SRS

1. Introduction

This section gives a scope description and overview of everything included in this SRS document. Also, the purpose for this document is described and a list of abbreviations and definitions is provided.

1.1 Purpose

The purpose of this document is to describe the requirements for the “StegSleuth” software. A detailed description of system constraints, interface, and interactions with other external applications will be presented. This writeup is done primarily for the clients convenience and for the them to approve or change what will be developed.

1.2 Scope

The StegSleuth software is a desktop application with the express purpose of assisting Capture the Flag competitors in steganography related puzzles and challenges. Common complaints around steganography software is that they do not contain enough steganographic or cryptographic algorithms to satisfy the amount of cases. Competitors will usually have to go through various steganography programs to find the one that uses the specific algorithm that was used on the carrier, and even still the payload might be encrypted with an algorithm that the software does not support.

The goal of this application is to compile as many steganographic and cryptographic algorithms as possible so that competitors may only have to rely on one application. Stegsleuth will run primarily on UNIX systems as it will require access to Terminal commands on the backend. It will be open source and free to download primarily by Polytechnic University of Puerto Rico students as well as anyone else that desires to compete in CTF competitions.

Following the needs of a steganography application, users will be able to extract hidden files (images, text documents, etc.) from another carrier file, which in this case will be images. This will be done through steganographic algorithms such as least significant bit (LSB), which is the replacement of binary data to match those of the hidden file without altering the carrier in a noticeable way. Since this data may be encrypted then cryptographic methods will also be included for decryption and encryption of documents and text. File can be manually analyzed through either a text editor for its ASCII data or through a hexdump for raw binary data. Finally, in order to verify file integrity, users will be able to compare files through their hashes in the application.

1.3 Definitions, acronyms, and abbreviations

|  |  |
| --- | --- |
| Term | Definition |
| Steganography | The practice of concealing a file or message within another file. Usually done by changing certain bits in the carrier file to match those of the payload. |
| Encryption/Cryptography | The process of obscuring information to make it undecipherable. |
| Payload | The file that will be hidden. |
| Carrier | The file that will hold the hidden file. |
| LSB | Least Significant Bit, the rightmost bit in a byte. |
| ASCII | American Standard Code for Information Internchange, character encoding standard for electronic communication. |
| UNIX | A multiuser, multitasking computer operating system. |
| Hexdump | Hexadecimal view of file data |
| Hexadecimal | Positional numeral system with a base of 16 |
| Binary | Positional numeral system with a base of 2 |
| GUI | Graphical User Interface |
| Payload | Data being hidden in carrier file |
| Hash or Message Digest | A value created from a file passing through a hashing formula. Used to verify file integrity. |
| Capture the Flag (CTF) | Computer Security Competition |

1.4 References

Iglesias, P. F. (2018, November 15). #MundoHacker: Esteganografía, el arte de ocultar información sensible. Retrieved April 17, 2019, from <https://www.pabloyglesias.com/mundohacker-esteganografia/>

Scott, A. (2018, September 01). Simple Image Steganography in Python. Retrieved March 22, 2019, from <https://hackernoon.com/simple-image-steganography-in-python-18c7b534854f>

PyQt5 Reference Guide¶. (n.d.). Retrieved April 7, 2019, from <https://www.riverbankcomputing.com/static/Docs/PyQt5/>

Welcome to PyCryptodome's documentation¶. (n.d.). Retrieved April 27, 2019, from <https://pycryptodome.readthedocs.io/en/latest/>

1.5 Overview

The remainder of this document includes three chapters. The second chapter is a definition of system functionality, constraints and assumptions for the final product. The third chapter focuses on the requirements for the application.

2.Overall Description

This section will give an overview of the application. The application will be explained on how it functions and how it interacts with other systems. At the end, constraints and assumptions will be listed and described.

2.1 Product Perspective

The product will be a desktop application with a frontend GUI. The primary function of the system is to allow the user to select a file as the carrier for a steganographic analysis be it for extraction or insertion of a payload. Many different types of algorithms exist for this process, with most of the available ones involving changing the least significant bits of the carrier to match those of the payload. The application will allow the user to select from various algorithms to achieve this.

As an extra measure of confidentiality people sometimes encrypt the payload before storing it in the carrier. If encrypted hidden information is stored in the carrier but is not decrypted, then what will be extracted will be an indecipherable mess. For this reason, the application will have the option to either encrypt the payload before being hidden in the carrier or decrypting it after being extracted. Since a large amount of cryptographic algorithms can be used for this, the application will allow the user to choose one from a selection and to insert a key with which to encrypt or decrypt.

In the event that the user would like to verify the integrity of a supposed carrier file against a similar file, the application would help in that regard. In order to achieve this, comparisons are done through hashing values. The user will select two files with the same extension and select a hashing algorithm from the various options (MD5, SHA, SHA265, etc.) which will get the message digest for both items. If both files have the same value, then they are truly equal. If both values are different then they are different.

Sometimes a payload could be something as simple as some text hidden in the data of the carrier. In order to verify this, two methods of viewing a text output will be done. One method is by showing the carriers text output in a regular text box. After receiving the text output, the user will be able to either scroll through the data or do a string search for something specific. The other method of verifying the data is through a hexdump. Through the hexdump, the user will be able to view both the hexadecimal data of the file and the ASCII values as well.

2.1.1 System Interfaces

The system is designed with contestants in mind, so speed and accessibility are the primary concerns. To achieve this, the amount of buttons needed to click and screens to go through are being kept to a bare minimum.

2.1.2 User Interfaces

The software will consist of a starting splash screen, followed by a file explorer for choosing the desired image carrier. Once the image is chosen, the user may decide if they will either be encoding or decoding payloads into the carrier. The payload may either be a file or text.

2.1.3 Hardware Interfaces

The users of the application will need access to a computer or a virtual machine running a Linux distribution, with the recommendation being the latest build of Ubuntu. They will need a keyboard in they decide to encode text, while the rest of the functions can be handled with the mouse. In the event that the user does not have a physical keyboard, they will need to use a virtual keyboard provided by the operating system. And in the event that the user does not have access to a mouse, they should be able to navigate throughout the application with tabs on the keyboard.

2.1.4 Software Interfaces

The software that will interact with the system are the following:

|  |  |
| --- | --- |
| Software | Description |
| Terminal Emulator | Interface that allows the application to run required functions as a background process. |
| PyQT GUI | A graphical user interface will be available. |

2.1.5 Memory Constraints

|  |  |  |
| --- | --- | --- |
| Item | Minimum Memory | Recommended Memory |
| Cryptographic Functions | 2GB | 4GB |
| Steganographic Functions | 2GB | 4GB |

2.1.6 Operations

There are two types of users that will be interfacing with the application, and they are the following:

* **CTF Contestants:** Will use the application to solve steganography puzzles during competitions. These require to be done quickly and have all available tools accessible on a short notice.
* **Cybersecurity Students:** Will use the application to learn about steganography and cryptographic concepts.

2.1.7 Site Adaptation Requirements

The application is intended to work in Linux distributions with an available packet manager to download required terminal commands in the case that they are not present by default.

2.2 Constraints

The application is mostly limited by the file types that the user will choose. Depending on the progress being made on the project, once image steganography is resolved in its entirety then other file types will be added progressively. Image steganography algorithms are highly available and simple to implement, so that is what it will start by. For the first version of the application, we are limiting ourselves to .jpg, .png, and .bmp file types.

Since the application will be developed with a mindset that the user’s primary goal is to participate in Capture the Flag competitions, this application is being developed for an audience who uses Linux distributions as their Operating System of choice for these events. As such, the application is being designed with the Python programming language for the Linux line of operating systems.

Terminal commands are a big part of this application, as functions such as the string searches and the cryptographic algorithms run through it. As such, making sure that these functions are installed in the users computer is important. In the event that the necessary commands are not present, a stable internet connection and user authorization must be present in order to install them.