

YOUTUBE: RECOMPRESSION EFFECTS

by

COLE MICHAEL WHITECOTTON

B.S., University of Colorado at Denver, 2015

A thesis submitted to the  
Faculty of the Graduate School of the  
University of Colorado in partial fulfillment  
of the requirements for the degree of  
Master of Science  
Recording Arts Program

2017

© 2017

COLE MICHAEL WHITECOTTON

ALL RIGHTS RESERVED

This thesis for the Master of Science degree by

Cole Michael Whitecotton

has been approved for the

Recording Arts Program

by

Catalin Grigoras, Chair

Jeff M. Smith

Jeff Merkel

Date: December 16, 2017

Whitecotton, Cole Michael (M.S., Recording Arts Program)

YouTube: Recompression Effects

Thesis directed by Assistant Professor Catalin Grigoras

## **ABSTRACT**

YouTube, founded in 2005, is the leading digital video streaming site on today's internet. There are many streaming sites today that allow users to upload and share multimedia content, but YouTube is the only one with some estimated one-billion unique monthly users. Many of these user-created videos can contain important information that may pertain to a crime or other event of interest. Prior to uploading, digital video files can be encoded with embedded data that can be important to forensic analysts. This data is unfortunately stripped by YouTube's encoding algorithms for every video that is uploaded, which is concordant with YouTube's neither being a file-sharing website nor allowing for the easy download of video content from its site. This, along with lossy compression, makes analysis of YouTube videos difficult.

This research aims to take a deeper look at what information, if any, can be gleaned from video files that are downloaded from YouTube using third-party options. Across five different tests, videos were created and uploaded to YouTube and the videos were downloaded using downloader programs. These files were then altered, compared, and analyzed in order to better understand the behavior of YouTube and the downloader tools.

Chapter 1 provides a quick primer to digital files and information that can be contained therein. Chapter 2 describes some of the underlying frameworks and background information about YouTube. Chapter 3 details each of the five tests, including information,

methods and materials, and results. Finally, Chapter 4 recaps each of the tests' results and some more general findings along with providing some points for potential future research.

The form and content of this abstract are approved. I recommend its publication.

Approved: Catalin Grigoras

This is dedicated to both of my parents, without whom I would not have developed my desire to learn and figure out how the world around me works.

## **ACKNOWLEDGEMENTS**

Thank you to Catalin and Jeff for helping me find my way into this crazy and evolving world of digital forensics. I have continued to find that the more I learn from you guys, the more I realize how little I actually know. I am thankful for the opportunities the National Center for Media Forensics has afforded us and the incredible learning environment that was created by all of you.

Leah and Emma, you have both been incredibly helpful throughout the whole program and especially in making sure I was on top of all of these deadlines!

Thank you, Jeff Merkel, for all of the amazing and crazy in-depth explanations on the physics of audio and your genius with Max and Live. I love seeing your crazy projects and hope to see many more.

Finally, thank you, James Ross, for your invaluable help with everything over the past year. This research would never have been completed without your help.

# TABLE OF CONTENTS

## CHAPTER

I.	INTRODUCTION .....	1
	Metadata/File Structure .....	1
	FFmpeg .....	3
	Downloaders .....	4
	Previous Research .....	5
II.	YOUTUBE FRAMEWORK .....	7
	Encoding .....	7
	DASH .....	7
	API .....	8
III.	DATA OVERVIEW .....	9
	Test 1 – ClipGrab v youtube-dl .....	10
	Test 2 – How does YouTube handle different codecs and containers? ... ..	15
	Test 3 – What additional information appears over time? .....	17
	Test 4 – What differences appear when comparing videos of varying length? .....	19
	Test 5 – What do the owners of YouTube accounts have access to download vs the public? .....	20
IV.	DISCUSSION .....	24
	Future Research.....	25
	REFERENCES.....	27

## LIST OF FIGURES

### FIGURE

1. Example of data contained in a digital video file .....	2
2. Example MediaInfo v17.10 display .....	3
3. FFmpeg video-stream hash command .....	4
4. ClipGrab UI .....	10
5. ExactFile report, Test 1 .....	12
6. FFmpeg video-stream hash values, Test 1 .....	12
7. MediaInfo v17.10 display, Test 1 .....	13
8. 010 Editor comparison display, Test 1 .....	13
9. Youtube-dl query output .....	18
10. Youtube-dl fragment query .....	19
11. YouTube download UI .....	21
12. Youtube-dl “best” format display .....	21
13. Google archive UI .....	22
14. ExactFile report, Test 5 .....	23

## **LIST OF ABBREVIATIONS**

API – Application Programming Interface

CCN – Content-Centric Networking

CMSS – Collaborative Multi-Scale Scheduling

DASH – Dynamic Adaptive Streaming over HTTP

EXIF – Exchangeable Image File Format

HTML5 – HyperText Markup Language 5

HTTP – Hyper Text Transfer Protocol

ISO/IEC – International Organization for Standardization/International Electrotechnical  
Commission

MD5 – Merkle-Damgard 5, hash value algorithm

MPD – Media Presentation Description

PRNU – Photo Response Non-Uniformity

SWGDE – Scientific Working Group on Digital Evidence

UI – User Interface

USB – Universal Serial Bus

XML – Extensible Markup Language

XMP – Extensible Metadata Platform

# CHAPTER I

## INTRODUCTION

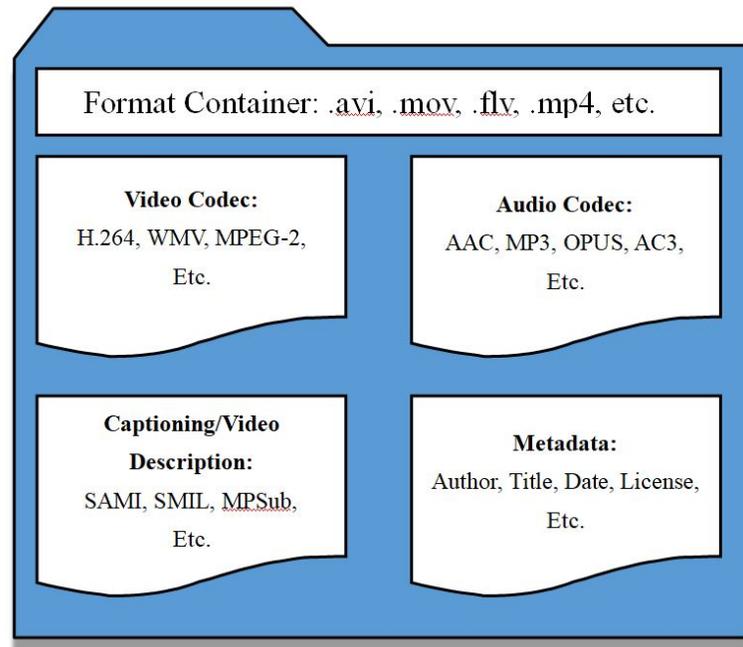
One research aspect of this thesis is the analysis of a few tools available for downloading streaming videos from YouTube. There are many websites available that allow for the downloading of videos and many of them use youtube-dl (which will be discussed in more detail later) in their core scripts. Because this research relies on these tools, an understanding of the underlying technologies is needed. This introduction will explain the basics of digital file structure and some of the tools used for analysis.

### **Metadata/File Structure**

All files created by digital cameras contain information about the files created. This information is important for any program that needs to open and read the file, but it can also contain information that is unique in some way to that camera—sometimes even specific to an individual camera. In the case of digital video files, there are two levels of information.

The top level is the format container (e.g., .avi, .mp4, etc.). The information in this level is vital for playback of the video as it contains information on what codec is needed to play the file and informs the software being used in order to properly display the video. It may also contain non-essential but useful information like metadata for the file, time-related information, etc. The next level of information is the video-stream itself. This is the actual bit-by-bit encoded video that the information in the first layer instructs the software to read.

Figure 1 is an example of a typical video file.



*Figure 1. Example of data contained in a digital video file*

A major part of the information contained in a digital file is the metadata. Metadata contains information that pertains to the digital file. This can be any information that the manufacturer of the hardware and software used to create the files would like to include (e.g. make and model name/numbers, serial numbers, version numbers, dates and times, etc.), as well as the information that is vital for playback of the file. Many modern cameras use EXIF (Exchangeable Image Format) or XMP (Extensible Metadata Platform) formatting for this metadata, which makes it easy for tools to read and organize the data for specific types of analysis. Figure 2 is an example of this data organized and displayed by a program (MediaInfo v17.10).

```

Codec ID           : dash (iso6/avc1/mp41)
File size          : 1.17 GiB
Duration           : 33 min 10 s
Overall bit rate   : 5 039 kb/s
Encoded date       : UTC 2017-10-02 21:22:07
Tagged date        : UTC 2017-10-02 21:22:07

Video
ID                 : 1
Format             : AVC
Format/Info        : Advanced Video Codec
Format profile     : High@L4
Format settings    : CABAC / 2 Ref Frames
Format settings, CABAC : Yes
Format settings, RefFrames : 2 frames
Codec ID           : avc1

```

*Figure 2. Example MediaInfo v17.10 display*

Metadata is invaluable information for analysis of files with known origins, but because this data can be altered quite easily, it cannot always be relied upon for accurate forensic analysis. Most social media services strip all of this data away from the original files in an effort to save space by minimizing files sizes, maintaining their users' privacy, and oftentimes, to add their own metadata tags for use by their proprietary software platforms [1]; YouTube is no exception.

## **FFmpeg**

FFmpeg is incredibly powerful software that works cross-platform and allows for the creation, conversion, and streaming of digital video and audio files. The program is open-source and provides its own libraries whenever technically possible; it is also provided free of charge and can be downloaded directly from the developer's website.

SWGDE (Scientific Working Group on Digital Evidence) document "Technical Notes on FFmpeg" [2] describes the use of FFmpeg's frame-level MD5 function where steps are provided to produce an MD5 hash of each individual frame of a video file, compile these into text files, and then compare them in order to verify that the videos are identical.

Another, more efficient way of determining the precise duplication of encoded video or audio is using FFmpeg to calculate the MD5 of the entire decoded stream; the stream-hash calculation. Figure 3 below shows the command which will return the MD5 hash of the video-stream contained in a video file.

```
C:\Users\Cole\Videos\New folder>ffmpeg -loglevel error -i "06 ClGr_Hyperlapse.mp4.mp4" -map 0:v -f md5 -  
MD5=11390f29a876aa0f6fe61d1e27f3f2ff
```

*Figure 3. FFmpeg video-stream hash command*

If two different video files with two different containers, e.g. .avi and .mov, contain the same video-stream, then the video-stream hash calculations of each will be the same; this same analysis can be used on audio as well. Calculating and comparing the stream hash values is very useful in verifying the integrity of multimedia content and whether it has changed regardless of any changes to the file metadata or structure. Furthermore, it has been rigorously tested and validated for forensic purposes at the National Center for Media Forensics.

## **Downloaders**

There are countless applications and websites that allow users to download videos from streaming sites such as YouTube, Vimeo, etc. Two common programs are deserving of a closer look due to their popularity and feature-set: youtube-dl and ClipGrab.

Youtube-dl is a free, open-source program. Youtube-dl does not offer a UI and is instead a command line only program. Thanks to the openness of the program and the use of open-source python libraries, youtube-dl has become a project that has been worked on and added to by multiple programmers. This has led to youtube-dl being a much more robust

program compared to the other program considered for this research: ClipGrab. While ClipGrab offers an easy-to-use UI and can easily download a few specific qualities of videos, youtube-dl allows access to all formats reported as available from YouTube's servers. The ease of setting up configurations and integrating youtube-dl into custom scripts has led this research to focus primarily on youtube-dl and its expected versus observed behaviors.

### **Previous Research**

The focus of previous research on the subject of YouTube videos has been primarily on the DASH video format itself. There has not been a focus on the collection of streaming video from YouTube intended for forensic analysis aside from the 2014 paper titled, "Source identification of high definition videos: A forensic analysis of downloaders and YouTube video compression using a group of action cameras," by Giammarrusco. In it, he details the effects of downloader tools and their compression algorithms on videos and it was discovered that some of the tools compress the videos themselves—many did this in differing ways. He explains that "one should test their downloading tools and select the method which will incorporate the least amount of compression." [3] This paper focuses on PRNU analysis and provided a lot of research on various downloader tools.

In their paper titled, "Forensic Analysis of Video File Formats," Gloe *et al.* [4] provide a systematic exploration of some popular container formats and the data contained within. They set up a number of test files and analyzed the metadata that was inherited directly from the cameras and then again after the videos had been edited. Their goal was to build a foundation to identify specific characteristics of this metadata in order to verify the authenticity of questioned video files.

Similarly, Hall's paper, "MPEG-4 Video Authentication Using File Structure and Metadata" [5], furthers this type of research and puts forth a guideline for how this analysis should be done. He provides detailed information on a number of analysis tools and compares their output of information in order to find which ones have the strongest forensic applications.

At the 2011 Global Telecommunications Conference in Houston, TX, Pu *et al.* presented an early look at the benefits and features of the upcoming MPEG-DASH format and how other companies were already leveraging DASH-related technologies to provide their streaming content; this includes Microsoft and Apple programs. In their presentation, they proposed their idea for a collaborative multi-scale scheduling algorithm (CMSS) in order to mitigate web server loads—among other benefits [6].

At the 2014 International Conference on Advanced Information Networking and Applications Workshops in Victoria, BC, Pereira *et al.* explained that video data accounts for over fifty percent of all internet traffic. This number will only rise as the proliferation of smart devices continues to grow. Because of the variability of network quality and type for these devices, the video content served on them has had to evolve to combat the issues of latency and varying speeds. The presentation focuses on the MPEG-DASH standard and explores some of the limitations as well as many of the benefits associated with it [7].

## **CHAPTER II**

### **YOUTUBE FRAMEWORK**

In his paper, Giammarrusco explains that it makes sense to think of the original video file uploaded to YouTube as the “master” video file that is used as the source for the various qualities and formats encoded by their algorithms. This means that higher-quality video files uploaded will allow for higher quality video files downloaded [3].

#### **Encoding**

All videos that are uploaded to YouTube will be re-encoded regardless of quality or format. This means that all metadata that may be contained within the original file will be lost, as it is not copied over to the newly-encoded versions. This is a significant fact that makes it impossible to find any specific information about the original uploaded file.

#### **DASH**

YouTube engineers began experimenting with HTML5 around 2010, but only made it the default for video delivery at the very beginning of 2015 [8]. This development was accompanied by the abandonment of the Flash Video format and a move to the DASH (Dynamic Adaptive Streaming over HTTP) format. Published as ISO/IEC 23009-1:2014 (current update) [9, p. 06], DASH was developed under MPEG as an adaptive bitrate streaming format in which a multimedia file is split up into segments and streamed to clients via HTTP. These segments can contain any media data, but the spec provides information for use with two types of containers: MPEG-2 Transport Stream and ISO based media such as MP4, etc. DASH enhances the user experience by allowing for fast switching between resolutions based on internet connectivity, which can be variable.

DASH is codec-agnostic, allowing for different versions of each multimedia file to exist and be tied to and played through a single stream. This is accomplished with an XML file known as a media presentation description (MPD). The MPD contains segment information that directs the stream to the correct media file and allows for the stream (audio and video are served in two separate streams) to switch to variable resolutions at different times based on the network conditions, device capabilities, user preferences, and file availability; this is the crux of the adaptive bitrate streaming capabilities. DASH is also able to be used with any underlying application protocol such as HTTP, CCN, etc.

### **API**

Google, who purchased YouTube in 2006, provides a number of APIs for their services and the one provided for YouTube allows for easy uploading of videos. Settings for the videos, along with descriptions, can be set via the API and uploaded to an account as long as the API is activated and account information is provided [10]. Because YouTube is not a file sharing service, the API is one-way only: allowing for the upload of videos, but not the download—hence the need for third-party tools.

## **CHAPTER III**

### **DATA OVERVIEW**

The goal of this research is to help us understand the behavior of YouTube and provide a possible foundation for the acquisition of streaming videos. This foundation will help shape our best practices around the observed predictable nature of YouTube. Five areas of interest were identified as possible points of forensic interest and multiple videos were produced as test videos in order to control for and change specific variables (example information for these videos is provided in more detail under each test description). These variables were then tweaked, the videos uploaded and downloaded from YouTube, and then these new video files were compared to the original base files. The differences identified are indicative of the processes the tools use and can also elucidate what YouTube is doing with the encoding and storage of original video files.

The tests conducted include: (1) ClipGrab v youtube-dl; (2) observing how YouTube handles various codecs and container types and observing possible differences in the downloaded files; (3) observing what additional information appears when videos are up for extended time periods; (4) determining whether differences appear when comparing videos of varying length; and (5) determining what the owner of a YouTube account has access to download vs the public.

## Test 1 – ClipGrab v youtube-dl

ClipGrab and youtube-dl aim to provide similar services for downloading streaming video/audio. Both have the ability to download different formats of both audio and video streams, however, ClipGrab does not download more than one format at a time; it also does not provide much detail about each available format. ClipGrab does, however, offer a UI (Figure 4) that makes it easy to download the highest quality available for a given video—which may be the quickest method for acquisition of video files in cases where speed is more important than thoroughness.

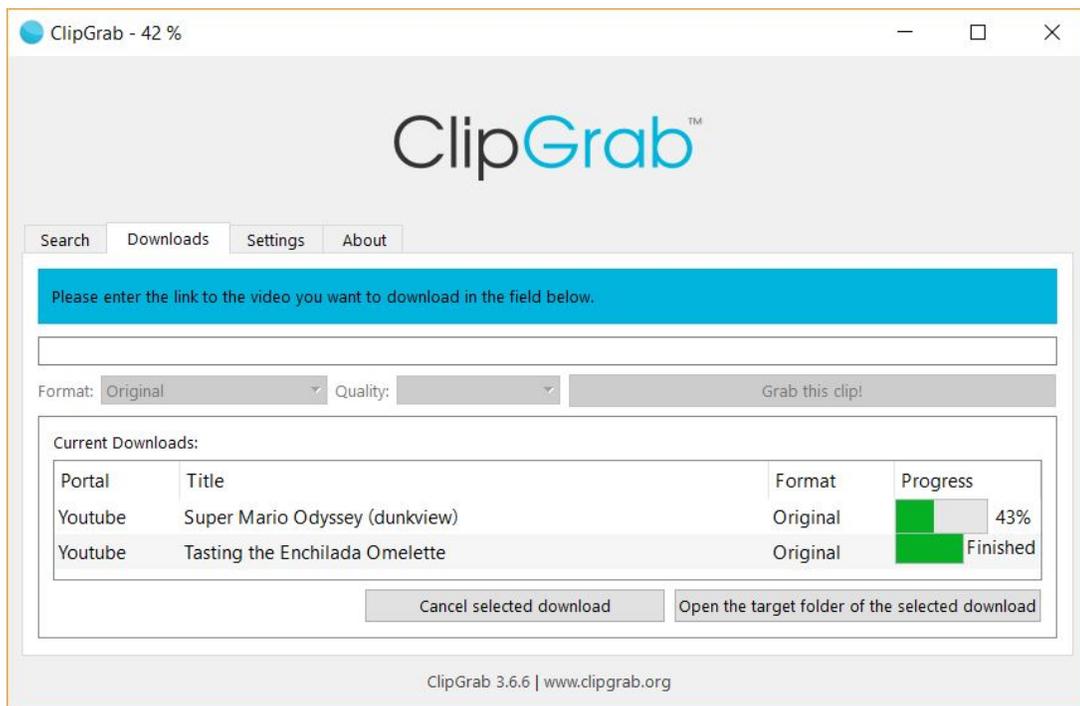


Figure 4. ClipGrab UI

Youtube-dl does not have a built-in UI and works only via command-line; while making it a bit less user-friendly, this does make it easy to integrate into scripts and run with other programs. It is also possible to see all formats that are available along with details (bitrate, file format, etc.) for each file. This allows for targeted downloads of specific formats

and it is also possible to download and organize all available formats with a single command in youtube-dl.

The two main objectives of this test were to compare and contrast the two tools and their interfaces/options and to compare possible differences between downloaded videos from each tool.

### **Methods and Materials**

Three videos with three different resolutions (4K, 1080p, 720p) were uploaded to YouTube via YouTube's web dashboard. All three videos were shot with a Samsung cell phone and transferred to a PC via USB. Each tool was then used to perform various functions including querying, downloading, and organizing each of the three videos.

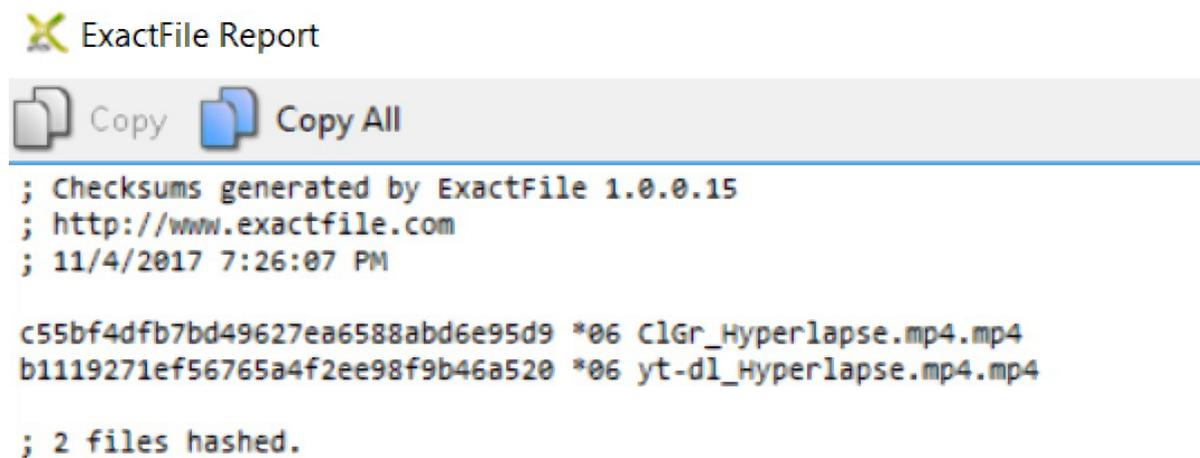
With ClipGrab, the main software settings were set to default. When selecting which video to download, the "Format" box was set to original. This ensured that the tool was not re-encoding the video files with its own compression. The "Quality" box was then set to the highest available quality for each video.

When downloading with youtube-dl, a query was made for each video in order to view which formats were available. The highest quality audio and video streams were then chosen for download and the tool downloaded and combined both streams with just the one command.

Each of these downloaded files was then analyzed with MediaInfo, 010 Editor, FFmpeg, and ExactFile.

## Results

When looking at the same quality video downloaded from both tools, the files themselves are not the same, but that does not mean the encoded video data is not the same. The files are different sizes on-disk and using ExactFile (Figure 5) to compare the MD5 hash values of each matching pair of videos shows that the files contain different information (i.e. the hashes don't match).



The screenshot shows a window titled "ExactFile Report" with a toolbar containing "Copy" and "Copy All" buttons. The report text is as follows:

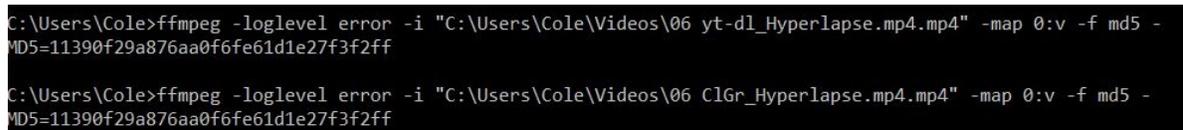
```
; Checksums generated by ExactFile 1.0.0.15
; http://www.exactfile.com
; 11/4/2017 7:26:07 PM

c55bf4dfb7bd49627ea6588abd6e95d9 *06 ClGr_Hyperlapse.mp4.mp4
b1119271ef56765a4f2ee98f9b46a520 *06 yt-dl_Hyperlapse.mp4.mp4

; 2 files hashed.
```

Figure 5. ExactFile Report, Test 1

However, using FFmpeg to calculate the MD5 hash value of the video-stream shows that the video content is identical (Figure 6). This means that the mismatch of information comes from data other than the video-stream itself.



The screenshot shows a terminal window with the following commands and output:

```
C:\Users\Cole>ffmpeg -loglevel error -i "C:\Users\Cole\Videos\06 yt-dl_Hyperlapse.mp4.mp4" -map 0:v -f md5 -
MD5=11390f29a876aa0f6fe61d1e27f3f2ff

C:\Users\Cole>ffmpeg -loglevel error -i "C:\Users\Cole\Videos\06 ClGr_Hyperlapse.mp4.mp4" -map 0:v -f md5 -
MD5=11390f29a876aa0f6fe61d1e27f3f2ff
```

Figure 6. FFmpeg video-stream hash values

MediaInfo reports the same core information for both videos (Figure 7), with the only differences being the date and times listed. Youtube-dl retains the encoding date and times, while ClipGrab removes this data.

```

Video
ID : 1 : 1
Format : AVC : AVC
Format/Info : Advanced Video Codec : Advanced Video Codec
Format profile : High@L4 : High@L4
Format settings, CABAC : Yes : Yes
Format settings, RefFrames : 2 frames : 2 frames
Codec ID : avc1 : avc1
Codec ID/Info : Advanced Video Coding : Advanced Video Coding
Duration : 1 min 12 s : 1 min 12 s
Bit rate : 4 330 kb/s : 4 330 kb/s
Width : 1 920 pixels : 1 920 pixels
Height : 1 080 pixels : 1 080 pixels
Display aspect ratio : 16:9 : 16:9
Frame rate mode : Constant : Constant
Frame rate : 30.000 FPS : 30.000 FPS
Color space : YUV : YUV
Chroma subsampling : 4:2:0 : 4:2:0
Bit depth : 8 bits : 8 bits
Scan type : Progressive : Progressive

```

Figure 7. MediaInfo v17.10 display, Test 1

Going deeper into the files with 010 Editor (Figure 8) shows where the real differences in the two files are. The metadata contained in each file differs dramatically—this is where the extra file size and differences in file hash values come from. Since the stream hashes match, though, the video content is not altered independently (not being re-encoded) by either tool and is a reliable forensic representation of what is on the YouTube server.

Result	Address A	Size A	Address B	Size B
Difference	0h	2C18DEh	0h	2C18DEh
Match	2C18DEh	8h	2C18DEh	8h
Difference	2C18E6h	10h	2C18E6h	10h
Match	2C18F6h	8h	2C18F6h	8h
Difference	2C18FEh	10h	2C18FEh	10h
Match	2C190Eh	8h	2C190Eh	8h
Difference	2C1916h	10h	2C1916h	10h
Match	2C1926h	8h	2C1926h	8h
Difference	2C192Eh	28h	2C192Eh	28h
Match	2C1956h	8h	2C1956h	8h
Difference	2C195Eh	10h	2C195Eh	10h
Match	2C196Eh	8h	2C196Eh	8h
Difference	2C1976h	10h	2C1976h	10h
Match	2C1986h	8h	2C1986h	8h
Difference	2C198Eh	10h	2C198Eh	10h
Match	2C199Eh	8h	2C199Eh	8h

Figure 8. 010 Editor comparison display, Test 1

It should also be noted that none of the 4K videos tested were found to be downloadable via ClipGrab. When downloading 4K videos, ClipGrab reported 1440p as the highest resolution available. Querying these same videos with youtube-dl shows that 4K video is indeed available. It was later found that ClipGrab can download 4K webm formatted videos with some modifications of the settings, though 4K and higher resolution DASH videos are not available. It is best practice to primarily use youtube-dl unless technical capability is a limitation.

The differences between these tools can be distilled as ClipGrab being able to offer an easy-to-use UI at the cost of extensive options, while youtube-dl offers extensive options but is only available as a command-line program and can also allow for higher resolution downloads (when available). These differences can determine which may be best to use in certain circumstances.

## **Test 2 – How does YouTube handle different codecs and containers?**

There are a wide variety of different video codecs and containers that various devices can use to create digital videos. Most devices can change their quality and size settings to a dozen or more different options. YouTube provides a guideline for videos [11] that are uploaded to the site and these guidelines include preferred resolutions and containers. This test was to find out how YouTube handles the most common codecs and containers—not just those on the recommended list.

### **Methods and Materials**

One video was made with a laptop webcam set to the highest resolution and quality available. FFmpeg was then used to produce different versions of this video, some re-wrapped in different containers and some encoded with different codecs. The re-wrapped files were produced as .avi, .m4v, .mkv, .mp4, .wmv, .flv, and .3gp, all containing the original compressed video-stream. The other videos were encoded with XVID, MJPEG, AMV, h.264, h.263, h.265, JPG2000, and MPEG-2 codecs.

Next, each video was uploaded to YouTube and downloaded using youtube-dl. 010 Editor and MediaInfo were then used to analyze the files to determine what differences existed between the downloaded videos. And finally, FFmpeg was used to calculate and compare the MD5 hash values of the video-streams from each pair of videos.

### **Results**

YouTube did not reject any of the formats or codecs uploaded and, in all circumstances, it re-encoded each of the videos into YouTube's standard encoded formats.

Since each of the re-wrapped videos uploaded to YouTube contained the same stream of the original video, the downloaded video files shared matching video data-streams, i.e. the videos that were uploaded with different containers but had the same exact video-stream information shared this same property when downloaded from YouTube's servers. The encoding algorithm YouTube used to re-encode the videos is consistent across multiple containers.

The metadata of all video files (including DASH, mp4, etc.) downloaded with youtube-dl also contain date and time information that is related to the date and time each file was originally uploaded and encoded. The new formats that appear after the initial upload have a different time associated with them as they are encoded at a later time.

### **Test 3 – What additional information appears over time?**

During the course of other tests, it was observed that more formats were available for videos uploaded sometime in the past. These formats were not available when first uploaded and it was unclear when these formats became available. In order to find what may trigger these extra encodings, a script was written to monitor when these new formats became available. Other variables were also controlled-for as the observations were made—including how many views, which browser was used, and which operating system was used.

#### **Methods and Materials**

A script was written in python using Google’s “YouTube Data API” [10] and youtube-dl libraries that allowed for a video to be uploaded and then queried with youtube-dl at hour intervals. The sample video was made and uploaded to YouTube and a query was made as soon as the video was available to view. Youtube-dl returned a total of fourteen available audio/video formats. The script then ran a query every hour for three days and the results were compared to each other. The video was then viewed multiple times from different browsers, different systems, and at different qualities. Between each new viewing, the script was run for a minimum of two days.

#### **Results**

Comparing these queries revealed that only one new format appeared and it was within the first hour. Figure 9 shows this new format (WEBM, 640x360).

[INFO] AVAILABLE FORMATS FOR NIOIZG1FT2Q:				[INFO] AVAILABLE FORMATS FOR NIOIZG1FT2Q:			
FORMAT CODE	EXTENSION	RESOLUTION	NOTE	FORMAT CODE	EXTENSION	RESOLUTION	NOTE
139	M4A	AUDIO ONLY		139	M4A	AUDIO ONLY	
140	M4A	AUDIO ONLY		140	M4A	AUDIO ONLY	
134	MP4	202X360		134	MP4	202X360	
160	MP4	82X144		160	MP4	82X144	
137	MP4	608X1080		137	MP4	608X1080	
133	MP4	136X240		133	MP4	136X240	
135	MP4	270X480		135	MP4	270X480	
264	MP4	810X1440		264	MP4	810X1440	
136	MP4	406X720		136	MP4	406X720	
266	MP4	1080X1920		266	MP4	1080X1920	
17	3GP	176X144		17	3GP	176X144	
36	3GP	136X240		36	3GP	136X240	
18	MP4	202X360		18	MP4	202X360	
22	MP4	406X720		43	WEBM	640X360	
				22	MP4	406X720	

Figure 9. Youtube-dl query output

This same analysis was conducted on a video that had been available for over nine months and the formats available were the same.

Due to the constraints of this research, the longest period of time that any of the videos created for this test has been available on YouTube is just under ten months. At the time of this data collection, no new formats have appeared on any of the videos that have been up for this amount of time. Comparing the video files downloaded shortly after the initial upload and the files downloaded at the 10-month mark shows that they still contain the same video-stream and are identical. An interesting note here is that the youtube-dl queries and downloads did not cause the YouTube view-counter that is displayed on the video's page to change for any of the videos tested.

## TEST 4 – What differences appear when comparing videos of varying length?

The DASH video format allows for some variance in the video-stream (e.g. how the video and audio streams get split into fragments and how many different format options are available). Youtube-dl can possibly shed light on how YouTube’s encoding algorithms decide on these options.

### Methods and Materials

Three different lengths of video were chosen to represent long (33 minute), short (one minute), and medium (eighteen minute) lengths. The highest quality DASH format videos were then downloaded with youtube-dl and analyzed with 010 Editor. Youtube-dl reports how many fragments each video is split into as it is downloading and 010 Editor confirms this.

### Results

Other than file size, the only real difference between the videos was the number of fragments in the DASH videos. The containers were identical across all three videos. The number of fragments was different—as expected. Figure 10 shows the 33-minute video has 375 fragments, the eighteen-minute video 211 fragments, and the one-minute video fifteen fragments.

```
[youtube] hxd1T1T8fvY: Downloading MPD manifest
[dashsegments] Total fragments: 375
[download] Destination: C:\Users\Cole\Videos\04 1080_Landscape.mp4\00001-04 1080_Landscape.mp4.mp4
[download] 100% of 1.17GiB in 05:19
[dashsegments] Total fragments: 211
[download] Destination: C:\Users\Cole\Videos\Work 720p\00001-Work 720p.mp4
[download] 100% of 7.58MiB in 00:23
[youtube] fmRKYQF_xMs: Downloading MPD manifest
[dashsegments] Total fragments: 15
[download] Destination: C:\Users\Cole\Videos\06 Hyperlapse.mp4\00001-06 Hyperlapse.mp4.mp4
[download] 100% of 37.60MiB in 00:13
```

Figure 10. Youtube-dl fragment query

## **TEST 5 – What do the owners of YouTube accounts have access to download vs the public?**

Google offers two ways to download videos that you upload to your own YouTube account: from the YouTube account interface and (since 2012) through a Google service that allows you to download all account data as an archive. If either of these methods allows the user to download original material, it could be immensely helpful for forensic analysis and possibly other areas of investigation.

### **Methods and Materials**

Using the same account that has been used for the rest of this research, videos were downloaded through the two methods available. 010 Editor and FFmpeg were then used to analyze these videos in order to find possible differences.

### **Results**

Even though a dozen or more different encodings of any one video are being stored on YouTube's servers and are available to stream, downloading a video from the YouTube interface only allows the owner to download the video as an MP4 at 720p or lower resolution—regardless of what the original uploaded video resolution may have been. Figure 11 shows an example of the YouTube download interface found in video manager.

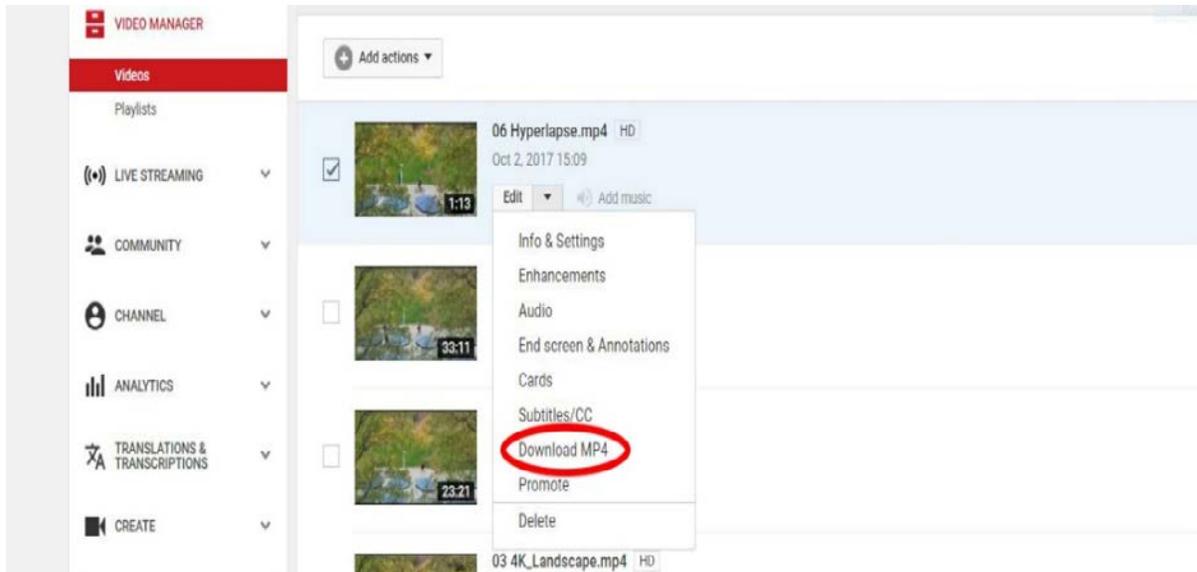


Figure 11. YouTube download UI

If the original video was already below 720p, then whatever the original resolution was is what gets downloaded. If the video was higher than 720p, then 720p is downloaded. This video matches what youtube-dl labels as “best” when querying (Figure 12).

36	3gp	320x180	small , mp4v.20.3, mp4a.40.2
43	webm	640x360	medium , vp8.0, vorbis@128k
18	mp4	640x360	medium , avc1.42001E, mp4a.40.2@ 96k
22	mp4	1280x720	hd720 , avc1.64001F, mp4a.40.2@192k (best)

Figure 12. Youtube-dl “best” format display

The finding of note here is that the video downloaded directly through the owner’s YouTube account is the exact same video downloaded when downloading the “best” video with youtube-dl; the video-streams have matching hash values.

When downloading video files through the archive option provided by Google, the video files downloaded are the same files originally uploaded. This archive option is meant as a way for users to download all of their files associated with their Google account. This includes any files that are stored in their Google Drive, Calendars, Google+, YouTube, etc. The files are downloaded as either .zip or .tgz files at a maximum of 50GB each. Each of the

videos downloaded also includes an HTML file that has metrics for each of the videos (e.g. views, likes, etc.). The archive downloaded for this research did not include any of the videos that had been deleted from this account, so it seems that only videos that are currently available in the account holder's video manager are available to download. Figure 13 is an example of the archive UI.

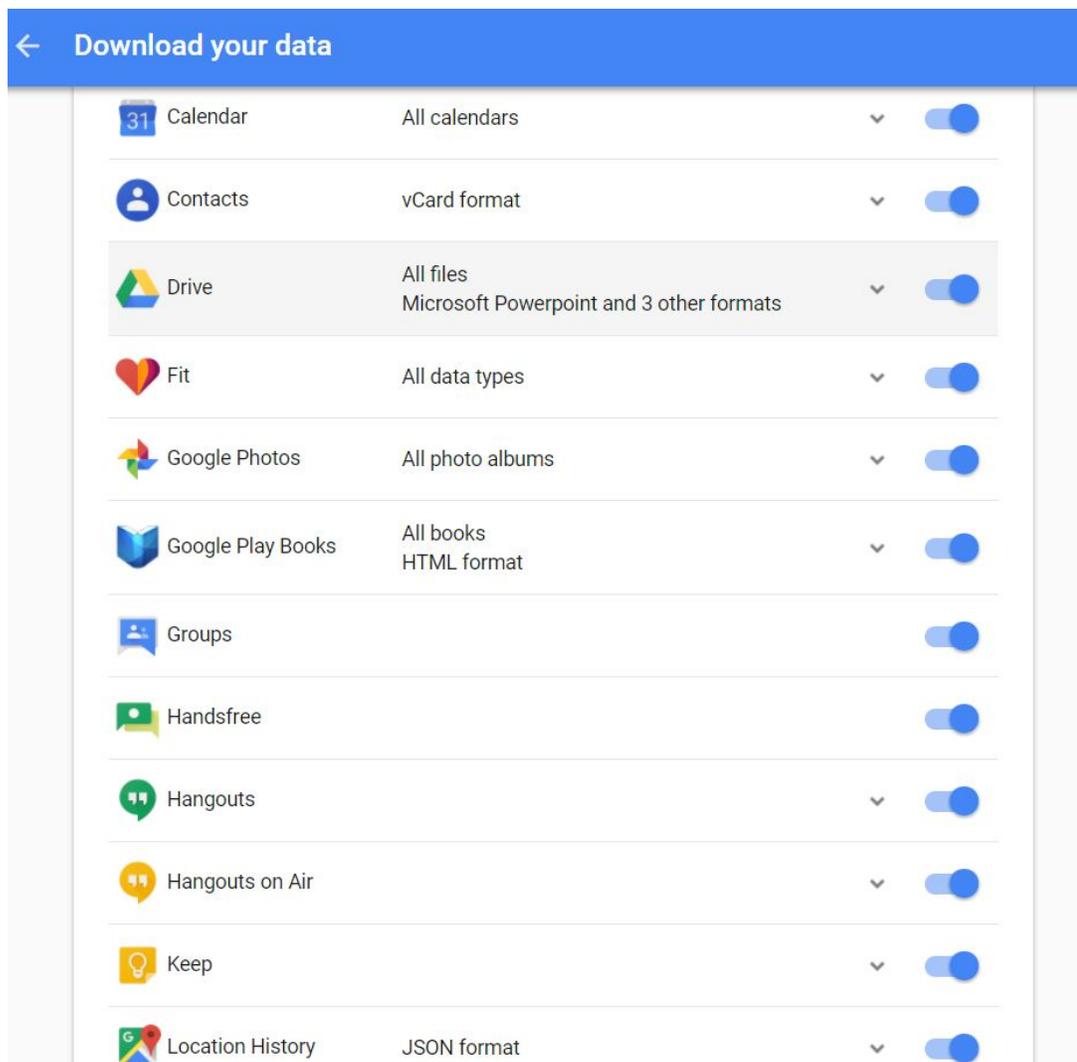


Figure 13. Google Archive UI

A comparison of an original file's hash values and the archive-downloaded hash values reveals that the hash values match (i.e. the files are identical.) This means that the account holders have access to retrieve their original video files. Figure 14 shows the matching hash values for an original video file (ORIG) and archive-downloaded file (ARCHIVE), as well as a file downloaded using the YouTube browser interface (Direct) and the "best" video (yt-dl) listed by youtube-dl.

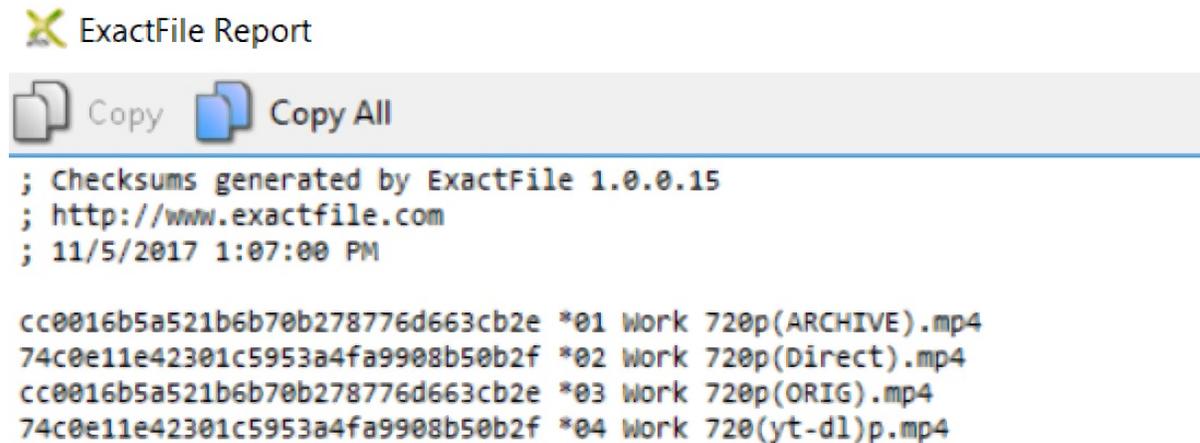


Figure 14. ExactFile Report, Test 5

## CHAPTER IV

### DISCUSSION

During Test 1's comparison of ClipGrab with youtube-dl, it was found that some 4K videos were not seen as available to download with ClipGrab without changes to settings (and even then, no DASH formats above 1440p were available). These 4K and higher resolutions videos were still available with youtube-dl, though. This leads to the best practice of 1) using youtube-dl to query all available formats for the video and 2) downloading the highest quality dash video and dash audio available and using youtube-dl to mux with the following command:

```
% youtube-dl -f '136+140' https://youtu.be/vid-ID-string
```

where the example formats 136 and 140 are the video and audio format codes, respectively; this will ensure that the highest-quality video and audio available will be downloaded. Either of the audio or video format codes may be omitted if only one of them is required, (e.g., the video contains no audio of interest or vice versa.)

The codec and container analysis in Test 2 found that YouTube re-encodes all videos uploaded with the same set of encoding algorithms, regardless of original codec or container—even those that match their recommended specs. This produces the same video-stream even when different containers are used. Also, metadata contained in the video files downloaded from YouTube's servers contains date and time information that correlates to the original encoding time of the videos.

In Test 3, where videos were observed over a period of time to see if new formats became available, it was discovered that only one new format was found to be available

beyond those that were available immediately upon uploading. This new format was made available within an hour of uploading and no new encodings were observed thereafter.

Test 4, a look at videos of varying lengths, found that there were no differences in the containers or other stored metadata in longer videos vs shorter videos. However, the longer a video is, the greater the number of fragments there will be in the DASH-formatted videos.

Finally, the findings in Test 5 (what do owners have access to download vs the public) can be very important for forensic analysis and investigation. Understanding that the original files can be obtained via the owner or owner's credentials lends more possibilities for important evidence acquisition. The results of these earlier tests highlight the lack of information that is retained when videos are encoded by the YouTube algorithms. In all cases, any original embedded data is lost—including evidence of editing programs. The only way to recover this information is with access to the archive by the owner of the account. If this is possible, the original video uploads can be obtained along with any data that might have been contained within.

More generally, this research has found that youtube-dl can be a very powerful tool for obtaining digital downloads of streaming videos. Understanding the limitations of the tool and knowing its base behavior are important for understanding its forensic uses.

### **Future Research**

Some possible areas of interest for future research could be: a deeper look into what formats and options might be available after a video is up for extended periods of time (and as it gains popularity/views); a deeper look into the specific characteristics of the DASH

video fragments that can be downloaded with youtube-dl; and a look into what possible traces the editing features in YouTube's own UI leave in the video files.

During this research, contact was attempted with one of the developers of youtube-dl in order to discuss possible points of interest; specifically, how youtube-dl obtains the information it does (e.g., format codes for each available format, etc.). A deeper look into how youtube-dl obtains information from YouTube's servers would be helpful for future research.

## REFERENCES

- [1] N. K.-S. Ng, "Cell phone images in social media: An analysis of cellphone image structure before and after social media compression," M.S., University of Colorado at Denver, United States -- Colorado, 2013.
- [2] "SWGDE." [Online]. Available: <https://www.swgde.org/documents/Current%20Documents/SWGDE%20Technical%20Notes%20on%20FFmpeg>. [Accessed: 13-Nov-2017].
- [3] Z. P. Giammarusco, "Source identification of high definition videos: A forensic analysis of downloaders and YouTube video compression using a group of action cameras," M.S., University of Colorado at Denver, United States -- Colorado, 2014.
- [4] T. Gloe, A. Fischer, and M. Kirchner, "Forensic analysis of video file formats," *Digit. Investig.*, vol. 11, no. Supplement 1, pp. S68–S76, May 2014.
- [5] J. R. Hall, "MPEG-4 video authentication using file structure and metadata," M.S., University of Colorado at Denver, United States -- Colorado, 2015.
- [6] W. Pu, Z. Zou, and C. W. Chen, "Dynamic Adaptive Streaming over HTTP from Multiple Content Distribution Servers," in *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*, 2011, pp. 1–5.
- [7] R. Pereira and E. G. Pereira, "Dynamic Adaptive Streaming over HTTP and Progressive Download: Comparative Considerations," in *2014 28th International Conference on Advanced Information Networking and Applications Workshops*, 2014, pp. 905–909.
- [8] R. McCormick, "YouTube drops Flash for HTML5 video as default," *The Verge*, 27-Jan-2015. [Online]. Available: <https://www.theverge.com/2015/1/27/7926001/youtube-drops-flash-for-html5-video-default>. [Accessed: 17-Oct-2017].
- [9] "ISO/IEC 23009-1:2014(en), Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats." [Online]. Available: <https://www.iso.org/obp/ui/#iso:std:iso-iec:23009:-1:ed-2:v1:en>. [Accessed: 17-Oct-2017].
- [10] "YouTube Data API Overview | YouTube Data API," *Google Developers*. [Online]. Available: <https://developers.google.com/youtube/v3/getting-started>. [Accessed: 17-Oct-2017].
- [11] "Recommended upload encoding settings - YouTube Help," *Google Help*. [Online]. Available: <https://support.google.com/youtube/answer/1722171?hl=en>. [Accessed: 16-Nov-2017].