2021-06

CYBERCIEGE VIDEOS: FROM ENGLISH TO SPANISH; VÍDEOS DE CYBERCIEGE: DEL INGLÉS AL ESPAÑOL

Rosario, Hector M., Jr.

Monterey, CA; Naval Postgraduate School

http://hdl.handle.net/10945/67806

NAVAL
POSTGRADUATE
SCHOOL

MONTEREY, CALIFORNIA

APPLIED CYBER OPERATIONS
CAPSTONE REPORT

CYBERCIEGE VIDEOS: FROM ENGLISH TO SPANISH; VÍDEOS DE CYBERCIEGE: DEL INGLÉS AL ESPAÑOL

by

Hector M. Rosario Jr.

June 2021

Advisor:	Cynthia E. Irvine
Co-Advisor:	Michael F. Thompson

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** June 2021	**3. REPORT TYPE AND DATES COVERED** Applied Cyber Operations Capstone Report

4. TITLE AND SUBTITLE CYBERCIEGE VIDEOS: FROM ENGLISH TO SPANISH; VÍDEOS DE CYBERCIEGE: DEL INGLÉS AL ESPAÑOL	**5. FUNDING NUMBERS**
6. AUTHOR(S) Hector M. Rosario Jr.	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

CyberCIEGE, a video game developed by the Naval Postgraduate School, supports cybersecurity awareness and education. A set of popular educational movies accompanies the game. The CyberCIEGE video collection was initially developed in English, thus limiting the diversity of its user population. In addition, the video format of the original movies (SWF) is being phased out due to security concerns associated with SWF video players. This capstone addresses the need for increased diversity of those familiar with cybersecurity basics by further introducing the Spanish-speaking community to 21st-century cybersecurity concepts. The CyberCIEGE movies were translated into Spanish to reach a broader audience while retaining the technical meaning of the concepts discussed. To contain costs, we demonstrate that it is possible to use open-source tools freely available for the Linux Operating System platform to produce reliable movies in both English and Spanish for web streaming. Recordings were created with Audacity and integrated as separate tracks to each corresponding movie via OpenShot. Afterward, the movies were exported to the MP4 file format for web streaming. In addition, the original English language movies were directly converted to MP4 file format via OpenShot. Detailed documentation ensures the repeatability of our processes. This work has increased the longevity of the CyberCIEGE video collection while expanding its viewership to a larger, more diverse audience.

14. SUBJECT TERMS CyberCIEGE, educational videos, Spanish translation	**15. NUMBER OF PAGES** 79
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

CYBERCIEGE VIDEOS: FROM ENGLISH TO SPANISH; VÍDEOS DE CYBERCIEGE: DEL INGLÉS AL ESPAÑOL

CPO Hector M. Rosario Jr. (USN)

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED CYBER OPERATIONS

from the

NAVAL POSTGRADUATE SCHOOL
June 2021

Reviewed by:

Cynthia E. Irvine Advisor	Michael F. Thompson Co-Advisor

Accepted by:

Alex Bordetsky
Chair, Department of Information Sciences

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

CyberCIEGE, a video game developed by the Naval Postgraduate School, supports cybersecurity awareness and education. A set of popular educational movies accompanies the game. The CyberCIEGE video collection was initially developed in English, thus limiting the diversity of its user population. In addition, the video format of the original movies (SWF) is being phased out due to security concerns associated with SWF video players. This capstone addresses the need for increased diversity of those familiar with cybersecurity basics by further introducing the Spanish-speaking community to 21st-century cybersecurity concepts. The CyberCIEGE movies were translated into Spanish to reach a broader audience while retaining the technical meaning of the concepts discussed. To contain costs, we demonstrate that it is possible to use open-source tools freely available for the Linux Operating System platform to produce reliable movies in both English and Spanish for web streaming. Recordings were created with Audacity and integrated as separate tracks to each corresponding movie via OpenShot. Afterward, the movies were exported to the MP4 file format for web streaming. In addition, the original English language movies were directly converted to MP4 file format via OpenShot. Detailed documentation ensures the repeatability of our processes. This work has increased the longevity of the CyberCIEGE video collection while expanding its viewership to a larger, more diverse audience.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

GNU	GNU’s Not Unix
HTML5	Hypertext Markup Language Revision 5
MPEG	Moving Picture Experts Group
MP4	MPEG-4 Part 14
SWF	Small Web Format, originally, Shockwave Flash

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would first like to thank God for the strength and imparted resilience in the midst of many challenges faced while conducting this work. This capstone is dedicated to Master Chief Carlton Harris. He inspired me to aim for more extraordinary things regardless of the prospects and is the reason behind being accepted at Naval Postgraduate School. I draw comfort in that his leadership is well represented.

I thank my family and especially my wife, Aileen, who has been by my side from the very start despite the geographical separation during my time at NPS. I am fortunate to have her unwavering support during this endeavor, and I am forever grateful. This work would not have been possible without Dr. Cynthia Irvine and Mr. Michael Thompson's support and guidance, both of whom patiently helped me turn unorganized data into a finished project. I am deeply grateful for their invaluable guidance and support during the challenging times faced while conducting this research. Last but not least, I thank Lieutenant Commander Eric Regnier for his consistent support throughout this research and beyond while at Naval Postgraduate School. It was indeed an honor to give back as a humble token of my gratitude.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

CyberCIEGE is a computer game-like learning tool that was developed to create and increase cybersecurity awareness and education. The tool develops cybersecurity skills by allowing users to create and manage networks of computers. The tool includes extensive on-line help, including video tutorials [1]. CyberCIEGE has been used by hundreds of educational institutions worldwide, and to help train U.S. Government cyber professionals. The CyberCIEGE video tutorials are available for viewing from the CyberCIEGE web page, outside of the game.

The CyberCIEGE informational video library was mastered in Small Web Format (SWF). Sadly, after 20 years since its introduction, Adobe announced in July 2017 support for flash content would cease by the end of 2020. The news prompted software titans like Microsoft to phase out Adobe Flash Player from its products from that point forward [2]. For CyberCIEGE, this news means that students will no longer be able to view the SWF-formatted videos via the public CyberCIEGE web page, (though videos will continue to be available for viewing within the game's on-line help functions.)

Moreover, it is essential to inspire non-English speaking communities in learning cybersecurity concepts starting with Spanish-speaking societies. We seek to initially translate the CyberCIEGE instructional video collection to Spanish because it is the most spoken non-English language in U.S. households according to an analysis conducted by Statista and the folks at Statistical Atlas [3], [4]. As with any translation, transferring the same meaning into a new language can be challenging, which is why we pursue answering the following:

1. How can technically-driven cybersecurity educational videos be translated from English to Spanish?
2. Can English computing terms be translated into Spanish?

Due to budgetary constraints, the work performed must be conducted with open-source tools. This capstone report describes how SWF movies can be converted to MP4 media files by using open-source tools within the Linux operating environment. The report also describes the necessary steps to translate SWF-based movies (voice and caption) into Spanish. In order of utilization, we list the required tools to produce an MP4 video that can be streamed via the web.

Chapter II provides an overview of the reasoning behind choosing to work with open-source software instead of commercially-available tools. We briefly explore why it is essential to preserve the access of the web-based CyberCIEGE video library. Furthermore, Chapter II discusses the importance of translating the video library to benefit the Spanish-speaking population as a starting point.

The requirements and processes described in Chapter III were developed in the following manner assuming the SWF videos are kept locally. First, we play SWF movies with the Gnash command-line tool. Once the video starts playing, Simple Screen Recorder is used to capture both the video and audio. When necessary, we employ the OpenShot video editor to import and edit the captured content, add subtitles, and mask anything within the video we wish to cover. When required, we use Inkscape to transform OpenShot titles into other objects like shapes to mask unwanted visuals and to create subtitles. Finally, once an OpenShot project is exported to MP4 and to the large file sizes OpenShot produces, we reduce the video sizes with HandBrake. Whenever a video needed to be translated into Spanish, Audacity was used to record and edit the narrated audio.

Chapter IV describes particularly pertinent features of the open source tools used for video format conversion and for the language translation process. Chapter V provides a summary and suggestions for future work.

II. MOTIVATION AND OVERVIEW

This chapter provides an overview of the CyberCIEGE game, and then exposes language barriers Spanish-speaking individuals experience in learning cybersecurity concepts and how CyberCIEGE intends to close the language gap. Moreover, we go over the obsolescence that web-based CyberCIEGE videos face as they rely on SWF, which will cease to operate by the end of 2020. We discuss using MP4 as a viable alternative to SWF videos and how the conversion process can be accomplished at no cost by employing open-source software tools.

A. CYBERCIEGE AND ITS TUTORIAL VIDEOS

CyberCIEGE is an educational game project centered on teaching information assurance concepts. Each game scenario is organized into stages; one stage builds upon the other in a fashion similar to the SimCity build-as-you-go concept [5]. Within this virtual environment, administrator or network defender decisions are rewarded or penalized monetarily based on the results, good or bad, of such judgments. The objective of CyberCIEGE is to engage students in reflecting on their failures or successes as they interact with the more than twenty scenarios provided within the game [6].

CyberCIEGE is complimented with movies that can be accessed within the game or separately through the CyberCIEGE web page. The videos or movies, as termed by CyberCIEGE, describe cybersecurity technical concepts for a broad audience. The CyberCIEGE video collection explains concepts like network filters, firewalls, encryption, and public key infrastructure (PKI), to name a few. Presently, there are 13 different videos, ranging from the fundamentals of information assurance to more advanced topics such as PKI installed roots and Virtual Private Networks (VPN) connection profiles [7]. Each movie is between 2:59 and 12:53 minutes long. CyberCIEGE game development was sponsored by the U.S. Navy and the National Science Foundation (NSF). CyberCIEGE has been requested by hundreds of educators, and used by students from many nations worldwide [8].

B. EXPANDING CYBER INTO A GROWING DEMOGRAPHIC

As the 21st century progresses, so does the reliance on the cyber domain. The inherent risks associated with all matters cyber is not exclusive to industry insiders. Users of all backgrounds face cyber threats. This reality requires that users develop computer and cyber literacy skills to interact within cyberspace safely. In the United States, as of 2019 the Hispanic population reached over 60 million, which is 10 million more than in 2010 [9]. Regarding the U.S. Navy and Marine Corps, Hispanics comprise slightly over 20% of the Navy and close to 40% for the Marine corps, according to a Council of Foreign Relations study [10].

A long standing goal of CyberCIEGE developers is to address language barriers, starting with students whose primary language is Spanish. The idea is to support diversity commencing with Spanish as a starting point due to evident demographic trends. The objective of this research is to develop a process to translate existing CyberCIEGE instructional movies into Spanish. Thus all CyberCIEGE movies can be translated and, as the processes are improved, and more multilingual volunteers partner with CyberCIEGE, its videos can be translated to more languages.

Cyberwarfare has become a more integral part of Navy and Marine daily operations. By using CyberCIEGE to broaden the understanding of cybersecurity fundamentals to non-native speaking individuals, the two military branches are bound to benefit from a future generation of motivated cyber warriors.

C. OBSOLESCE OF THE SWF FILE FORMAT

As early as 2010, Adobe Systems, Inc., revealed in their Adobe Product Security Incident Response Team (PSIRT) Blog, that their Flash Player is susceptible to remote code execution. CVE-2010-1297 is identified as the malicious code compromising, among others, their Flash Player application [11]. The National Vulnerability Database (NVD) published by the National Institute of Standards and Technology (NIST), describes the CVE-2010-1297 vulnerability in more detail. By manipulating SWF content dependent upon the authplay.dll, an Acrobat Player dynamic link library, bad actors can remotely execute malicious code to conduct denial of service attacks [12].

In a July 25, 2017 announcement, Adobe Systems, Inc. shared its intention to discontinue Adobe Flash by the end of the year 2020. Adobe cited emerging open standards like HTML5, WebGL, and WebAssembly as the main reason for withdrawing its support from its flash application [13]. Nevertheless, according to a Wired Magazine report, aside from obsolescence, a significant factor for the decision is the developer's constant battle in patching Flash security holes [14]. Regardless of the motivating factors, the reality is that a decision has been made to discontinue SWF support. Browsers by the end of December 2020 will cease to support flash content, as announced by Microsoft [15]. These factors alone negatively impact the CyberCIEGE project. Presently, CyberCIEGE relies heavily on flash to deliver its movies via the web. Without the web-based movies, students will be unable to view the tutorial videos outside of the game. Preemptive measures need to be implemented.

Obsolescence is the motivating factor in switching to an updated industry standard. If a direct SWF replacement is not feasible, converting to a file format like MP4 should be pursued. The potential downside could be losing all SWF functionality, such as interactive buttons (fast forward, play/pause, and rewind), currently present within the flash movies, and perhaps quality and(or) file size.

THIS PAGE INTENTIONALLY LEFT BLANK

III. REQUIREMENTS AND CONVERSION PROCESSES

The objective of the project was to convert web-based CyberCIEGE videos in SWF format to an up-to-date format suitable for web browser-based video streaming. Two major categories of requirements were defined. The first was related to the cost and general accessibility of the tools and the second category related to the tools' technical requirements.

We hoped to employ a single open-source application, if available, to directly convert SWF files to the HTML5 industry-standard. Otherwise, we intended to pursue the MP4 file format as the potential video output. In either case, the requirements were for web browsers to support streaming movies without requiring special plug-ins. The format chosen had to be compatible with CyberCIEGE in-game playing as well. If a single open-source software program or suite for video conversion could not be found, then a combination of programs would suffice to achieve the desired results. To examine the possibilities, software selection considerations were determined.

A. FREELY AVAILABLE OPEN SOURCE TOOLS

For many years, CyberCIEGE has been operated and maintained with little to no funding. Thus, costs associated with the conversion of the web-based videos must be minimized. We expect that the SWF videos will be converted to the MP4 video file format. It would be infeasible to invest in specialty video conversion software. A cost-neutral solution consists of using open source tools that can achieve the same results one would expect from specialized proprietary (and costly) solutions.

Not all purportedly free video processing software is actually free of restrictions and constraints. Some developers tout their software as freeware, but impose unexpected restrictions when video files are saved or exported. The limitations may include the superposition of the developer's logo on the video, limited choice of resolutions, restrictions on the video's length, or deactivation of the product's full features.

For this project, the software tools used must be: free to use; redistributable to other machines used in the project; fully functional; short of an end of life cycle, non-expiring; and Linux compatible. We purposely focused on tools designed to execute as applications on the Linux operating system for two reasons. First, because Linux itself is freely available, and, second, because of the prevalence and wide variety of open-source, freely available software designed for this platform. Finding software that follows the GNU Not Unix (GNU) licensing guidelines is a key factor in ensuring the material produced will not be restricted or limited in any way. The three main GNU categories of unrestricted software licenses are: GNU AGPLv3, GNU GPLv3, and GNU LGPLv3. Software distributed with any of these will suffice for our project [16].

B. TECHNICAL REQUIREMENTS

Initially, we hoped to employ a single open-source application that could directly convert SWF files to the HTML5 industry-standard. If this was not possible, we intended to pursue the MP4 file format as the potential video format. In either case, a project requirement was that the resulting movies support web browser video streaming without requiring special plug-ins. If a single open-source program or software suite could not be found, then a combination of programs would suffice. All software had to be freely available and without licensing restrictions. To examine the possibilities, the main software selection criteria were determined.

The tool or combination of tools to be used had to be able to perform the following functions.

1. **Opening or Playing SWF media files.** Having the ability to play SWF files is crucial. The tool chosen must be able to play SWF media, both audio and video, without video distortion.

2. **Voice recording.** Any software that can record voice in MP3 format will suffice. The program should allow clear voice recording and save the recorded audio in the MP3 file format for maximum compatibility with the video editing software.

3. **Masking SWF objects and content that do not transfer when capturing video.** If certain interactive functionality of the CyberCIEGE videos cannot be exported or converted to MP4, any interactive elements embedded in SWF videos no longer in use should be masked. The software considered must have the ability to mask unwanted content, including the English subtitles.

4. **Creating subtitles.** The tool must produce subtitle file extensions that can be imported into the video editing program to be used. The subtitling program should allow the text to be tailored so that the writing appears sharp relative to the video resolution; subtitles should be legible with edges that are not blurred.

5. **Video editing.** The video editing tool should allow several media elements to be imported for simultaneous manipulation via a single interface. After editing the video, the program should export the final product in MP4 video format.

6. **Small file size**. The CyberCIEGE videos produced in this project will be streamed online as MP4 media. This will require that the file size be small even though MP4 videos tend to be larger than comparable SWF media files. Whichever video editing program is selected will have to export final videos with sizes comparable to their original SWF counterparts. If the ability to control the size of the files is not available within the video editing tool, an auxiliary utility must be sought.

7. **Ease of use.** Most media creation software requires extensive training or familiarization to take advantage of the software's full capabilities. If manning is taking into account, mastering all pertinent applications to create elaborate products takes time. Therefore, to process as many movies as possible before flash content is no longer supported, the software should be self-explanatory. This entails having a user-friendly interface,

self-descriptive terminology, intuitive menu items, and industry-standard features.

8. **Compatibility.** Many consider Hypertext Markup Language revision 5 (HTML5) to be a direct replacement for SWF in terms of media handling. HTML5 can support video and audio without relying on browser plug-ins. HTML5 is independent of Adobe Flash and Java, meaning that it is capable of delivering a full media experience on its own [17]. However, since developing HTML5 skills takes time to learn and master, we decided to convert the CyberCIEGE video collection to MP4, while keeping in mind the December 2020 Adobe Flash end of life support deadline.

   MP4 video files are ideal for streaming over the web [18]. First, for a distraction-free learning experience, CyberCIEGE requires that video content play uninterrupted even during Internet bottlenecks. Second, since the h.264 video codec is the predominant world standard, it is only logical to choose MP4 as the preferred file output when converting SWF media files [19]. Third, transforming SWF files into MP4 does not require in-depth knowledge of the Hypertext Markup Language as is the case for most HTML5 software tools.

9. **Documentation and Community Support.** Documentation explaining the capabilities of the software considered in this project is not essential, but desired. Generally, most tools are self-explanatory by associating their standard Graphical User Interface (GUI) buttons and menu items with those of similar software tools. We prefer using software with manuals, user guides, and/or video tutorials, explaining capabilities beyond what can be accomplished by making guesses based upon experience with similar products. This is particularly helpful when a tool is multi-purpose. However, this project does not seek to improve upon open software processes.

In summary, our efforts focus on proving that MP4 output created with open-source tools for CyberCIEGE is satisfactory for producing web-based videos.

C. SOFTWARE TEST RESULTS

Although more tools were tested, as shown in Tables 1, 2, and 3, we only discuss the software tools used in this project. These tools are discussed in the order in which they were used to produce the desired results.

1. Gnash

Gnash was the simplest program to use for playing SWF media. The only requirement is that the user be familiar with basic Command Line Interface commands to navigate directories and execute the program. Although Gnash, in its simplest form, functions as a plug-in for web browsers, in this project, the tool was used as a standalone program [20]. The performance was consistent, meaning there were no software crashes or compatibility error messages observed. Refer to Appendix C for the workflow on how to play SWF files with Gnash.

2. Simple Screen Recorder

Simple Screen Recorder captured video without issues from the beginning of the project. Initially, the SWF content captured did not contain audio, under the premise that there would be noticeable sound quality issues when opting to capture audio with Simple Screen Recorder. Eventually, sound comparisons were made between the output produced by VLC and Simple Screen Recorder. We also conducted more capturing tests without audio, then compared the sound against the output from OpenShot (discussed further in item 4).

We discovered that OpenShot caused the audio distortion observed and not audio processing the audio produced by Simple Screen Recorder. After this discovery, the Simple Screen Recorder tool was configured to capture video and audio. As a result, the original workflow was simplified, and notes were made that the volume within OpenShot should be significantly reduced before exporting the final product. For more details on how Simple Screen Recorder is used in this project, refer to Appendix C.

3. Audacity

Audacity proved vital in delivering clear and seamless audio output. Changes in voice pitch, narration pace, background noise, narration errors, and disruptions were not anticipated during the planning phase. These items were appropriately addressed with Audacity's audio editing toolset. Moreover, with Audacity, it was possible to record in multiple sessions and merge the content so that it gave the impression that every narration was conducted in a single recording. Audacity was incorporated as an improvement to the original workflow. Details of the Audacity editing techniques used in this project can be reviewed in Appendix D.

4. OpenShot

Ultimately, it was determined that it is best to use OpenShot for importing audio and captured videos. Then build custom shapes that will cover the inoperable play/pause, fast forward and rewind buttons, and English subtitles from the original SWF videos. Finally, producing and synchronizing subtitles from within OpenShot was found to be much easier than using an external subtitling application. Refer to Appendix D for the full list and procedures of how OpenShot is used in this project.

5. Inkscape

As tested, we successfully transformed OpenShot titles into other objects by selecting Inkscape as an advanced OpenShot editing tool. In this project, Inkscape is used to mask inoperable play buttons embedded in the CyberCIEGE movies and cover English subtitles before exporting to MP4. The application is not used as a standalone tool. Instead, Inkscape is employed as a plug-in from within the OpenShot video editing program [21]. The only observed issue was that Inkscape must be closed in order to continue using other OpenShot functions.

Moreover, objects aligned within Inkscape could not be align-matched in OpenShot until Inkscape was closed. The somewhat tedious process of opening and closing Inkscape had to be repeated as many times as necessary to correctly align the Inkscape objects intended to mask unwanted features. Refer to Appendices C and D to learn more about the Inkscape workflow.

6. HandBrake

HandBrake was used to dramatically reduce the size of the MP4 output files produced by OpenShot. HandBrake performed well, reducing MP4 file sizes averaging 90MB to an average of 8MB. Consistent results could be achieved by selecting a configuration scheme that significantly reduced file sizes, while maintaining video quality comparable to the original product. Refer to Figures 1 and 2 for illustrations of a video frame before and after reducing the size of a sample CyberCIEGE MP4 movie. Workflow details can be referenced in Appendix E.

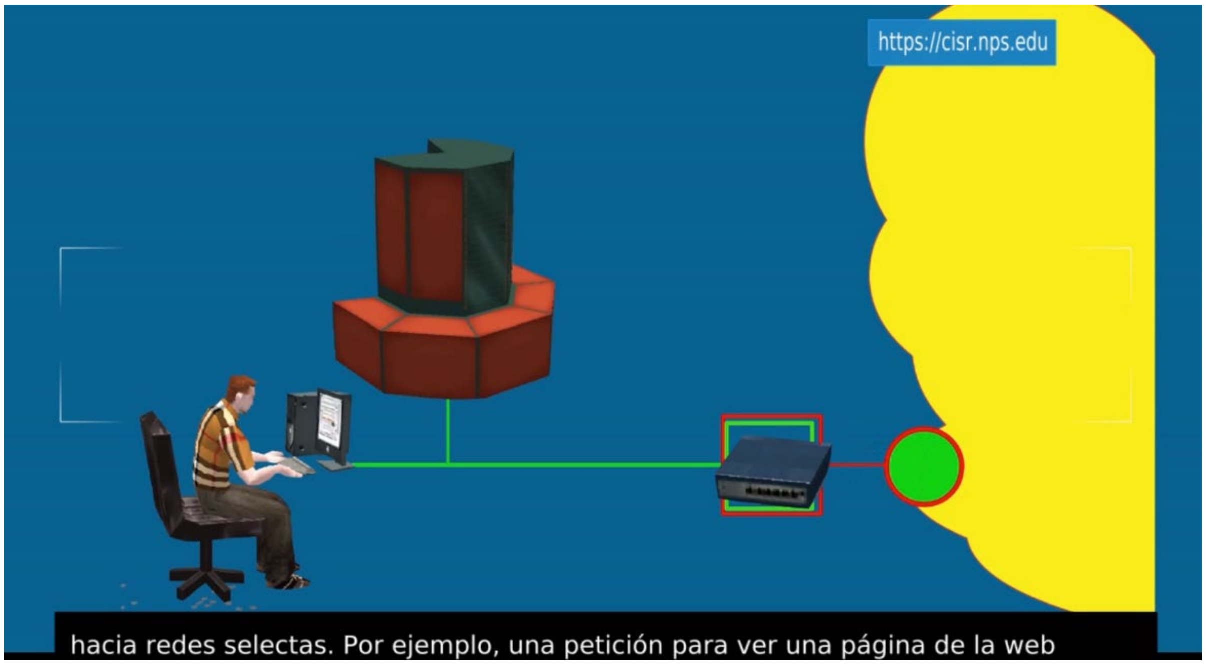

https://cisr.nps.edu
hacia redes selectas. Por ejemplo, una petición para ver una página de la web

Figure 1. Frame from an original MP4 video created with OpenShot totaling 107MB in size

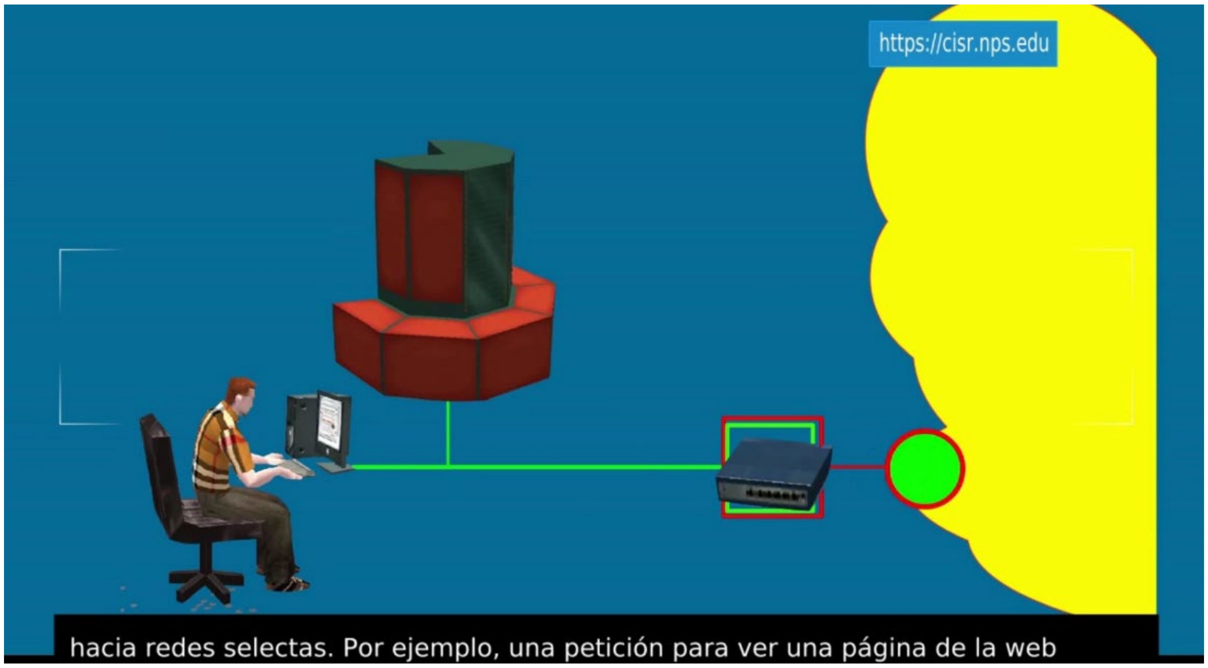

https://cisr.nps.edu
hacia redes selectas. Por ejemplo, una petición para ver una página de la web

Figure 2. Frame of the same video after reducing the file size to 8.95MB with HandBrake

D. SOFTWARE SELECTION CRITERIA

Tables 1, 2, and 3 illustrate the criteria used to evaluate video playing and editing software, sound recording software, and subtitling software for this project. The columns in each table list each selection criterion for a particular software type. The software to be used must allow exporting the end product to MP4 format without sacrificing the quality of the video to be processed in a significant way.

We were unable to find a single open-source software application offering everything needed to produce a Spanish version of a CyberCIEGE video. A suite of software tools that enabled unrestricted video editing allowing media files and text to be combined. Other software for this project consisted of applications for recording audio and correcting imperfections, creating subtitles, and masking English subtitles.

Table 1. Video player and editor software

Software	Open Source	Documentation, Tutorials	SWF File Compatibility	Skill Level Requirement	Community Support	Ubuntu 18.04 Compatible
SWFTools	Yes	Limited	Inconclusive	Advanced	None	No
VLC Media Player	Yes	Yes	Sound only	Moderate	Limited to developer	Yes
Gnash Flash Player	Yes	Yes	Yes, Video and Audio	Moderate	None	Yes
Flash Develop	Yes	Yes	Yes	Advanced	Yes (Forums)	Yes
OpenShot	Yes	Yes, Video tutorials and written documentation	No	Low	Yes	Yes
HandBrake	Yes	Yes	No	Moderate	Yes	Yes

Table 2. Sound recording software

Software	Open Source	Audio Volume Control	Sound Editing Capability	Industry Standard CODEC Support	Community Support	Ubuntu 18.04 Compatible
Linux Sound Recorder	Yes	No	No	Yes	N/A	No
VLC Media Player	Yes	No	Theoretically	Yes	Yes	Yes
Audacity	Yes	Yes	Yes	Yes	Yes	Yes

Table 3. Subtitling software

Software	Open Source	Skill Level Required	Documentation	Video Editor / File Extension Compatibility	Community Support	Ubuntu 18.04 Compatible
Subtitle Composer	Yes	Low	No	No	No	Yes
Subtitle Editor	Yes	Moderate	Yes (Online Tutorial)	No (Not to MP4)	Yes	Yes
Subtitled	Yes (Basic version only)	Advanced	Yes (Basic Tutorials)	No	No	Yes
OpenShot*	Yes	Yes	No (Not for subtitling)	Yes (As MP4 output)	Yes	Yes

*OpenShot is not a subtitle creator or editor but was found to be useful in creating subtitles.

E. CHAPTER SUMMARY

In this chapter, software requirements, selection criteria, and processes were discussed. Various software packages were compared and the specific applications used in this project were discussed. In the next chapter, we will discuss features of the selected software applications that were particularly pertinent to this project.

IV. PERTINENT FEATURES OF OPEN SOURCE APPLICATIONS USED IN THIS PROJECT

In this chapter, we discuss the results obtained when comparing software tools. Refer to Tables 1, 2, and 3 for our selection criteria overview, and Section E of Chapter III for more details. Here, we expand on the reasons each of these programs was chosen.

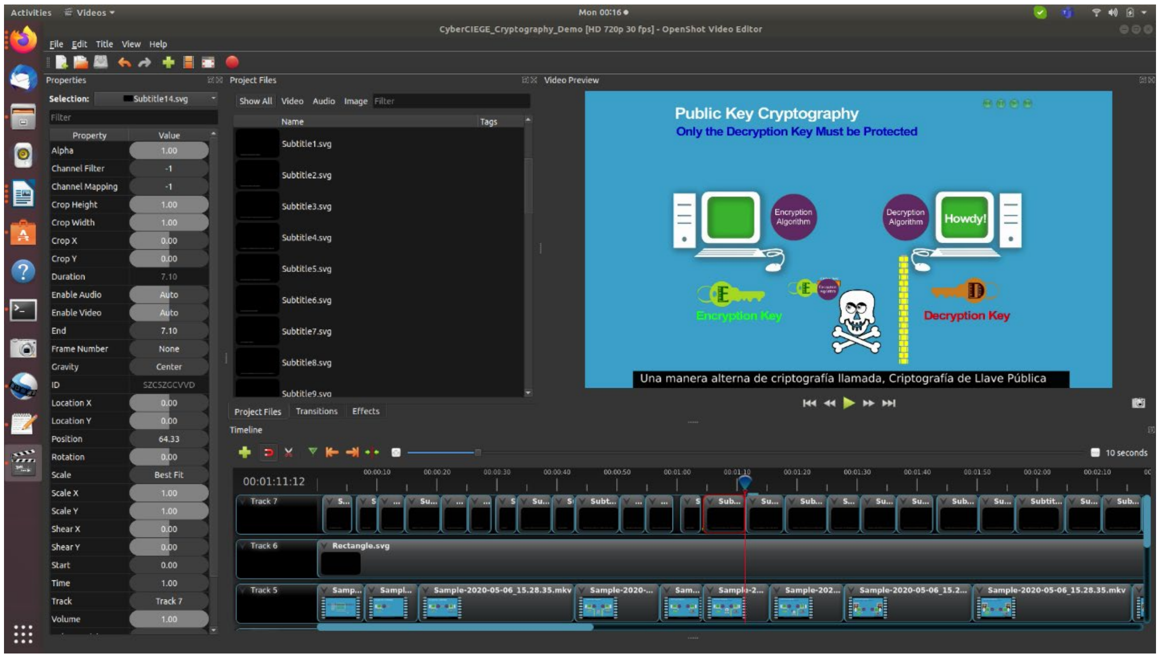
The detailed processes for creating the new videos are described in the Appendices. Appendix A shows how to convert SWF files to MP4. Appendix B lists the necessary steps for creating Spanish versions of CyberCIEGE videos. Appendix C is an improved workflow for converting SWF videos to MP4, where different steps are required to accomplish the task. Appendix D is an improved version of Appendix B; it requires fewer steps to create the Spanish version of CyberCIEGE videos. Appendix D also replaces the default Linux Sound Recorder application with Audacity. Appendix E lists the software application and necessary steps used to shrink the size of MP4 videos.

A. OPENSHOT

OpenShot is the only identified open-source video editing program capable of combining the different media elements produced by other software tools into a new video recording [22]. It can combine video, audio, images, shapes, and text created with the Inkscape application, which serves as a plug-in within OpenShot. More importantly, the program can export the final product in the standard MP4 file format. Figure 3 illustrates the interface for OpenShot. During testing, we discovered a bonus feature in OpenShot: the ability to create subtitles. We originally envisioned using OpenShot solely for video editing and exporting the final product to MP4. However, after discovering that OpenShot can be used to create subtitles, we used OpenShot for this purpose as well. Afterward, the workflow was modified to use OpenShot for creating subtitles. At first, using OpenShot for subtitling was time consuming because it was not developed for this purpose.

Initially, three different subtitling applications, as shown in Table 3, were tested. Not only were the applications not intuitive enough, we did not find how to export the result to MP4. We searched the web for alternatives and ultimately, our research led us to experiment with OpenShot's title feature. Even after performing several steps to alter OpenShot title templates, creating subtitles with OpenShot was more user-friendly than employing a subtitling program.

We opted to continue our project by using OpenShot for subtitling because the subtitling tools in Table 3 are not compatible with OpenShot. Since OpenShot is not compatible with file extensions created with the subtitling programs listed in Table 3, the text within the subtitling programs had to be saved as images. Afterward, the subtitles were imported to OpenShot, but the text was distorted or pixelated. At times the imported subtitles blurred the surrounding pixels. Therefore, adding subtitles with any of the subtitling programs listed in Table 3 after exporting an OpenShot MP4 video, was not possible. Refer to Appendix B and Appendix D for instructions on how to create subtitles within OpenShot.



Selection: Subtitle14.svg

Property	Value
Alpha	1.00
Channel Filter	-1
Channel Mapping	-1
Crop Height	1.00
Crop Width	1.00
Crop X	0.00
Crop Y	0.00
Duration	7.10
Enable Audio	Auto
Enable Video	Auto
End	7.10
Frame Number	None
Gravity	Center
ID	SZCSZGCVVD
Location X	0.00
Location Y	0.00
Position	64.33
Rotation	0.00
Scale	Best Fit
Scale X	1.00
Scale Y	1.00
Shear X	0.00
Shear Y	0.00
Start	0.00
Time	1.00
Track	Track 7
Volume	1.00

Figure 3. OpenShot Video Editor during an editing session

OpenShot was used for three purposes in this project:

1. Subtitles

Initially, the intent was to use specialized software like Subtitle Editor or a similar tool to compose Spanish subtitles. We found it unwieldy to synchronize the text to the narration in a continuous track interface. In contrast, by modifying OpenShot titles selectable at the top of the program, it was easier to create subtitles. However, it was necessary to pre-install and use the Inkscape plug-in to customize the title text. The subtitling process is described in Appendix D, Section D.iii. For a detailed description of Inkscape refer to Subsection 3 of Appendix D.

2. Masking

All original CyberCIEGE movies contain embedded buttons to rewind, fast forward, and play/pause the videos. The interactive buttons are rendered useless when capturing SWF videos for later conversion to MP4. Therefore, it was necessary to mask the buttons so that the viewer is not distracted or disappointed when attempting to use those features. Buttons were masked by using OpenShot's Inkscape plug-in to convert a title into a shape filled with a color that approximately matched the movie background. Refer to Appendix C, Section ii for workflow details. For a detailed Inkscape description, refer to Subsection 3 of Chapter IV.

3. Video Editing

The primary reason for choosing OpenShot was to compose videos exported as MP4 movies without licensing restrictions. We used OpenShot to incorporate subtitles, embed shapes covering what we want to mask, add background music and narrations, manipulate visuals to match narration pace, and truncate unwanted portions of captured videos.

B. AUDACITY

Audacity is versatile for recording, correcting prolonged pauses while narrating, removing background noise between sentences, and adjusting the volume when needed. It is a powerful application because audio can be duplicated, isolated by channel, and exported to the codec of choice, yet it is easy to install and operate; the Audacity interface is intuitive. Refer to Appendix D Section C for audio editing details.

C. INKSCAPE

Inkscape is a vector graphics design application used to create artwork, logos, illustrations, etc., all of which can be scaled [23]. Within the scope of this project, Inkscape is used to create subtitle templates. Although it is possible to use the program as a separate application, we chose to employ Inkscape as a plug-in for OpenShot. Inkscape allowed us to take an OpenShot title template and convert it into a subtitle template. With Inkscape it was possible to adjust and change the title's text size, position in the video's viewable area, font type, color, and effect. These are desired features that OpenShot fails to provide on its own.

With Inkscape, it is also possible to design shape templates that can be incorporated in OpenShot videos to mask unwanted content. Using Inkscape, we created shape templates (circles, squares, or rectangles) by manipulating OpenShot text-based title templates. Refer to the Inkscape workflow listed in Appendix D for more details.

D. SIMPLE SCREEN RECORDER

It was impossible to find a functional open-source SWF manipulation application that can import SWF movies and convert them to MP4 video format. Alternately, the flash video content can be played and captured from the screen. For the alternative approach, Simple Screen Recorder was used to capture the animation because it is simple to use, captures video with quality nearly identical to the original source as tested, and exports its recordings in a format that OpenShot can import for editing. As the name suggests, the program only captures the screen. Simple Screen Recorder does not support editing of the captured footage as certain proprietary screen capture tools do. The program was used to capture and export the video content into the MP4 file format and nothing more. Appendices C and D delineate how Simple Screen Recorder is used in this project.

E. HANDBRAKE

While testing OpenShot, regardless of the MP4 video codec selected, it was not possible to produce output files of a small enough size suitable for online streaming and distribution. By using HandBrake, the size of MP4 files could be reduced from an average of 90 megabytes (MB) to an average of 8MB without a notable degradation of the video quality. The size reduction was primarily attained by reducing the frames per second from 30 fps to 15 fps. Refer to Appendix E to learn more about the HandBrake workflow.

F. GNASH FLASH PLAYER

Gnash is a GNU flash (SWF) player, chosen primarily for its non-proprietary licensing and ability to play SWF files without restrictions. This project only used Gnash to play SWF files and does not use any options Gnash may offer. Refer to Appendix C for program execution details.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSION

We conclude that when translating into Spanish, it is possible to coherently discuss cyber matters with the same context and ease as the original language. We discovered that more open-source software tools are needed than initially anticipated to create Spanish versions of the original CyberCIEGE movies or to simply convert from SWF to MP4. Moreover, we observed that many separate software tools needed to be combined to create the same end result as with retail specialty media creation programs which was time-consuming.

By utilizing open source tools, we obtained consistent results where the video and audio quality is adequate, and the videos are suitable for web streaming. We did not encounter issues when attempting to play the MP4 media files produced with standalone or web applications. Overall, we proved that it is possible to transform SWF videos to MP4 without issues. We also confirmed that it is possible to translate cybersecurity concepts into Spanish while still retaining the meaning of the topics discussed.

Once the workflow is practiced a few times, the inconvenience of using several programs becomes irrelevant. CyberCIEGE and other budget-conscious organizations seeking to update or replace their flash media library will be able to do so without significant or any expense.

The ultimate benefit to the Navy from this work is its support for diversity, i.e., the ability to produce educational material that has been translated into the second most common language in use within the United States. With CyberCIEGE movies in two languages, more members of the Naval community will be able to appreciate the importance of cybersecurity.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A. PROCEDURES FOR CONVERTING SWF VIDEO FILES TO MP4 FORMAT

This section lists the steps required to convert an SWF file to MP4 format with open source tools. The process does not consider efficiency but rather obtaining results with freely available tools listed in this section. The outlined steps are a reflection of what ultimately allowed us to convert a CyberCIEGE SWF video file to MP4.

A. GNASH

The purpose of Gnash within the scope of the project is to play CyberCIEGE SWF video files for capture with the Simple Screen Recorder software. It is a very simple process as demonstrated in the following steps:

1. Open a terminal window (Command Line Interface)
2. Browse to the folder where the SWF file resides
3. Initiate the SWF file playing session by typing gnash followed by the name of the file
4. Press Enter key
5. Maximize the window (do not worry if the file begins to play but do ensure that step B.12 has been completed)
6. Let the content play in a loop until an entire sequence is captured by the Simple Screen Recorder
7. Upon completion, the player may be closed

B. SIMPLE SCREEN RECORDER

In the context of this project, Simple Screen Recorder is used to capture SWF video content. In continuation are the necessary steps to complete an SWF video capture task:

1. Open the Simple Screen Recorder application by clicking Ubuntu’s Show Applications menu button and partially or completely typing the word “simple” in the search bar
2. Press Continue
3. Select Record a fixed rectangle under the Video input
4. Press select rectangle
5. Select the viewable portion of the video to be captured and release the cursor when the desired area has been selected (viewable area may vary between videos)
6. At the bottom where it says Audio input, deselect the Record audio checkmark
7. Press Continue
8. Specify file name and location in the Save as: menu item
9. If not already selected, under Container, change the file type to MP4 via the dropdown arrow. All other default options can remain unchanged
10. Press Continue to move to the screen capture prompt
11. Before initiating the screen capture, ensure the SWF file is accessible ready for execution or already playing
12. Press Start recording
13. Minimize the Simple Screen Recorder software and remove from view the mouse cursor
14. Upon fulfillment of the desired content to be captured, press Save recording to stop and save the screen capture task

NOTE: It is important to minimize all other content during the capture procedure including the mouse pointer to avoid undesired items from reflecting in the final product.

C. VLC MEDIA PLAYER

The VideoLan Community (VLC) Media for the purposes of this project is to extract and export the audio from SWF files to MP3 format. In continuation are the necessary steps to extract the audio from SWF media files:

1. Open VLC Media Player
2. Select Media > Convert/Save
3. Under the File tab press the Add button
4. Browse to the SWF file location
5. At the bottom of the pop-up window where it says Files of type, change the default selection to All files via the drop-down arrow
6. Once the desired media file which in this case is an SWF file is selected, press Open
7. To export the SWF file to another media format, where it says Save / Convert at the bottom, this must be changed to Concert by means of the drop-down arrow
8. If not selected by default the desired output must be changed to Audio – MP3
9. Choose a destination directory
10. Press Save then Start for the audio extraction to take place.

D. OPENSHOT VIDEO EDITOR

The premise of using OpenShot during the SWF file conversion process is to combine separately processed audio and video into a single MP4 video output file. Masking non-functional buttons that some original SWF videos may have that do not translate to MP4 is also a feature utilized. The following steps describe the process:

i. ***Importing video and audio***

1. Once OpenShot has been opened press the green plus (+) sign or select the File menu item followed by Import Files
2. Browse to the file location where the video output produced with Simple Screen Recorder is located
3. Select the file and press Open
4. Repeat steps 2, and 3 to import the audio file created with VLC Media Player
5. Right-click the audio track and when the pop-up menu loads, select Display > Show Waveform (this step is optional but helps in the editing process)
6. Once the video and audio tracks have been edited, aligned, and synchronized, press the large red button or select the File menu item then, Export Project > Export Video
7. Enter a file name in the File Name text box
8. Chose a file destination folder if the default destination must differ from the suggested path
9. It is recommended to leave all default settings under the Simple tab unchanged to avoid video and audio distortions and synchronization issues
10. Press Export Video to initiate the video output process

ii. ***Masking inactive or irrelevant SWF video content***

1. Click the Title menu potion and click title
2. Under Choose a Template select the TITLE (Camera border) template
3. Where it says Advanced press Used Advanced Editor
4. Press the Ctrl + A keystrokes to select all then press the delete key to remove the invisible content on the template

5. To the left of the program there are several icons. Select the 7th icon from top to bottom or simply press the F4 key to create a rectangle
6. Press F1 to activate the select tool again and touch the rectangle
7. Select Fill and Stroke located in the right pane to apply and gradually change colors as needed
8. The shape may be altered by dragging the shape’s arrows as needed. For greater precision, the shape may be manipulated by entering pixel values in the X, Y, W, and H text boxes at the top of the program
9. Exit the program then press Save when prompted

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. PROCEDURES FOR TRANSLATING SWF VIDEOS INTO ANOTHER LANGUAGE

This section lists the detailed process to produce an MP4 video by using a combination of open source tools. The focus is to produce a quality end product without resorting to commercially available offerings. The applications and steps outlined does not consider if the process is efficient. Although the project focuses in translating CyberCIEGE videos into Spanish, the steps delineated in this section apply to any language.

A. GNASH

Gnash is utilized in this project to play CyberCIEGE SWF video files while the Simple Screen Recorder software captures the content. It is a very simple process as demonstrated in the following steps:

1. Open a terminal window (Command Line Interface)
2. Browse to the folder where the SWF file resides
3. Initiate the SWF file playing session by typing gnash followed by the name of the file
4. Press Enter key
5. Maximize the window (do not worry if the file begins to play but do ensure that step B.12 has been completed)
6. Let the content play in a loop until an entire sequence is captured by the Simple Screen Recorder
7. Upon completion, the player may be closed

B. SIMPLE SCREEN RECORDER

In the context of this project, Simple Screen Recorder is used to capture SWF video content. In continuation are the necessary steps to complete an SWF video capture task:

1. Open the Simple Screen Recorder application by clicking Ubuntu's Show Applications menu button and partially or completely typing the word "simple" in the search bar
2. Press Continue
3. Select Record a fixed rectangle under the Video input
4. Press select rectangle
5. Select the viewable portion of the video to be captured and release the cursor when the desired area has been selected (viewable area may vary between videos)
6. At the bottom where it says Audio input, deselect the Record audio checkmark
7. Press Continue
8. Specify file name and location in the Save as: menu item
9. If not already selected, under Container, change the file type to MP4 via the dropdown arrow. All other default options can remain unchanged
10. Press Continue to move to the screen capture prompt
11. Before initiating the screen capture, ensure the SWF file is accessible ready for execution or already playing
12. Press Start recording
13. Minimize the Simple Screen Recorder software and remove from view the mouse cursor
14. Upon fulfillment of the desired content to be captured, press Save recording to stop and save the screen capture task

NOTE: It is important to minimize all other content during the capture procedure including the mouse pointer to avoid undesired items from reflecting in the final product.

C. LINUX SOUND RECORDER

The purpose of using the default Linux Sound Recorder software is to record any narrations with or without a script with simplicity. The simple steps required to fulfill this requirement in the project are listed as follow:

1. Click the Show Applications button to the lower left corner of Linux
2. In the search text box partly or fully type "sound" for the Sound Recorder icon to sow in the results
3. Open Sound Recorder and press Record once the video to be translated has commenced playing to start narrating at the pace of events viewed in the original video
4. To stop the recording process press Done
5. Recordings are automatically saved to the Home > Recordings directory if browsing via the File Manager or /home/username/Recordings/ if navigating via a terminal window

NOTE: In step 3 narrating while the original video plays is only a suggestion and not a requirement to minimize instances where extra editing may be required to compensate for language pace differences. It is recommended to fully narrate the script even if the original video ends. The purpose of playing the video while narrating is to provide visual cues for dynamically spoken narrations that help maintain the same volume and tone as the original language.

D. VLC MEDIA PLAYER

The VideoLan Community (VLC) Media for the purposes of this project is to extract and export the audio from SWF files to MP3 format. In continuation are the necessary steps to extract the audio from SWF media files:

1. Open VLC Media Player
2. Select Media > Convert/Save
3. Under the File tab press the Add button

4. Browse to the SWF file location

5. At the bottom of the pop-up window where it says Files of type, change the default selection to All files via the drop-down arrow

6. Once the desired media file which in this case is an SWF file is selected, press Open

7. To export the SWF file to another media format, where it says Save / Convert at the bottom, this must be changed to Concert by means of the drop-down arrow

8. If not selected by default the desired output must be changed to Audio – MP3

9. Choose a destination directory

10. Press Save then Start for the audio extraction to take place.

E. OPENSHOT VIDEO EDITOR

The premise of using OpenShot during the SWF file conversion process is to combine separately processed audio and video into a single MP4 video output file. Masking non-functional buttons that some original SWF videos may have that do not translate to MP4 is also a feature utilized. The following steps describe the process:

i. Video editing

1. Once OpenShot has been opened press the green plus (+) sign or select the File menu item followed by Import Files

2. Browse to the file location where the video output produced with Simple Screen Recorder is located

3. Select the file and press Open

4. Repeat steps 2, and 3 to import audio and image files

5. As audio files are imported and dragged to a track, right-click the audio track. When the pop-up menu loads, select Display > Show Waveform (this step is optional but helps in the editing process)

6. Organize tracks in the following order starting at the top:

7. Subtitles track

8. Shape(s) track masking unwanted content

9. Video track

10. Narration audio track

11. Background music audio track

12. Complimentary sound effects audio track(s)

13. Once the video and audio tracks have been edited, aligned, and synchronized, press the large red button or select the File menu item then, Export Project > Export Video

14. Enter a file name in the File Name text box

15. Chose a file destination folder if the default destination must differ from the suggested path

16. It is recommended to leave all default settings under the Simple tab unchanged to avoid video and audio distortions or synchronization issues

17. Press Export Video to initiate the video output process

ii. Masking inactive or irrelevant SWF video content

1. Click the Title menu option and click title

2. Under Choose a Template select the TITLE (Camera border) template

3. Where it says Advanced press Used Advanced Editor

4. Press the Ctrl + A keystrokes to select all then press the delete key to remove the invisible content on the template

5. To the left of the program there are several icons representing shapes. For instance, select the 7th icon from top to bottom to create a rectangle or simply press the F4 key. If a circle needs to be created it can be clicked or the F5 key can be pressed to do the same

6. Press F1 to activate the select tool and touch the shape before attempting to modify it

7. Select Fill and Stroke located in the right pane to apply and gradually change colors as needed

8. The shape's size and position may be altered by dragging the shape's arrows as needed. For greater precision, the shape can be manipulated by entering pixel values in the X, Y, W, and H text boxes at the top of the Inkscape program

9. Exit Inkscape then press Save when prompted

10. Upon returning to the Titles pop-up window, where it says File Name, replace the automatically generated name with a self-descriptive naming convention distinguishing the modified title template from a text-based title

11. Press Save to add the newly created shape

NOTE: Inkscape must be installed as a prerequisite before modifying a title.

iii. OpenShot for Subtitles

OpenShot is not a subtitle editor however, it has been identified as the best tool for the task because it allows seamless text integration with the other imported media elements. The required steps are as follow:

1. Launch OpenShot by selecting Ubuntu's Show Applications icon and partially or completely typing openshot

2. Select the Title menu option then Title or press the CTRL + T keystrokes

3. Once the Titles window opens, select the Bar 3 template or preferably, Camera border which is the template used for this project

4. Type a preferred file name related to the project at hand followed by a number as many instances of the same title template will be used to create a subtitle effect

5. Refer to step iv.1

6. In the Line 1 text box enter or copy/paste a portion of the translated text not to exceed 75 characters in length including spaces

7. Before saving make sure that the file name has a number increment at the end after the first or prior number has been used. For example, Subtitle1, Subtitle2, etc.

8. Press Save

9. Recommended: create new subtitle files with enough text to cover spoken words for timing purposes as the video progresses. Creating subtitles ahead of time can make it harder to align written text with the narration

10. To crate subsequent subtitles, the previously created subtitle may be used as a template to avoid performing steps 6.a through 6.e as follows:

11. Right-click the latest created subtitle

12. Change the file name's number to the next increment

13. Perform step 5.f and 5.h

NOTE: If performing a copy/paste function ensure it is done from a text editor and not from a word processing application in order to prevent hidden special characters from pasting into OpenShot's title function.

iv. Inkscape plug-in for OpenShot

The role of Inscape in this project is to personalize OpenShot titles to resemble subtitle letters in terms of shape, size, position and screen length. Inscape must be preinstalled. The following steps are required to align subtitles to the video edit in progress:

1. Upon completion of step iii.4 select the Use Advanced Editor option
2. If not configured by default, change the font size to 30pt, make sure under Font > Font family sans-serif is selected, ensure CSS Normal and Face Regular under Style are chosen, and Align left is selected
3. The textbox dimensions and position should be configured at the top menu as follow for best fit in relation to the total viewable area based on a 1920 x 1080 (16:9) resolution:
4. W: 1380.664
5. H: 40.312
6. X: 204.405
7. Y: 12.698
8. Upon completion of the above parameters Inscape may be closed after shaving the changes
9. Refer to step iii.6

NOTE: This step is only required once per project or if switching from subtitle style from another within the same project.

v. Synchronizing videos to match translated narration length

OpenShot makes it possible to freeze, slow down or accelerate frames. Pacing differences between languages are addressed by employing the effects previously listed. The following procedures can be performed to dissect a video multiple times without degrading it:

1. Create scenes to delay or accelerate them

2. Pause the video at a point when an alteration needs to be made
3. Ensuring that the vertical red line play progress indicator is at the desired time, pause the video and press the scissors icon to establish the start of a separate scene
4. Resume the video until the point when the scene matches what is narrated in the new language
5. Once more pause the video then press the scissors icon again to establish the end of the scene
6. Right-click the scene that was cut from the rest of the video and when the pop-up menu displays, browse to Time > Slow > Forward then press 1/2x. If the segment needs to be accelerated browse to Time > Fast > Forward then press 2x
7. Create scenes to freeze frames
    - (a) Pause the video at a point when an alteration needs to be made
    - (b) Ensuring that the vertical red line play progress indicator is at the desired time, pause the video and press the scissors icon to establish the start of a separate scene
    - (c) Resume the video until the point when the scene matches what is narrated in the new language
    - (d) Once more pause the video then press the scissors icon again to establish the end of the scene
    - (e) Right-click the scene that was cut from the rest of the video and when the pop-up menu displays, browse to Time > Freeze then select the amount of seconds listed as appropriate.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C. IMPROVED PROCEDURES FOR CONVERTING SWF VIDEO FILES TO MP4 FORMAT

This section lists the improved process for converting an SWF video file to MP4 format. After thorough workflow testing it was determined that VLC Media Player is not needed. No distinguishable sound quality change was observed when opting to capture sound as well with Simple Screen Recorder. Refer to sub section B for the Simple Screen Recorder process outline. For procedures on how to shrink the size of MP4 files produced with OpenShot, refer to Appendix E.

A. GNASH

The sole purpose of Gnash within the scope of the project is to play CyberCIEGE SWF video files. It is a very simple process as demonstrated in the following steps:

1. Open a terminal window (Command Line Interface)
2. Browse to the folder where the SWF file resides
3. Initiate the SWF file playing session by typing gnash followed by the name of the file
4. Press Enter key
5. Maximize the window (do not worry if the file begins to play but do ensure that step B.12 has been completed)
6. Let the content play in a loop until an entire sequence is captured by the Simple Screen Recorder
7. Upon completion, the player may be closed

B. SIMPLE SCREEN RECORDER

In the context of this project, Simple Screen Recorder is used to capture SWF video content. In continuation are the necessary steps to complete an SWF video capture task:

1. Open the Simple Screen Recorder application by clicking Ubuntu's Show Applications menu button and partially or completely typing the word "simple" in the search bar
2. Press Continue
3. Select Record a fixed rectangle under the Video input
4. Press select rectangle
5. Select the viewable portion of the video to be captured and release the cursor when the desired area has been selected (viewable area may vary between videos)
6. At the bottom where it says Audio input, ensure the Record audio checkmark is selected
7. Press Continue
8. Specify file name and location in the Save as: menu item
9. If not already selected, under Container, change the file type to MP4 via the dropdown arrow. All other default options can remain unchanged
10. Press Continue to move to the screen capture prompt
11. Before initiating the screen capture, ensure the SWF file is accessible ready for execution or already playing
12. Press Start recording
13. Minimize the Simple Screen Recorder software and remove from view the mouse cursor
14. Upon fulfillment of the desired content to be captured, press Save recording to stop and save the screen capture task

NOTE: It is important to minimize all other content while capturing including the mouse pointer to avoid undesired items from reflecting in the final product. The SFW video can play continuously in the background while Simple Screen Recorder is configured to start a new capturing session. Extra non-continuous video sequences can be edited out afterwards.

C. OPENSHOT VIDEO EDITOR

The premise of using OpenShot during the SWF file conversion process is to import captured videos and exporting them to MP4 after editing out unwanted scenes. Masking non-functional buttons or other unwanted content no longer relevant after capturing SWF videos is also accomplished with OpenShot. The following steps describe the process:

i. Importing video files for editing

1. Once OpenShot has been opened press the green plus (+) sign or select the File menu item followed by Import Files
2. Browse to the file location where the video output produced with Simple Screen Recorder is located
3. Select the file and press Open
4. Unwanted scenes can be deleted by dragging the vertical red line that tracks the progress when a video is played, and releasing or adjusting as necessary while a video is paused. It is helpful to play the video while observing the time elapsed, then dragging the red line up to the end of the clip that needs to be deleted then release
5. Click the scissors icon and click the video as close as possible to the red line to cut the undesired clip
6. Right-click the video clip to be deleted and select Remove clip or simply click the clip then press the delete key
7. After all undesired clips have been deleted, right-click the remaining clip
8. Separate the video's audio into a separate track by right-clicking the video track and choosing Separate Audio > Single Clip (all channels) or Multiple Clips (each channel) as appropriate

9. Right-click the audio track and when the pop-up menu loads, select Display > Show Waveform (this step is optional but helps in the editing process)

10. Adjust the audio track volume by selecting it and scrolling down to the Volume option listed in the Properties pane. Double-click the numerical value and manually type a number above or below the default. For example, 0.75 represents 75% volume. If the Properties pane is not open, it may be activated by right-clicking a track then selecting Properties

11. To export the project, press the large red button at the top of the program or select the File menu item then, Export Project > Export Video

12. Enter a file name in the File Name text box

13. Chose a file destination folder if the default destination must differ from the suggested path

14. It is recommended to leave all default settings under the Simple tab unchanged to avoid audio/video synchronization issues

15. Press Export Video to initiate the output process

NOTE: When exporting the final product with OpenShot, sound quality degrades slightly. This issue can be overcome by adjusting the volume level to a lesser degree. Adjustment requirements will vary.

ii. Masking inactive or irrelevant SWF video content

1. Click the Title menu option and click title

2. Under Choose a Template select the TITLE (Camera border) template

3. Where it says Advanced press Used Advanced Editor

4. Press the Ctrl + A keystrokes to select all then press the delete key to remove the invisible content on the template

5. To the left of the program there are several icons representing shapes. For instance, select the 7th icon from top to bottom to create a rectangle or simply press the F4 key. If a circle needs to be created it can be clicked or the F5 key can be pressed to do the same

6. Press F1 to activate the select tool and touch the shape before attempting to modify it

7. Select Fill and Stroke located in the right pane to apply and gradually change colors as needed

8. The shape's size and position may be altered by dragging the shape's arrows as needed. For greater precision, the shape can be manipulated by entering pixel values in the X, Y, W, and H text boxes at the top of the Inkscape program

9. Exit Inkscape then press Save when prompted

10. Upon returning to the Titles pop-up window, where it says File Name, replace the automatically generated name with a self-descriptive naming convention distinguishing the modified title template from a text-based title

11. Press Save to add the newly created shape

NOTE: Inkscape must be installed as a prerequisite before modifying a title.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D. IMPROVED PROCEDURES FOR TRANSLATING SWF VIDEOS INTO ANOTHER LANGUAGE

This section reflects the improved process by which SWF videos can be converted to MP4 with open source tools. Two major changes have been made from the workflow delineated in appendix B. Firstly, VLC Media Player is no longer required. The other changed made consists of replacing the Linux Sound Recorder software with Audacity to edit the narrations and eliminate sound imperfections.

A. Gnash

Gnash is utilized in this project to play CyberCIEGE SWF video files while the Simple Screen Recorder software captures the content. It is a very simple process as demonstrated in the following steps:

1. Open a terminal window (Command Line Interface)
2. Browse to the folder where the SWF file resides
3. Initiate the SWF file playing session by typing gnash followed by the name of the file
4. Press Enter key
5. Maximize the window (do not worry if the file begins to play but do ensure that step B.12 has been completed)
6. Let the content play in a loop until an entire sequence is captured by the Simple Screen Recorder
7. Upon completion, the player may be closed

B. Simple Screen Recorder

In the context of this project, Simple Screen Recorder is used to capture SWF video content. In continuation are the necessary steps to complete an SWF video capture task:

1. Open the Simple Screen Recorder application by clicking Ubuntu’s Show Applications menu button and partially or completely typing the word “simple” in the search bar
2. Press Continue
3. Select Record a fixed rectangle under the Video input
4. Press select rectangle
5. Select the viewable portion of the video to be captured and release the cursor when the desired area has been selected (viewable area may vary between videos)
6. At the bottom where it says Audio input, ensure the Record audio checkmark is selected
7. Press Continue
8. Specify file name and location in the Save as: menu item
9. If not already selected, under Container, change the file type to MP4 via the dropdown arrow. All other default options can remain unchanged
10. Press Continue to move to the screen capture prompt
11. Before initiating the screen capture, ensure the SWF file is accessible ready for execution or already playing
12. Press Start recording
13. Minimize the Simple Screen Recorder software and remove from view the mouse cursor
14. Upon fulfillment of the desired content to be captured, press Save recording to stop and save the screen capture task

NOTE: It is important to minimize all other content while capturing including the mouse pointer to avoid undesired items from reflecting in the final product. The SFW video can play continuously in the background while Simple Screen Recorder is configured to start a new capturing session. Extra non-continuous video sequences can be edited out afterwards.

C. Audacity

The purpose of using the Audacity recording software is to record any narrations with or without a script and to correct sound imperfections, and background noise. The simple steps required to fulfill this requirement in the project are listed as follow:

i. Basic setup

1. After opening Audacity go to Edit > Preferences and under Devices > Recording ensure that pulse and 2 (Stereo) is selected
2. Under Recording if not already selected ensure that in Play through other tracks while recording (overdub) is check-marked. Do the same for Enable under Sound activated Recording
3. Press OK

ii. Recording

1. At the top of the program press the red button or go to Tracks > Add New > Stereo Track if there is a need to adjust default parameters before recording
2. Perform a test recording, observing the equalizer at the top. If the equalizer level reaches the orange or red color there is a good change the playback will reflect some sound distortion. In this case microphone distance relative to the narrator and recording volume would have to be tweaked for optimal sound quality
3. To stop the recording process, press the black square at the top of the program

iii. Removing sound anomalies or noticeable background noise that cannot be corrected with the Noise Reduction tool

1. Identify a portion of the sound track that has a flat or minuscule sound wave

2. Select a small portion of the sound wave (track) by clicking and holding the mouse button, then dragging the cursor until the small portion is selected.

3. Go to Edit and select Copy or use the Ctrl+C keystroke combination to copy the sound wave sample

4. Repeat step 2 and paste to substitute the selected area by going to Edit and selecting Paste or using the Ctrl+V keystroke combination

iv. ***Delete prolonged pauses between words***

1. For better precision zoom the sound track by choosing the View menu item and selecting Zoom > Zoom in (optional)

2. Select the bad portion of the sound wave (track) by clicking and holding the mouse button, then dragging the cursor until the bad portion is selected.

3. Press Delete or at the menu choose Edit > Delete

NOTE: It is advisable to select fractions of a second at a time then replaying the portion of the audio track being edited to maintain a natural speech pace.

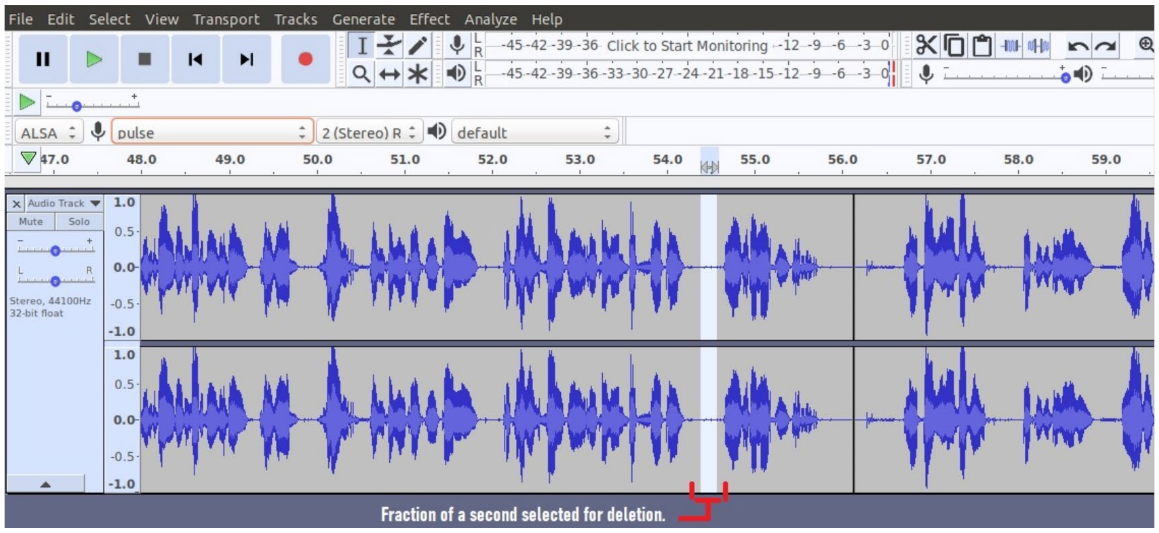


Figure 4. Example of an extended silence gap being selected for deletion to maintain a uniform narration flow.

D. OPENSHOT VIDEO EDITOR

The premise of using OpenShot during the SWF file conversion process is to combine separately processed audio and video into a single MP4 video output file. Masking non-functional buttons that some original SWF videos may have that do not translate to MP4 is also a feature utilized. The following steps describe the process:

i. Video editing

1. Once OpenShot has been opened press the green plus (+) sign or select the File menu item followed by Import Files
2. Browse to the file location where the video output produced with Simple Screen Recorder is located
3. Select the file and press Open
4. Repeat steps 2, and 3 to import audio and image files
5. When the video that was captured is dragged to a track do the following:
6. Right-click the track
7. At the pop-up menu browse to Separate Audio > Single Clip (all channels)
8. Right-click audio track that was separated from the video
9. Select properties
10. Scroll down the Properties pane to Volume
11. Double-click the default value to the right of Volume and replace it with 0.0
12. Press Enter
13. As audio files are imported and dragged to a track, right-click the audio track. When the pop-up menu loads, select Display > Show Waveform (this step is optional but helps in the editing process)
14. Organize tracks in the following order starting at the top:
15. Subtitles track

16. Shape(s) track masking unwanted content

17. Video track

18. Narration audio track

19. Background music audio track

20. Complimentary sound effects audio track(s)

21. Once the video and audio tracks have been edited, aligned, and synchronized, press the large red button or select the File menu item then, Export Project > Export Video

22. Enter a file name in the File Name text box

23. Chose a file destination folder if the default destination must differ from the suggested path

24. It is recommended to leave all default settings under the Simple tab unchanged to avoid video and audio distortions or synchronization issues

25. Press Export Video to initiate the video output process

ii. Masking inactive or irrelevant SWF video content

1. Click the Title menu option and click title

2. Under Choose a Template select the TITLE (Camera border) template

3. Where it says Advanced, press Used Advanced Editor

4. Press the Ctrl+A keystrokes to select all then press the delete key to remove the invisible content on the template

5. To the left of the program there are several icons representing shapes. For instance, select the 7th icon from top to bottom to create a rectangle or simply press the F4 key. If a circle needs to be created it can be clicked or the F5 key can be pressed to do the same

6. Press F1 to activate the select tool and touch the shape before attempting to modify it

7. Select Fill and Stroke located in the right pane to apply and gradually change colors as needed

8. The shape's size and position may be altered by dragging the shape's arrows as needed. For greater precision, the shape can be manipulated by entering pixel values in the X, Y, W, and H text boxes at the top of the Inkscape program

9. Exit Inkscape then press Save when prompted

10. Upon returning to the Titles pop-up window, where it says File Name, replace the automatically generated name with a self-descriptive naming convention distinguishing the modified title template from a text-based title

11. Press Save to add the newly created shape

NOTE: Inkscape must be installed as a prerequisite before modifying a title

iii. OpenShot for Subtitles

OpenShot is not a subtitle editor however, it has been identified as the best tool for the task because it allows seamless text integration with the other imported media elements. The required steps are as follow:

1. Launch OpenShot by selecting Ubuntu's Show Applications icon and partially or completely typing OpenShot

2. Select the Title menu option then Title or press the Ctrl+T keystrokes

3. Once the Titles window opens, select the Bar 3 template or preferably, Camera border which is the template used for this project

4. Type a preferred file name related to the project at hand followed by a number as many instances of the same title template will be used to create a subtitle effect

5. Refer to step iv.1

6. In the Line 1 text box enter or copy/paste a portion of the translated text not to exceed 75 characters in length including spaces

7. Before saving make sure that the file name has a number increment at the end after the first or prior number has been used. For example, Subtitle1, Subtitle2, etc.

8. Press Save

9. Recommended: create new subtitle files with enough text to cover spoken words for timing purposes as the video progresses. Creating subtitles ahead of time can make it harder to align written text with the narration

10. To crate subsequent subtitles, the previously created subtitle may be used as a template to avoid performing steps 6.a through 6.e as follow:

11. Right-click the latest created subtitle

12. Change the file name's number to the next increment

13. Perform step 5.f and 5.h

NOTE: If performing a copy/paste function ensure it is done from a text editor and not from a word processing application in order to prevent hidden special characters from pasting into OpenShot's title function.

iv. Inkscape plug-in for OpenShot

The role of Inscape in this project is to personalize OpenShot titles to resemble subtitle letters in terms of shape, size, position and screen length. Inscape must be preinstalled. The following steps are required to align subtitles to the video edit in progress:

1. Upon completion of step iii.4 select the Use Advanced Editor option

2. If not configured by default, change the font size to 30pt, make sure under Font > Font family sans-serif is selected, ensure CSS Normal and Face Regular under Style are chosen, and Align left is selected

3. The textbox dimensions and position should be configured at the top menu as follow for best fit in relation to the total viewable area based on a 1920 x 1080 (16:9) resolution:
4. W: 1380.664
5. H: 40.312
6. X: 204.405
7. Y: 12.698
8. Upon completion of the above parameters Inscape may be closed after shaving the changes
9. Refer to step iii.6

NOTE: This step is only required once per project or if switching from subtitle style from another within the same project.

v. Synchronizing videos to match translated narration length

OpenShot makes it possible to freeze, slow down or accelerate frames. Pacing differences between languages are addressed by employing the effects previously listed. The following procedures can be performed to dissect a video multiple times without degrading it:

1. Create scenes to delay or accelerate them
2. Pause the video at a point when an alteration needs to be made
3. Ensuring that the vertical red line play progress indicator is at the desired time, pause the video and press the scissors icon to establish the start of a separate scene
4. Resume the video until the point when the scene matches what is narrated in the new language
5. Once more pause the video then click the scissors icon again to establish the end of the scene

6. Right-click the scene that was cut from the rest of the video and when the pop-up menu displays, browse to Time > Slow > Forward then press 1/2x. If the segment needs to be accelerated browse to Time > Fast > Forward then press 2x
7. Create scenes to freeze frames
8. Pause the video at a point when an alteration needs to be made
9. Ensuring that the vertical red line play progress indicator is at the desired time, pause the video and press the scissors icon to establish the start of a separate scene
10. Resume the video until the point when the scene matches what is narrated in the new language
11. Once more pause the video then click the scissors icon again to establish the end of the scene
12. Right-click the scene that was cut from the rest of the video and when the pop-up menu displays, browse to Time > Freeze then select the amount of seconds listed as appropriate.

APPENDIX E. PROCEDURES FOR REDUCING THE SIZE OF MP4 FILES CREATED WITH OPENSHOT

A. HANDBRAKE TO SHRINK VIDEOS FOR WEB STREAMING

HandBrake is used in this project to dramatically shrink the size of the MP4 output files produced by OpenShot. Reducing the file size of MP4 videos allows them to be streamed online. The following steps show the process:

1. After opening HandBrake click File > Open Source or click the Open Source icon located in the upper left of the program
2. Browse to the location where the MP4 file created with OpenShot was saved
3. Select the file and click Open
4. In the Summary tab ensure that MPEG-4 (avformat) and Align A/V Start are selected
5. In the Video tab verify or select the following parameters:
6. Video Encoder – H.264 (x264)
7. Framerate – 15
8. FR slide bar – 22
9. Constant quality – leave default value
10. All other values – leave as is
11. If not already displayed, change the file extension at the bottom where it says Save as: to .mp4. Type a different file name than the one suggested to avoid the original from being overwritten
12. On the lower right where it says To: click the drop down arrow and browse to the location where you wish the output file to be if not satisfied with the suggested location
13. Press the Start Encoding button at the top to start the process

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] B. Cone, C. Irvine, M. Thompson, T. Nguyen, “A Video Game for Cyber Security Awareness and Training,” *Computers and Security,* vol. 26, no. 1, pp. 63–72, February, 2007. [Online]. https://nps.edu/documents/107523844/117287768/FISSEA_CyberCIEGE_PreConf.pdf/312fed4e-b6cb-4a23-b7a0-a8faed038881?t=1534887969000

[2] K. Yurieff, “So Long, Flash: Adobe Will Kill Plug-in by 2020,” CNN Business, July 25, 2017. [Online]. Available: https://money.cnn.com/2017/07/25/technology/adobe-killing-flash/index.html

[3] E. Duffin, “Languages Spoken (at home) Other Than English in the United States by Number of Speakers in 2019,” Statista, September 21, 2020. [Online]. Available: https://www.statista.com/statistics/183483/ranking-of-languages-spoken-at-home-in-the-us-in-2008/#statisticContainer

[4] Statistical Atlas, “Languages in the United States,” September 4, 2018. [Online]. Available: https://statisticalatlas.com/United-States/Languages

[5] Naval Postgraduate School, “Center for Cybersecurity and Cyber Operations,” Accessed August 20, 2020. [Online]. Available: https://nps.edu/web/c3o/

[6] M. Thompson and C. Irvine, “Active Learning with the CyberCIEGE Video Game,” Naval Postgraduate School, Monterey, CA, USA, 2011 [Online]. Available: https://www.usenix.org/events/cset11/tech/final_files/Thompson.pdf

[7] Naval Postgraduate School, “CyberCIEGE,” Accessed: August 15, 2020. [Online]. Available: https://nps.edu/web/c3o/cyberciege

[8] M. Thompson and C. Irvine, “Active Learning with the CyberCIEGE Video Game,” Naval Postgraduate School, Monterey, CA, USA, 2011 [Online]. Available: https://www.usenix.org/events/cset11/tech/final_files/Thompson.pdf

[9] L. Noe-Bustamante, M. Lopez, and M. Krogstad, “U.S. Hispanic population surpassed 60 million in 2019, but growth has slowed,” *Pew Research Center*, July 7, 2020. [Online]. Available: https://www.pewresearch.org/fact-tank/2020/07/07/u-s-hispanic-population-surpassed-60-million-in-2019-but-growth-has-slowed/

[10] Council on Foreign Relations, “Demographics of the U.S. Military,” July 13, 2020. [Online]. Available: https://www.cfr.org/backgrounder/demographics-us-military

[11] Adobe Product Security Incident Response Team (PSIRT), “Update to Security Advisory for Adobe Reader, Acrobat and Flash Player,” June 7, 2010. [Online]. Available: https://blogs.adobe.com/psirt/?p=98

[12] National Institute of Standards and Technology, “CVE-2010-1297 Detail,” September 18, 2017. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2010-1297

[13] Adobe Blog, “Flash & the Future of Interactive Content,” July 25, 2017. [Online]. Available: https://blog.adobe.com/en/publish/2017/07/25/adobe-flash-update.html#gs.nkugdx

[14] B. Barrett, “Adobe Finally Kills Flash Dead,” Wired, July *25*, 2017. [Online]. Available: https://www.wired.com/story/adobe-finally-kills-flash-dead/

[15] S. Gopinath, “End of support from Microsoft in December 2020,” Microsoft, September 4, 2020. [Online]. Accessed: https://blogs.windows.com/msedgedev/2020/09/04/update-adobe-flash-end-support/

[16] Choose a License. “Licenses.” Accessed September 2, 2020. [Online]. Available: https://choosealicense.com/licenses/

[17] A. Wood, “HTML5 Basics for Everyone Tired of Reading About Deprecated Code,” HTML, October 31, 2020. [Online]. Available: Available: https://html.com/html5/

[18] H. Soffar, “MP4 File Format Uses, Features, Advantages and Disadvantages,” Online Sciences, May 12, 2017. [Online]. Available: https://www.online-sciences.com/technology/mp4-file-format-uses-features-advantages-and-disadvantages/

[19] J. Ozer, “What Is H.264?,” Streaming Media, April 4, 2011. [Online]. Available: https://www.streamingmedia.com/Articles/ReadArticle.aspx?ArticleID=74735

[20] R. Savoye, “Gnash Reference Manual,” GNU’s Not Unix, January 27, 2010. [Online]. Available: http://www.gnu.org/software/gnash/manual/gnashref.html

[21] J. Thomas, “Improved Title Editor!,” OpenShot Video Editor, July 2, 2009. [Online]. Available: https://www.openshot.org/blog/2009/07/02/improved-title-editor/

[22] OpenShot Video Editor, “OpenShot User Guide,” November 5, 2020. [Online]. Available: https://www.openshot.org/user-guide/

[23] Inkscape, “Overview.” Accessed December 23, 2020. [Online]. Available: https://inkscape.org/about/overview/

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California