Encryption in Steganography

Ash Olson
CS3100 X01
Utah Valley University
Orem, UT
10987957@uvu.edu

***Abstract*—This project is a programming application built with Python. It allows users to input images and hide messages in them, as well as extract them through a steganographic algorithm. It's a great platform for users to share cute photos with secret messages for friends.**

***Index Terms*—Python, Flask, Steganography, Encryption, Cryptograpy, etc.**

I. INTRODUCTION

Sometimes people have secrets that they don't want anyone to know. This could be anything from a secret crush to a government secret! If confessed in a text, anyone could glance at the phone of either user. This paper explores the implementation of one way to counteract this situation using fun methods, focusing on the advantages, disadvantages, and privacy implications of this approach.

II. TECHNOLOGY

This interactive encryption application leverages Flask to support encryption algorithms coupled with steganography as means to maintain user privacy. Flask, a module provided by Python, offers a streamlined and efficient way to develop simple web applications for those who prefer using Python for their server-side host.

III. ADVANTAGES

1. Enhanced Security: Putting aside all other factors, the main purpose of using this application is to protect user data from any unwelcome peepers. There are several layers of privacy used to protect these messages. First, there's the fact that the message is virtually impossible to extract from the encoded image without knowing the exact location of the hidden bits. Next is the fact that if anyone sees these messages, they would never think to check them for messages, especially considering that they likely don't know what steganography is. The image is also never uploaded to any sort of database and cleared on every new instance, preventing any possible exposure due to unauthorized access or breaches in security. Finally, if none of that is enough to convince someone of the security of this product, the actual "encryption" part isn't just hiding the message within the image. The message is first encrypted using a completely unique key created per image by referencing certain bits in the image and a tedious process to generate it.

2. Ease of Use: The encryption and decryption functions can be completed within moments of submitting user photos and messages, downloading the new image straight to their computer. This convenience enhances user experience and encourages higher user engagement.

3. User Experience: One of the reasons why the concept of steganography was chosen for this project is the allowance of user creativity, community engagement, as well as the suspense of waiting to see the reaction of your friend—whether fellow middle-school nerd or high clearance government employee—to the image, the message, and the novelty of what is for many unseen technology.

4. Simple Upkeep Procedures: Once the initial application and algorithm are developed and usable, there isn't much else to be done to keep the application working and progressing. After deploying to a public website, the anticipated future changes mainly consist of UI/UX improvements as well as increasing security.

IV. DISADVANTAGES

1. Privacy Concerns: There are three main privacy concerns that may come up as the boundaries of the application are pushed. The first is the security of the data handling as well as the security of the web app itself, including any insecurity in the web app connection, safe downloading uploading and downloading practices, and user anonymity. Next is if the encryption algorithm is exposed, in which case user data is at risk of exposure. The last is if unwanted lurkers get their hands onto your image and know about this application, they can steal your precious message and leave you none the wiser. Something to counteract that could include allowing a key phrase to be inserted for the intended users to exchange.

2. Credibility: It is very likely that when a user discovers this application, there may be some doubts as to the privacy of the overall service. Seeing as there would be little reputation for the reliability of this application, users may be reluctant to trust the application with more important messages, leaving them to just play around on the application or simply discard the idea.

3. Reliability: There are many ways for a nonsensical message to come out of an inputted image. If the image is changed at all, it changes the bits holding the key and message, giving no output. It has to be kept as a PNG so as not to lose any data, and it cannot work if compressed. A user may also input random images, possibly thinking that they're encoded, and receive a completely random message that they may take a bit too seriously.

V. PRIVACY IMPLICATIONS

Implementing Flask and any chosen deployment platform in this web app raises privacy considerations. As a developer, it is crucial to adhere to the principle of least privilege and only request the permissions necessary for the application's functionality. This begins by reading all of the fine print on any module implemented in order to protect user privacy.

This application must be transparent about its data collection, usage, and storage practices. Clear privacy policies should be provided, explaining how user data is handled, who has access to it, and how long it is retained. Compliance with relevant privacy laws and regulations, such as the General Data Protection Regulation (GDPR) or the California Consumer Privacy Act (CCPA), is essential to build user trust and avoid legal repercussions.

VI. CONCLUSION

Encryption coupled with steganography offers a creative, engaging, and secure way to ensure the data and privacy of users within this web application. By protecting data in an unsuspected location, users can benefit from its ease of use and enhanced security. However, it is crucial to consider the privacy implications and potential drawbacks with any unknown factors within each level of protection.

To mitigate privacy concerns, the Application must prioritize transparency, adhere to privacy best practices, and comply with relevant regulations. By striking a balance between user convenience and data protection, the application can provide a seamless and secure user experience while respecting user privacy.

As web development continues to evolve, it is crucial to understand the changing advantages, disadvantages, and privacy implications, so that developers can make informed decisions when implementing their applications, ensuring a secure and user-friendly experience for their users.

VII. FEATURES

The application has the following features:

- User Experience: Implementation encourages user interaction with an engaging experience that encourages others to join the party!
- Ease of Use: Speedy algorithms allow for an all encompassing experience within moments!
- Enhanced Security: Layer upon layer of data protection relieves those who've been looking their whole life for real privacy!

VIII. INSTALLATION

To install the application, clone the repository and navigate into the directory. Required: Python 3.8 or higher, Flask 3.0 or higher (and Flask dependencies), and any modern web browser (To skip this lengthy installation process, you may just type "pip install flask" into your command line interface and all necessary dependencies will be included.) Then, run the application using the command: "python server.py" and go to the address returned.

IX. USAGE

After starting the application, you may choose an image to input and a message to hide to get an encrypted image, as well as input an encrypted image and receive the message.

X. COMPONENTS

The application consists of several components:

- Encryption: This component has an input space for each an image and a message, and when you press the "Encrypt" button it downloads the new image.
- Decryption: This component has an input space for an image as well as an output field to display the original message when you press the "Decrypt" button.

XI. LICENSE

The project is licensed under the MIT license.