Data Security Using Images-StegTool

Summer Internship Report

Submitted by

Anesh Parvatha
N090977

Jyothiram Pekala
N090990

Anil Kumar
N090537

in partial fulfillment of Summer Internship for the award of the degree of

Bachelor of Technology

IN

Computer Science and Engineering
RGUKT Nuzvid Campus

RGUKT Nuzvid Campus
Rajiv Gandhi University of Knowledge Technologies
Nuzvid, Krishna District, Andhra Pradesh - 521202

August 15, 2014

Summer Internship 2014
Project Approval Certificate

Department of Computer Science and Engineering

CMC LIMITED

The project entitled "Data Security Using Images" submitted by Mr. Anesh Parvatha, Mr.Jyothiram Pekala , Mr. AnilKumar Karri is approved for Summer Internship 2014 program from May 2014 to July 2014, at CMC LIMITED, Hyderabad.

Prof Mr. Imteyaz Hussain
Project In-charge

Place: CMC LIMITED, Hyderabad
Date: August 15, 2014

Declaration

We declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

_____

Mr. Anesh Parvatha
RGUKT - Nuzvid

_____

Mr. Jyothiram Pekala
RGUKT - Nuzvid

_____

Mr. Anilkumar Karri
RGUKT - Nuzvid

Date: _____

Acknowledgement

We the summer intern team of development on Data Security Using Images, a tool for message security, are overwhelmed in all humbleness and gratefulness to acknowledge our depth to all those who have helped us to put our ideas and assigned work, well above the level of simplicity and into something concrete.

We would like to express our special thanks of gratitude to **Mr.Imteyaz Hussain**, In-Charge of our project, for providing us an opportunity to work under one of his esteemed projects, constantly motivating for doing better and showing complete confidence in our work.With the help of his valuable suggestions, guidance and encouragement, we all were able to complete our tasks properly and with satisfaction.

We are also very thankful to our Guide **Mr.Vikas** for their valuable guidance. Their every word seemed as a precious advice to us. Due to sharing of their experiences of real world situations, we were able to manage the project smoothly. Because of their support and enthusiasm we used to get inspiration to do work. Also in the process we were able to learn other technical and non-technical things from them and we consider ourselves to be very fortunate to have them as our Project Guides.

We would like to thank all administrative people for making our stay here in CMCLimited,Hyderabad a pleasant memory and for all other administrative help.
Finally we also like to thank all other fellow internmates working in different projects for helping us at small problems as well as critical junctures.

Abstract

The objectives of data security are to prevent eavesdropping to get access of data and in case is stolen, make it difficult to understand the stolen data. These objectives are met through different approaches of data security. One of the approach is Stegnography.
Stegnographic techniques are useful to convey hidden information by using various types of typically transmitted text data as cover for concealed communication. Stegnography is a data-hiding/hidden-communication method that uses digital imagery as a cover signal.

The present project deals with design and Implementation of Data Security over corporate network using LSB(Least Significant Bit) algorithm for image encryption with stegnography. The image encryption is incorporated into the compression algorithm for transmission over a data network.

Cryptography is a method of mathematical encoding used to transform messages into an unreadable format in and effort to maintain confidentiality of data.

In corporate network of clients/server need to interact with each other and parallel maintain the confidentially of data. Before transmission of the encrypts it using the LSB algorithm and is then sent on the network. The receiver before reading the file needs to decrypt it using the same algorithm to get the actual data. The keys needed to encrypt/decrypt data are pre-known to specific sender and receiver.

Contents

Chapter 1

Software Requirement Specification

1.1 Introduction

This document includes overall Design, Software Requirements and Features of "Data Security Using Images". We made exact and accurate list of Software Requirements to develop this tool by following the methodologies of Software Engineering.This time we followed one of the Iterative Watterfall Model for the Development of this project.

1.2 Document Purpose

The purpose of this document is to present a detailed description of the "Data Security Using Images". It will explain the purpose and features of the system and what the system will do and also explain how the various modules work and how they communicate with each other for the successful working of the application.

1.3 Product Scope

The development of the project Data Security Using Images is based on stegnographic features to the IMAGE file and text file data and also provides encryption /decryption features.We named this is as Stegtool. Stegtool is a Component suite that can be used as a component in any application to provide the security to the text file. Stegtool provides several functions.
It will take the TEXT file,IMAGE file and secret key as input and gives the Encrypted Image file as output based on the LSB(Least Significant Bit) algorithm.
It will take the Encrypted Image and key as input and will give decrypted TEXT file.

- Easy User Interface
- High data Security
- Encrypt and Decrypt Algorithm

1.4 Definitions, Acronyms And Abbreviations

Term	Definition
Unified Modeling Language(UML)	Programming Language used for object-oriented software development.
UML Diagram	It is a partial graphical(view) representation of a model of a system under design, implementation, or already existence.
Cryptography	The science of analyzing and deciphering codes and ciphers and cryptograms.
Stegnography	Steganography is a method of hiding large quantities of information to be hidden inside a file, while making no perceivable changes to that file's contents.
Pixel	A pixel is the fundamental unit of digital imagery. It is the smallest point whose color can be controlled.
BIT-Planes	Each color channel R, G, and B in the RGB colorspace is represented by a number, and this number is represented by a number of bits.
LSB(Least Significant Bit)	While it is possible to embed data into an image on any bit-plane, LSB embedding is performed on the least significant bit(s).
Cease Cipher	It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet.
XOR encryption-Decryption	A string of text can be encrypted by applying the bitwise XOR operator to every character using a given key. To decrypt the output, merely reapplying the XOR function with the key will remove the cipher.
Java Swing	The thing that makes it easy to code Is the thing that makes It hard to control is the layout Manager. Layout Manager objects control the size and location of the widgets in a Java GUI.

Table 1.1: Definitions,Acronyms and Abbreviations

Chapter 2

Overall Description

2.1 Project Perspective

The Project main aim is to prevent eavesdropping to get access of data and in case is stolen, make it difficult to understand the stolen data.Establish Data Security over corporate network using LSB(Least Significant Bit) algorithm for image encryption with stegnography. The image encryption is incorporated into the compression algorithm for transmission over a data network.

2.2 Product Functionality

- Embed:
- Encrypt the data in the message with the help of XOR-Encryption technique by using key.
- Embed data into Image by using LSB(Lease Significant Bit) algorithm.
- Extract:
- Extract the data in the Image pixels.
- Decrypt the data from the text with the help of XOR-Encryption technique by using key.

2.3 Users And Characteristics

User:

Embed: User should select the image in which data want to be written then select the message text file then select image formate then choose the directory to output image and then give the password to protect it.

Extract: User have to select the image then give the protected password and then choose the directory to which output text file has to create.

2.4 Operating Environment

As we are designed it using java as of course java is portable language, It will be work in any platform that which contains java software. This is highly portable tool which is platform independent.

2.5 Design and Implementation Constraints

- No specific constraint for Operating System. But System must contain java.

2.6 User Documentation

A detailed user manual and possibly in-line help will be provided with in the application itself.

2.7 Assumption And Dependencies

- The development phase should not exceed the deadline.
- The product must be reliable,portable & paltform independent.
- The Application must be user friendly with clean and easy to use front-end.
- From the very start of this project we are aware of time constraints so the main emphasis is on extensibility and parallel development. We shall try our best to ensure that project deadlines are met.

Chapter 3

Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interface

Application Front-End must be user-friendly. The user interface shall be designed using various components available in Java Swing such as popups and well designed buttons, dropdowns,alerts.

3.1.2 Software Interface

This Application is a paltform independent and easy to configure and deployable. Any Operating system with Java support is enough to run this tool.

3.2 Functional Requirements

Functional requirements describe the interactions between the system and its environment independent of its implementation. The environment includes the user and any other external system with which the system interacts. The following are the functional requirements for the proposed system.

- The system takes text file,image fiel and secret key as input to the system.
- After that it will generate encrypted text by using Encrypted algorithm.
- The encrypted data will be embed into image file by using Hiding Algorithms.
- The hidden data is decoded in order to obtain final encrypted data, which has to be represented in text format.

The above-mentioned requirements show only the possible interactions between the system and its external world. The above description does not focus on any of the implementation details.

Chapter 4

Non Functional Requirement

Non-Functional requirements describe the user visible aspects of the system that are not directly related with the functional behavior of the system. Non-functional requirements include quantitative constraints, such as response time or accuracy.

- The user has to supply the key,file,image to the algorithms.
- **User interface and Human factor:** The user need not be technically sound, as these interfaces are explanatory.
- **Documentation:** User manual is provided to the user for using the system.
- **Hardware Considerations:** This is the Minimum hardware to run the corresponding this software. The system will not interact with other hardware system.
- **Performance Characteristics:** The operations done in this system is according to the algorithm. Here the system just executes those operations . So, here the performance will depend on the efficiency of the algorithm used.
- **Error Handling and Extreme Conditions:** There may be cases where the user not providing sufficient input parameters for the method he selected. In those cases exception should be raised.

Chapter 5

Design Document and Implementation

5.1 Resource Requirements

5.1.1 HW Requirements

The following hardware configuration is required for this project:

- Minimum RAM Required: 64 MB
- Minimum Capacity Required:
- Internal: 4 MB
- External: 2 to 32 GB

5.1.2 SW Requirements

Following software are required for this project:

- Operating system : Windows , Linux
- Software : JDK 1.6

5.2 Software Development Life Cycle Model

The systems development life cycle(SDLC), or software development life cycle in systems engineering, information systems and software engineering, is a process of creating or altering information systems, and the models and methodologies that people use to develop these systems.

In software engineering the SDLC concept underpins many kinds of software development methodologies. These methodologies form the framework for planning and controlling the creation of an information system

A software development process is a structure imposed on the development of a software product. Similar terms include software life cycle and software process. There are several models for such processes, each describing approaches to a variety of tasks or activities that take place during the process. Some people consider life cycle model a general term and software life cycle development a specific term.

Iterative and Incremental development is at the heart of a cyclic software development process developed in response to the weaknesses of the waterfall model. It starts with an initial planning and ends with deployment with the cyclic interactions in between. Incremental development slices the system functionality into increments(portions). In each increment, a slice of functionality is delivered through cross-discipline work, from the requirements to the deployment. The unified process groups increments/iterations into phases: inception, elaboration, construction, and transition.

- Inception identifies project scope, risks, and requirements(functional and non-functional) at a high level but in enough detail that work can be estimated.
- Elaboration delivers a working architecture that mitigates the top risks and fulfills the non-functional requirements.
- Construction incrementally fills-in the architecture with production-ready code produced from analysis, design, implementation, and testing of the functional requirements.
- Transition delivers the system into the production operating environment. Each of the phases may be divided into 1 or more iterations, which are usually time-boxed rather than feature-boxed. Architects and analysts work one iteration ahead of developers and testers to keep their work-product backlog full.

The system has been developed using **Iterative Waterfall Model**.

This model is most often used in the following scenarios:

- Requirements of the complete system are clearly defined and understood.
- Major requirements must be defined; however, some functionalities or requested enhancements may evolve with time.
- There is a time to the market constraint.
- A new technology is being used and is being learnt by the development team while working on the project.
- Resources with needed skill set are not available and are planned to be used on contract basis for specific iterations.
- There are some high risk features and goals which may change in the future.

The waterfall model is a sequential design process, often used in software development processes, in which progress is seen as flowing steadily downwards(like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance.

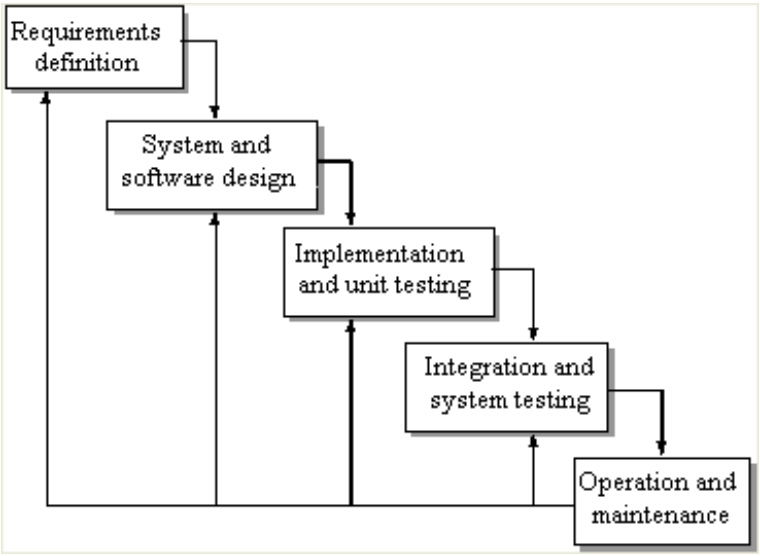

Figure 5.1: Iterative Waterfall Model

Advantages

- Simple goal. Simple to understand and use.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.
- Easy to manage. Each phase has specific deliverable and a review.
- Works well for projects where requirements are well understood.
- Works well when quality is more important than cost/schedule.
- Customers/End users already know about it.

Disadvantages

- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- No working software is produced until late in the life cycle.
- Risk and uncertainty is high with this process model.
- Adjusting scope during the life cycle can end a project
- Not suitable for complex projects
- Not suitable for projects of long duration because in long running projects requirements are likely to change.

- Integration is done as a "big-bang" at the very end, which doesn't allow to identify any technological or business bottleneck or challenges early.
- Attempt to go back 2 or more phases is very costly.
- Percentage completion of functionality could not be determined in mid of the project because each functionality is undergoing some phase.
- Very risky, since one process can start before finishing the other.

5.3 High Level Design Document

5.3.1 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system.
A use-case diagram can help provide a higher-level view of the system. It has been said before that "Use case diagrams are the blueprints for your system". They provide the simplified and graphical representation of what the system must actually do.

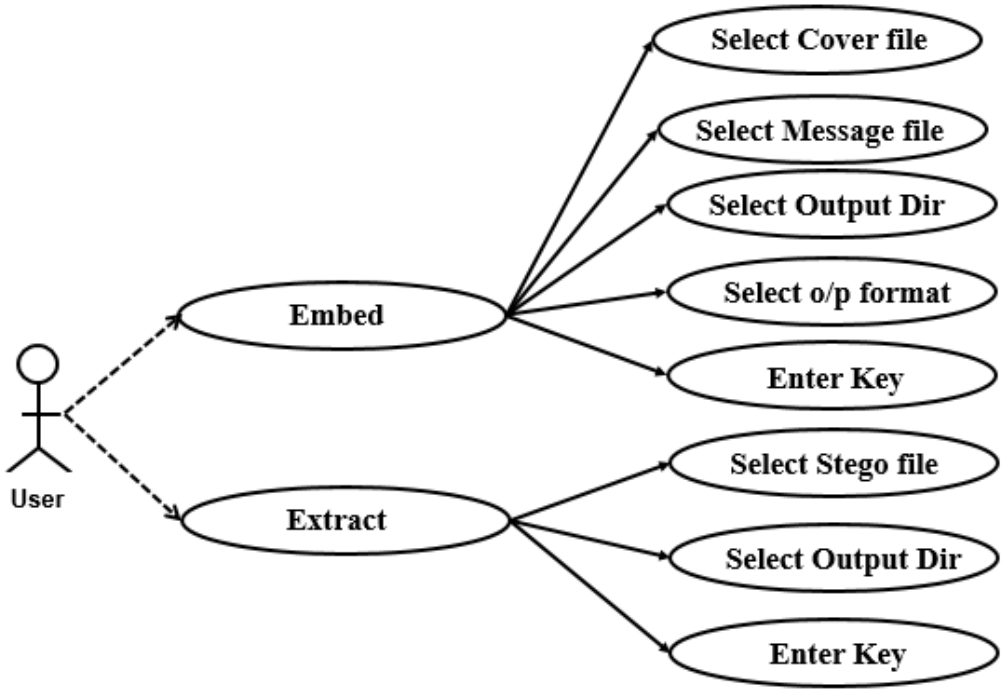

Figure 5.2: Use Case Diagram

5.3.2 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organisational processes (i.e. workflows).Activity diagrams show the overall flow of control.

Activity diagrams are constructed from a limited number of shapes, connected with arrows.The most important shape types:

- rounded rectangles represent actions;
- diamonds represent decisions;
- bars represent the start (split) or end (join) of concurrent activities;
- a black circle represents the start (initial state) of the workflow;
- an encircled black circle represents the end (final state).

Arrows run from the start towards the end and represent the order in which activities happen.

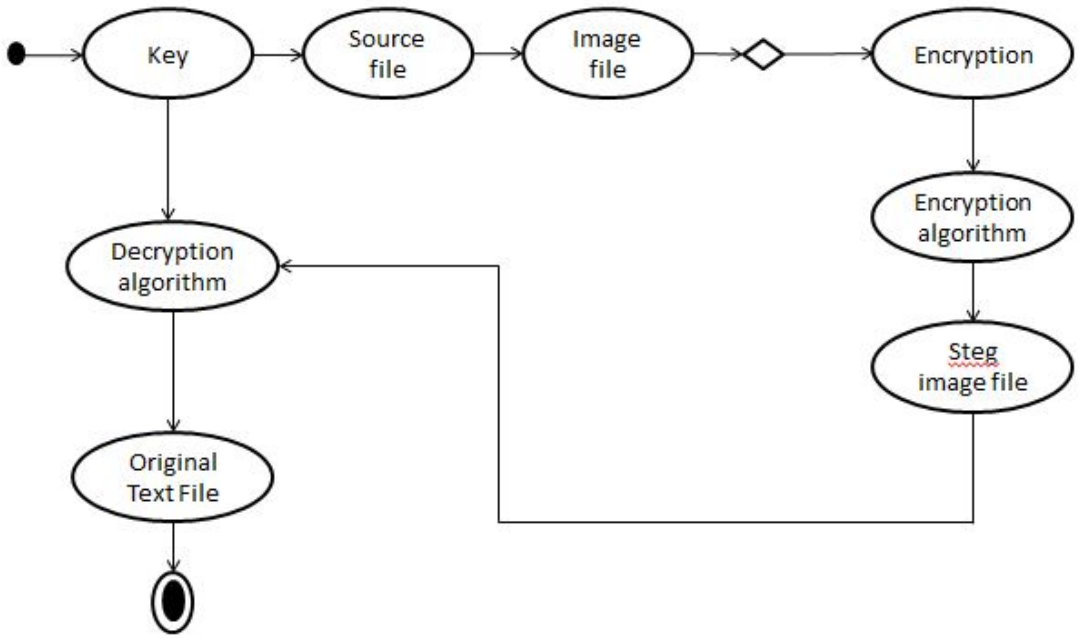

Figure 5.3: Activity Diagram

5.3.3 Class Diagram:

a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.
In the diagram, classes are represented with boxes which contain three parts:

- The top part contains the name of the class. It is printed in Bold, centered and the first letter capitalized.
- The middle part contains the attributes of the class. They are left aligned and the first letter is lower case.
- The bottom part gives the methods or operations the class can take or undertake. They are also left aligned and the first letter is lower case.

In the design of a system, a number of classes are identified and grouped together in a class diagram which helps to determine the static relations between those objects. With detailed modelling, the classes of the conceptual design are often split into a number of subclasses.

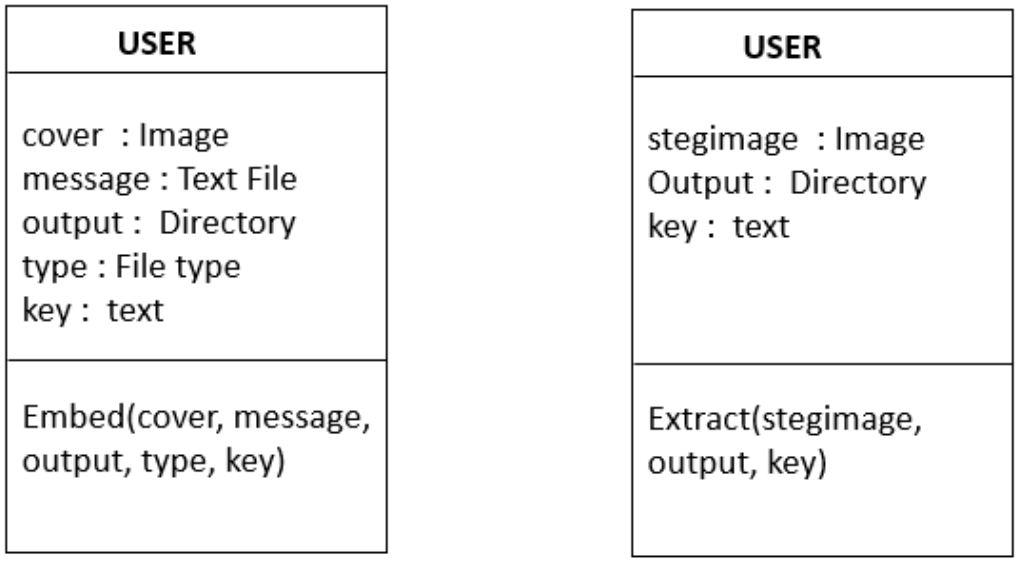

Figure 5.4: Class Diagram

5.3.4 Sequence Diagram:

A 'Sequence diagram' is an interaction diagram that shows how processes operate with one another and in what order. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios.

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

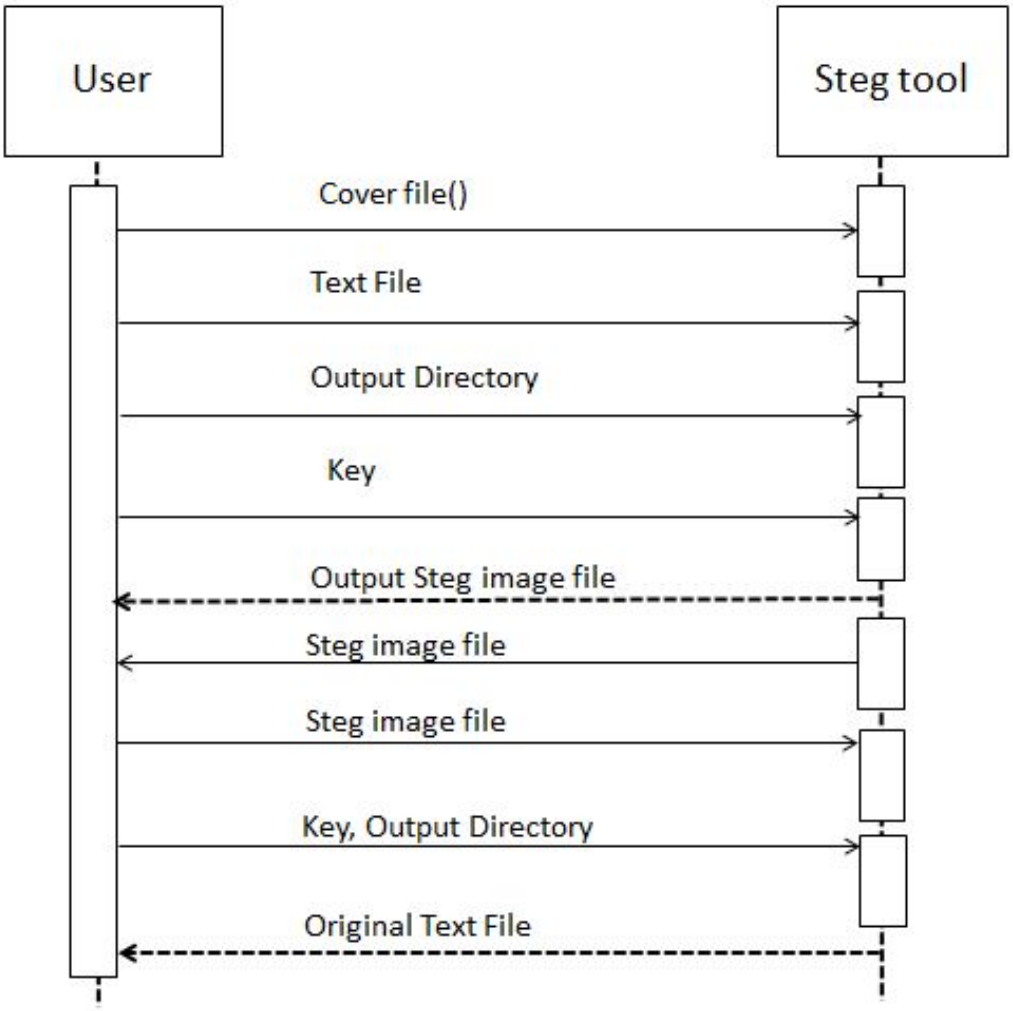

Figure 5.5: Sequence Diagram

Chapter 6

Testing

Software testing is an investigation conducted to provide information about the quality of the product or service under test. Software testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks at implementation of the software. Test technique includes the process of executing a program or application with the intent of finding software bugs. Software testing can be stated as the process of verifying and validating that a software program/application/product:

- Meet the requirements that guide its design and development
- Work as expected
- Can be deployed with the same characteristics.

Software testing can be implemented at any time depending on the testing method employed in the development process. However, most of the test effort traditionally occurs after the requirements have been defined and the coding process has been completed. It is observed that fixing a bug is less expensive if found earlier in the development process. Although in the agile approaches most of the test effort is, conversely, on-going. As such, the methodology of the test is governed by the software development methodology adopted.

6.1 Testing technique used

Several testing techniques have been performed on the system such as unit testing, integration testing and system testing.

6.1.1 Unit Testing:

Unit testing, also known as component testing refers to tests that verify the functionality of a specific section of code, usually at the function level. Unit testing is a software development process that involves synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time, and costs.

6.1.2 Integration Testing

The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Integration testing works to expose defects in the interfaces and interaction between integrated components (modules).

Black-box testing- It is a method of software testing that examines the functionality of an application(e.g. what the software does) without peering into its internal structures or workings.

White-box testing - It is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. In this testing, we performed Boundary Value Analysis.

6.1.3 System Testing

System testing tests a completely integrated system to verify that it meets its requirements. In addition, the software testing should ensure that the program, as well as working as expected, does not also destroy or partially corrupt its operating environment or cause other processes within that environment to become inoperative.

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. In this testing we performed Alpha testing.

Chapter 7

User Manual - StegTool

7.1 Introduction

StegTool v0.1 is a Java Application that hides text into an image and then extract the data form that image using hiding algorithms.

7.2 Flow of StegTool

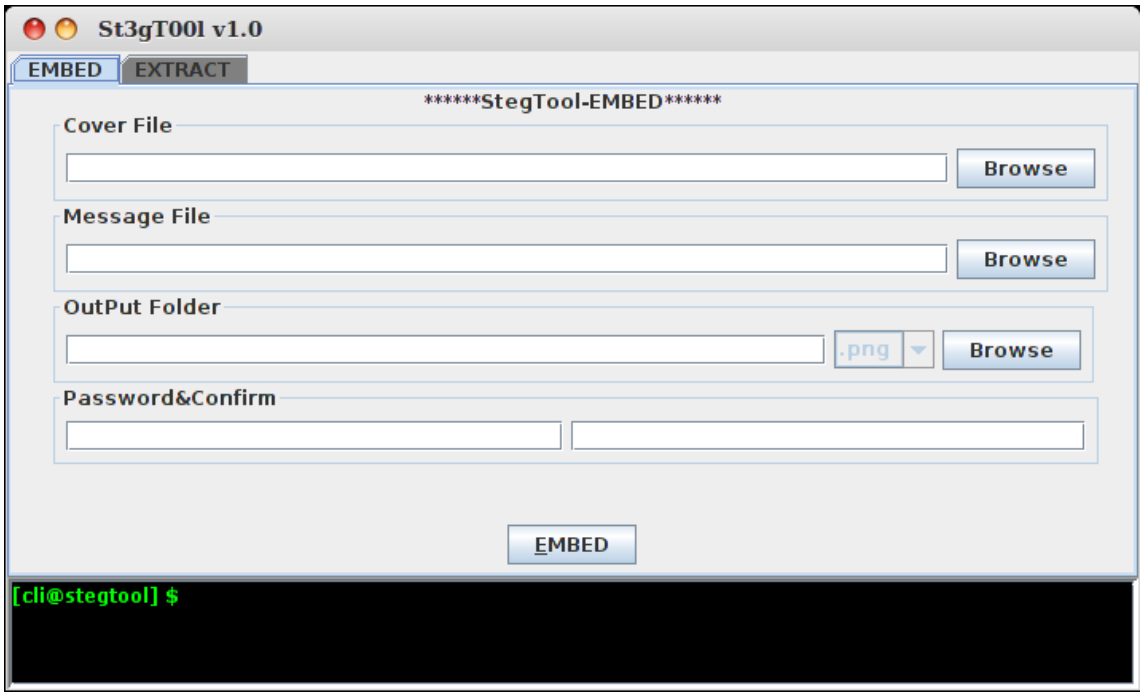

Figure 7.1: StegTool Front end

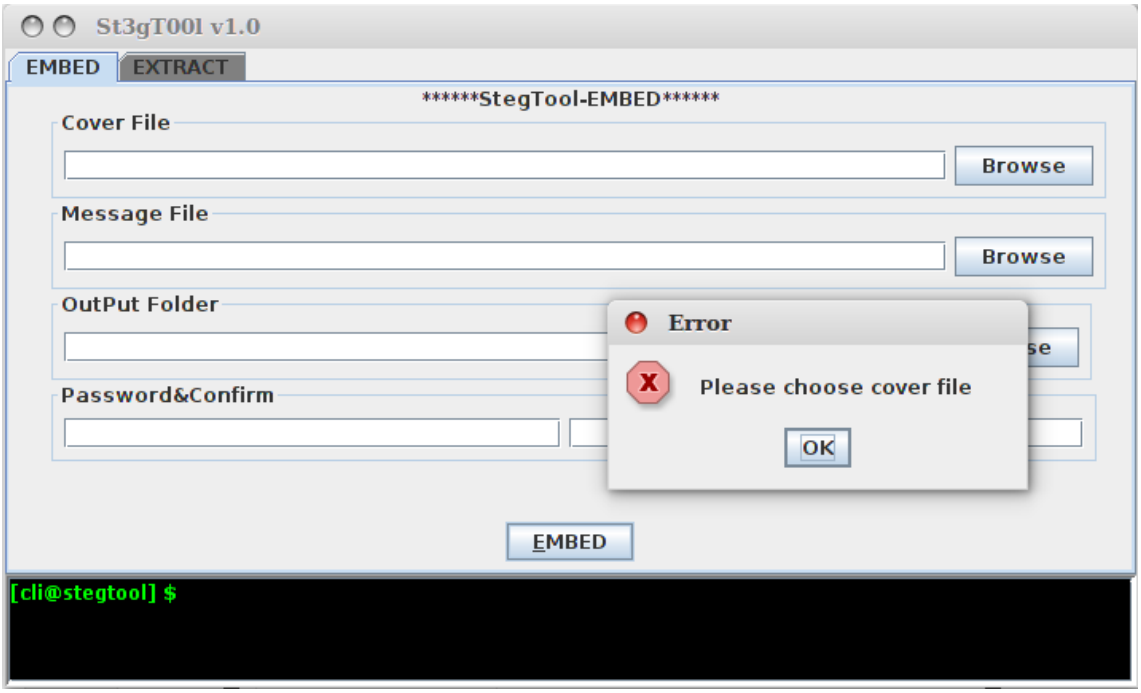

Figure 7.2: Error to select cover file embed

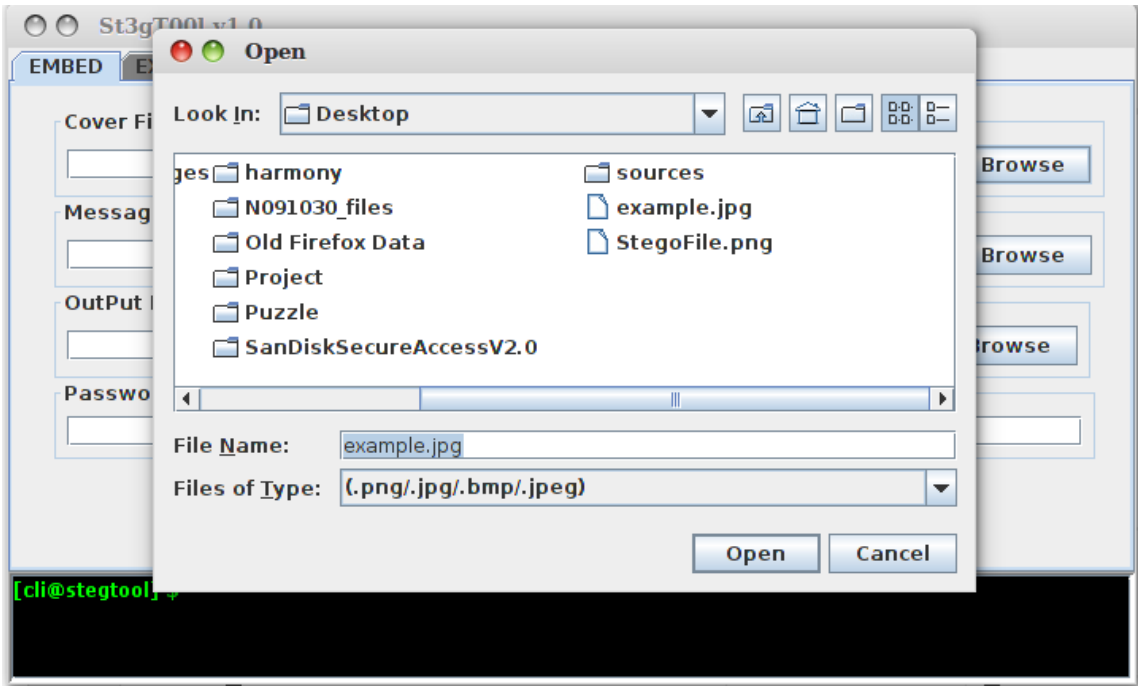

Figure 7.3: Selecting cover file to EMBED

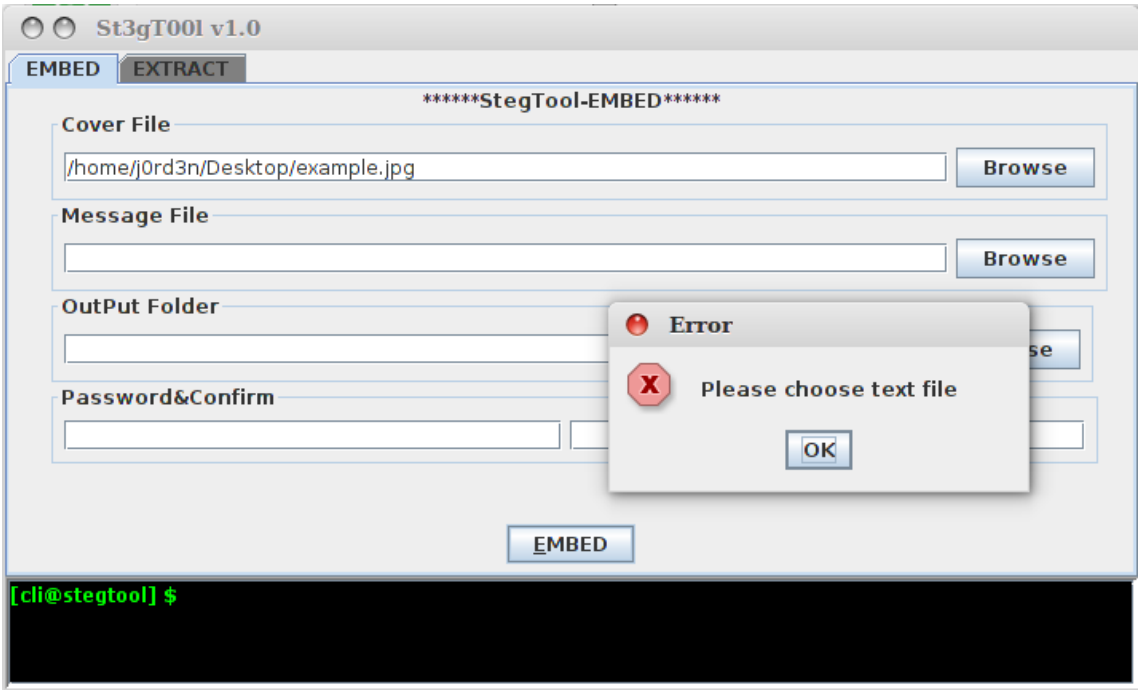

Figure 7.4: Alert to select message file[EMBED]

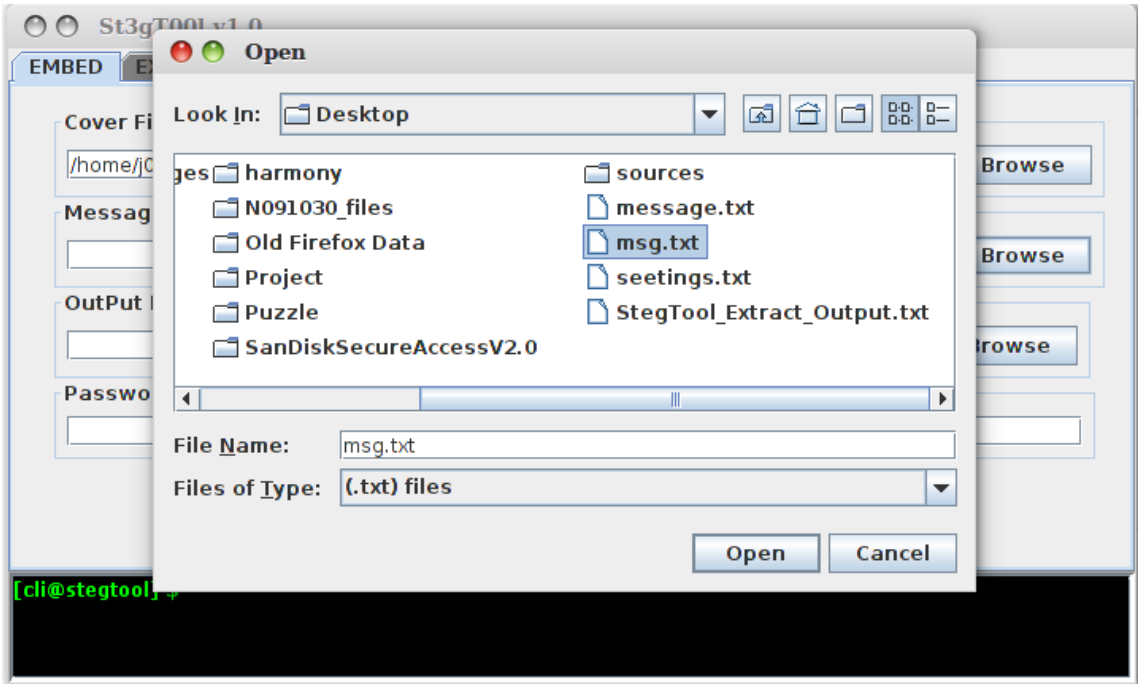

Figure 7.5: Selecting message file[EMBED]

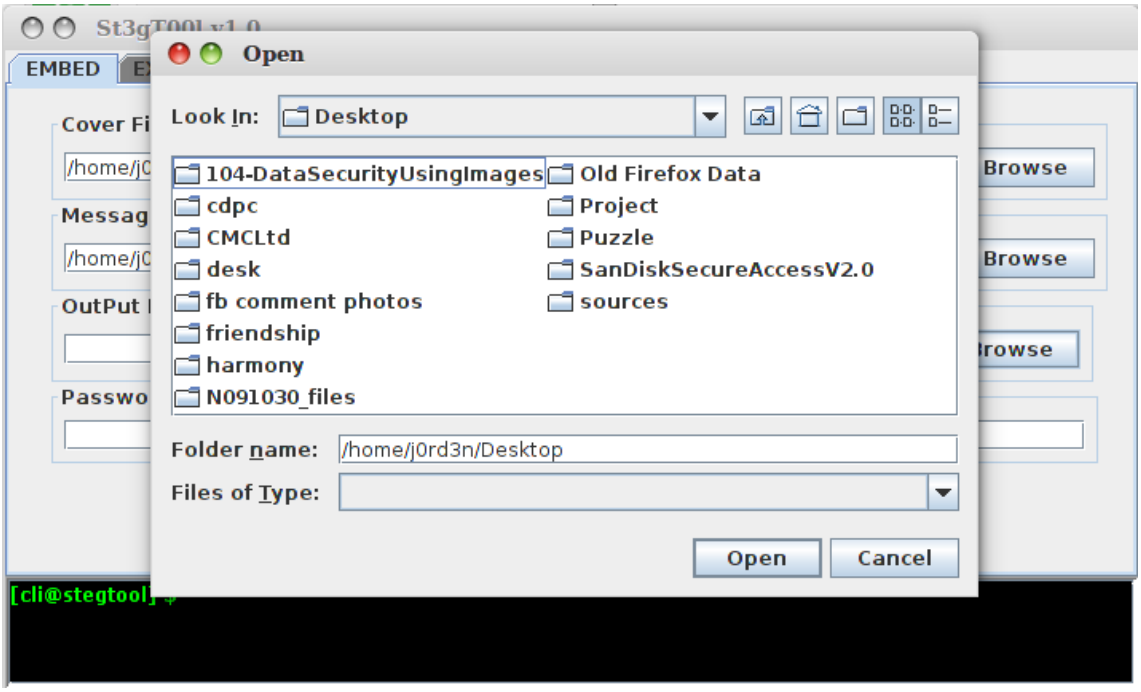

Figure 7.6: Selecting Output Directory[EMBED]

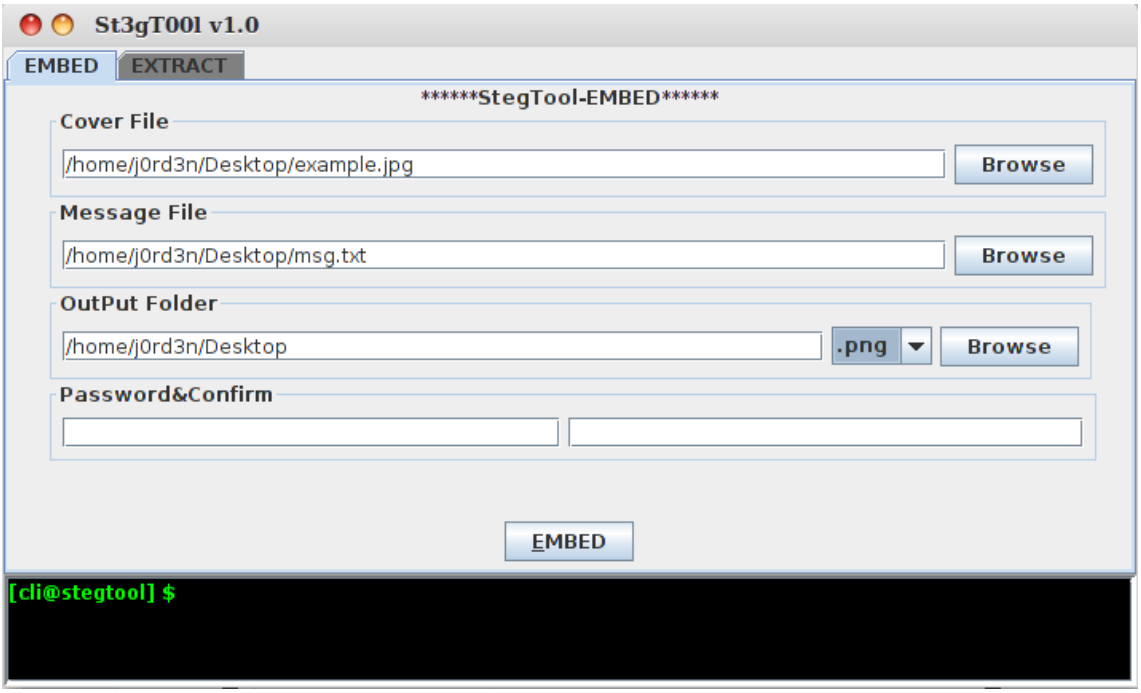

Figure 7.7: Selecting Output Image Format[EMBED]

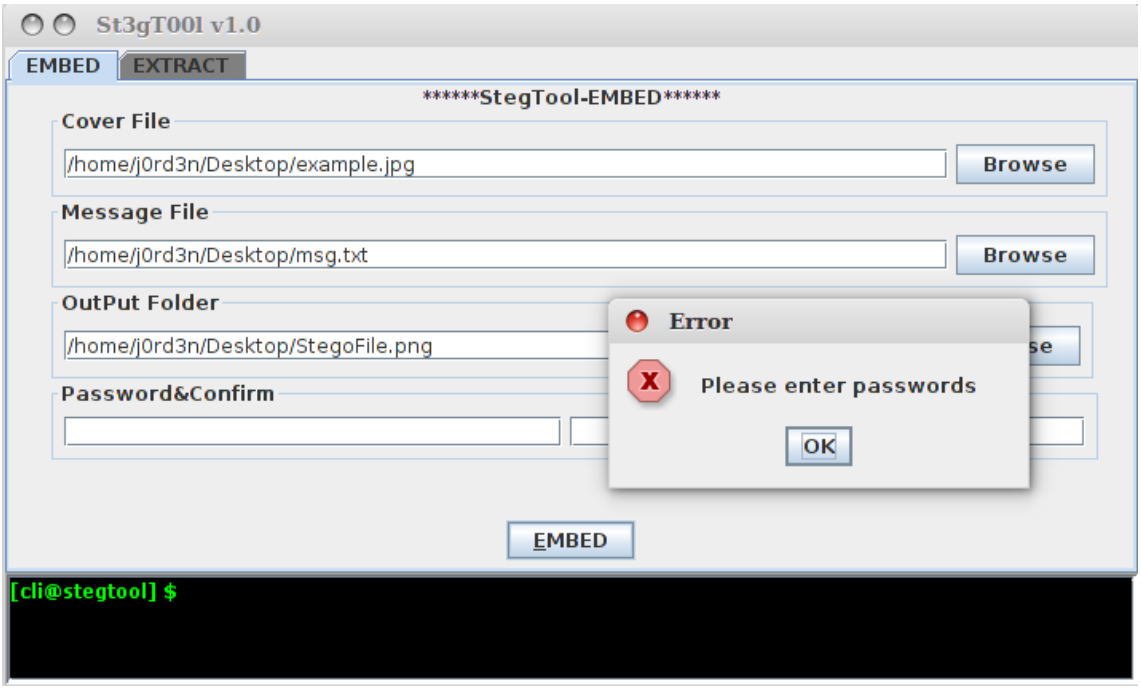

Figure 7.8: Alert to enter passwords

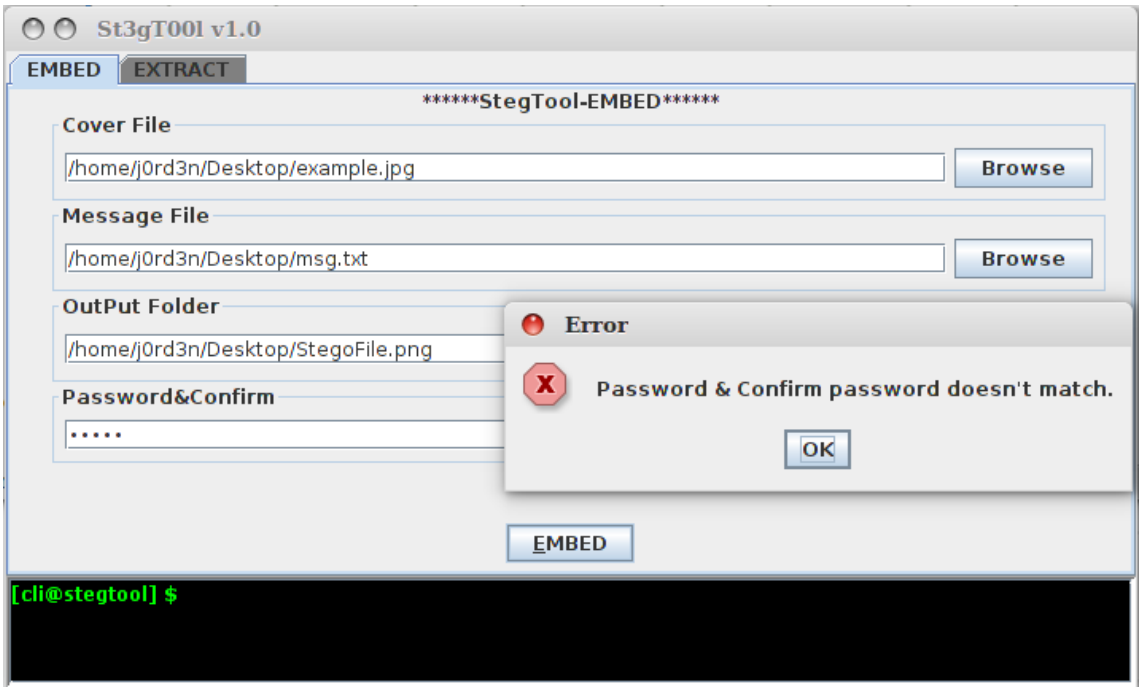

Figure 7.9: Warning! Bothe passwords should be same.

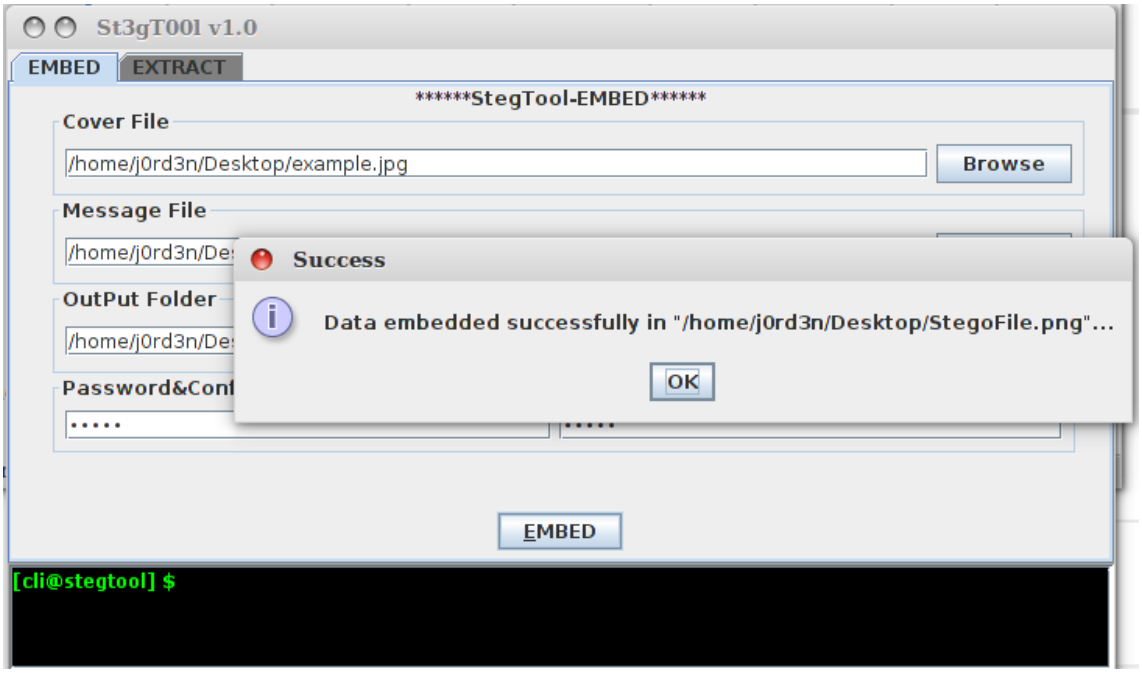

Figure 7.10: Success Message Data embed successfully

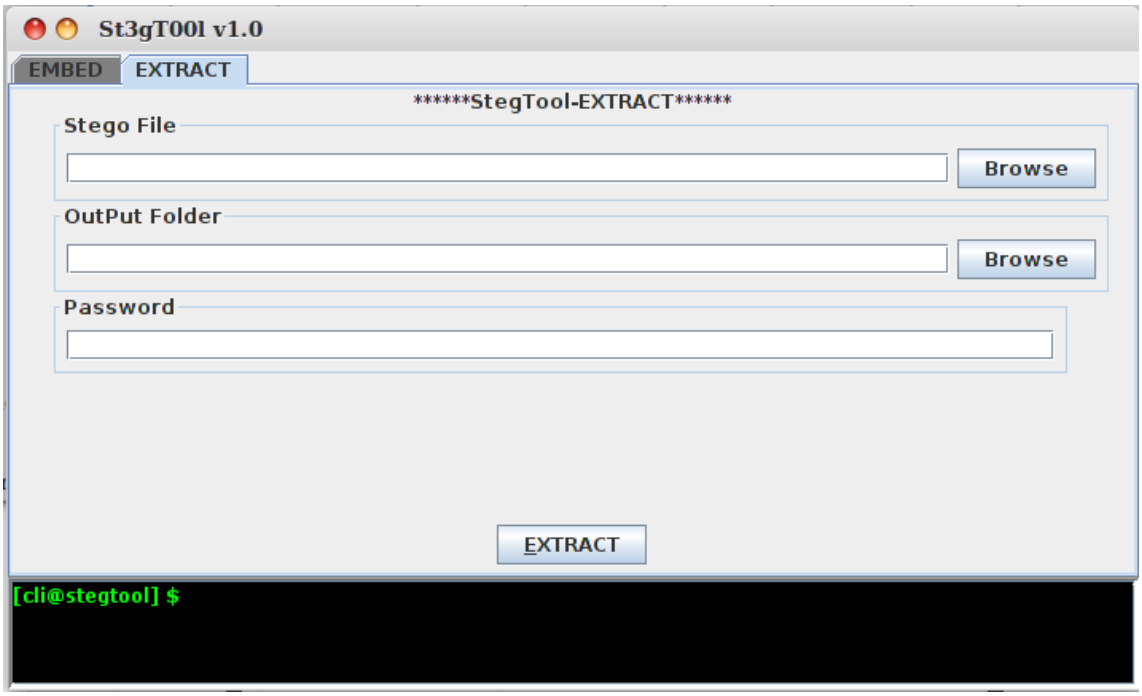

Figure 7.11: Extract Panel

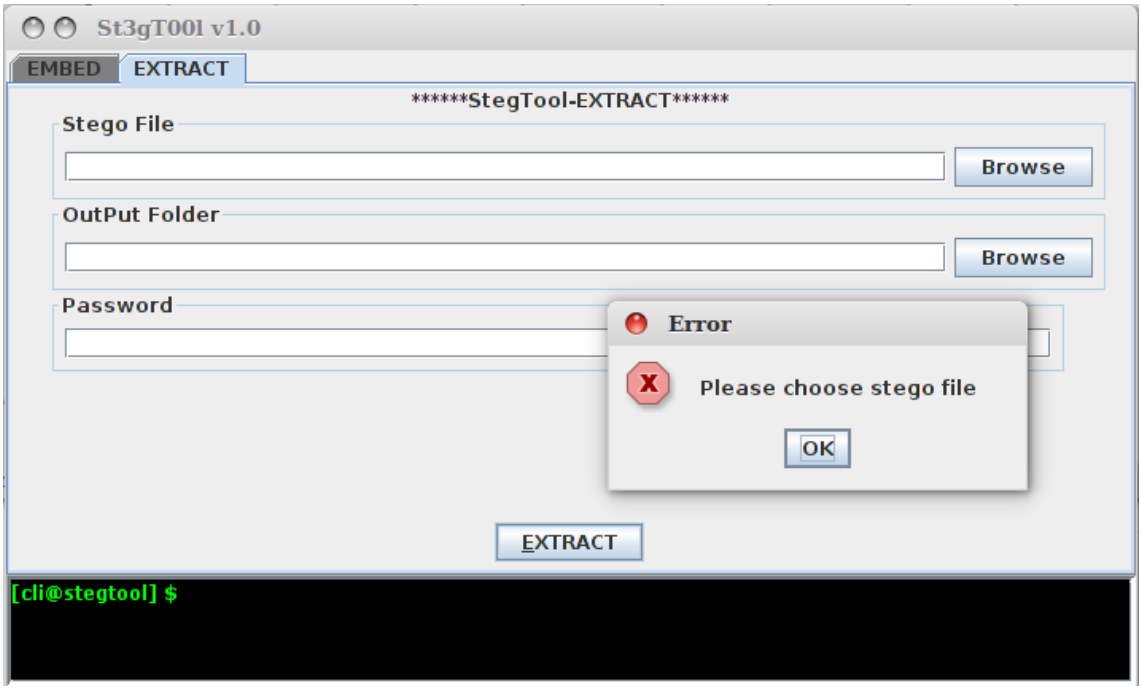

Figure 7.12: Select Stego File to Extract data

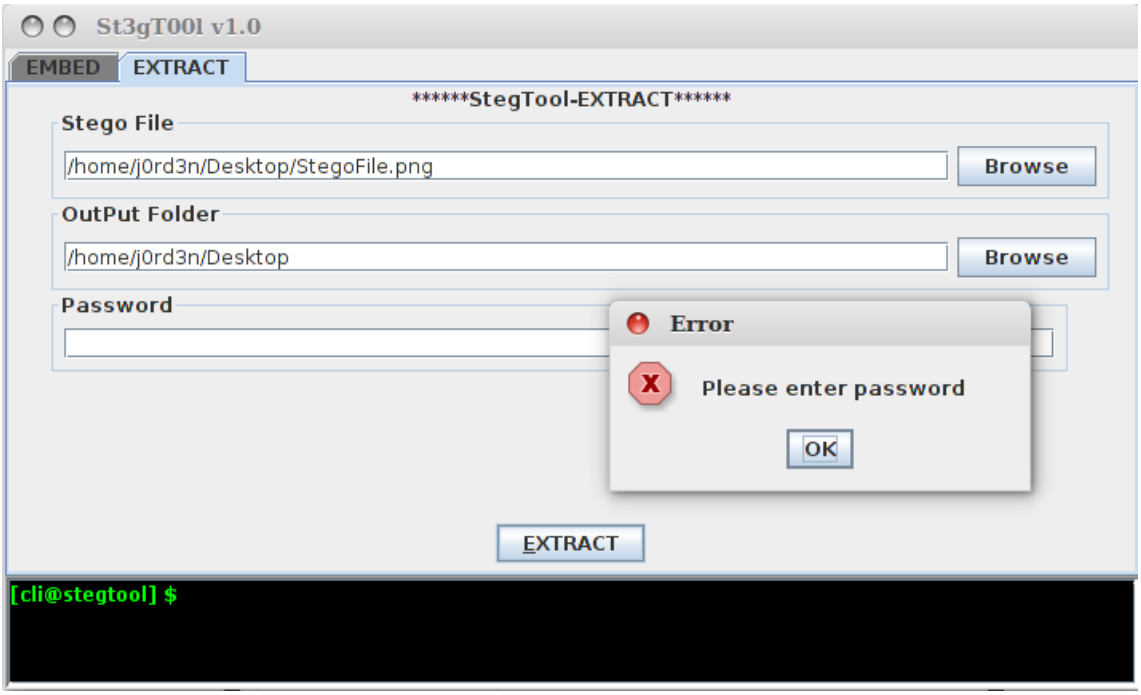

Figure 7.13: Alert! Enter password

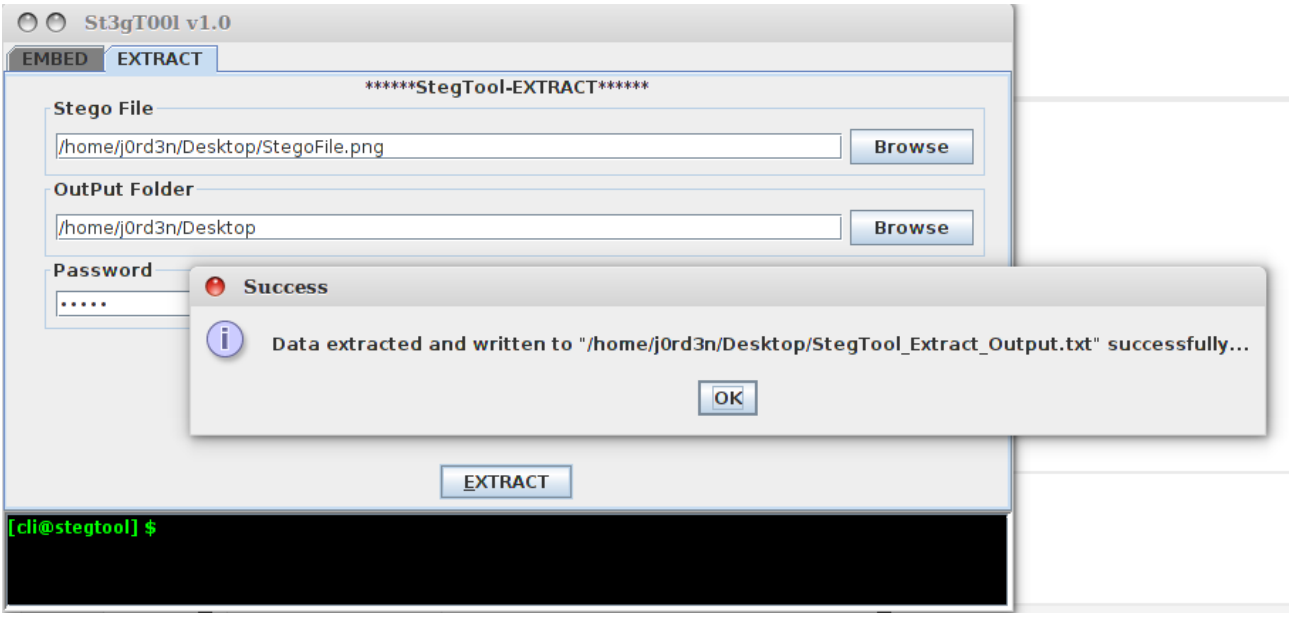

Figure 7.14: Success Message. Data extracted successfully.

Chapter 8

Conclusion

You know nowadays privacy become the main aspect so hiding the data place important role.Here our tool will do that.We use several encryption algorithms to hide the data and decryption algorithms to get hidden data with a protected key.

Although we have been successful in implementing our desired aim of the project functionalities, In Future we will come-up with new version of this in order to overcome the limitations in the current system.

8.1 Future Enhancement

- Extending some other algorithms to embed.
- Improving Command Line Interface with more options.
- Improving simulation to get automated result depends on the procedural steps

Chapter 9

References

9.1 Web References

- Java Tutorial Website, http://tutorialspoint.com/
- Java Documentation Website http://docs.oracle.com/
- Stack Overflow http://stackoverflow.com/