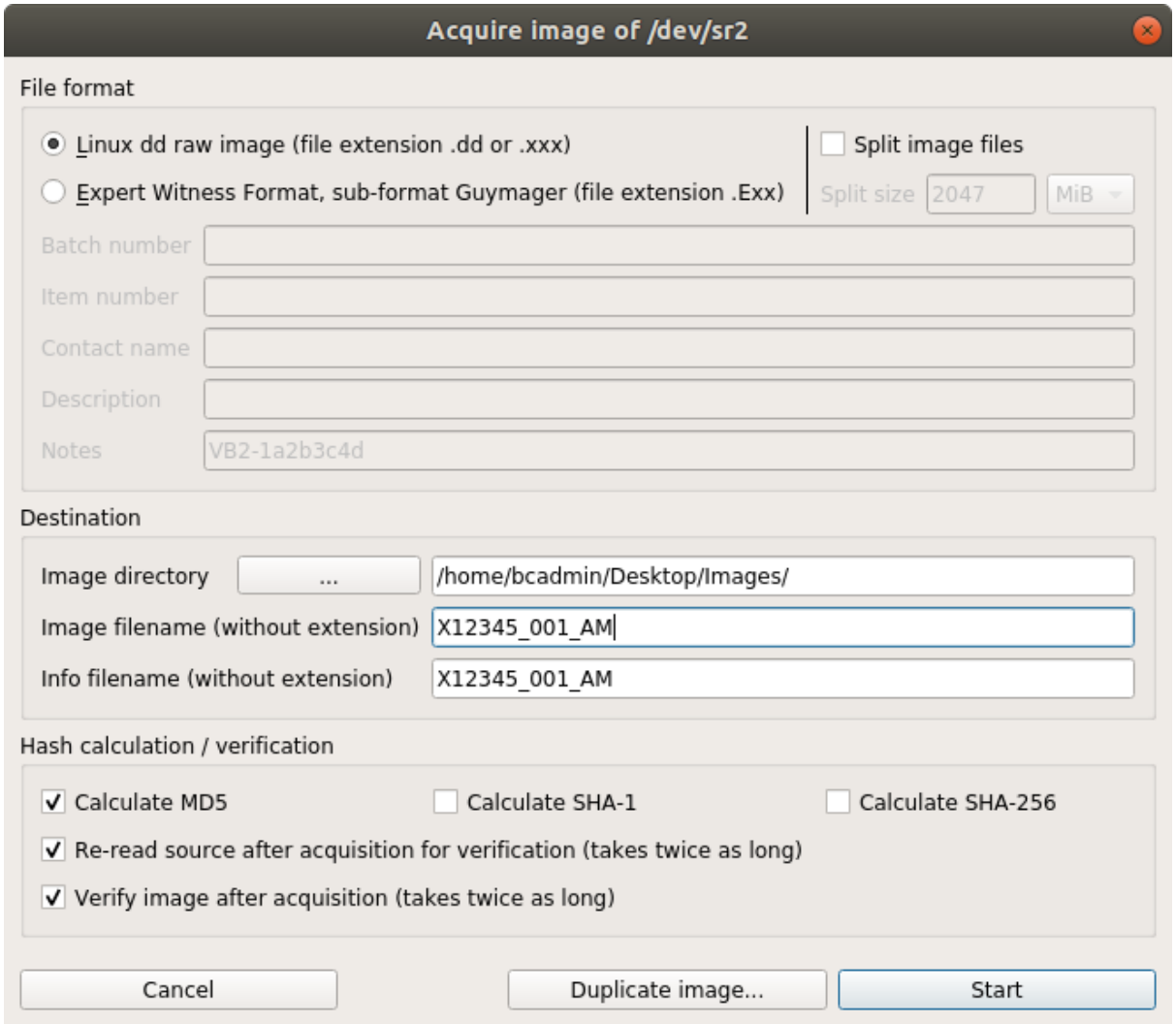


5. Open Guymager and locate the attached storage device in the list. To do so, you can use the device name identified in the previous step and known information about the attached drive (e.g. model, size).

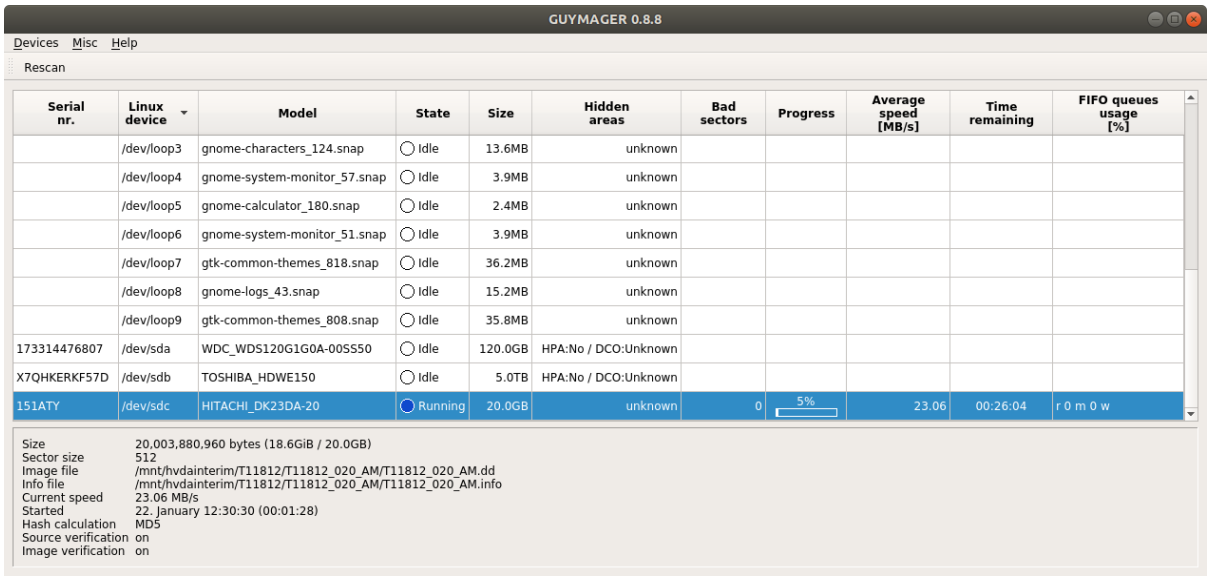


6. Right click the device in the list and select 'Acquire image'.

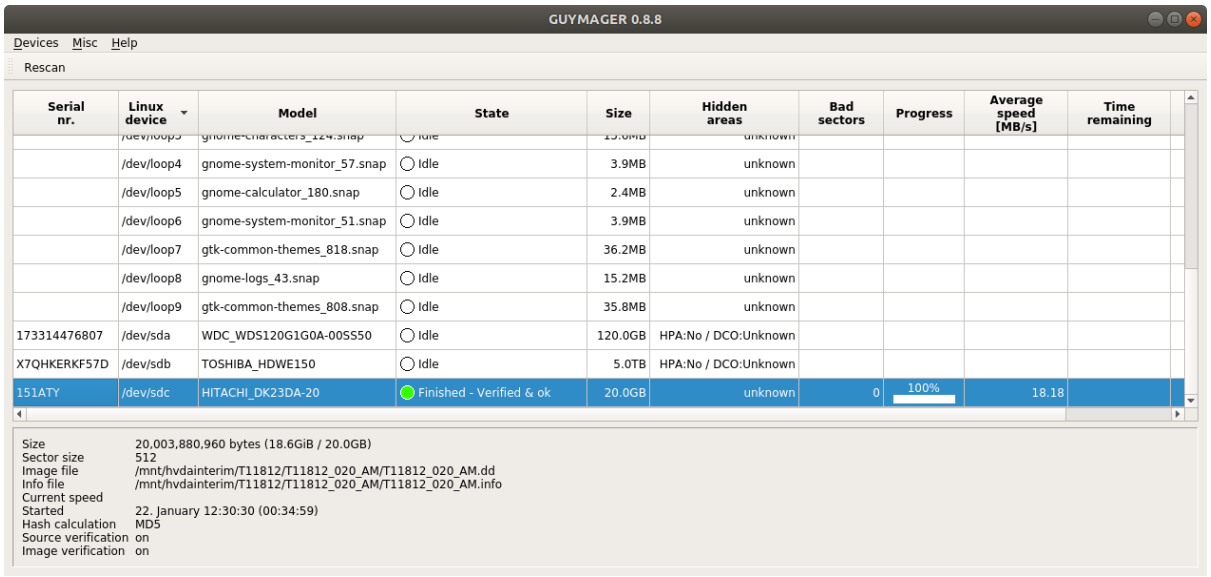
7. Ensure the options pictured in the image below are selected and enter a suitable location and filename (use the component name and type).



8. Wait for imaging to complete.



9. When Guymager reports that imaging is complete, ensure that the verification status of the image is OK and no bad sectors have been encountered. A plain text log file is generated alongside the image, which includes the checksum value generated, and can be saved in the artwork folder under Structure_and_Examination/Media/Disk_Imaging.



10. Apply image quality control procedures (see Image Quality Control).

4.5 Drive Imaging Troubleshooting

4.5.1 Enclosed Mac Computers

For some enclosed Mac computers (e.g. some models of Mac Mini and MacBook) it may be difficult to access the internal hard drive without damaging the case or other components. Use the Target Disk Mode (TDM) feature (usually boot with T key held down, but this should be confirmed for the operating system version in question) to boot into a special mode which

allows a connected computer to directly access the internal storage. Sometimes write-blockers may prevent TDM from functioning, in which case you can use a software write-blocker or a utility for blocking automatic mounting.

Alternatively, it may be possible to boot from a 'live' USB stick running a suitable operating system (e.g. BitCurator). However, this does not guarantee write-blocked access and should be used as a last resort.

4.5.2 Drive Read Errors

If you have problem with read errors / bad sectors preventing validation of a disk image, you can try using the recovery tool ddrescue. This tool carries out multiple read attempts on problematic sectors, which may make retrieval of bad sectors possible (e.g. by eventually picking up a weak magnetic signal). Before using ddrescue, ensure that your drive is not mechanically failing (see below). Where retrieval is not possible using imaging tools, specialist recovery companies may still be able to recover data.

4.5.3 Mechanical Problems and Drive Failure

If a drive is not powering on, is not recognised by the host computer, or is making an unusual noise when powered on, there may be a problem with the internal mechanism. In these cases, recovery tools like ddrescue are unlikely to help recover data and may further damage the drive. Instead, the drive should be sent to a specialist recovery company who can repair or replace the drive internals.

4.5.4 Integrated Storage

For computer systems where data is stored on memory that is integrated into the circuit board (e.g. EPROM, soldered NAND), it is unlikely that you will be able to image their contents unless a suitable interface is built into the board.

5 Image Quality Control

This section describes the quality control (QC) workflow which we use to ensure that images are valid, usable and sufficiently documented.

5.1 Image QC Workflow

1. Ensure the image you have created has been verified against the source media and no bad sectors or read errors have been encountered. Refer to the optical media and drive media troubleshooting sections for more information on dealing with verification failures and bad sectors.

    1.1. If using the Guymager workflow described above, both verification and bad sector counts are reported in the GUI when image acquisition finishes.

1.2. If you use a tool without verification features (i.e. not Guymager), you will need to verify manually by generating and comparing MD5 checksums for the image and its source media.

1.3. If creating BIN/CUE images you will not be able to generate a checksum for the source, so you should instead create the BIN/CUE image twice and compare checksums.

2. At this point, a bagged (see Bagger page) safety copy of the image should be created, as the next steps may alter data.

3. Use fiwalk to capture an XML description of the contents of the image.

4. Mount each of the filesystems contained within the image and confirm that the contents are accessible. See detailed guidance in Mounting.

5. If working with a disk image acquired from a computer system, emulate the system using the disk image as a base software environment. See Emulation page for further guidance.

6. Archive the safety copy of the image using our normal workflow for digital files.

6 Imaging Tools

6.1 Image Acquisition

6.1.1 Guymager

Guymager is an open-source disk imaging tool for Linux with a GUI. This tool should be used for disk imaging storage media in most cases — see Optical Imaging Workflow / Drive Imaging Workflow for a detailed guidance.

The primary advantage of Guymager over other tools is that it incorporates checksum integrity verification and automatically generates documentation about the process (including device information). It is also relatively fast and allows multiple images to be made simultaneously. Guymager offers two image format options, DD/RAW and EWF/E01.

6.1.2 FTK Imager

FTK is a proprietary disk imaging tool for Windows (where it has a graphical interface) and Mac OS (command-line only). It incorporates forensic imaging features similar to Guymager's such as automatic validation of images.

6.1.3 cdrdao

cdrdao is a Linux command-line tool which can be used to capture BIN/CUE disk images of complex optical media like multi-session and multi-track discs. This tool creates the equivalent of a CUE file in the TOC format, which can be converted to CUE using the included `toc2cue` command, as described below.

cdrdao images one session at a time, creating two files in the process: a BIN file which is a raw binary dump of the data; a TOC file which indexes the contents of the BIN.

The following command can be used to create a BIN/TOC image of an optical disc:

```
$ cdrdao read-cd --read-raw --datafile <image_filename.bin>
<toc_filename.toc>
```

The above command always defaults to session 1, so for multi-session discs you need to add the session option and specify a session number:

```
$ cdrdao read-cd --read-raw --session <session_number> --datafile
<image_filename.bin> <toc_filename.toc>
```

To create an ISO file from a BIN file you can use a tool called bchunk. This tool requires the index file to be in the CUE format, so you must first convert the TOC to a CUE using toc2cue:

```
$ toc2cue <toc_filename.toc> <cue_filename.cue>
```

You can then split the BIN up into ISO files, one for each track, using bchunk (you may need to install the bchunk package):

```
$ bchunk <image_filename.bin> <cue_filename.cue> <output_filename>
```

6.1.4 dd

dd is a command line tool for copying data from one location to another. This includes the ability to create disk images by specifying the source device as an input and an image file as the output. This tool does not incorporate verification of images, and this should be carried out manually using a suitable tool (such as `md5sum` on Linux). Caution should be taken when using dd as you risk causing irreversible damage by writing over device data if an incorrect drive name is used.

6.1.5 ddrescue

ddrescue is an extension of dd for the recovery of data from damaged sources. It works by repeatedly attempting to read from a bad sector, which can be particularly effective when recovering data from damaged optical media.

Example usage:

```
$ ddrescue -b 2048 -r 5 /dev/sr0 imagefile.iso mapfile.txt
```

In this example, data is being recovered from a DVD which has a block size (-b) of 2048, with 5 retries (-r). Progress is recorded in the mapfile, allowing ddrescue to continue focusing on bad sectors in further retries. The command can be re-run with the same image file and map file to continue recovery.

Adding the direct access option (-d) skips the kernel cache which may allow more data to be recovered:

```
$ ddrescue -d -b 2048 -r 5 /dev/sr0 imagefile.iso mapfile.txt
```

6.1.6 Mac Disk Utility

Note: the 'full' imaging features described below were removed from Disk Utility in Mac OS 10.11 (El Capitan).

Mac Disk Utility is a general purpose tool for imaging included with Mac operating systems. While it is suitable for creating ISO images, in its default state it lacks features for generating RAW disk images. In versions of Mac OS prior to 10.11 (El Capitan), the following command can be entered in Terminal to unlock additional imaging options:

```
$ defaults write com.apple.DiskUtility advanced-image-options 1
```

This includes the option to create 'full' disk images, which are RAW disk images in a DMG wrapper.

6.2 Structural Analysis

6.2.1 lsblk

lsblk is a simple command-line utility which lists the storage devices currently connected to the Linux system. Use this tool to find the Linux device identifier for the device you are analysing or imaging (e.g. /dev/sr0 or /dev/sdb).

It is called by typing the following at the command-line:

```
$ lsblk
```

Typically optical drives will be given an identifier in the format /dev/sr# while hard drives and flash storage devices will be given an identifier in the format /dev/sd#.

6.2.2 disktype

disktype is a command-line utility for analysing the structure of storage volumes of various kinds, including optical discs, hard drives and disk images.

It is called using the following command:

```
$ sudo disktype <image/device name>
```

The output of the program is a description of the internal structure of the storage volume, including partition table type and partition size, name and filesystem.

For example, the disktype output below indicates that we have a disk image file containing two partitions using different filesystems, a recovery FAT16 partitions and a Windows NTFS partition. It also uses the MBR partition table and is therefore likely to be of the BIOS boot type.

```
Regular file, size 37.25 GiB (40000000000 bytes)
DOS/MBR partition map
Partition 1: 39.19 MiB (41094144 bytes, 80262 sectors from 63)
  Type 0xDE (Dell Utility)
  FAT16 file system (hints score 5 of 5)
    Volume size 39.10 MiB (40994816 bytes, 20017 clusters of 2 KiB)
    Volume name "DellUtility"
Partition 2: 37.21 GiB (39950184960 bytes, 78027705 sectors from
80325, bootable)
  Type 0x07 (HPFS/NTFS)
  Windows NTLDR boot loader
  NTFS file system
    Volume size 37.21 GiB (39950184448 bytes, 78027704 sectors)
```

6.2.3 mmls

mmls is a command-line utility which outputs a table of partition data for a storage volume.

It is called using the following command:

```
$ sudo mmls <image/device name>
```

As an example, the following output presents partition table information for the same image as the disktype example above:

```
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   0000000062   0000000063   Unallocated
002:  000:000   0000000063   0000080324   0000080262   Dell Utilities FAT
(0xde)
003:  000:001   0000080325   0078108029   0078027705   NTFS / exFAT (0x07)
004:  -------   0078108030   0078124999   0000016970   Unallocated
```

6.2.4 fdisk

fdisk (when used with the -l option) is not very useful for understanding the detail of disk structure, but it is the only tool which identifies the logical and physical sector size of the attached volume.

It is called using the following command:

```
$ sudo fdisk -l <image/device name>
```

For example:

```
Disk /media/bcadmin/91f4893f-72ed-4924-8fe4-
5e10b2f9c76c/DiskImages/T13645/T13645_019_Astro_DellUSB.dd: 37.3
GiB, 40000000000 bytes, 78125000 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: 0x41ab2316

Device
Boot Start      End  Sectors  Size Id Type
/media/bcadmin/91f4893f-72ed-4924-8fe4-
5e10b2f9c76c/DiskImages/T13645/T13645_019_Astro_DellUSB.dd1
63    80324    80262 39.2M de Dell
/media/bcadmin/91f4893f-72ed-4924-8fe4-
5e10b2f9c76c/DiskImages/T13645/T13645_019_Astro_DellUSB.dd2 *
80325 78108029 78027705 37.2G  7 HPFS
```

6.2.5 cd-info

This program is a useful way to find out more about optical media and the optical drive you are using. It can be used to identify the type of disc, and the number of tracks and sessions it contains.

Example usage:

```
$ cd-info /dev/sr0
```

For DVDs you need to add an option:

```
$ cd-info /dev/sr0 --dvd
```

6.2.6 cdrdao disk-info

A command-line program included in the cdrdao package, disk-info provides a concise description of the characteristics of the disc, including the number of sessions it contains.

Example usage:

```
$ cdrdao disk-info /dev/sr0
```

6.3 Content Analysis

6.3.1 fiwalk

fiwalk is a program which analyses the contents of a storage volume. This includes identification of partitions, filesystems, files (including format and location). It outputs a structured description of the content of the volume as Digital Forensics XML (DFXML), including MD5 checksums for each file. This is useful in case mount access to a filesystem is lost in the future, as the byte locations of the files contained can be used to extract them (e.g. using dd).

fiwalk can be launched in two ways, either via the BitCurator Report and Access tool (GUI) if using BitCurator, or via the command-line.

Example command-line usage, outputting the results to an XML file:

```
$ fiwalk -X <XML file name>.xml <image file>
```

6.4 Mounting

In some cases, simple images can be mounted using the operating system GUI (e.g. by double clicking on the file). Ubuntu 18.04 (including BitCurator) and 20.04 include a GUI tool, called simply *Disks*, which can be used to mount images. For more complex images with multiple partitions, you may need to manually mount the partition using the commmand-line tools described below.

6.4.1 mount

mount is a Linux command-line tool for mounting the filesystems contained in disk image partitions. In order to mount a specific partition, you need two things beyond the disk image itself:

- the byte offset at which the partition starts, which can be identified using disktype, mmls, fdisk or similar;
- a mount point, which is a folder (usually placed in the /media/ or /mnt/ directories) where the filesystem will be made accessible.

These can be fed into the relevant parts of the following command, which mounts a partition as a read-only virtual storage device:

```
$ sudo mount -o loop,ro,offset=<byte offset of partition> <image
file> <mount point>
```

For example:

```
sudo mount -o loop,ro,offset=209735680 image.dd /mnt/partition
```

In order to mount some filesystems, you need to also install an additional program and call that extension in the mount command using the -t option.

6.4.1.1 Mounting NTFS filesystems

Install NTFS tools:

```
$ sudo apt-get install ntfs-3g
```

Then use the following syntax to mount:

```
$ sudo mount -t ntfs-3g -o loop,ro,offset=<byte offset of partition> <image
file> <mount point>
```

6.4.1.2 Mounting HFS+ filesystems

Install the HFS tools:

```
$ sudo apt-get install hfsprogs
```

Then use the following syntax to mount, making sure to include the size of the partition (which you can get from disktype and other analysis tools):

```
$ sudo mount -t hfsplus -o ro,loop,offset=<byte offset of
partition>,sizelimit=<size of partition in bytes> <image file> <mount
point>
```

For example:

```
$ sudo mount -t hfsplus -o ro,loop,offset=209735680,sizelimit=499248103424
image.dd /mnt/hfspartition
```

6.4.2 fmount

fmount is a command-line tool which automates aspects of the mounting process at the cost of precision, as it does not allow you to target specific partitions. It mounts images as read-only by default.

6.4.3 Mounting Windows Extended Partitions

Some older Windows formatted hard drives (MBR) may include what is known as an 'extended partition', which shows up as a partition with the type `W95 Ext'd (LBA)` in disktype and other tools.

These partitions should be mounted in Windows environments (including Windows 10).

7 Further Reading

7.1 General Imaging Resources

Presentation and resources from MoMA's *Peer Forum I: Disk Imaging* (December 7-8, 2017 at MoMA). URL: https://www.mediaconservation.io/disk-imaging

Disk imaging workflows from the Guggenheim's *Conserving Computer-Based Art Initiative.* URL: https://www.guggenheim.org/conservation/the-conserving-computer-based-art-initiative

Eddy Colloton, Jonathan Farbowitz, Flaminia Fortunato and Caroline Gil (2019) *Towards Best Practices In Disk Imaging: A Cross-Institutional Approach.* Electronic Media Review, Volume Six: 2019-2020. URL: http://resources.culturalheritage.org/emg-review/volume-6-2019-2020/colloton/

Eddy Colloton. *Denver Art Museum Disk Imaging Workflow*. URL: https://static1.squarespace.com/static/50cccb35e4b0cc6f589d467d/t/5c8d11daeef1a12f1f966f51/1552749020181/Denver+Art+Museum+Disk+Imaging+Workflow.pdf

7.2 Optical Media Imaging Resources

Angela Dappert, Andrew Jackson and Akiko Kimura (2011). Developing a Robust Migration Workflow for Preserving and Curating Hand-held Media. *Proceedings of the 8th International Conference on Digital Preservation*, iPRES 2011, Singapore, November 1-4, 2011. URL: https://arxiv.org/ftp/arxiv/papers/1309/1309.4932.pdf

Alex Duryee. *An Introduction to Optical Media Preservation*. URL: https://www.weareavp.com/wp-content/uploads/2014/04/OpticalMediaPreservation.pdf

Johan van der Knijff (2015). *Preserving optical media from the command-line*. URL: https://www.bitsgalore.org/2015/11/13/preserving-optical-media-from-the-command-line

Johan van der Knijff. *CD and DVD imaging and quality control notes*. URL: https://gist.github.com/bitsgalore/1bea8f015eca21a706e7#file-notescdimaging-md

Alice Prael (2016). *To Image or Copy -The Compact Disc Digital Audio Dilemma.* URL: https://campuspress.yale.edu/borndigital/2016/12/20/to-image-or-copy-the-compact-disc-digital-audio-dilemma/