Preserving Write-Once DVDs

Producing Disc Images, Extracting Content, and Addressing Flaws and Errors

An Analytic Report by *George Blood Audio Video Film* for the Library of Congress

April 2014

Delivered as a work product under the terms of Library of Congress contract OSI12T0014

George Blood Audio Video Film
21 West Highland Avenue
Philadelphia, PA 19118
215-248-2100

Table of Contents

Introduction

George Blood Audio Video Film (GBAVF) has reformatted write-once, authored DVDs for the Veterans History Project (VHP), a unit within the American Folklife Center at the Library of Congress, on two occasions between 2013 and 2014. On the first occasion, GBAVF encountered a number of technical issues and provided informal reports detailing them to the Library. As part of a 500-disc reformatting project began in 2013, the Library asked GBAVF to document the issues they encountered and to report on the steps they took to remedy the problems. This report is the result of that effort, offering an overview of the range and extent of the issues, as well as describing the corrective tools and processes that were used. The VHP collections include more than 10,000 DVDs, and this report also offers some ideas and recommendations that pertain to workflow elements that may lend themselves to automation.

The Library's approach to reformatting the DVDs entails two actions. The first is to create ISO disc image files from the optical discs, a process henceforth referred to as "cloning." Since these ISO files contain all of the data on the disc, and retain the logical structure of this data, the Library considers these ISO files as *archival master files*.[1] The second action, henceforth referred as "extracting," is to extract the underlying digital video content from the ISO files. To the degree possible, the video and audio from the ISO files is left in the encoding "as found." However, as described in this report, there are circumstances that require modest transcoding in order to yield a playable video file. The Library considers this extracted content to represent *production master files*.[2]

This report contains general information about reformatting optical disc media with a focus on DVD-Video discs and the processes of cloning and extracting the data. Additionally, the report contains a detailed explanation of the methods used be GBAVF, a summary analysis of our findings, and advice about automating the process. Footnotes are used throughout the report to explain a number of the technical terms mentioned and discussed. Additionally, an online glossary of terms related to DVDs is available on Jim Taylor's DVD demystified website, which can be found here: http://dvddemystified.com/glossary.html.

Notes on the preservation of optical disc media

Most specialists agree that optical disc media, although inexpensive and easy to use, does not support long-term data management. The Optical Storage Technology Association conservatively estimates that the shelf life of a CD-R or CD-RW to be between five and ten years. This lifespan can be lengthened or shortened by environmental factors such as temperature and relative humidity, as well as technological factors such as the quality of the recording media stock and recording equipment. A disc is generally considered to have reached

[1] http://www.digitizationguidelines.gov/term.php?term=archivalmasterfile
[2] http://www.digitizationguidelines.gov/term.php?term=productionmasterfile

the end of its life when the data is no longer readable by an optical drive. However, as Kevin Bradley mentions in his UNESCO report[3] on the subject, the lack of standards governing playback drives means that while a disc may be readable by one drive, it may not be readable by another. Given the short shelf life of this type of media, as well as the falling cost of hard disc drive and tape backup solutions, many institutions with large collections of optical disc objects have established initiatives to migrate data held from optical discs to into more sophisticated repository systems.

This migration, however, may encounter issues that stand in the way of success. GBAVF has had extensive experience in the migration of content from CD-Rs. We have seen wide variation in manufacturing quality and longevity between discs from different manufacturers and even from batch to batch from the same manufacturer. There is significant variation across brands and product lines. In fact, the greater variation actually makes it easier to predict failures because these failures have a normal (Gaussian) distribution. Homogenous collections are subject to sudden failure, often unpredictable, due to an unknown variable.

Although our experience with DVD-Rs is more limited, we have seen some of the same variations in outcome. Since nearly every disc in the VHP collection was made by a different hand--they are oral history recordings from many individual makers--the variation in the collection provides a good test bed for the study of flaws and errors.

CDs and DVDs are both physically comprised of a reflective metal layer and a polycarbonate substrate layer. In the case of CD-ROM and DVD-ROM discs, the data is written into the disc by imprinting a series of pits and lands into the reflective data layer. A laser light beam is shot at the disc, and reflected at different intensities depending on whether it hits a pit or a land. The data encoded onto a disc encoded as a series of binary digits, with the edge of each pit constituting a 1, and the flat areas between or within pits constituting a 0.

The same concepts are applied to CD±R/W and DVD±R/W discs, though the physical pits and lands are replaced by a reflective data layer and a dye layer. When a disc is “burned”, a light beam burns the dye layer, changing its reflective properties. The interference in the light’s reflection as it passes between burned and unburned areas acts the same as it does passing between lands and pits on a manufactured disc. This interference is encoded as a binary 1.

CDs and DVDs differ in how these layers are physically organized. On CDs, the reflective layer is on the top side of the disc, whereas the reflective layer on DVDs is in the center of the disc, surrounded by the polycarbonate substrate on either side. This layout makes DVDs are more resistant to physical data and tracking errors, while at the same time making them more sensitive to errors caused by surface contamination. This sensitivity is ameliorated through the use of more sophisticated error correction methods. Advancements in error correction technology allow DVD error correction to fix far more problematic bits, though this is offset by the smaller and more densely packed pits in the DVD data layer. Thus, DVDs are physically

[3] The report, published in 2006, is titled *Risks Associated with the Use of Recordable CDs and DVDs as Reliable Storage Media in Archival Collections - Strategies and Alternatives*

more robust than their predecessors, though they are still vulnerable to damage and contaminants. Along with error correction, the inclusion of a second data layer in DVDs also allows for much larger data capacity than CDs. Dual Layer DVD-ROM discs work by having a second data layer behind the first, where the first is made of a semi-translucent material. This allows the light beam to pass through the first layer when it is focused on the second layer. This same technology has been applied to DVD±R discs, though they are generally more expensive than their single-layer counterparts.

Producing disc images: tools and techniques to create the ISO files

DVDs in the VHP collection are *authored*, meaning that they have been packaged using the same structure that a professional distributor would use to make feature films playable on home DVD players. The technical specifications for these entertainment products was developed by SONY and Philips, and the resulting DVDs contain an array of related and interdependent files inside of a VIDEO_TS folder, including multiple examples with filename extensions like VOB, IFO and BUP[4]. A clone of the bundle copied onto magnetic media such as a hard disc drive, with the original relationships and logic intact, is called an ISO disc image. This image is a complete and authentic copy of the content carried on the original optical disc, which is why many consider ISO image files to be archival master files. It is the case, however, any authoring or coding flaws contained on the original disc are carried over to the cloned image.

Playing an ISO image file requires DVD player software or its equivalent, with the VLC[5] application being the most common cross-platform choice. Although this outcome may be well suited to, say, the interactive products of the entertainment industry, it may be less apt for VHP and other archival programs where the custodial interest is in the underlying video footage.

The Library asked GBAVF to deliver the extracted video in the form of MPEG-2 files. From our perspective at GBAVF, the MPEG-2 format initially appeared to be a good choice since it is the native format for video on DVD-Video discs, the idea being that the derivative copy does not require transcoding the compressed MPEG-2 to another format, such as MPEG-4. However, as this project proceeded, a number of concerns arose about the MPEG-2 format, both on the encoding and decoding side. Please see the section titled *Encoding and Decoding MPEG-2 Files* under the heading *Additional Considerations* for more details on issues surrounding the MPEG-2 standard.

It is also important to note that while it is expected that the discs being cloned will be formatted as DVD-Video discs, it is likely that a number will not conform to this format. Another possible format is DVD-ROM. DVD-ROM discs can contain arbitrary data. If a DVD-ROM contains video files, they can be played off the disc when it is inserted into a computer. A DVD-ROM is not

[4] For an explanation of the structure and purpose of these files, please see Appendix G.

[5] VLC is a powerful tool for viewing the content of video files that may be difficult to or impossible to view using other playback systems. However, with its penchant for playing anything thrown at it, VLC will ignore the logical-structure or interoperability issues in a file that might make it otherwise unplayable. Be aware of this when using to VLC to perform quality control on access files, which should generally be interoperable throughout a wide range of playback systems.

playable by a standard DVD Player, even if the video on it is encoded in the MPEG-2 format. This format can be cloned to ISO files in order to create digital archival master files with the same methods used to clone DVD-Video files. The method to extract the program content from the ISO files is different in this case as discussed in the report.

The following sections describe methods used by GBAVF to clone the DVD discs and then to extract their video content. These sections will describe the methods and tools used, as well as provide details about our experience with these tools. The tools selected for this report were chosen to display a wide range of features, price points, and user interactivity. This is meant to provide institutions of various sizes with the information they need to develop cloning and extraction workflows specific to their needs, with a focus on affordability, transparency, and cross-platform compatibility. We hope that this will enable institutions of various sizes to develop reformatting workflows specific to their needs. Please remember that the methods discussed are not exhaustive, as there are a number of tools and methods not described in the report.

Additionally, a detailed walkthrough of how to clone ISO files from a DVD-Video using a Mac is available in Appendix C.

Findings from the 2013-2014 reformatting job

A deeper summary of the DVDs with issues is provided in the Defect Summary and Advisory section on page 20. At a high level, these are the findings:

- 500 DVDs were reformatted
- Of these, 49 were problematic or abnormal in some way. The remaining 451 were reformatted without difficulty.
- Of the 49 issues, 15 were associated with cloning the discs. 34 issues were associated with the extraction of the underlying content.
- Of the 15 that had problems during the cloning process, nine had physical damage, and three did not. These three likely had some sort of coding error in the data that made the imaging process difficult, though they were eventually imaged properly.
- Of the 34 discs that had to be extracted using workarounds, 31 were DVD-ROM discs instead of DVD-Video
- The remaining three of the 34 discs exhibited other coding errors that made it difficult , though not impossible, to extract the video data from the images.
- At least some, if not all, content was recovered from every disc.

In general, our finding for this job was that the life-or-death action is the production of the ISO disc image. Although there were challenges and complications in the extraction and assembly of the underlying content, for the most part once the ISO image existed, extraction and assembly were always successful. There were cases in which the extracted video files exhibited corruption, such as poor audio/video sync, or having sections of unplayable content. However, More often than not however, it went smoothly in all but three cases, which required additional manual effort to extract a working video.

Cloning and extracting data from discs

GBAVF has used a number of applications and methods to clone the archival master files from the discs, and to extract the production master files from the ISO files. Both Command Line Interface (CLI) and Graphical User Interface (GUI) applications were used in the cloning and extracting processes. This section contains brief reviews of all of the tools used by GBAVF to clone discs.

The first three tools used to clone the archival master files, hdiutils, dd, and ddrescue are CLI applications.[6] While command line applications are not ideal for use by novice users due to their lack of graphical interaction, we chose to examine these tools first because they are both free and provide a good base for automation processes. Throughout the report, any text that appears in `Courier New` is meant to represent terminal commands, and any text that appears in `[square brackets]` is a placeholder for text to be filled out by the user. In this case the square brackets themselves do not need to be typed, but the information they represent needs to be present. For more information on using and accessing the command line on a Mac computer please see Appendix A

.

The rest of the programs are GUI tools. All of these programs can be used with any DVD compatible optical disc drive, however MediaGrabber is intended for use with a RipStation disc-loading robot.

hdiutil

This free command line application[7] is included with Mac OS X from version 10.0 to the latest version (10.9 at the time this paper is being written), and does not require separate installation. It is used to clone a copy of the data on the disc and save to an ISO file in a specified location. If hdiutil runs into a problem while cloning the image, it will report an error. The following is the typical command given to hdiutil to clone an ISO image from a disc:

```
$ sudo hdiutil makehybrid -iso -udf -verbose -o [output path] [disc path]
```

This command can be broken up into the following chunks:

- `sudo` This allows the command to be run by an admin user, ignoring any permissions issues that might otherwise cause problems with the cloning process.
- `hdiutil` This is the command to run hdiutil.
- `makehybrid -iso -udf` This is the option to clone the disc. The `-iso` and `-udf`[8]

[6] Unix is the operating system which Mac OS X is built on top of. Every Mac computer has a Unix terminal that can be run by the user in order to run command line applications

[7] The manual for `hdiutil` can be found here: https://developer.apple.com/library/mac/documentation/Darwin/Reference/ManPages/man1/hdiutil.1.html

[8] The "-udf" option, as described by the **hdiutil man** page: "Generate a UDF file system. This file system can be present on an image simultaneously with HFS+, ISO9660, and Joliet. UDF is the standard

flags tell the program to clone the ISO with a UDF file system.

- `-verbose` This tells the program to output as much information about the process as possible. This information can be helpful in troubleshooting and problems, and has no effect on the cloning process.
- `-o [output path]` This indicates where the ISO file will be clones to.
- `[disc path]` This is the path of the disc being cloned.

We initially ran hdiutil by itself, however in the interest of automation methods we wrote a shell script, DVD_isorip.sh[9] to make it easier for users to input the necessary information. See Appendix H for more information on this script. When using this script, the user only needs to type in the disc name, rather than the entire command. If this script were used in creating the ISO it was notated in the metadata that was delivered to the Library.

When using hdiutil it is important to specify the file system of the output image. Unlike other tools, hdiutil will not pass the input file system to the output automatically. This allows the user a great deal flexibility in creating images with varying file systems, though this flexibility may not be desired while working in a preservation paradigm[10]. When working with DVDs, the output file system should typically be UDF.

dd

dd[11] is a free command tool and is included with every Mac computer. The program's main function is to copy file data from one location to another. Because the input and output files can be set arbitrarily, dd can be used to copy an entire disc or volume to a specified output location. Thus, if a user sets the input directory of dd to the contents of the an optical drive, the resulting output file with be a bit-for-bit copy of the optical disc. The following string is a typical terminal command given to dd in order to copy a disc to an iso file:

```
$ dd if=[disc path][12] of=[output path]
```

Here is a breakdown of the string:

- `dd`: This runs dd
- `if=`: This specifies the "input file", which must contain the path of the optical disc being cloned
- `of=`: This specifies the "output file" which must contain the path of the desired output.

---

interchange format for DVDs, although operating system support varies based on OS version and UDF version."

[9] A shell script is an automation script that performs command line tasks. We used DVD_isorip.sh to control hdiutil, and make the user input slightly more efficient and user friendly. The script changes nothing about how the tool works, but rather how the user interacts with the tool.

[10] See the discussion "ISO File Systems" on page 25 for more information.

[11] The manual for dd can be found here: https://developer.apple.com/library/mac/documentation/Darwin/Reference/ManPages/man1/dd.1.html

[12] In this case, the disk path must be in the form "/dev/disk1". The path of the disk drive may vary slightly depending on the hardware of the computer being used. The path of the optical disk drive being used can be found by typing `drutil status` into terminal. Please see Method 1 of Appendix C for more details

This method will quit if it runs into an error. In order to get dd to continue through an error you must use the following command:

```
$ dd if=[disc path] of=[output path] conv=sync,noerror
```

The `conv=sync,noerror` part of the string tells dd to continue if it finds an error, and to pad the bad sector[13] with null data in order to maintain the proper sector size.

While dd is rather simple and generally reliable, it should not be trusted to make archival ISO files due to its unreliable error reporting and handling. From our experience, dd does not clearly describe errors it encounters, and will deal with unreadable sectors of data by replacing all of the bits with null data. This makes it incredibly easy for dd to alter the data contained in the archival ISO file without the user being fully aware of the process. With this in mind, dd should only be used on discs that are known to be in good condition.

ddrescue

ddrescue[14] is a command line tool and is not included with Mac OS X, but can be downloaded for free. This program is described as a "data recovery tool", which makes it ideal for dealing with damaged discs.

If ddrescue finds an error on a disc it will try a number of methods to recover the data, making it a good option for trying to recover data from a disc that couldn't successfully be read by hdiutil. However depending on the type and severity of the defect, ddrescue may take hours, days, or even weeks to fully recover a damaged disc. Our experience at GBAVF showed that simply allowing ddrescue to run on a damaged disc for a few hours allowed us recover a majority, if not all of the data in the damaged areas.

Of the 500 discs that were processed, ddrescue was used on 18 discs. Of these discs, 12 were fully recovered using ddrescue, and six were partially recovered. For these six remaining discs, the size of the unrecoverable data sectors was between 130kb and 3221mb. In the context of a 4.7gb disc, 130kb represents about 0.0026% and 3221mb represents about 66.9257% of the data on the disc. In some cases, ddrescue was left running on a damaged disc over the course of many days, with the hopes that more data could be recovered if the program was given more time to work with the disc. We found that while more data could be recovered through this method, it was minimal (on the order of bytes per day) and that there is in fact a point at which damage or corruption renders the data inaccessible by any means. We were able to extract

[13] "Sector" refers to an organizational unit of data. On disk media, data is organized and divided into sectors, also known as blocks. Different types of media organize data into different size sectors or blocks. On DVDs, the sector size is 2048 bytes. When cloning information from one disk to another, be it a CD-ROM, a Hard Disk Drive, a DVD, etc., it is important to maintain the proper sector size in order to properly represent the data.

[14] Not to be confused with dd_rescue, yet another command line disk recovery tool. ddrescue and dd_rescue are similar and both actively maintained and useful tools. The manual can be found here: http://www.gnu.org/software/ddrescue/ddrescue.html

playable video files from the discs in which ddrescue was able to recover all but a very small portion of the data (less than a percent), while this was not possible with the discs in which significant data was unrecoverable.

In general, we were able to make ISO images and playable MPEG-2 videos from discs with smaller unrecoverable data sectors. However, as the size of the error sectors increased, so did the number of video artifacts and corruption. Below is a terminal command that will run ddrescue, though a more detailed explanation of installing and running the program can be found in Appendix C:

```
$ ddrescue -b 2048 -r4 -v [disc path] [output ISO path] [output log path]
```

- `-b 2048:` This specifies the data sector or block size in bytes. Since the sector size for DVDs is 2048 bytes, this helps ddrescue maintain the proper data structure as it corrects errors.
- `-r4:` ddrescue This specifies that ddrescue can retry a bad sector up to four times. This can be changed according to however much time you have available to let ddrescue run on a single disc.
- `-v:` This puts ddrescue in “verbose” mode, which makes it display more information about what it is doing.

While the log path is optional, it is crucial to the error correction of a disc. ddrescue stores information about the errors in the log file. If ddrescue is run on a disc consecutive times, the log will inform ddrescue of the location of the errors that were found the previous time it was run, so no time needs to be wasted trying to recover good data.

The main consideration to keep in mind while using ddrescue is that its error correction processes make the time it takes to clone an image difficult to predict. A damaged disc may take hours to rip, while a disc in good condition can be cloned in a matter of minutes. If ddrescue is used in an automated workflow, users should be sure to use flags that specify the amount of retries ddrescue can perform, or possibly tell it to skip errors it finds the first time around. If errors are found, a second workstation can be used to run ddrescue for as long as it needs to in order to record the data on the disc. This would allow the main workstation to run at a steady pace through DVDs, leaving the automated workflow uninterrupted.

ddrescue can even be used to recover a single images from two damaged discs, given that they are damaged in different areas. Using the log file, ddrescue can try to recover the erroneous data from a second disc, jumping immediately to the data that it could not recover from the first. This process is described in more detail in the Fifth Method of Appendix C.

MediaGrabber

MediaGrabber is a software package that can be used along with a Ripstation disc cloning robot. MediaGrabber offers support for cloning Audio CDs, Data CDs, Video DVDs, Data DVDs

and Blu-ray discs. As with the previous cloning methods discussed, the primary focus of MediaGrabber is to clone the data from the disc to a drive. MediaRipper comes with three modules: Audio, Video, and Data. This report is focused on the Video module, which is capable of cloning to either a VIDEO_TS folder or to an ISO file.[15] The information in these ISO files is organized according to the original file system of the disc. This means that the outputted ISO files can contain information encoded in Joliet, UDF, Rock Ridge, or ISO 9660. Whatever the specified output format is, the data from these discs can either be cloned to network drive via 1Gb/s Ethernet or to an external drive via USB. If the end goal is to have these ISO fed into an automation system that transcodes the video data, then the Ethernet 1Gb/s provides an ideal solution for placing the output files in a network watch folder.

There are different pricing tiers available, each with their own set of features. There are two levels of RipStation robots available, one with two optical drives and one with four optical drives. At their base price of $3295 and $3995 respectively, both of these robots come with the audio cloning software, with the Video software being a separate purchase at $495. If an institution plans to be processing DVD-Video discs regularly, then the more cost effective option is to purchase the robots with the MediaGrabber software, which includes video cloning software, as well as Blu-ray cloning software. This puts the total prices of the two and four drive robots at $3795 and $4295 respectively.

All RipStations have a 300 disc load capacity. The two-drive units can process an estimated 30 discs per hour, while the four-drive units can process 50. This means that a four-disc unit would be able to process an entire bin in six hours, allowing an operator to start two full bins during an eight-hour shift. However, a two-drive unit could only be loaded with 240 discs in order to allow an operating to start two bins during the course of an eight-hour shift.

Any bad discs are dropped in a reject bin, and notes about the bad disc are collected in the output log. Bad discs are generally unreadable by the drive and software. This may happen if the disc is damaged beyond readability, or if the data on the disc is corrupt. A disc could also be rejected due to user error, such as if the disc were loaded upside-down.

If the output format is set to ISO, the software will clone an ISO image of any readable disc fed to the drive, regardless of the disc format or type. This means in a case where DVD-Video DVDs are expected, but Data DVDs are fed to the robot, the discs will not be rejected, and the output will still be ISO files. The log files state the format of the disc, which can be used to guide any sort of automation or scripting being run on the files. The information about the format of the disc can be used to route the output ISO files to different watch folders, which expect certain formats and process the ISOs accordingly.

[15] This is assuming the disk is not a commercial DVD, which is protected by a copy protection scheme. Commercial DVDs are protected in this manner as a way of protecting the intellectual content. DVDs containing non-commercial content, such as the oral histories from the VHP project, do not have a copyright protection scheme, and thus will not be affected by this. Additionally, John McGrath, a representative at MF Digital, stated that software could be purchased that would circumvent this protection scheme, but doing so is against the law and could not be supported by them.

Mac DVDRipper Pro (MDRP)

MDRP is a simple GUI program for Mac computers. The program is simple, with a slick interface and very simple options. At its simplest, MDRP will clone an ISO image from a DVD, though it can also be made to extract derivative video files as well. The user can request that the extraction follow the cloning by using MDRP's "batch" method.[16] The tools available are quite limited, however. The extraction is limited to 64 bit M4V files, and the encoding options are all presets rather than user-adjustable parameters. The user can, however, chose to extract the audio track as a separate AC3 file.

In its default mode, MDRP will not display a log of its actions or any errors. If the program is able to successfully clone the data from a disc there is not user dialog besides the alert that the cloning process has finished. Additionally, the error correction for MDRP appears to be on the same level as ddrescue, as the ISO files created by MDRP were bit-for-bit identical to those created by ddrescue, even in the case of minor errors and data recovery.[17] In cases where a disc was severely damaged beyond readability for most programs, MDRP was able to create a playable image and reported no errors. However, without a log file it is difficult to say what MDRP does to create readable images from unreadable discs.

At $25 a copy, MDRP presents a relatively inexpensive and easy to use option for cloning DVDs. However, it's lack of options and processing data makes it undesirable for use in a preservation environment depending on what an institution is looking for in its preservation masters. The processes MDRP is using to recover the errors are not clear to the user, and it may be adding in data that was never on the disc in the first place. Thus, the ISOs it creates from damaged discs may contain information not to true to the original disc. A program like ddrescue will never add in data that it did not find from the disc, at the expense of creating ISO files that may not play well. The flip-side however, is that MDRP is capable of cloning images of severely damaged discs. With this in mind, MDRP may be best used as a backup, in case a disc cannot be transferred by another method, with the caveat that the data on the image is not precisely the data on the original disc.

Toast

Toast 11 Titanium is a media toolkit for the Mac platform. It can be used to duplicate discs, clone images from discs, create disc images from files on a hard disc, write images to discs, as well as encode video and audio for writing to Audio CDs or DVD-Video. For the purpose of cloning DVDs to image files the Copy mode is used. Toast's Copy mode can be used to copy the data from one disc directly to another, or to clone an image file to a hard drive. Unfortunately, the image files created by Toast do not have a .iso extension, but rather have a

[16] In MDRP's vocabulary, "batch" does not refer to processing multiple disks in a batch, but rather performing multiple processes on a single disk. MDRP does not support any sort of automation or batch processing of multiple disks.

[17] This was verified by comparing the checksums of the ISO files against one another. When comparing the ISO files created by one program against another, both the checksum for the ISO file and the checksum of the mounted image were compared against one another. In any case where it is stated the output files were identical, it means that the checksums for both the ISO files and the mounted images were identical.

.toast extension. However, a .toast image of a disc created by Toast is identical to a .iso image created by ddrescue according to a checksum comparison. Thus, an extra renaming step will need to be taken in order to get the images from Toast to look and act like typical ISO files, but the results are the same.

Toast deals with problem discs by aborting the transfer upon the detection of a problem unless the “Use disc Recover” box is ticked. Upon ticking this box the user is informed that disc recovery may take a very long time. Much like ddrescue, Toast will spend as much time as it needs to recover as much as it can if given the chance.

Toast can be purchased for $80, which can be considered relatively expensive compared to other programs. The reason for this price is that it includes a number of tools that can be used for other purposes, with the Copy feature being just one of many. This makes Toast a useful investment for an institution looking to make use of its other features, but overkill for any institution simply looking for a disc cloning utility. Additionally, the lack of useful log files along with output files that deviate from typical ISO files makes Toast a poor decision in an automated environment.

CloneDVD

A PC based GUI app, CloneDVD has a number of features that can be used to copy DVDs, clone ISO files, and write DVD-Video discs. For the purpose of cloning ISO files, the Write Existing Data mode should be used. In this mode, the user defines a source (the DVD drive), an output format (ISO/UDF Format), and an output destination. This program does have a log window, which is visible during the cloning process.

CloneDVD costs about $75 per license, and as with Toast this price is likely due to the many other features included with the software that are unrelated to cloning ISO images. The process and interface are both very simple, but as with MDRP there is a lack of options, though the log window does display useful information to the use. This program has its advantages, but it is not an ideal choice for an institution looking to preservation-grade disc cloning due to its lack of user adjustable options.

ImgBurn

This is a free GUI app for the PC platform. The program has a number of modes, including Read, Build, Write, Verify and discovery, with the Read and Verify modes being the most useful for the purposes of cloning DVDs. In Read mode, the user is presented with a number of options to clone a DVD. Begin by selecting “Create image file from disc.” Next, select a destination path and then begin the cloning process. The Verify mode allows users to verify the cloned disc image against the disc itself. There are also a great deal of options and parameters available to the user with this software, including the ability to create MD5 checksums of the image upon cloning.

During any process, ImgBurn provides the user with a log window, which displays information about the disc. A master log file can be saved in a user-defined location, which can be set in the “File Locations” tab of the settings window. This log file includes information about the disc

media, the cloning processes, and the output file names. There a “batch mode” in which causes the disc tray to open when the cloning process finished or is aborted, and starts again automatically upon loading another disc. The program can also be controlled by command line processes. These features, along with an operation available for switching the discs, could provide a solid foundation for automating the cloning and extracting processes.

DVD Decrypter

DVD Decrypter is very similar to ImgBurn. It is a free PC-based GUI app that provides a number of tools for cloning DVDs. The program supports a number of user-adjustable options and parameters and includes a log which can be saved to an arbitrary location. However, unlike ImgBurn, DVD Decrypter is focused on cloning DVDs and does not include tools for other actions such as writing or analyzing discs, though it does offer some unique tools in this regard. One of these tools is the ability to rip all of the VOB files on the disc to a single concatenated VOB file. This single VOB contains all of the video data contained on the disc, starting with any existing menus, and moving forward through the program content in the order that it appears on the disc. Depending on the needs of the institution, these could serve as archival master files.[18]

DVD Decrypter is an excellent tool for any institution looking for and free yet comprehensive tool for cloning DVDs. It lacks the batch processing support that ImgBurn provides, but has a number of very useful tools and user-adjustable settings that would make it useful in a number of environments, including preservation.

Additional Consideration for disc Cloning and Error Correction

A number of the programs mentioned in this section will react to errors with discs differently. The previous section describes how some of these programs react to errors, though this section compiles what is known about each program’s error handling.

hdiutil - Will stop cloning if it runs into an error. In verbose mode it will give the user information about the process, though it does not describe errors thoroughly.

dd - Will stop if it encounters an error. However, if run with the option `conv=sync,noerror` it will skip erroneous sectors, jumping to the next sector. Once it has read everything it can it will return to the error sectors and do what it can to recover them. If it cannot recover a sector it will pad the error section with null data in order to maintain the proper sector size and ensure readability of the data.[19] This process can be very slow and take days if the errors are severe enough.

ddrescue - ddrescue will, by default, attempt to recover the data in a damaged area. It does this by splitting the errors into chunks and processing those chunks piece by piece. The idea is that

[18] For a more detailed discussion of this topic, see the Additional Considerations section on page 25.
[19] It is unclear from the dd manual whether this process is simply zero padding (filling the error the section with 0s) or some other method. If you wanted to ensure that the erroneous sectors are in fact zero padded, it would be beneficial to use dcfldd, a tool built on top of dd by Nick Harbour for use with forensic data recovery and analysis. This program is very explicit about the fact that it zero pads erroneous sectors, and can be found here: http://dcfldd.sourceforge.net/

ddrescue can isolate the errors as it works through the data, and will keep splitting up data until it has isolated the errors and has processed all of the non-error data. This is why as ddrescue splits the error chunks up, the output in Terminal will report that the size of the error will decreases as the amount of errors increases. Unlike dd, ddrescue will NOT replace data it cannot retrieve with zeroes. This makes the ddrescue more appealing from a forensics perspective, though it may create completely unusable files.

MediaGrabber - If an error is encountered, MediaGrabber will slow down the drive speed and attempt to recover the data. If damage on the disc makes the data unreadable, or if the data is corrupt beyond readability on the disc then the disc will be rejected. In the past, MF Digital has written special software for clients that will rip all of the data on a DVD at all costs. This option may be possible for institutions with the resources to invest in this type of solution, but the data pulled from these discs will still be corrupt, appearing as visual corruption and artifacts in the video files.

Mac DVDRipper Pro (MDRP) - Due to a lack of log files, it is unclear as to what MDRP is doing when it encounters a damaged disc. If the disc is only slightly damaged, it will pull off all of the information properly. On a heavily damaged disc, MDRP will hang on an error, though eventually it will move along. The images that MDRP creates form the damaged discs are playable, though the video data is visibly corrupted where the errors occurred.

Toast - If an error is encountered, toast will stop cloning and tell the user to retry using disc recovery. Upon ticking the “Use disc Recovery” box the user is told that the process may take a very long time.

CloneDVD - If the disc is damaged, a dialog will appear telling the user that the program could not read the disc, and suggests that the disc be cleaned and reinserted. The dialog will also display which file the error was in. This information is also displayed in the log window. Using this information, the user could possibly use another program with superior error correction to extract the file in question.

ImgBurn - In the case that ImgBurn can work around the error, it will appear in the log but the program will continue as normal. If ImgBurn finds an error that it cannot work around a window will appear informing the user that an error was found, giving the user the options to Ignore, Retry, or Cancel. The user can also set the amount of software and hardware retries or to ignore errors in the settings menu, though it is unclear what methods the program uses to correct errors.

DVD Decrypter - The error handling for DVD Decrypter appears to be identical to ImgBurn, and it has the same options for setting the amount of retries. However, it is possible that the way the software deals with the errors under the hood is different.

Extracting the video content: tools and techniques to create video files

Once an ISO file is cloned from a disc, the information in the ISO can be used to extract derivative video files. Inside of an ISO file clone from a well-formed DVD-Video disc is a VIDEO_TS folder. Inside of this folder is a number of files, including VOB files.

As part of the transfer project, The Library asked GBAVF to provide Production Master files from the ISO files in the form of MPEG-2[20] video files. Due to the interoperability issues common with MPEG-2's many variants, we initially had trouble creating MPEG-2 files that were compatible with the Library's playback systems. This section details four of the programs used to extract these files. Throughout this process, we noticed that some programs could easily deal with VOB files while others could not. In some cases, high bitrate MP4 files had to be extracted as intermediate files[21] in order to extract consistent and robust MPEG-2 files from VOB files. The four tools discussed in the following section are StreamZ, Compressor, FFmpeg and MPEGStreamclip.

StreamZ is Digital Rapid's proprietary software, meant to be used with their hardware encoders. The software boasts a flexible and robust feature set, including the ability to encode to multiple codecs concurrently during true stream encoding. Compressor is a video encoding program developed by Apple. It has long been included as part of the Final Cut Studio suite to be used as a backend for exporting and compressing video from Final Cut Pro, though it can also be purchased as a stand-alone application.

FFmpeg is a command line tool that allows users to encode and convert a number of video and file formats. It is an open source project with a very large community of contributing programmers and advocates. While the tool is incredibly powerful, because it exists entirely in a command line environment it is not ideal for novice users. MPEG Streamclip is a GUI tool that is built on top of FFmpeg. It has many of the same functionalities as FFmpeg, but they are easily accessible to the user through GUI menus and windows. The downside however, is that there is far less flexibility with MPEG Streamclip, making it less ideal for automation. Both MPEG Streamclip and FFmpeg allow users to convert VOB files inside of the ISO files to MPEG-2 files. These tools also support the concatenation of multiple video files (VOBs) into a single MPEG-2 file, a crucial aspect of this project (See Appendix D and F), though FFmpeg supports the automation of this concatenation while MPEG Streamclip requires extra steps and file renaming.

StreamZ

The StreamZ software extracts highly interoperable generic MPEG-2 files, and also has watch folder capabilities. While the watch folder and automation features are somewhat basic, we have been able to use them as part of a widely automated system. This program has trouble working with VOB files. The output is very inconsistent when it works with these types of files, often outputting MPEG-2 files without sound, or files that are not recognized by video files by

[20] At this time, the MPEG-2 codec for QuickTime must be purchased for $19.99 from the Apple Store. The codec for Windows Media Player can be downloaded from CNET for free.
[21] These files are created by concatenating the VOB files in the ISO and encoding them as high bitrate MP4 files. Please see Appendix F for a discussion on methods of VOB concatenation.

any software. However, if given high bitrate MP4 files the StreamZ makes very consistent MPEG-2 files. The major drawbacks of this software are that it is expensive, requires proprietary hardware to run, and is only available for on the Windows platform. At GBAVF we are able to automate a mixed Mac/PC environment compliments of a SAN running StorNext that handles the cross platform storage issue. This sort of infrastructure allows us to take full advantage of StreamZ's watch folders for automating a number of processes. However, this infrastructure is both expensive and complex, making StreamZ a wonderful solution for institutions with available resources, but a poor solution for smaller institutions looking for a simple solution to extract MPEG-2 files.

Compressor

MPEG-2 Files made with Compressor were also compatible with multiple systems. Compressor has inconsistent output when given VOB files though, like StreamZ, it works fine when dealing with high bitrate MP4 files. The default MPEG-2 encoder extracts files that are made to be burned to DVDs, which includes an .m2v file and an .ac3 file. Compressor can be told to make multiplexed MPEG-2 files out of an arbitrary input, however a custom setting must be created. The custom setting we used can be seen in Appendix E. Compressor also offers both batch processing and automation support. The former being relatively simple to setup and the latter being somewhat complicated. Performing batch processes in Compressor is as easy as dragging all of the input files into compressor, dragging in a destination setting, dragging in a encoding setting, and letting it run. Automation can be enabled through droplets, drop folders or Compressor's command line interface. Compressor droplets are very simple to set up in Compressor 4[22], and drop folders are possible, though significantly more complicated in earlier versions.[23] All versions also support command line processes, which allow the user to use command line scripting automation to run encoding processes. However, we have yet to experience consistent output with this method, and every video that was extracted using this method seemed to have a single green frame at the beginning. Compressor is far more affordable than StreamZ, costing only $50 from the Mac App Store. Unfortunately, on top of only being available for on the Mac platform, the software is also only available for computers running Mac OS 10.9 or above.[24] Therefore, while the software is promising from an encoding standpoint, it should only be used by institutions that are either up to date, or small enough to successfully upgrade their system without suffering from OS migration issues.

Handbrake

Handbrake is a popular GUI tool used for extracting video files from ISO files or optical discs. The program works on Mac, Linux and Windows platforms, is free, easy to use, and takes advantage of multithreaded encoding for very fast extraction times. The program supports a

[22] See instructions here: http://help.apple.com/compressor/mac/4.0/en/compressor/usermanual/index.html#chapter=28%26section=1%26tasks=true

[23] See instructions here: http://backtotheedit.com/2011/06/compressor-files-with-a-watch-folder/

[24] After spending a great deal of time on the phone with Apple support I was assured that they could not provide me with a .dmg of an older version. Additionally they could not tell me whether a license purchased for a current version of the software would work with an older version of the software.

queue, in which the user can place several files to be transcoded or extracted. While the encoding options are fully customizable, the available presets are fairly comprehensive. The extracted video files can either be output in MP4 or MKV containers, and can be transcoded using either the MPEG-4 or MPEG-2 codecs via FFmpeg, or the H.264 codec via x264. The user can also specify the codec for the audio tracks, with the options being AAC, HE-AAC, AC3, MP3 or Native. Handbrake will by default encode the output video to the same frame size as the source. The output frame size can be adjusted by the user, though Handbrake has trouble creating a larger frame than the source. This means in the case that the source ISO has video encoded at 352x240[25], Handbrake will not be able to create a 720x480 derivative video file from it.

Handbrake's MPEG-4 files play well on a number of a platforms and programs, but the MPEG-2 files it creates do not play in QuickTime on OS X 10.6.8, nor do they play in Windows Media Player on Windows 7, though they will play in VLC.

Handbrake would be best utilized at an institution looking to create MPEG-4 derivative video files directly from discs, or from ISO files, though the lack of upsizing options can cause problems for an institution looking to create uniform access files. In the case that ISO files cloned from DVDs are the source, the encoding queue can be used to save a great deal of time and effort. It is not advised that Handbrake's MPEG-2 files be used as access files, unless thorough testing can is done beforehand to make sure the files will work well with the Digital Asset Management Systems (DAMS)[26] and playback systems in question.

MPEG Streamclip

MPEG Streamclip can extract robust and consistent MPEG-2 files from a number of source formats, including VOB files (either individually or concatenated), high bitrate MP4s, and various others. MPEG Streamclip's major strength with regards to this project is its ability to batch process files. The user is able to create a batch list of files to be processed, and let MPEG Streamclip run through these files for however long it takes. Unfortunately, the idiosyncrasies of DVD file names make it difficult to run this process on multiple ISO files at once. MPEG Streamclip depends on the file names for its conversion, and since VOB files typically have the same name across different DVDs (VTS_01_X.VOB) the batch method will end up overwriting the output files. With this in mind, MPEG Streamclip's batch processing provides better results when working with files with unique names and paths. However, the fact that MPEG Streamclip

[25] The frame size of NTSC (the video standard used in most of North and South America) MPEG-2 video content encoded for DVD-Video must be either 720x480, 704x480, 352x480, or 352x280. Depending on the authoring software that created a DVD, the video content could be encoded at any of these. While the video content in the archival master ISO files should be kept consistent with the original content, institutions wishing to create access files with consistent attributes should keep in mind that they may need to resize the original content in order to do so.

[26] A Digital Asset Management System refers to the information and technology systems that manage the ingesting, archiving, and disseminating of digital media objects. Some DAMS are created in-house by institutions, while others are created by contractors or third-parties. While some video formats may seem ideal for an institution's purposes, those same formats may not be supported by their DAMS. Institutions should be certain that any video formats they plan to use are compatible with their DAMS before investing resources in the creation of those formats.

is both free and cross-platform compatible makes a very useful tool for smaller institutions, with the caveat that due to the default names of the input files from the ISO images using it may become a rather labor-intensive process.

FFmpeg

The last program used so far is FFmpeg. The string used to extract access-ready MPEG-2 files is as follows:

```
$ ffmpeg -i [input file] -vcodec mpeg2video -pix_fmt yuv420p -
me_method epzs -threads 4 -r 29.97 -g 15 -s 720x480 -aspect 4:3 -b
5000k -bt 300k -acodec mp2 -ac 2 -ab 192k -ar 48000 -async 1 -y -f vob
[output file]
```

This string transcodes the video output with an array of specific parameters to ensure a normalized and high quality output. Here is as breakdown of the flags used

- `-i [input file]`: specifies the input file
- `-vcodec mpeg2video`: Unlike the previous string, this one specifies that the output be encoded as MPEG-2. Since the video in an ISO cloned from a DVD-Video file is always encoded as MPEG-2, this is not absolutely necessary in the context of extracting MPEG-2 files from Video-DVD ISO. However, this flag allows this string to be used on most any other video files that you may wish to encode to the MPEG-2 format.[27]
- `-me_method epzs`: This sets the motion estimation[28] method to “epzs”, which is a method that is compatible with FFmpeg’s mpeg2video encoder.
- `-threads 4`: This is used to do multithreaded encoding. In our system, the computers used to render the derivative video files have four cores,[29] thus we used 4 for the thread-count. The number can be changed, or the flag can be omitted entirely, though it can greatly increase encoding performance if used properly.
- `-r 29.97`: This sets the output frame rate to 29.97 frames per second.
- `-g 15`: Sets the GOP size to 15.
- `-s 720x480`: This sets the output frame size to 720x480.

[27] A number of more complex video formats, such as JPEG2000 wrapped in MXF, require more complex flags in order to transcode them properly.

[28] Motion Estimation is a process of compression by which the encoding system uses the visual information of adjacent frames to throw away visual data it doesn’t need to fully render a moving image. For example, If a sequence in a video involves a person walking across a static background, then using motion estimation the encoder does not need to re-render the background for every frame. Rather, It can use the same visual information for the background in each frame, only rendering the information that changes between frames. This allows encoders to compress the size of video files without losing image quality. MPEG-2 videos for DVD-Video already use motion estimation, and the inclusion of it in this FFmpeg string serves the purpose of keeping the methods that perform the compression consistent throughout each file for the purpose of greater interoperability.

[29] A multi-core processor is a single physical unit with multiple processors inside of it. Multi-core processing, or multithreading refers to using algorithms to optimize performance by spreading the workload across multiple cores. Many video encoding processes can take example of multithreading, which splits the video into chunks, allowing each core to manage a different chunk simultaneously.

- `-aspect 4:3`: This sets the output aspect ratio to 4:3.
- `-b 5000k -bt 300k`: This sets the video bitrate to 5Mbps, with a tolerance of 300Kbps.
- `-acodec mp2`: As with the last string, this transcodes the audio to mp2 audio.
- `-ac 2`: This sets the audio channels to two.
- `-ab 192k`: This sets the audio bitrate to 192Kbps.
- `-ar 48000`: This sets the audio sampling rate to 48,000 kHz.
- `-async 1`: This sets the A/V sync method.
- `-y`: This forces the output to overwrite if the outpath already exists.
- `-f vob`: This sets the output format to vob. This must be set or the audio will not play properly.
- `[output file]`: This specifies the path of the output file.

This method is a variation of another method originally developed by Jason Priebe, the Director of Technology at CBC New Media Group and an active member of the FFmpeg community.[30]

While FFmpeg's command line interface, while daunting for novice users, provides a great deal of flexibility and automation. It is both free and cross-platform compatible, which makes it an ideal tool for smaller institutions looking for a cost-effective way to extract video. Of course, any institution looking to use these tools will need to invest either time or capital in a system to automate FFmpeg.

Extracting video from DVD-ROM discs

Unlike with DVD-Video discs, the videos on DVD-ROM do not conform to any specific format or standard. These discs are not playable by standard DVD players, and the video content must be played by accessing the files through a computer's optical drive. The process of cloning an image of these discs is identical to the cloning a Video-DVD. The process of extracting the video content in its native format is as simple as dragging the video file from the disc or the mounted ISO image to its destination. However, since it is likely that an institution may wish to have all of its extracted video content in the same format for the sake of consistency, the video files on a DVD-ROM disc will likely need to be transcoded. In this case, any of the aforementioned transcoding methods in the previous section can be used to transcode the video file to the desired format. It should be kept in mind however, that these types of discs will not fit nicely into an automated workflow designed to extract video from DVD-Video discs, and will have to be dealt with manually.

Defect summary and advisory

Of the 500 DVDs that were cloned, 49 were found to be problematic at the cloning stage. Of these 49 discs, nine were determined to be problematic due to their physical condition and 34

[30] For more information, visit his website at http://smorgasbork.com/component/content/article/35-linux/97-real-time-mpeg-2-encoding-with-ffmpeg

were determined to be problematic due to coding errors.

Physical Errors

These types of errors occur when a physical abrasion on the polycarbonate layer reflects the beam improperly, causing a lost or improper reading of the data. Throughout our experience in working with optical disc media we found there to be three tiers of physical damage:

- **Tier 1 - Light Damage:** Superficial scratches, scuffs, fingerprints, and dust. This type of damage generally goes unnoticed during processing as the error correction algorithms built into DVD reading work around them.
- **Tier 2 - Medium Damage:** Medium to large scratches. This type of damage will cause sections of the disc to be unreadable by typical reading applications. Using recovery tools such as ddrescue, or using a combination of tools will aid greatly in the recovery of the damaged sections; polishing the disc may help as well.
- **Tier 3 - Heavy Damage:** Large gouges and chips in the polycarbonate layer, flaking reflective metal layer, deteriorating dye, radial scratches. This type of damage will render the data in the damaged section completely unreadable. Depending on the size and location of the damaged data sectors it may still be possible to create a working ISO file.

Optical discs are designed with powerful error detection and correction methods. DVD error correction algorithms use Reed-Solomon Product Codes to correct errors. These correction algorithms and are constantly running in the background while a disc is being read, which allows discs with minor scratches or dirt to be read without problems. However, certain types of physical damage are beyond the capabilities of error correction. The maximum correctable burst length for a DVD is 2200 bytes, which corresponds to 4.6mm physically. This means that if a 4.6mm long scratch across consecutive bits will cause an error uncorrectable by error correction.

One of these discs found to have physical errors, record 030815, could be readily identified as being in poor physical condition. The disc had deep gouges outside of the data track. After the initial attempt to clone the image using hdiutil failed, the operator attempted to clone the image using ddrescue. While ddrescue was able to recover far more data from the damaged disc than hdiutil after running for several weeks, it was still unable to recover the information fully.[31]

There are other cases, such as record 025587, where the disc is in relatively good physical condition but still cannot be read by optical drive and clone using hdiutil. In these cases, where the disc was lightly damaged by small scratched but lacked any serious physical flaws such as gouges or a damaged data layer, ddrescue was able to recover all of the data on the disc. ddrescue's method of splitting sectors of data and reading them in reverse if necessary allows it to work around physical damage or defects.

[31] It is in these cases that the ddrescue log becomes extremely valuable, as it keeps a record of known good and bad sectors of data. ddrescue can use this information to ignore known good sectors and focus on known bad sectors.

Most physical errors will be present on the disc before they are processed, though it is possible for discs to sustain physical damage during handling. In order to reduce the risk of damage it is important to maintain a dust and dirt free working environment. Nitrile and lint free cotton gloves reduce the risk of damage to the disc by way of fingerprints or scratches, and it is suggested that technicians use them when handling optical disc media.

Physical Intervention

Error caused by scratches are problematic, though not necessarily hopeless. It is possible to smooth out these scratches with the proper use of an abrasive. This type of physical intervention was not attempted on any discs from the VHP collection, though the processes were tested on other discs. Plastic polish works the best, though metal polish or even toothpaste will work.[32] Begin by cleaning the disc with cold water and buffing it dry with a microfiber cloth. Make sure to buff the disc from the center to the edge, and not in a circular motion, as buffing in a circular motion can cause scratches that follow the path of the laser. Once the disc is dry, apply the abrasive, allow it to dry, and buff it away.

Polishing the disc will not help if a scratch is deep enough to damage the reflective metal layer, or the dye layer of a writable disc. If this type of damage is larger than the maximum correctable burst length then it is beyond repair. This type of damage is more common on CDs, as the reflective metal layer is located at the edge of the polycarbonate layer where the label is applied over a very thin and soft lacquer coating. On DVDs, the reflective metal layer is embedded inside of the polycarbonate layer, which offers more protection to the data from physical damage.

Coding Errors

The 39 other fails at this stage were due to coding errors. Coding errors thus far have come in two different types. The first is that the discs are not properly encoded as DVD-Video discs, and instead are Data DVDs containing video files. The second type of error is that, due to some error in the encoding, the disc is not readable by the tools we are using, despite lacking any physical evidence of damage.

Record 024782 exhibited the first type of error, and could not be cloned and extracted with the established method because the disc was a Data DVD with containing a .wmv video file. In any case where the DVD contains a video file, this file can be converted to an MPEG-2 in order to maintain consistency within the digital collection. However, due to the likelihood that many different types of video files could appear on these discs, a decision tree should be developed to guide users through the process of converting an arbitrary video file to an MPEG-2.

[32] Only the white, lightly gritty toothpaste works for this purpose. The smooth gel toothpastes will not work, as they are not abrasive enough to polish the polycarbonate layer of the disk.

Record 037269 exhibited the second type of error, as it showed no signs of physical defect yet would not play normally and could not be cloned at all using hdiutil. This behavior led us to the conclusion that it suffers from a coding defect rather than a physical defect. Our initial examination was unable to clarify exactly what caused the defect, though as we move through the collection we hope to collect more data about these DVDs to make a clearer diagnosis of the issue. Despite the unclear source of the problem, we were able to successfully clone the image using ddrescue and extract a playable MPEG-2 file using MPEG Streamclip.

Other errors

Record 030894 exhibited another type of issue, in that the content was shot at a 16:9 (often referred to as “widescreen”) aspect ratio. In a typical context, this is not a problem. While the aspect ratio of DVD-Video content is typically 4:3, it is possible for 16:9 content to be encoded onto and played from a DVD-Video. This is achieved by squeezing the pixels in the 16:9 video to fit in a 4:3 frame. If the squeezed 16:9 video were to be played back in a 4:3 ratio, the image would appear horizontally “squished” to the viewer. To get around this, the decoder on the DVD player is told to stretch the pixels out, thus displaying the correct 16:9 frame. DVDs with these characteristics are referred to as “Anamorphic DVDs”.

Dealing with anamorphic files in most encoding tool is not complicated. The user must specify that the aspect ratio of the output file be 16:9 rather than the typical 4:3. While this is simple enough, it may be difficult to incorporate into automated systems that are running a specific encoding profile. Thus, it is important to be able to detect whether the file is anamorphic or not before sending it off to be extracted. This can be done with any tool that can read technical metadata from media files and return it as text in a command-line environment. MediaInfo and FFmpeg are both useful tools in this regard. What follows are two command-line strings that will return the aspect ratio of a VOB file in the form of “x:y”. This can then be used in automation tools to route the video to the correct encoding process.

MediaInfo:

```
$ mediainfo --language=raw -f [input.VOB] | grep
DisplayAspectRatio/String | awk '{print $3}'
```

FFmpeg:

```
$ ffmpeg -i [input.VOB] 2>&1 | grep DAR | awk '{print $11}'
```

This process will only work if the disc is correctly encoded at 16:9. It is possible for the disc to be improperly encoded, so that while the video content was shot 16:9, the content was encoded onto the disc as a 4:3. If this is the case, automated systems that can detect the aspect ratio will fail to correctly extract the video, and the output will be squished.

Advice about possible automation

We have found that a number of the processes involved in cloning the discs and converting the images can be automated. There are, however, a number of limitations with the tools we have tested so far that are preventing us from creating a fully automated process.

We have found that hdiutil can be run as many times on one machine as that machine has optical drives, and that the process of cloning the discs only puts a small load on the CPU of the computer being used. This means that it is possible to hook up as many optical drives as the machine has USB ports in order to clone multiple discs at once. The main limitation of this strategy is that hdiutil requires the disc name to clone the image. DVDs contain a Primary Volume Descriptor which functions as a title. Some DVD authoring software will default to titling discs “My_DVD” or “My_Disc”, or “Video_Disc”. This can be problematic if there are two discs with the same name. In this case, the discs will be unable to mount with the same name, and they will be auto-incremented by the operating system to avoid the issue. For example, if there are three discs with the title “MY_DISC” mounted simultaneously, the first will be named “MY_DISC”, the second “MY_DISC-1”, and the third “MY_DICS-2”. This prevents the cloned ISOs from overwriting each other, but adds a layer of confusion to the process. The best solution to this issue to have an alert operator managing the clone station.

As mentioned earlier, the Ripstation robots have a 1 Gigabit/second connection which can be used to drop the output ISO files in a network watch folder. From here, automated scripts can be used to extract, concatenate, and transcode the VOB files in these ISOs. An institution willing to invest time and R&D resources could certainly create fully automated cloning, extracting, and nesting workflow using a Ripstation robot.

Due to the range of errors that may be encountered while cloning and extracting, users should be aware that any automation put in place will likely run into discs that it cannot process. In these cases, the discs should be set aside to be dealt with manually. It is possible to invest resources in an automated system that can handle all of these errors, though it is often a better use of resources to build a robust automation that can properly handle discs that exhibit the expected behavior, leaving the problem discs to be processed by hand. With this in mind, it is important to have a thorough method of detecting problematic discs, whether the processes are automated or manual. Training Operators and Quality Assurance staff to recognize, identify and diagnose potential problem discs is crucial to a successful process.

Additional Considerations

ISO File Systems

Data that is held on optical discs vary in much the same way an HDD can be formatted to be NTFS, HFS+, WFS, exFAT, and so on. The file system on an optical disc may change depending on the type of disc and the data on it. It is important to keep the file system of the original disc in mind when cloning the ISO image

As a part of our tests, we cloned a single disc using the cloning method programs discussed earlier. We also cloned the same disc to a variety of file systems (HFS+, ISO, Joliet, and UDF) using hdiutil. Once the ISOs were cloned, we generated checksums for each ISO file. A number of cloning methods created ISO files with different checksums. However, we found that generating checksums on the VOB data contained by the ISO after mounting the image showed that data held by each ISO file was still identical regardless of file system selection. This

suggests that the video data in the ISOs remains intact regardless of the file system or cloning method.

Our findings were that both dd and ddrescue maintain the original file system of the disc in the ISO file without any extra user input. However, hdiutil must be told what the output file system should be through the use of flags (thus, the inclusion of the -udf flag in our sample string). ISO files cloned by dd and ddrescue are in not identical, which can be seen by comparing the checksums generated from each file.

Additionally, the data inside the ISO files is not changed when the file system of the ISO is changed. This can be demonstrated by using hdiutil to convert an ISO from a Joliet file system to a UDF file system and generating checksums on the VOB files contained by the mounted images. In this case, the VOBs will have identical checksums, indicating that the data remains intact despite the change in file system.

This means that is possible to adjust the file system of the of the ISO to provide maximum compatibility between an ISO and the user environment without compromising the data. However, it is recommended that the ISO files initially be created using the same file system as the original disc whenever possible.

Mounting ISO Files

In order to access the data in an ISO file, the ISO must be mounted. Mounting an ISO makes your computer act as if the ISO is on a disc that has been inserted in an optical drive, allowing the user to access the information via a virtual drive.

Mounting an ISO file on a Macintosh is very simple. The ISO simply needs to be double-clicked on as if it were a standard executable file. This will automatically mount the ISO, allowing the user to access the data. This can also be completed through the command line with the following command:

```
$ hdiutil mount [path to iso file]
```

Mounting an ISO file on a Windows computer is significantly more complex, though it can be done. Windows operating systems cannot mount ISO files without extra software. Virtual CloneDrive is a freeware program that will allow you to mount ISO files.[33] Once this program is installed and running, any ISO file can be mounted on your computer simply by right-clicking the file and selecting “Open With -> Mount files With Virtual CloneDrive”.

[33] The program can be downloaded here: http://www.slysoft.com/en/download.html

Encoding and Decoding MPEG-2 Files

Audio Challenges

One challenge with this strategy concerns the audio on a DVD-Video file. The most commonly used encoding Dolby Digital (AC-3) is not supported within the MPEG-2 Program Stream[34]. Simply extracting Dolby Digital from the DVD-Video disc is possible; however, due to technical issues in data framing, the audio and video will play out of sync when not supported by the multiplexing designed into the DVD, which, among other things, controls for the synchronization.

For this reason the AC-3 is transcoded to MPEG-2 *audio* for highest compatibility in the MPEG-2 program file.

Encoder Challenges and Preservation Considerations

The process of creating MPEG-2 files from VOB became a major challenge due to the limitations in support for the MPEG-2 codec, as well as user expectations for the files. In the end, the user expectations acted to inform which tools were used for encoding the MPEG-2 files.

If the user defines the MPEG-2 files as *production masters* then the encoding process will be different than if the files are defined as *access copies* or some other sort of digital derivative. The main difference between the paradigms of *preservation* and *access* being that preservation files should be as true to the original format as is possible, potentially at the expense of easy access; while access files may be changed or manipulated to allow for broader interoperability across systems.

With this difference in mind, our first approach to creating MPEG-2 files from the ISO files was to use FFmpeg to extract MPEG-2 files form the VOB files inside of the ISO files. The FFmpeg string used to do this transformation is as follows:

```
$ ffmpeg -i [input file] -vcodec copy -acodec mp2 -q:a 0 [output file]
```

This string copies the video data without transcoding it to the output file, and transcodes the audio to MP2 without adjusting the quality at all. This means that the video portion is completely untouched, while the audio portion is adjusted to fit within the MPEG-2 program stream, as well as to avoid playback issues caused by the AC-3 audio format. AC-3 is a proprietary format requiring a license from Dolby Laboratories, Inc. The license fee is normally paid when a DVD is manufactured. No fee is paid when a DVD is cloned to an ISO image. To have the playback codec, each user station would have to pay a license fee.

The following is a breakdown of the flags used in this string:

[34] MPEG-2 Program Stream (MPEG-PS) refers to the standard by which the data in VOB files is organized. They contain the multiplexed audio and video data, as well as timing data, in a standardized structure which is decoded by the DVD player upon playback. This stream was designed by the Moving Picture Experts Group (MPEG) based on the needs of optical media.

- `-i [input file]`: Specifies the input file.
- `-vcodec copy`: Specifies that the vcodec for the output file should copy the input file. Because all VOB files contain an MPEG-2 program stream, if this string is used with a VOB file is will always put out digital video file encoded with the MPEG-2 standard.
- `-acodec mp2 -q:a 0`: This forces the audio output to be transcoded to MPEG-2 audio. Lynda Schmitz Fuhrig at the Digital Services Division of the Smithsonian Institute Archive has found that if left to its own devices, FFmpeg will always encode the audio of a VOB to MPEG-2.[35] With that in mind, this string chooses to define the output as such. The `-q:a 0` portion specifies that the quality of the output audio should not be adjusted.
- `[output file]`: This specifies the path of the output file.

While this method is certainly the most preservation-friendly approach, the files extracted by this method are not access ready and present a number of playback issues across multiple platforms. These files had significant playback errors when opened with QuickTime X and Quicktime 7 on Mac OS X 10.5 and 10.6, as well as Windows Media Player on Windows XP and Windows 7. The most general playback error across these systems is that the files do not playback with sound or video. In some cases, one element will playback while the other does not. In other cases, both elements play back, but begin to fall out of sync. There have been cases where files extracted with this method do playback on these systems, though it has been difficult to find any consistent details between these files that would indicate what about them makes them play back while others do not. This experience demonstrates the range of interoperability issues associated with MPEG2 industry wide. One would expect the modality of interoperability failure to be common, that the files would always behave the same way, for example with video playback but no audio playback, or audio and video playing out of sync. This is not the case. Despite literally bit-for-bit reformatting of the video, and identical processing of the audio, the behavior varies.

The second approach is to extract files that are geared towards general access, and leaving the ISO file to act as the sole preservation file. As The Library has requested MPEG-2 files, our efforts were focused on creating MPEG-2 files that can be played on an arbitrary system. The FFmpeg string discussed on page 20 of this report addresses this access concern, and was used in order to create MPEG-2 files that played consistently across multiple platforms. However, this was done at the cost of changing many attributes of the original video, making it no longer useful as a preservation file.

All of the methods for extracting video files from ISOs mentioned in this paper were focused on creating access ready files. The aforementioned methods have solved the problem of creating MPEG-2 files that play across multiple systems. However, it is worthwhile for institutions to investigate potential access file formats and choose the one that works best with their existing

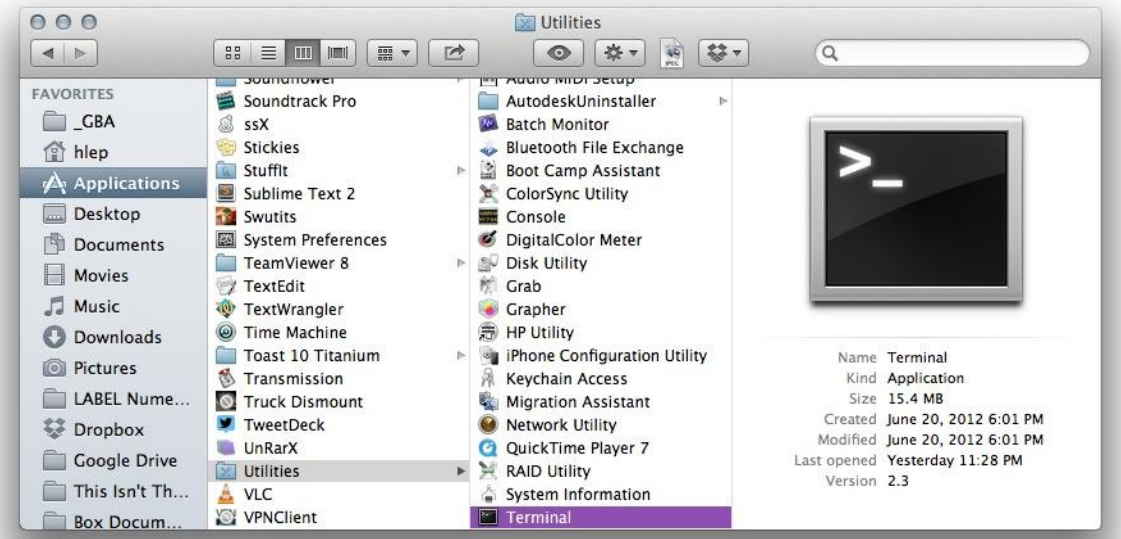
[35] Whether this is due to the proprietary license on the AC-3 format, or the sync/framing issues thereof is unclear.

infrastructure. All of the programs discussed in this report are capable of creating MPEG-4 video files, which generally have less interoperability and playback issues than MPEG-2 files. Any institution looking to extract access ready files from DVD-Video discs should make sure to consider what it is they need out of their access files before embarking on a large-scale migration initiative.

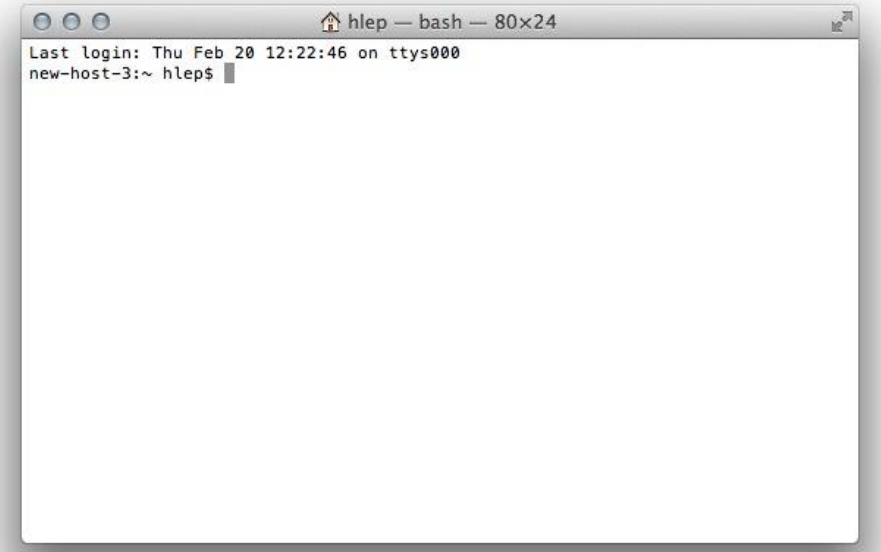
Appendix A : Understanding, Accessing, and Using the Command Line on a Mac

In the early days of computing all computers existed solely in the Command Line Interface (CLI). Early operating systems, such as MS-DOS and UNIX offered only text-based interfaces, in which the user was expected to control the processes through text commands. Upon Xerox PARC's development of the first Graphic User Interface (GUIs) in 1973, users became able to interact with the computer processes in the visually intuitive manner that we are familiar with today. CLIs and GUIs are both abstractions of the actual processes a computer is performing. However, CLIs are generally less abstracted from the internal processes, and allow for more power and flexibility at the expense of ease-of-use. In fact, many GUI programs are built on already existing CLI programs (MPEG Streamclip being built on top of FFmpeg being an example).

On Mac computers the command line can be accessed through a program named Terminal. Terminal can be found in the Applications/Utilities/ folder (see image below), or by typing "Terminal" into the Spotlight search bar.



Once terminal is opened, a CLI is available to the user. This is the window in the which all command line commands and programs are run from.



This window allows you to type commands and control your computer. You can highlight text in this window to copy and paste, but you cannot select the cursor position with the mouse. Rather, you will have to use the arrow keys on your keyboard to move the cursor. If you need to quickly move your cursor to either the beginning or the end of a command you can press Control-a or Control-e on your keyboard to move to both respectively.

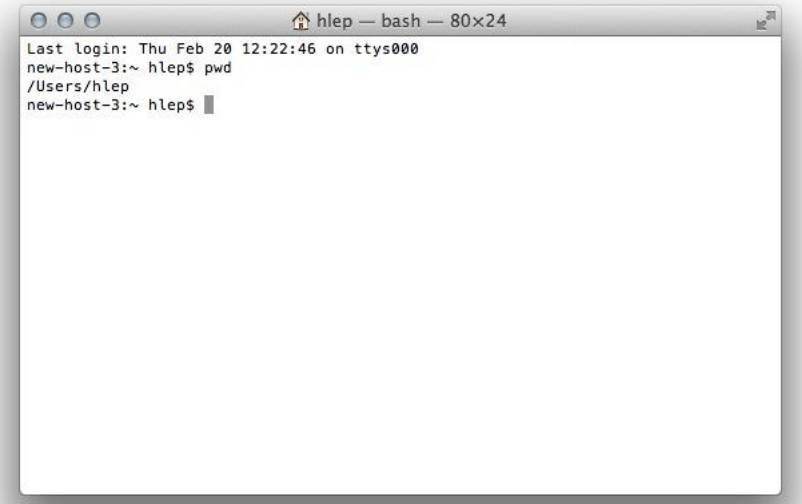
Throughout this paper, a number of commands will be discussed. These commands will appear in the following format:

```
$ commandname -flag
```

The commands will appear in `Courier New` font for the sake of readability. The `$` does not need to be typed and is not part of a command, it simply denotes that a new command is being entered. A flag is signified by a dash. Flags modify the command in some way. Some flags will output the information in a different format, or will change the processing methods of the command. If you wish to experiment with this interface, type the following command into the terminal window:

```
$ pwd
```

To Terminal, this command means *Print Working Directory*, which asks Terminal to print the directory that Terminal is currently inside.

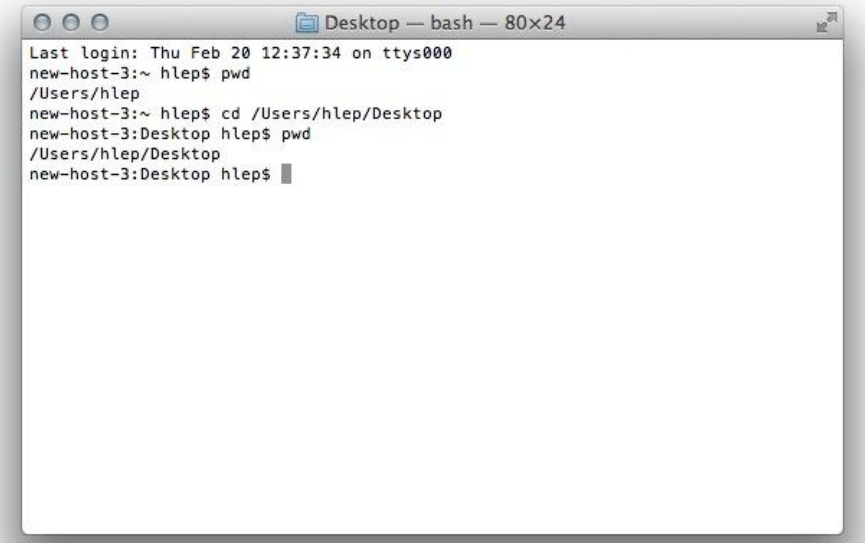


Terminal responds with a UNIX path, in which folder names are separated by slashes (/). To change the directory, use the following command

```
$ cd [new path]
```

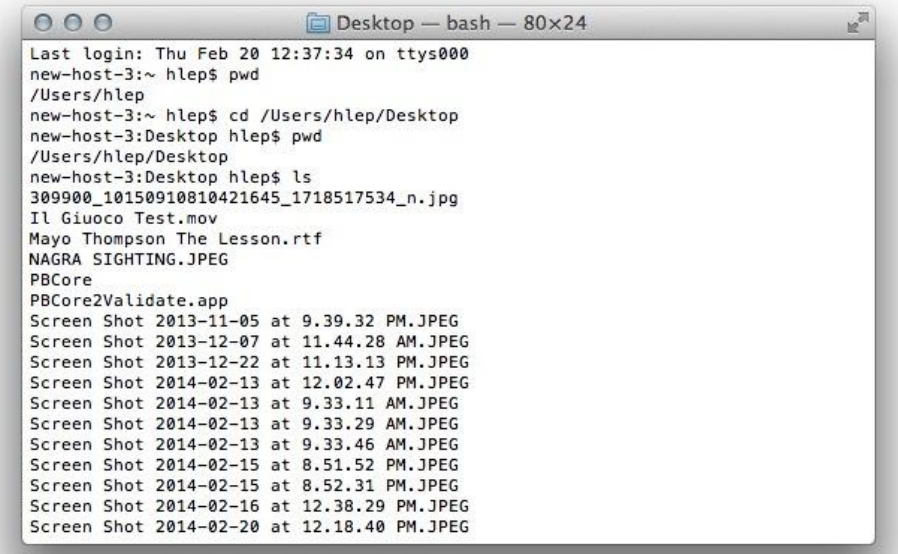
cd means *Change Directory*. Throughout this report, the parts of command line commands in `[square brackets]` are to be filled in by the user. If you wish to change the directory to you Desktop folder, you can use the following command

```
$ cd /Users/hlep/Desktop
```



In this case “hlep” is the user name. You can also list all of the files in the working directory with the ls command:

```
$ ls
```



Sometimes you will need to run a process as an administrator due to permissions. In order to do this you need to start the command with `sudo.` You will be asked for a password. If you do not know the password you will need to ask your IT systems administrator for it.

Another useful feature of Terminal is that it will automatically enter the path of any file or folder that you drag-and-drop into it. This is helpful since Terminal does not like space, and often you will find a space in a file path. If a folder path has a space in it and you want type out the folder path rather than drag-and-drop the file into the terminal window; you will have to either put quotes around the path, or escape the spaces with a backslash. Here is an example:

If you want to navigate to the directory /GBAUDIO/Project/Video/DVD Videos/ then it would be INCORRECT to type in the following command:

```
$ cd GBAUDIO/Project/Video/DVD Videos/
```

If you type this, Terminal will attempt to navigate to /GBAUDIO/Project/Video/DVD because of the space in the “DVD Videos” folder. Instead you will need to type in one of the following three command.

```
$ cd GBAUDIO/Project/Video/DVD\ Videos/
$ cd ‘GBAUDIO/Project/Video/DVD Videos/’
$ cd “GBAUDIO/Project/Video/DVD Videos/”
```

In the first example the backslash before the space tell terminal to render the space in the file path, rather than thinking what comes after the space is a new command. The other two examples use quotes to tell terminal to ignore the space. Any of these will work, however if you find yourself confused remember you can drag-and-drop a file or folder into the Terminal window to get its path.

Lastly, the `$ man` command will pull up the manual on any command. If you are ever lost with a command, you can type in `$ man [command name]` to see the manual of the command in question. For example, the following command will give you the manual on the `ls` command:

```
$ man ls
```

Try running `man` on any commands that are giving you trouble and take the time to read through the manual. This will help clarify any troubles you have with the command, as well as get you practice working with terminal.

Appendix B: Installing ddrescue on a Mac computer

This appendix has instructions for installing ddrescue on a Mac computer. As with the rest of the report, any text that appears in `Courier New` is meant to represent text typed into Terminal, and any text that appears in `[square brackets]` is a placeholder for text to be filled out by the user.

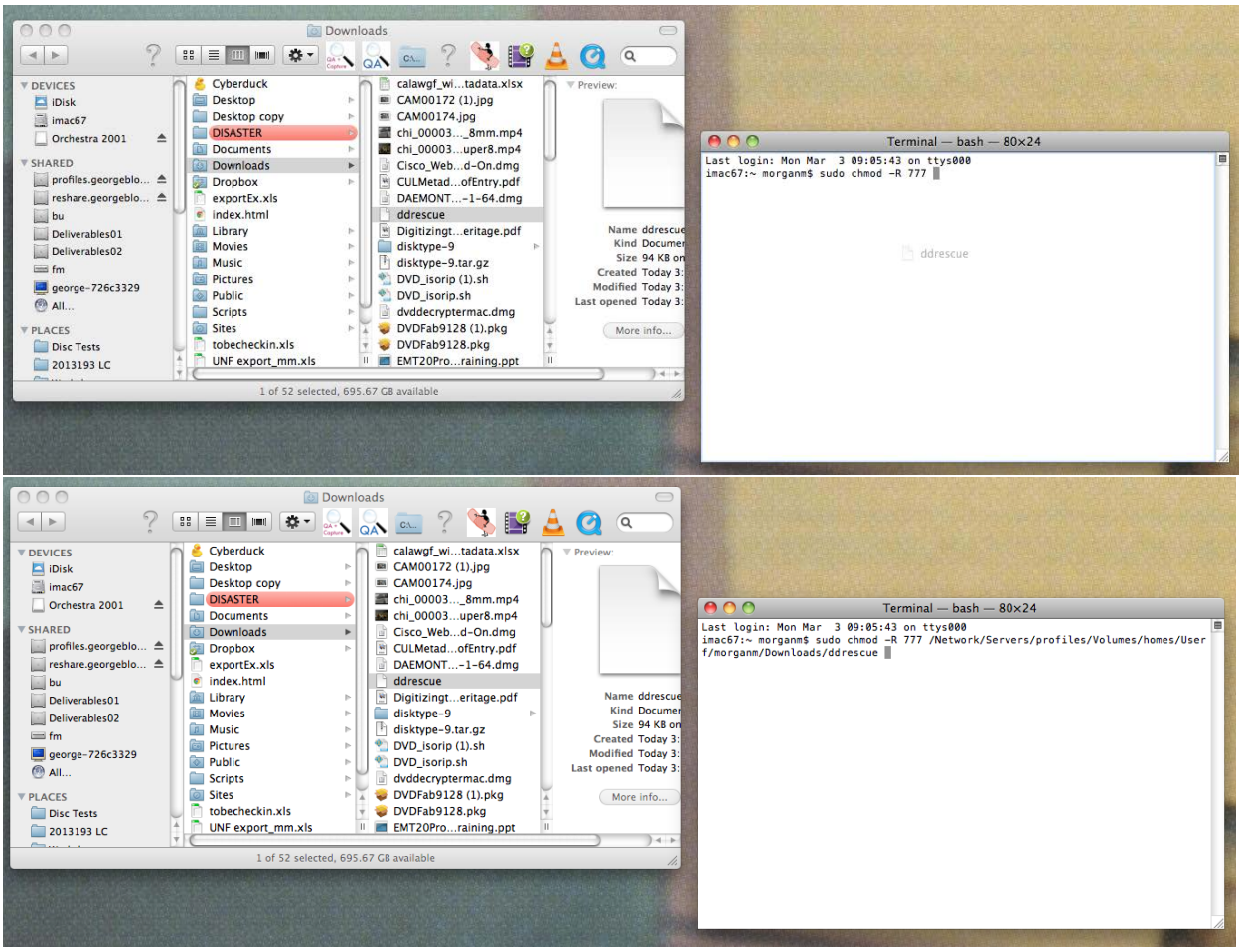
The ddrescue source code is available for download on the internet. However, compiling and installing it from source code is rather difficult. In order to get around this, we have provided a executable file for download here: www.georgeblood.com/scripts/ddrescue.

Note: in some browsers the link will download automatically, while in others you may have to right-click the link and select "Save Link As…"

Once you have downloaded the ddrescue file, you must open the permissions on it. You can do this using the `chmod` command, which can by done by typing the following into terminal:

```
$ sudo chmod -R 777 [path to ddrescue]
```

Remember that you can drag the file from Finder into the Terminal window to get the path to it. The two screenshots illustrate how this works:



Note: Running any command with `sudo` *before it causes the command to be run as an administrator. Terminal will prompt you for a password when you do this, so you will need administrative access to run these commands. Be careful when doing this, as it is possible to delete files that your computer needs to run, though you will be safe as long as you follow the directions closely. If you do not know the administrator password, please contact your IT System Administrator and ask them to help you with this process.*

Now it is time to move the file to your `/usr/bin` directory so it will be run every time you type `$ ddrescue` into the command line. To do this type the following into the command line:

```
$ sudo mv [path to ddrescue] /usr/bin/ddrescue
```

That's it. You should now be able to type `$ ddrescue` and see the following response:

```
ddrescue: Both input and output files must be specified.
Try 'ddrescue --help' for more information.
```

This means you have installed ddrescue properly. You can now type `$ ddrescue --help` to view the manual, or you can go to Appendix C for a walkthrough on how to use ddrescue cloning DVDs.

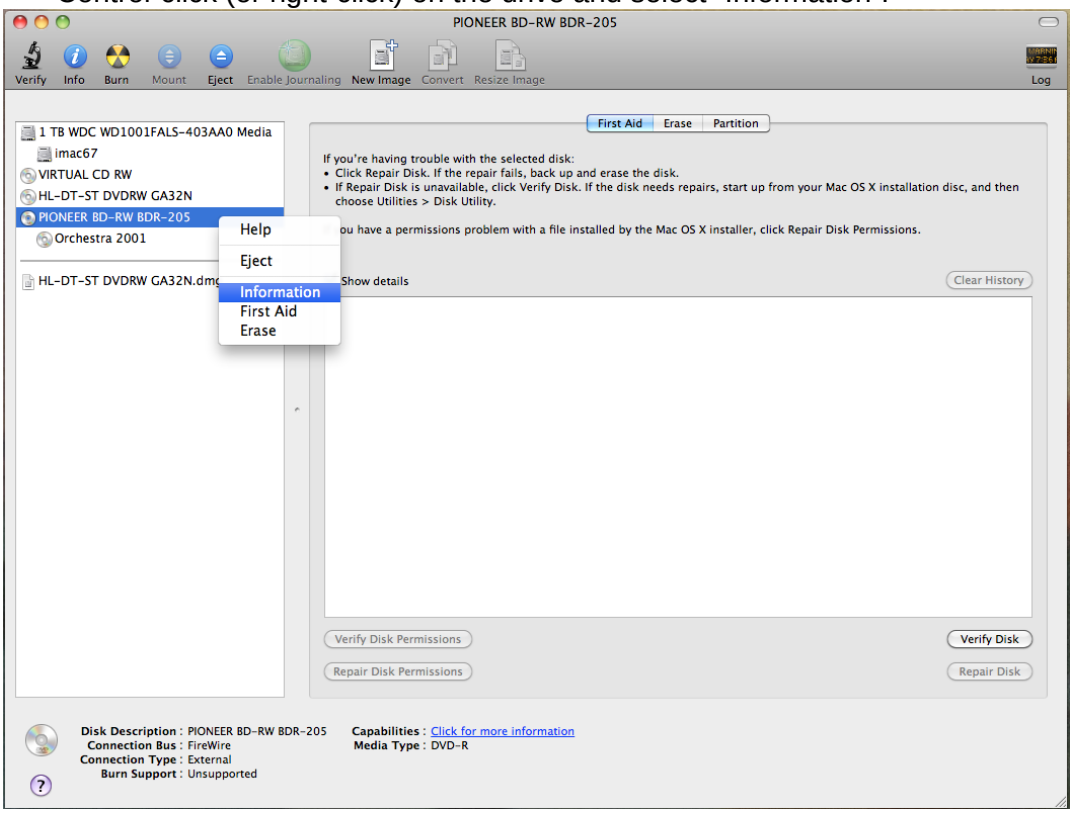
Appendix C: Cloning ISOs from DVD-Video discs

1. First Method: *Use ddrescue to make an image of a disc*

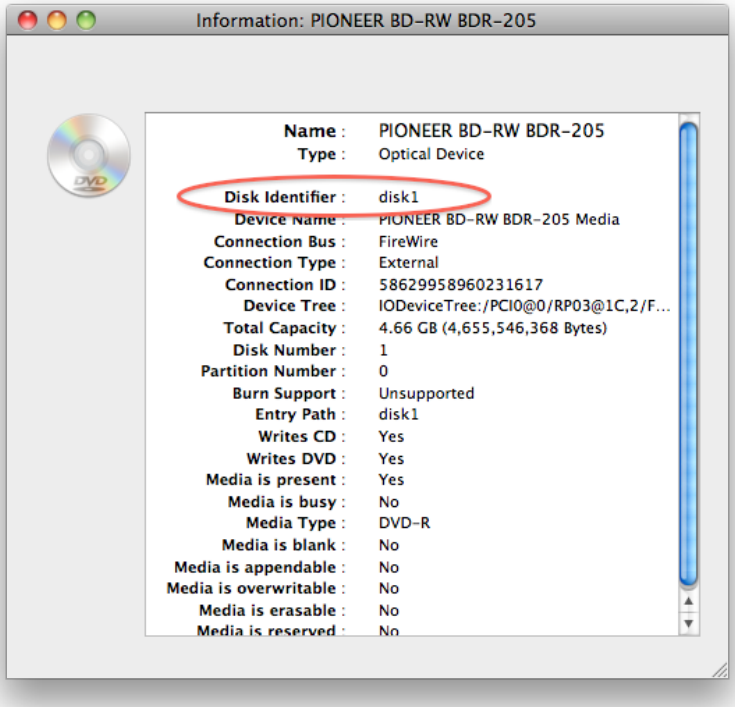
This method will run ddrescue in such a way that it attempts to fix any errors it finds during its first pass. This may take a very long time if the disc is damaged or the data is corrupted. If you wish for ddrescue to skip over errors please use the second method. As with the rest of the report, any text that appears in `Courier New` is meant to represent terminal commands, and any text that appears in `[square brackets]` is a placeholder for text to be filled out by the user. In this case the square brackets themselves do not need to be typed, but the information they represent needs to be present. For more information on using and accessing the command line on a Mac computer please see Appendix B.

- Load the DVD into an Apple computer.
- Launch Terminal.
    - For an introduction to opening and using Terminal, see Appendix A .
- Make sure that you have a binary of ddrescue in your /usr/bin folder. This will allow you to use ddrescue simply by typing `$ ddrescue` in the terminal window.
    - To see if it is installed, type `$ ddrescue` in the terminal window. If the response is `-bash: ddrescue: command not found` then you will need to install ddrescue, please see Appendix B.
- Find the name and path of your optical disc reader. There are two ways to do this, one using the Disk Utility GUI, and one using Terminal.
- Here is the first method, using Disk Utility:
    - Open Disk Utility. It is located at “/Applications/Utilities/Disk Utility”. You can also search for it in Spotlight if you have trouble locating it.
    - Select the disc drive you want to find the name of from the sidebar on the left side of the window.

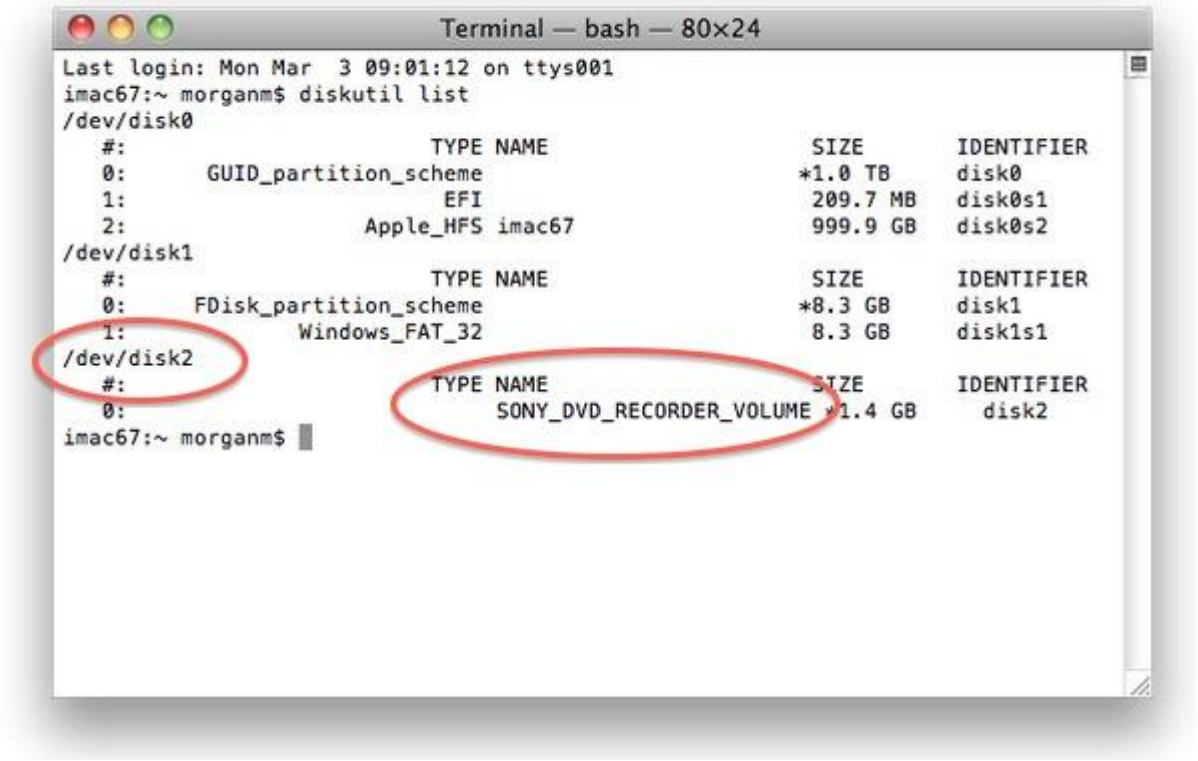
- Control-click (or right-click) on the drive and select “Information”.



- A window will appear that contains the name of the drive. In this example, the drive is named “disk1”.



    - This means that the path to this disc is “/dev/disk1”. The path to an optical drive will always be “/dev/[name of drive]”.
- Here is the second method, using Terminal:
    - Type the following command into Terminal:
    - `$ diskutil list`
    - This command will return a list like the following screenshot.



    - In this list, look for the device whose name is the name of the DVD that you have inserted, then look for the path of that device, which will be to the left and above the name. In the case of this screenshot, the path and name of the optical drive is `/dev/disk2`. This is typically what the name of the internal drive on an iMac will be, though it may change across machines. It is a good idea to check before you begin ripping. Throughout this walkthrough you will see `[path to optical drive]`, which refers in this case to the text string `/dev/disk2`
- Whichever method you choose to find the path of the optical drive, make sure to write it down and remember it. You will need to use it every time you run ddrescue.
- Unmount the DVD drive with the following:
- `$ diskutil unmount [path to optical drive]`
    - This will unmount the optical drive, making it disappear from the finder sidebar. This needs to be done so that ddrescue can access the disc drive.
- Run ddrescue with the following:
- `$ ddrescue -b 2048 -v [path to optical drive] [output ISO path] [output log path]`
    - `-b` 2048 defines the sector or block size, which is 2048 for DVDs.
    - `-v` puts ddrescue in verbose mode, so you can see what is happening.
    - ddrescue will name the output ISO whatever you specify in the output path and name, however the output ISO path should end in “.iso”. Also, it is generally a good idea to name the ISO file after the name of the disc, which in the case of

the screenshot above is SONY_DVD_RECORDER_VOLUME Here is an example of a well-formed path:

```
/Volumes/GBAUDIO/Video/Projects/SONY_DVD_RECORDER_VOLUME.iso
```

- The easiest way to create a new ISO file is to drag the folder you want it to be in into the Terminal window, hit the *backspace* button, add a slash (/) character, then type in the name of the output ISO file.
- The name of the ISO file can be whatever you want, though we suggest that you make it unique so that it does not get confused with or overwrite any other ISO files.

- the log path is optional, though it will allow you to run analysis on the log at a later time. In order to keep file names consistent, always be sure to make the log path the same as the ISO path, but with ".log" instead of ".iso" as the extension. Here is an example of a well-formed log path:

```
/Volumes/GBAUDIO/Video/Projects/SONY_DVD_RECORDER_VOLUME.log
```

- Allow ddrescue to run until it has completed. You will know it is done when the terminal window says `Finished.`

- Eject the disc with the following command:
- `$ drutil eject [path to optical drive]`
  - You will need to use this method to eject the disc, as the disc drive was unmounted earlier and the standard method of using the Eject button will not work.

Cloning the ISO from the disc will generally take between 5 and 10 minutes if the disc is in fair condition. However, if there is physical damage to the disc or it is otherwise difficult to read, this process may take hours or even days as ddrescue attempts many different methods to salvage data from the disc.

If you see that ddrescue has spent a considerable amount of time `splitting failed blocks` you may want to abort the process. You can safely quit ddrescue by pressing Ctrl+c in the terminal window and you will still have most of the salvageable data from the source disc intact. If ddrescue is spending a long time (hours or more) trying to split failed blocks it is a clear sign that some of the source disc is particularly difficult to read or damaged in some way. Quitting ddrescue here means that not all of the data from the disc has been cloned, but there is also no guarantee that ddrescue will be able to save all of the data even if it ran for a number of weeks. As long as you have saved a log file, which keeps tracks of the errors that ddrescue could not salvage, you can cancel the process and re-attempt cloning the disc using Method 3.

2. Second Method: *Run ddrescue, skipping errors*

This method will skip all erroneous sectors. Data about these bad sectors will be kept in the log file, which can be referenced at a later time by ddrescue. This method should only be used in an automation workflow, where ddrescue is not expected to run for a very long time. This method should not be expected to create working ISO files from damaged discs. For very damaged discs, please refer to the third method.

- Load the DVD into an Apple computer.
- Launch Terminal.
- Unmount the DVD drive with the following:
- `$ diskutil unmount [path to optical drive]`
- Run the following command:
- `$ ddrescue -b 2048 -n -v [path to optical drive] [output ISO path] [output log path]`
    - `-n` tells ddrescue to skip skipped blocks.
- ddrescue will tell you if there are any errors, but will not try to recover split sectors.
- Eject the disc with the following command:
- `$ drutil eject[path to optical drive]`

3. Third Method: *Polish the disc and run ddrescue with multiple retries to try and rescue errors*

This method is to be used if a number of errors are found and the first method is unable to recover them all. Be sure to include a path to the log, which ddrescue will use to locate the bad sectors from the previous attempt.

- Rinse the DVD with cold water and buff it dry with a microfiber cloth.
    - Make sure to buff the disc from the center to the edge, and not in a circular motion, as buffing in a circular motion can cause scratches that follow the path of the laser.
- Once the disc is dry apply plastic polish (metal polish or toothpaste will work as well).
- Allow the polish it to dry, and buff it away.
- Load the DVD into an Apple computer.
- Launch Terminal.
- Unmount the DVD drive with the following:
- `$ diskutil unmount [path to optical drive]`
- Run the following command:
- `$ ddrescue -b 2048 -r4 -v [path to optical drive] [output ISO path] [output log path]`
    - `-r4` tells ddrescue to retry a total of 4 times. This can be adjusted however you like, but realize that more retries will take considerably longer, and you may never recover all of your data.
- Eject the disc with the following command:
- `$ drutil eject[path to optical drive]`

4. Fourth Method: *Run dd to create a working image with missing data.*

With some special flags, dd can be made to fill in data that it cannot recover. This will cause data to be lost, though it may be able to create a readable image when ddrescue cannot. This is because it will pad the bad sectors in order to maintain a correct file structure.

- Load the first DVD into an Apple computer.
- Launch Terminal.
- `$ dd if=[path to optical drive] of=[output path] conv=sync,noerror`
- dd expects the same path type as ddrescue, though it will not display any user dialogs until it is finished, which will take about 5 to 10 minutes.

5. Fifth Method: *Run ddrescue on two damaged copies to create one functioning ISO*

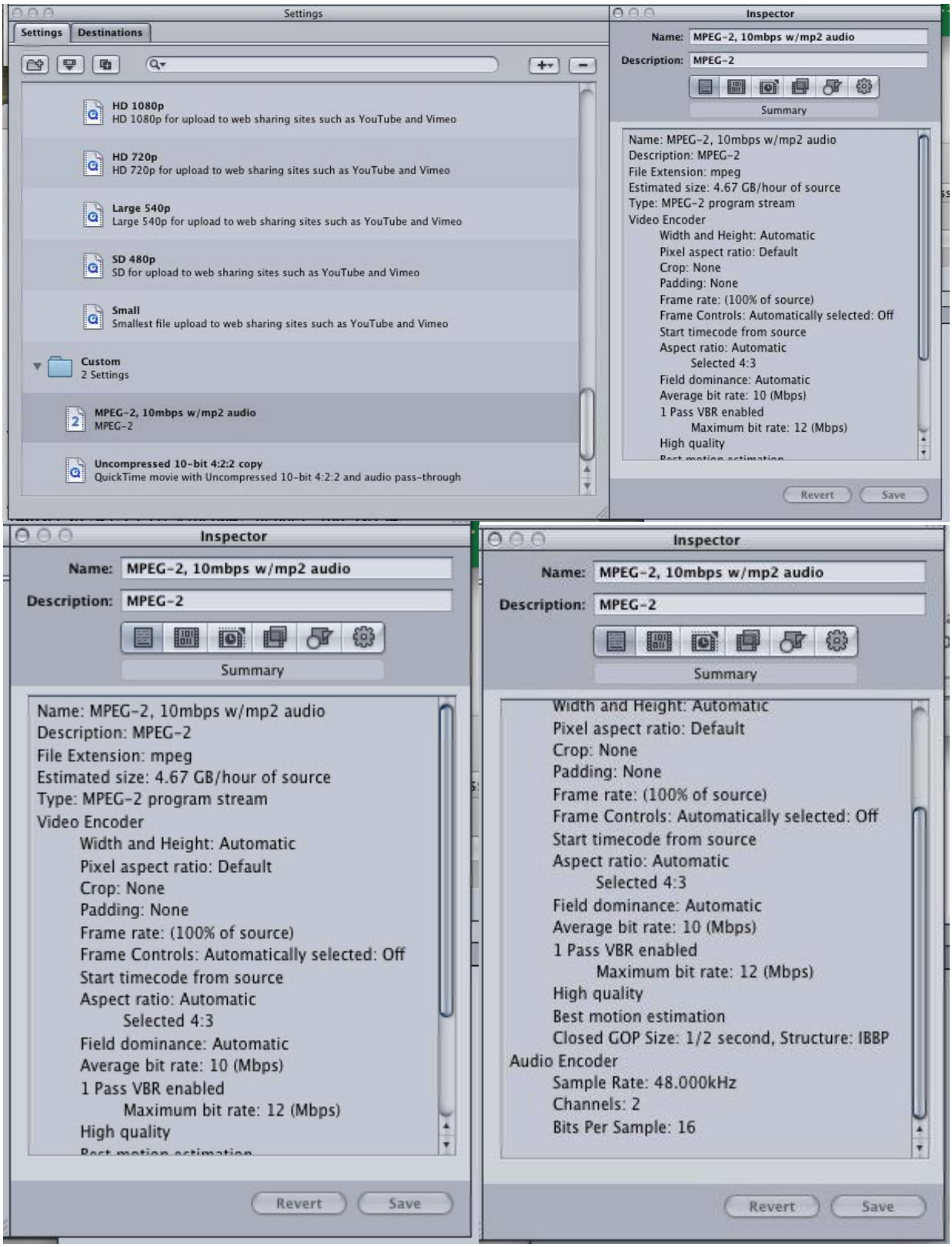
This method can only be used if there are multiple copies of a disc. If both discs are too damaged to clone a functioning ISO file, but are damaged in different areas, it may be possible to recover an ISO by combining the data from both of them into a single functioning ISO. ddrescue will try to create an ISO file from the first disc, and keep a log of the sectors it couldn't recover. When you run ddrescue on the second disc, it will immediately attempt to rescue the bad sectors from the first disc, compiling the information from both discs into a single ISO file. Keep in mind that this only works if both discs are identical.

- Load the first DVD into an Apple computer.
- Launch Terminal.
- Unmount the DVD drive with the following:
- `$ diskutil unmount [path to optical drive]`
- Run the following command (just like the second method):
- `$ ddrescue -b 2048 -n -v [path to optical drive] [output ISO path] [output log path]`
- If the first disc fails, try the following command (similar to the third method):
- `$ ddrescue -b 2048 -v [path to optical drive] [output ISO path] [output log path]`
- Eject the disc with the following command:
- `$ drutil eject[path to optical drive]`
- If you still cannot get a working image, insert the second copy of the disc.
- Unmount the DVD drive with the following:
- `$ diskutil unmount [path to optical drive]`
- Run the following command:
- `$ ddrescue -b 2048 -r1 -v [path to optical drive] [output ISO path] [output log path]`
- Eject the disc with the following command:
- `$ drutil eject[path to optical drive]`

Appendix D: Using MPEG Streamclip to Extract MPEG-2 files from the VOBs inside the ISO Images

- Mount the ISO Image on a transfer machine. On an Apple computer this means double-clicking the icon in Finder, which will mount the contents of the image just like an external hard drive in /Volumes.
    - Most of the time the mounted version of the image will appear in the Finder sidebar. If it does not appear right away you may need to adjust your Finder preferences.
- Open MPEG Streamclip.
- Open the MPEG Streamclip Batch List. List menu --> Batch List
- Open the mounted ISO image as you would a DVD in MPEG Streamclip, using File menu --> Open DVD…
    - If there is more than one title on the DVD, MPEG Streamclip will prompt you to select which DVD movie you want to open from a pull down menu. If there is only one title on the DVD this prompt will not appear.
- A dialog will appear asking, "Would you like to open this DVD, or batch convert it?" Click the "To Batch" button.
- A dialog will appear prompting you to choose a task.
    - Select "Convert to MPEG with MP2 Audio" from the pull down menu.
    - Check the "Fix timecode breaks" box.
    - Check the "Do not skip any frame" box.
    - When done, click "OK."
- A dialog will appear prompting you to select the destination folder. You cannot set output file names when using the Batch List; MPEG Streamclip forces you to use the input filenames, which, when working with ISO images made from DVDs, are always "VTS_01_01" or a sequentially numbered variant.
    - *WARNING*: If you are working with multiple instances of MPEG Streamclip on multiple computers, you will inevitably end up with duplicate filenames. If the discs are being cloned to a network drive there is a possibly of files being overwritten during this process. Being aware of this issue and having an attentive operator will help avoid files being overwritten.
- Note: If Apple's MPEG-2 playback codec is not installed on the computer a dialog will appear warning you that "The Apple QuickTime MPEG-2 Component is not installed…" with a number of additional details. If this is the case, you will not be able to see the video you have opened, but you can still extract the MPEG-2 video and transcode the audio because neither of these actions require said playback component.
- Click "Continue."
- The task will appear in the Batch List. If there was more than one movie on the DVD, repeat the process for each movie until they have all been queued in the Batch List.
- Click "Go" at the bottom of the Batch List and wait for MPEG Streamclip to finish the extraction and audio transcoding.
- Repeat the process for each ISO Image.

Appendix E: Compressor Settings for MPEG-2 Encoding

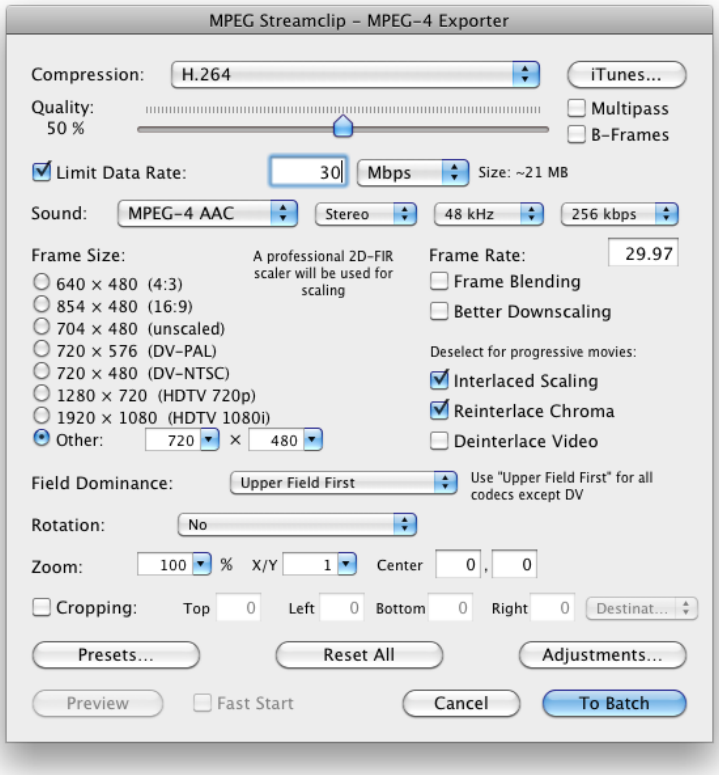


Appendix F: Methods of VOB Concatenation

In order to completely extract the video information in an ISO file and transcode it to any other video format, the VOB files in the ISO must be concatenated into a single VOB file. This single VOB file can then be transcoded to the desired output file type. This appendix includes three methods of concatenating VOB files.

Using MPEG Streamclip

The first method is to do so manually with MPEG Streamclip. This method has proven to be robust, though rather time consuming. The steps for this method are laid out in Appendix D, which explains how to concatenate the VOB files and extract MPEG-2 files from them. The only difference for creating any other sort of file from the VOB files would be to change the settings described in item **6a** of Appendix D. If you would like to extract high bitrate MP4 files for use as intermediate files, select “Export to MPEG-4” and use the following settings:



In the case that you want the MP4 files to be your final output, you can use the same settings, but with the Data Rate set to 3.5 Mbps.

Using FFmpeg

The second method is to use FFmpeg to concatenate the VOB using the `concat` method. This is a sample string from FFmpeg's online documentation:[36]

```
$ ffmpeg -f concat -i mylist.txt –acodec copy -vcodec copy [output]
```

In this example, `mylist.txt` is a text file containing a list of video files to be concatenated. The `–acodec copy –vcodec copy [output]` string specifies that the output is a video file whose audio and video format is the same as the input files. FFmpeg's `concat` method can be used with any video file, not just VOB files.

Peter Finkel at the Smithsonian Institution Archives has developed a .bat script to concatenate VOB files and transcode them to MPEG-2 files. The script copies the VOB files to a specified location and concatenates them into a single file, keeping the audio and video formats intact. That script is available here:

http://www.georgeblood.com/scripts/ffmpeg.bat

Be aware that this script has a few dependencies; including that the checksum utility JackSum be installed in a specific location. It may take some tweaking in order to have it run on another system. With this in mind, it may be more useful to institutions as a reference for how to build automation systems specific to their own platform.

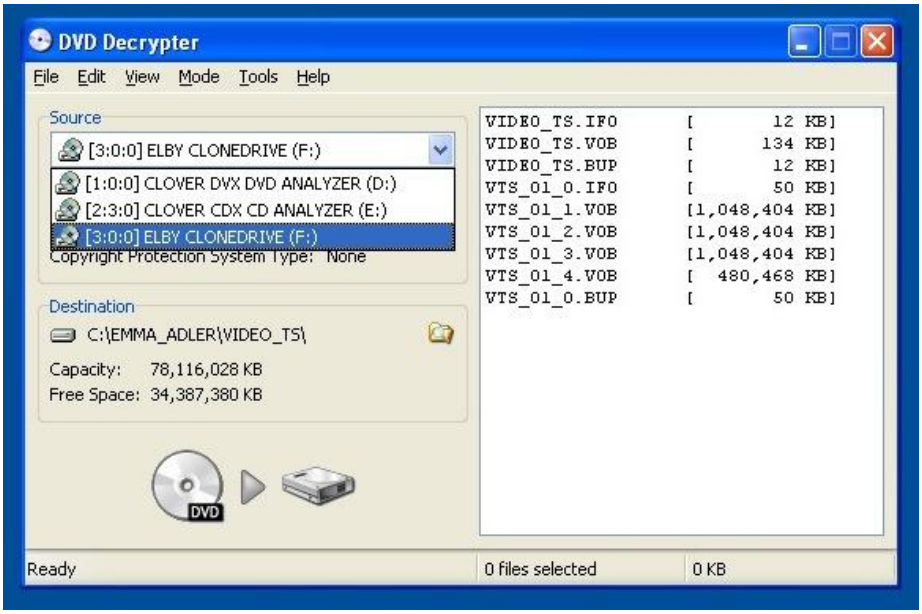
Using DVD Decrypter

Lastly, DVD Decrypter has an option to automatically concatenate the VOB files on a DVD-Video disc, and should be considered by any institution running on a Windows platform. The following are the steps to perform this process:

- Mount the ISO File (You will need a program such as Virtual CloneDrive installed in order to do this) or insert a DVD-Video disc.
- Open DVD Decrypter.

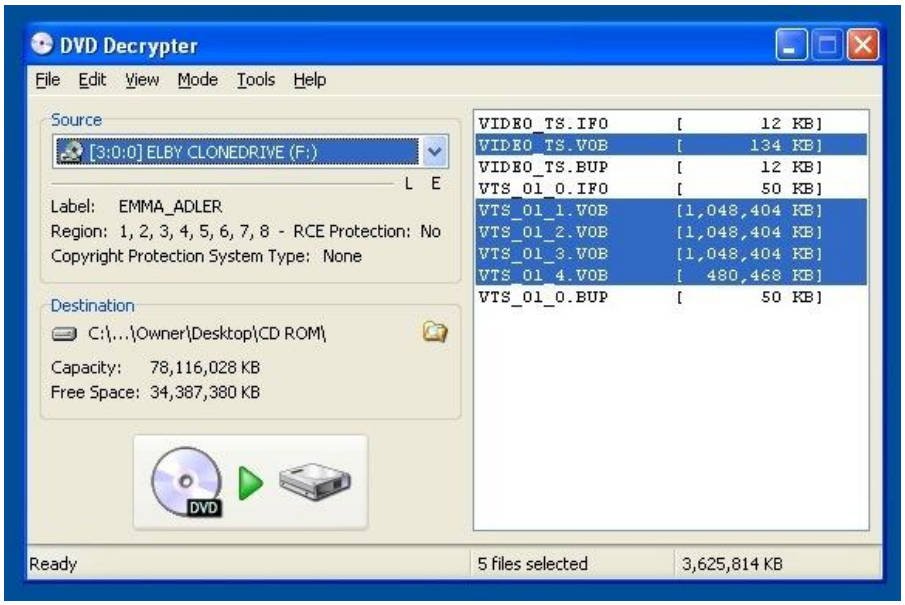
[36] http://trac.ffmpeg.org/wiki/How%20to%20concatenate%20(join,%20merge)%20media%20files

- Select the mounted drive from the Source drop-down menu.

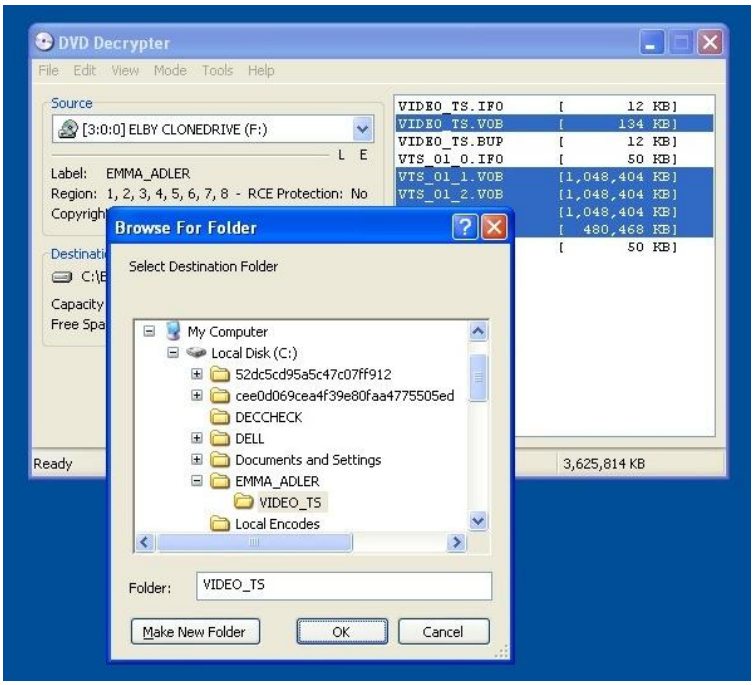


- Click on “Mode” at the top of the window and select “File”.

- Click on “Edit” at the top of the window and select “Select VOB Files”.



- Click on the folder icon under “Destination” and select an output directory.

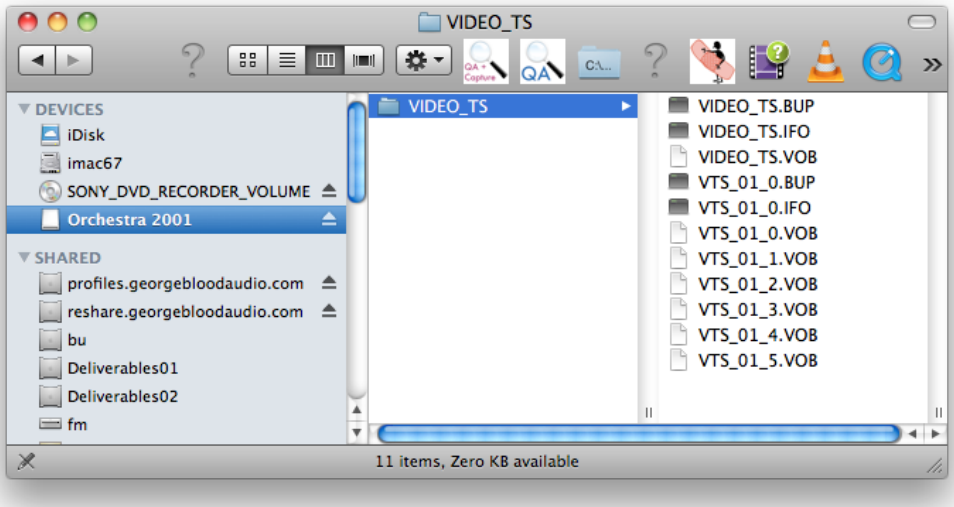


- Start the process by clicking the button at the bottom of the screen that has a DVD with a green arrow pointing to a hard disc. This begins the cloning process.

- When the process is finished you will have a single file named VTS_01_1.VOB that contains the video information from all the VOB files in the VIDEO_TS folder. This VOB file can now be transcoded to the desired output format.

Appendix G: The Structure of a VIDEO_TS folder

If you insert a DVD-Video disc into your optical drive, or mount an ISO image of a DVD-Video disc, and browse the disc as a directory, you will see that it contains a VIDEO_TS folder. This folder contains all of the data that makes the DVD playable in your DVD player, including the menu data, navigation data, video program data, and timing data.[37]



Inside of the VIDEO_TS folder there are three types of files, IFO, BUP, and VOB. The IFO files contain information that control playback and navigation on the DVD. The BUP is a backup version of the IFO, and contains the same information, which is read in the case that the IFO is damaged or unreadable. Both of these files must be present on a DVD for it to be compliant to the DVD-Video standard.[38]

The VOB (Video Object) file format is subset of the MPEG-2 file format. It contains Presentation Data and part of the Navigation Data in a multiplexed stream. The Presentation Data includes the video, audio and subtitles, while Navigation Data consists navigation and playback control (such as not permitting the user to skip ahead during trailers). The video data is encoded as MPEG-2, while the audio can be encoded as Linear PCM, AC-3, or MPEG. The VOB file format is subset of the MPEG-2 file format. A single VIDEO_TS folder often contains a number of VOBs, as each VOB is limited to 1GB according to the DVD-Video specification. VOBs on a DVD-Video disc have the file name VTS_XX_Y.VOB, where XX refers to the title number and Y refers to the part of the title. Any VOB where Y is 0 is a menu, and all other VOBs contain the main program. There can be up to 99 titles and 10 parts.[39] If each VOB file is concatenated to one another in order, the resulting video file would contain all of the video data on the disc.

---

[37] *Principles of Digital Audio,* Ken C. Pohlmann

[38] http://www.dvd-replica.com/DVD/vmgguide.php

[39] *Video Demystified: A Handbook for the Digital Engineer: 5th Edition*, Keith Jack

Appendix H: Working with DVD_isorip.sh

DVD_isorip.sh is a shell script used for cloning images from DVD. It reads the currently mounted optical disc and creates an ISO image using the following string:

```
$ hdiutil makehybrid -iso -udf -verbose -o [output path] [disc path]
```

CD_isorip.sh is a similar script, though it is meant for use with CDs. It uses the following string:

```
$ hdiutil makehybrid -iso -joliet -verbose -o [output path] [disc
path]
```

Both scripts are available on the GBAVF website:

http://georgeblood.com/scripts/DVD_isorip.sh
http://georgeblood.com/scripts/CD_isorip.sh

Currently, the script must be run through the Terminal window. To run the script, navigate to the directory the script is in, by typing

```
$ cd [path to directory]
```

Once you have directed Terminal to the directory that the script is in, you can run the script by typing `./DVD_isorip.sh` into the terminal window.

Both of these scripts currently require that a specific folder path exists for the ISO to be cloned to. This folder path is specific to GBAVF's internal workflow, and will likely not be compatible with other institutions. However, institutions looking to create their own automation systems may find it helpful to see what the coding for an existing automation platform looks like.

Another script developed by GBAVF for the purpose of cloning ISO files is ddrescue_helper.app. This is an AppleScript application that can be run on any Macintosh computer. The script is meant to be simple to run. It asks the user to choose the ddrescue options they would like to use, type the /dev path of the optical drive, choose the output path of the ISO file, and enter name of the ISO file. From there, it automatically opens a Terminal window and runs ddrescue with the information given to it. The script is available here:

http://www.georgeblood.com/scripts/ddrescue_helper.zip

List of References and Contributors

The following is a list of resources and manuals referenced throughout the creation of this report. These resources provide a good starting point for anybody looking to do more in-depth research on the topic of cloning, extracting, and authoring DVD video content:

"Compressor 4 User Manual." *Compressor 4 User Manual*. Apple Inc., 2012 Web. 5 Jan. 2014. <http://help.apple.com/compressor/mac/4.0/en/compressor/usermanual/>.

"HDIUTIL(1) BSD General Commands Manual." *OS X Man Pages*. Apple Inc., 16 Aug. 2013. Web. 24 Sept. 2013. <https://developer.apple.com/library/mac/documentation/Darwin/Reference/ManPages/man1/hdiutil.1.html>.

"Send Files to Compressor With A Watch Folder." *Back to the Edit*. Back To The Edit, 30 June 2011. Web. Summer 2013. <http://backtotheedit.com/2011/06/compressor-files-with-a-watch-folder/>.

Berkeley Software Distribution. "DD(1) BSD General Commands Manual." *OS X Man Pages*. Apple Inc., 13 Jan. 1994. Web. 26 Sept. 2013. <https://developer.apple.com/library/mac/documentation/Darwin/Reference/ManPages/man1/dd.1.html>.

"The Unofficial DVD Specifications Guide." *The Unofficial DVD Specifications Guide*. DVD-Replica Media, LLC., 2000. Web. 13 Feb. 2014. <http://www.dvd-replica.com/DVD/>.

"FFmpeg Bug Tracker and Wiki." *FFmpeg WIki*. N.p., n.d. Web. 05 Oct. 2013. <http://trac.ffmpeg.org/wiki>.

"GNU Ddrescue Manual." *GNU Ddrescue Manual*. GNU, n.d. Web. 26 Sept. 2013. <https://www.gnu.org/software/ddrescue/manual/ddrescue_manual.html>.

Jack, Keith. *Video Demystified: A Handbook for the Digital Engineer*. Amsterdam: Newnes, 2007. Print.

The Moving Picture Experts Group. "MPEG-2 Systems." *Systems | MPEG*. The Moving Picture Experts Group, 1996. Web. 10 Jan. 2014. <http://mpeg.chiariglione.org/standards/mpeg-2/systems>.

"Understanding CD-R & CD-RW - Disc Longevity." *Understanding CD-R & CD-RW - Disc Longevity*. Optical Storage Technology Association, 2001. Web. Fall 2013. <http://www.osta.org/technology/cdqa13.htm>.

Pohlmann, Ken C. *Principles of Digital Audio*. 4th ed. New York: McGraw-Hill, 2000. Print.

Taylor, Jim H. *DVD Demystified.* 2nd ed. New York: McGraw-Hill, 2001. Print.

Taylor, Jim H. *Everything You Ever Wanted to Know about DVD: The Official DVD FAQ.* New York: McGraw-Hill, 2003. Print.

UNESCO. *Risks Associated with the Use of Recordable CDs and DVDs as Reliable Storage Media in Archival Collections: Strategies and Alternatives*. By Kevin Bradley. UNESDOC, 2006. Web. <http://unesdoc.unesco.org/images/0014/001477/147782e.pdf>.

The following is a list of persons who contributed information, code, or development to this report:

Morgan Oscar Morel of George Blood Audio Video Film was the principal author and compiler of this report. Additional contributors include:

- George Blood, President at George Blood Audio Video Film
- Brandon Foltz, Automation Developer and Video Quality Assurance Technician at George Blood Audio Video Film
- Biz Maher Gallo, Manager of Audiovisual Preservation at George Blood Audio Video Film
- Lynda Schmitz Fuhrig, Electronic Records Archivist at Smithsonian Institution Archives
- Peter Finkel, Volunteer at Smithsonian Institution Archives
- John McGrath, Sales Manager at MF Digital
- Robert Warnock, Technical Support Manager at MF Digital
- Jason Priebe, Director of Technology at CBC New Media Group
- Brendan Coates, Audio Digitization Technician at UC Santa Barbara, Department of Special Collections
- Henry Boon Kelly, CTO at DCA Inc.