SEMESTER PROJECT REPORT

IMAGE ENCRYPTION FOR SECURE FILE TRANSFER

Title: Parallel and Distributed Computing Course

Code: CEN 302:

Class: BS AI 05 (Spring 2024)

Course Instructor: Sadia Nazim

GROUP MEMBERS

NAMES	ENROLLEMENTS
SAIM CHISHTI	01-136221-045
TAHA HASNAT	01-136221-018

Table of Contents

1. Introduction

In the realm of secure data transfer, image encryption and decryption play a pivotal role. This report critically examines the process of image encryption and decryption using noise-based techniques. Utilizing tools such as Visual Studio 2022, OpenCV, and Qt, this project ensures secure image handling through encryption and subsequent decryption. The report delves into each stage of the process, providing a comprehensive understanding of the methodologies and their impact.

2. Loading the Image

2.1 Overview

Loading the image is the first step in the encryption process, where the image is read from a specified path and prepared for processing.

2.2 Key Contributions

1. **Image Preparation:** Ensures the image is correctly loaded into a matrix format suitable for processing.
2. **Foundation for Encryption:** Provides the base data required for subsequent encryption steps.

2.3 Methodology

The image is loaded using OpenCV's imread function, which reads the image file and stores it in a matrix. This matrix representation allows for pixel-level manipulation essential for encryption.

2.4 Insights

- **Reliability:** OpenCV's robust image handling capabilities ensure accurate image loading.
- **Compatibility:** The matrix format is compatible with a wide range of image processing operations.

3. Generating Noise

3.1 Overview

Noise generation involves creating a random noise matrix that will be used to encrypt the image through an XOR operation.

3.2 Key Contributions

1. **Reproducibility:** The use of a seed for the random number generator ensures that the noise can be reproduced for decryption.
2. **Security:** The random noise adds a layer of security to the encryption process.

3.3 Methodology

A noise matrix is created with the same dimensions as the image. Using a seeded random number generator, random noise values between 0 and 255 are generated for each pixel and each color channel.

3.4 Insights

- **Controlled Randomness:** Seeding ensures that the same noise can be generated again, which is crucial for the decryption process.
- **Security Enhancement:** Random noise makes the encrypted image difficult to interpret without the corresponding noise matrix.

4. Encrypting the Image

4.1 Overview

Image encryption is achieved by performing an XOR operation between each pixel's color channel value and the corresponding noise value.

4.2 Key Contributions

1. **Simplicity:** The XOR operation is straightforward yet effective for encryption.
2. **Reversibility:** XOR is a reversible operation, making decryption possible using the same noise values.

4.3 Methodology

Each pixel's color channel in the image is XORed with the corresponding value in the noise matrix, resulting in an encrypted image that appears visually different and contains noise.

4.4 Insights

- **Efficiency:** XOR operations are computationally efficient, ensuring quick encryption.
- **Robustness:** The encryption process is robust against simple attacks due to the added noise.

5. Saving the Encrypted Image and Noise Key

5.1 Overview

The encrypted image and the noise matrix (key) are saved to disk to facilitate secure storage and transfer.

5.2 Key Contributions

1. **Persistence:** Ensures that the encrypted image and noise key can be stored and retrieved later.
2. **Security:** The noise key is crucial for the decryption process, ensuring that only authorized users can decrypt the image.

5.3 Methodology

The encrypted image is saved using OpenCV's **imwrite** function, while the noise matrix is stored in a file using OpenCV's **FileStorage**.

5.4 Insights

- **Data Integrity:** Proper storage methods ensure the integrity of the encrypted image and noise key.
- **Accessibility:** The saved files can be easily accessed and used for decryption when needed.

6. Decrypting the Image

6.1 Overview

Decryption involves loading the encrypted image and noise matrix and applying the XOR operation again to retrieve the original image.

6.2 Key Contributions

1. **Reversibility:** Demonstrates the reversibility of the XOR operation used in encryption.
2. **Data Recovery:** Successfully recovers the original image using the noise key.

6.3 Methodology

The encrypted image and noise matrix are loaded from disk. The XOR operation is applied again between the encrypted image and the noise matrix to restore the original image.

6.4 Insights

- **Accuracy:** The decryption process accurately restores the original image when the correct noise key is used.
- **Dependability:** The process is dependable, ensuring that the original image can be consistently retrieved.

7. Front End Integration with Qt

7.1 Overview

Qt is used to create a graphical user interface (GUI) that allows users to load images, encrypt and decrypt them, and save the results.

7.2 Key Contributions

1. **User Accessibility:** Provides an intuitive interface for users to interact with the encryption and decryption system.
2. **Functionality:** Integrates all steps of the encryption and decryption process into a cohesive application.

7.3 Methodology

The GUI is developed using Qt, enabling users to perform tasks such as loading images, initiating encryption/decryption, and saving the results through a user-friendly interface.

7.4 Insights

- **Usability:** The Qt interface enhances the usability of the system, making it accessible to non-technical users.
- **Integration:** Seamlessly integrates the backend encryption and decryption processes with a front-end application.

8. Conclusion

The image encryption and decryption system developed in this project demonstrates a secure and efficient method for handling images. By using noise-based encryption and leveraging tools such as Visual Studio 2022, OpenCV, and Qt, the project ensures that images can be securely encrypted, transferred, and decrypted. The methodologies applied provide a robust framework for secure image handling, with significant implications for secure communications and data protection.

9. References

https://www.researchgate.net/publication/370434831_Compressed_Image_Encryption_for_Secure_Internet_Transfer