SHIVAJI UNIVERSITY

An Image Based Symmetric Key Cryptographic System

by

Author Name

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the
Department of Computer Science and Engineering
Sharad Institute of Technology College of Engineering,Yadrav-Ichalkaranji

April 2015

SHIVAJI UNIVERSITY

Abstract

Department of Computer Science and Engineering
Sharad Institute of Technology College of Engineering,Yadrav-Ichalkaranji

Doctor of Philosophy

by Author Name

Information sharing through digital media is a vital and challenging one. Those challenges are addressed with the help of cryptographic and stenographic techniques. In general, strength of the symmetric key cryptographic algorithms is based on keys. The objective of this paper is to propose an approach to avoid image distortion due to transmission noise while transferring the image to the receiver for generating the key. For key generation a structure similar to alphabetical trie is used, which is generated from a unique character set. AES algorithm is used for experimental purpose, which is well known for its block of randomized key bits based security and easy implementation.

Acknowledgements

We take this opportunity to express our profound gratitude and deep regards to our guide Mrs. M.N.Jadhav for his exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by him time to time shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to our Principal, Dr. S. A. Khot and our H.O.D., Prof. O.D.Joshi for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are obliged to all faculty and staff members of all departments of our Sharad Institute of Technology College of Engineering, Yadrav for the valuable information provided by them in their respective fields. We are grateful for their cooperation during the period of our assignment.

Lastly, we thank almighty, our parents, brothers, sisters, friends and our colleagues for their constant encouragement without which this assignment would not be possible.

Miss. Batte Priti Anil

Miss. Bhandare Komal Mohan

Miss. Patil Sonalee Annaso

Miss. Potdar Komal Dattaguru

Miss. Yadav Megha Bhupal

Contents

List of Figures

Chapter 1

INTRODUCTION

1.1 Overview:

Information sharing through digital media is a vital and challenging one. Those challenges are addressed with the help of cryptographic and steganography techniques. In general, strength of the symmetric key cryptographic algorithms is based on keys. Cryptology is the study of techniques which ensures secrecy and authenticity of data. Framing a secret code, which is known only to the intended users and which makes the information unpredictable for the eavesdroppers is known as cryptography.

The project is "An Image Based Symmetric Key Cryptographic System" provides security to secret message. This system is vulnerable to transmission noise and brute force attacks. We propose a new image based key generation algorithm, which generates dynamic and complex keys and avoids key sharing issues like transmission noise and brute force attack. First, get the unique character set from the user, which appreciates the security of key by having a dynamic nature. The structure similar to the alphabetical trie is formed using the above unique character set, which appreciates the complexity of key. We choose a non-volatile image in public web sites, which avoids sharing of image to generate the key. Then the alphabetical trie formed using the unique character set, is applied on the chosen image.

In this system two tyes of images are used. First image for key generation and second image for encrypted message wrapping. This second image is called as Stego Image and this is in the format of .png. For key generation both sender and receiver should use same image. Key length is 192 bit. For encryption and decryption purpose we used AES algorithm. This algorithm increases the key complexity without affecting the performance of the system up to a maximum level.

1.2 What is Cryptography?

Cryptology is the study of techniques which ensures secrecy and authenticity of data. Framing a secret code, which is known only to the intended users and which makes the information unpredictable for the eavesdroppers is known as cryptography[1]. Cryptography is a well known and widely used technique that manipulate information in order to crypt their existence. More specifically, cryptography protects information transforming it into an unreadable format.

The original text is transformed into a scramble equivalent text called cipher text and this process is called as Encryption. This is achieved via an Encryption Algorithm. Only those who possess a secret key can decrypt the cipher text into plaintext. Simply it scrambles a message so it cannot be understood. Cryptography deals with protecting information by encoding or transformation of data.

1.3 Why we need Cryptography?

We cannot think about a world without communication. With faster growth of internet, communication has become very easy. There are many methods in cryptography and steganography for providing secured communication. In secure communication, key generation phase has many challenges and this problem can be solved if the sender and the receiver share the key in any other form or if they generate the keys readily during the encryption and decryption separately.

1.4 Branches of Cryptology:

The two main branches of cryptology are cryptography and cryptanalysis. Cryptography deals with the study of design of such defending techniques. Cryptanalysis deals with the defeating study of finding loop holes to break such techniques, to recover information, or forging information that will be accepted as authentic.

1.5 Goals of Cryptography:

Providing confidentiality to the information is the main goal of cryptography. Also providing authentication, integration, non-repudiation are also appreciable goals for it. In order to achieve these goals we need strong cryptographic algorithms.

1.6 Classification of Cryptographic Algorithm:

Based on the key, the cryptographic algorithms are classified as Symmetric key cryptography and Public key cryptography[1]. Single key is used for encryption and decryption in symmetric key cryptography and in public key cryptography two keys are used, where encryption is done using public key and decryption is performed using private key.

Chapter 2

LITERATURE REVIEW

2.1 A New Approach for Improving Data Security using Iterative Blowfish Algorithm:

The paper is proposed on a software tool which considerably enhances the security by following an iterative approach depending upon sender need. It uses Blowfish algorithm included with iterative approach, which enhances the security provided by the algorithm when compared to the non-iterative approach[2].

2.2 Blowfish Algorithm:

Blowfish, a symmetric block cipher that uses a modified Feistel network structure, which has 16 rounds for encryption and decryption. The block size is 64 bits, and the key size is up to 448 bits. The strength of the Blowfish algorithm relies on its sub-key generation and its encryption. Blowfish is a block cipher which uses a variable-length key. It is well fitted for applications in which the key size does not change often. It is significantly faster than most encryption algorithms when implemented on 32-bit microprocessors with large data caches[2].

Blowfish cipher uses 18 each of 32-bit sun arrays commonly known as P-boxes and four Substitution boxes each of 32 bit size and having 256 entries each. It uses a Feistel cipher which is a general method of transforming a function into another function by using the concept of permutation.

2.3 A Novel Cryptographic Key Generation Method Using Image Features:

It is to increase the security in communication by encrypting the information with the key generated by using an image. This study proposes a novel algorithm for key generation using image features. It is a reliable and flexible method of key generation for information security. In this method the user need not remember the keys during encryption and decryption[3].

This study uses the Gray Level Co-occurrence matrix of an image to extract the Gray Level Co-occurrence properties of the image. A 56-bit sub-key is generated from the extracted Gray Level Co-occurrence properties. Then the key for encryption and decryption is to generate during the sub-key generated from the image. This proposed method is easy to implement. The key strange is better than others.

2.4 An Integrated Block and Stream Cipher Approach for Key Enhancement:

RivestCipher4 and Blow fish are the cryptographic algorithms which are very well known for their performance, simplicity, strong key generation. In this paper, this propose a method of combining block and stream cipher for increasing the key strength so that it will be very hard for the intruder to break the key and intruder will have no idea about the key formation from the combination of block and stream cipher. So it will lead to increased key complexity which obviously results the intruder nothing else than confusion and frustration[4].

2.5 Trie Structure:

Here we get the unique character set from the user, which appreciates the security of key by having a dynamic nature. The structure similar to the alphabetical trie is formed using the above unique character set, which appreciates the complexity of key. We choose a non-volatile image in public web sites, which avoids sharing of image to generate the key. Then the alphabetical trie formed using the unique character set, is applied on the chosen image.

2.6 Steganography:

The art and science of hiding information by embedding messages within other, seemingly harmless messages[1]. Steganography works by replacing bits of useless or unused data in regular computer files (such as graphics, sound, text, HTML, or even floppy disks) with bits of different, invisible information. This hidden information can be plain text, cipher text, or even images.

Steganography sometimes is used when encryption is not permitted. Or, more commonly, steganography is used to supplement encryption. An encrypted file may still hide information using steganography, so even if the encrypted file is deciphered, the hidden message is not seen.

For example, the hidden message may be in invisible ink between the visible lines of a private letter. The advantage of steganography over cryptography alone is that the cryptography is the practice of protecting the contents of a message alone, steganography is concerned with concealing the fact that a secret message is being sent, as well as concealing the contents of the message.

2.7 AES Algorithm:

AES (acronym of Advanced Encryption Standard) is a symmetric encryption algorithm[1]. The algorithm was developed by two Belgian cryptographer Joan Daemen and Vincent Rijmen.AES was designed to be efficient in both hardware and software, and supports a block length of 128 bits and key lengths of 128, 192, and 256 bits.

When you want to encrypt a confidential text into a decryptable format, for example when you need to send sensitive data in e-mail. The decryption of the encrypted text it is possible only if you know the right password.

AES is based on a design principle known as a substitution-permutation network, combination of both substitution and permutation, and is fast in both software and hardware.Unlike its predecessor DES, AES does not use a Feistel network. AES is a variant of Rijndael which has a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits. By contrast, the Rijndael specification per se is specified with block and key sizes that may be any multiple of 32 bits, both with a minimum of 128 and a maximum of 256 bits.

AES operates on a 4*4 column-major order matrix of bytes, termed the state, although some versions of Rijndael have a larger block size and have additional columns in the state. Most AES calculations are done in a special finite field. The key size used for an AES cipher specifies the number of repetitions of transformation rounds that convert the input, called the plaintext, into the final output, called the ciphertext. The number of cycles of repetition are as follows:

- 10 cycles of repetition for 128 bit keys.
- 12 cycles of repetition for 192-bit keys.
- 14 cycles of repetition for 256-bit keys.

Each round consists of several processing steps, each containing four similar but different stages, including one that depends on the encryption key itself. A set of reverse rounds are applied to transform ciphertext back into the original plaintext using the same encryption key.

Chapter 3

OBJECTIVE AND SCOPE

3.1 Objective:

- To propose a robust algorithm for key generation using image and unique character set.
- To propose Image Steganography.
- To use AES algorithm for encryption and decryption purpose.

3.2 Scope:

- This project is mainly used for Information Security purpose. Encryption and Decryption is done through image.
- While sending message from sender to receiver message is wrapped into the given image so hackers cannot see secret message.

3.2.1 Out of Scope

- To use of Image steganography rather than Audio and Video steganography.
- To use of .png file rather than .jpeg, .jpg, .bmp file.
- To use of key length 192 bit.
- To use of AES algorithm for encryption and decryption purpose rather than another algorithm.

Chapter 4

REQUIREMENT ANALYSIS

4.1 Hardware Requirements:

- PC/Laptop
- Wi-Fi/Internet Connection

4.2 Software Requirements:

- Operating System: Windows OS- XP or above (For 32-bit and 64-bit edition computer)
- Technologies:

  1. **Java:-**

     (a) **Description:**
     A high-level programming language developed by Sun Microsystems. Java was originally called OAK, and was designed for handheld devices and set-top boxes. Oak was unsuccessful so in 1995 Sun changed the name to Java and modified the language to take advantage of the burgeoning World Wide Web.

     (b) **Java:**
     It is a computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on

one platform does not need to be recompiled to run on another. Java applications are typically compiled to byte code that can run on any Java virtual machine (JVM) regardless of computer architecture.

(c) **Java platform:**

It is the name given to the computing platform from Oracle that helps users to run and develop Java applications. The platform does not just enable a user to run and develop a Java application, but also features a wide variety of tools that can help developers work efficiently with the Java programming language. The platform consists of two essential softwares:
1)The Java Runtime Environment (JRE), which is needed to run Java applications and applets;
2)The Java Development Kit (JDK), which is needed to develop those Java applications and applets.

(d) **Java Runtime Environment 7:**

The Java Runtime Environment (JRE) is a set of software tools for development of Java applications. It combines the Java Virtual Machine (JVM), platform core classes and supporting libraries.
JRE is part of the Java Development Kit (JDK), but can be downloaded separately. JRE was originally developed by Sun Microsystems Inc., a wholly-owned subsidiary of Oracle Corporation.

JRE consists of the following components:

- Deployment technologies, including deployment, Java Web Start and Java Plug-in.
- User interface toolkits, including Abstract Window Toolkit (AWT), Swing, Java 2D, Accessibility, Image I/O, Print Service, Sound, drag and drop (DnD) and input methods.
- Integration libraries, including Interface Definition Language (IDL), Java Database Connectivity (JDBC), Java Naming and Directory Interface (JNDI), Remote Method Invocation (RMI), Remote Method Invocation Over Internet Inter-Orb Protocol (RMI-IIOP) and scripting.
- Other base libraries, including international support, input/output (I/O), extension mechanism, Beans, Java Management Extensions (JMX), Java Native Interface (JNI), Math, Networking, Override Mechanism, Security, Serialization and Java for XML Processing (XML JAXP).

  - Lang and util base libraries, including lang and util, management, versioning, zip, instrument, reflection, Collections, Concurrency Utilities, Java Archive (JAR), Logging, Preferences API, Ref Objects and Regular Expressions.
  - Java Virtual Machine (JVM), including Java HotSpot Client and Server Virtual Machines.

2. **NetBeans IDE 7.4:-**

   (a) **Description:**
   NetBeans is an integrated development environment (IDE) for developing primarily with Java, but also with other languages, in particular PHP, C/C++, and HTML5.[3] It is also an application platform framework for Java desktop applications and others. NetBeans IDE 7.4 was released on October 15, 2013.
   The NetBeans IDE is written in Java and can run on Windows, OS X, Linux, Solaris and other platforms supporting a compatible with JVM. The NetBeans Platform allows applications to be developed from a set of modular software components called modules. Applications based on the NetBeans Platform (including the NetBeans IDE itself) can be extended by third party developers.

   (b) **NetBeans Platform:**
   Framework for simplifying the development of Java Swing desktop applications. The NetBeans IDE bundle for Java SE contains what is needed to start developing NetBeans plugins and NetBeans Platform based applications; no additional SDK is required. Applications can install modules dynamically.
   Any application can include the Update Center module to allow users of the application to download digitally signed upgrades and new features directly into the running application. Reinstalling an upgrade or a new release does not force users to download the entire application again.

   (c) **NetBeans IDE:**
   It is an open-source integrated development environment. NetBeans IDE supports development of all Java applications. Modularity, all the functions of the IDE are provided by modules. Each module provides a well defined function, such as support for the Java language and editing. NetBeans contains all the modules needed for Java development in a single download, allowing the user to start working immediately. Modules also

allow NetBeans to be extended. New features, such as support for other programming languages, can be added by installing additional modules.

Chapter 5

SYSTEM DESIGN

5.1 An Overview of The UML

The UML is a language for

- Visualizing
- Specifying
- Constructing
- Documenting

THE UML LANGUAGE

A language provides a vocabulary and the rules for combing words in that vocabulary for the purpose of the communication. A modeling language is a language whose vocabulary and rules focus on conceptual and physical representation of a system. A modeling language such as the UML is thus a standard language for software blueprints. In this context, specifying means building models that are precise, unambiguous, and complete. In particular, the UML addresses the specification of all the important analysis, design and implementation decision that must be made in developing and deploying a software intensive system. The UML is not a visual programming language, but its model can be directly connected to a verity of programming languages. This means that its possible to map from a model in the UML to a programming language such as java, cpp, or visual basic or even to tables in a relational database. Things that are best expressed graphically are done so graphically in the UML, whereas things that best expressed textually are done so in the programming language. A healthy software organization produces all sorts of artifacts in addition to raw executable code. These artifacts include requirements, architecture, design, source code, project plans, tests, prototypes, releases. The UML addresses the documentation of a systems architectureï£¡s and all of its details. The UML also provides for expressing requirements and for tests. Finally, The UML provides a language for modeling the activities of project planning and release management[5].

5.2 Goals of UML

The primary goals in the design of the UML were:

- Provide users with a ready-to-use, expressive visual modeling language so they can develop and exchange meaningful models. Provide extensibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development processes.Provide a formal basis for understanding the modeling language
- Encourage the growth of the OO tools market.
- Support higher-level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices

5.3 A Conceptual Model of The UML

To understand the UML, you need to form a conceptual model of the language, and this requires learning three major elements: the UML is basic building blocks, the rules that dictate how those building blocks may be put together, and some mechanisms that apply throughout the UML. Once you have grasped these ideas, you will be able to read UML models and create some basic ones. As you gain more experience in applying the UML, you can build on this conceptual model, using more advanced features of the language.

5.3.1 Building Blocks of The UML

The vocabulary of the UML encompasses three kinds of building blocks:

- Things
- Relationships

- Diagrams

These are the abstractions that are first-class citizens in a model; relationships tie these things together; diagrams groups interesting collections of things.

5.4 Diagrams in The UML

A diagram is the graphical presentation of a set of elements, most often rendered as a connected graph of vertices (things) and arcs (relationships). You draw diagrams to visualizing a system from different perspectives, so a diagram is a projection into a system. For all but the most trivial systems, a diagram represents an elided view of the elements that make up a system. The same element may appear in all diagrams. In theory, a diagram may contain any combination of things and relationships. The views that comprise the architecture of software ï£¡ intensive system. For this reason, the UML includes following diagrams:

- Use case diagram
- Class diagram
- Sequence diagram
- Deployment diagram

5.4.1 Use Case Diagram

A use case diagram is a diagram that shows a set of use cases and actors and their relationships. A use case diagram is a just special kind of diagram and shares the same common properties as do all other diagram-a name and graphical contents[5].

5.4.1.1 Contents

Use case diagrams commonly contain

- **Use Case**

  Use case is a description of a set of sequence of actions that a system performs that yields an observable result of value to a particular actor. A use case is rendered as an ellipse with solid lines usually including its name.

FIGURE 5.1: Use Cases

- **Actors**

  An actor represents a role that an outsider takes on when interacting with the business system. For instance, an actor can be a customer, a business partner, a supplier, or another business system and every actor has a name.

- **Dependency, generalization, and association relationships.**

  A dependency is a semantic relationship between two things in which a change to one thing may affect the semantics of the other thing.

FIGURE 5.2: dependencies

A *generalization* is a relationship in which objects of specialized elements (the child) are substitutable for objects of the generalized element.

An *association* is a structural relationship that describes a set of links, a link being connection among objects

Like all other diagrams, use case diagram may contain notes and constraints.

5.4.1.2 Common uses

Use case diagram typically contain in one of two ways.

- To model the context of the system

  Here system involves drawing line around the whole system and actors outside of the system and interact with it.

- To model the requirement of a system

  Here specifies what the system should do, independent of how that system should do.

5.4.1.3 Use-Case Scenarios

Use Case	**Description**
Encrypt	User is the main Actor/Sender in this phase. The main flow of events: 1. User gives input as character set 2. User clicks on Create Trie button. 3. System creates Trie Structure. 4. User clicks on Browse button and select an image. 5. User gives that selected image to the system. 6. System generates key for user. 7. Plain text is covers in image. 8. System encrypts Plain Text. 9. User sends stego image to the receiver.
Decrypt	User is the main Actor/Receiver in this phase. The main flow of events: 1. User gives input as character set. 2. User clicks on Create Trie button. 3. System creates Trie Structure. 4. User clicks on Browse button and select an image. 5. User gives that selected image to the system. 6. System generates key for user. 7. User gives Stego image to the system. 8. System encrypts Cipher Text. 9. User or Receiver receives the original plain text.

Table 5.1: Use Case Scenario Table

5.4.1.4 Use-Case Diagram

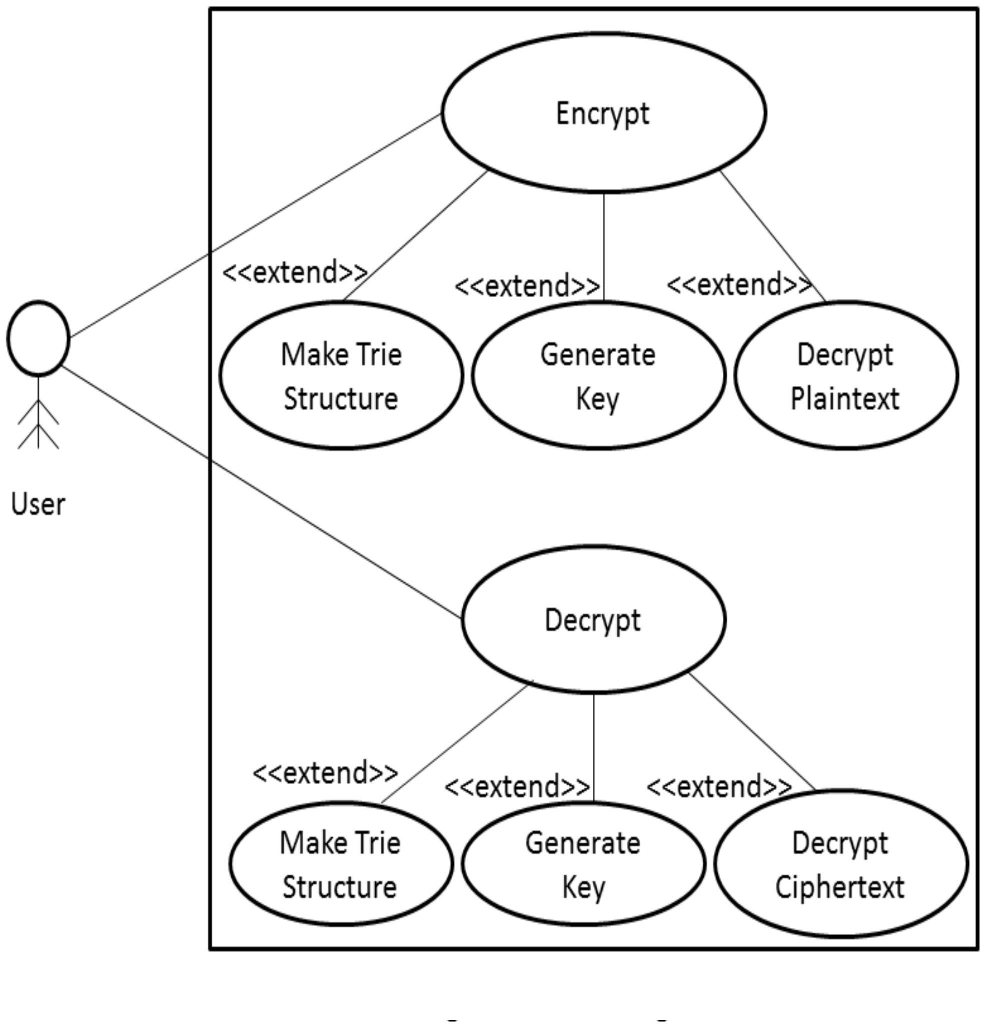


FIGURE 5.3: Use Case Diagram

5.4.2 Sequence Diagram

5.4.2.1 Contents

Sequence diagram commonly contains

- Objects
- Links
- Messages

5.4.2.2 Definition And Overview

A *sequence* diagram is an interaction diagram that emphasizes the time ordering of messages. A sequence diagram shows a set of objects and the messages sent and received by those objects. The objects are typically named or anonymous instances of classes, but may also represent instances of other things, such as collaborations, components, and nodes. You use sequence diagrams to illustrate the dynamic view of a system. An Actor models a type of role played by an entity that interacts with the subject (e.g., by exchanging signals and data), but which is external to the subject (i.e., in the sense that an instance of an actor is not a part of the instance of its corresponding subject). Actors may represent roles played by human users, external hardware, or other subjects. Note that an actor does not necessarily represent a specific physical entity but merely a particular facet (i.e., "role") of some entity that is relevant to the specification of its associated use cases[5].

Sequence diagram have two features that distinguish them from collaboration diagrams.

- First, there is the object lifeline. An object lifeline is the vertical dashed line that represents the existence of an object over a period of time. So these objects are at the top of the diagram. With their lifelines drawn from the top of the diagram to the bottom
- Second, there is the focus of control. The focus of control is a tall, thin rectangle that shows the period of time during which an object is performing an action, either directly or through a subordinating procedure. The top of the rectangle is aligned with the start of the action; the bottom is aligned with its completion and also it can be marked by replay message.

5.4.2.3 Sequence Diagrams

Sequence Diagram for Encryption:

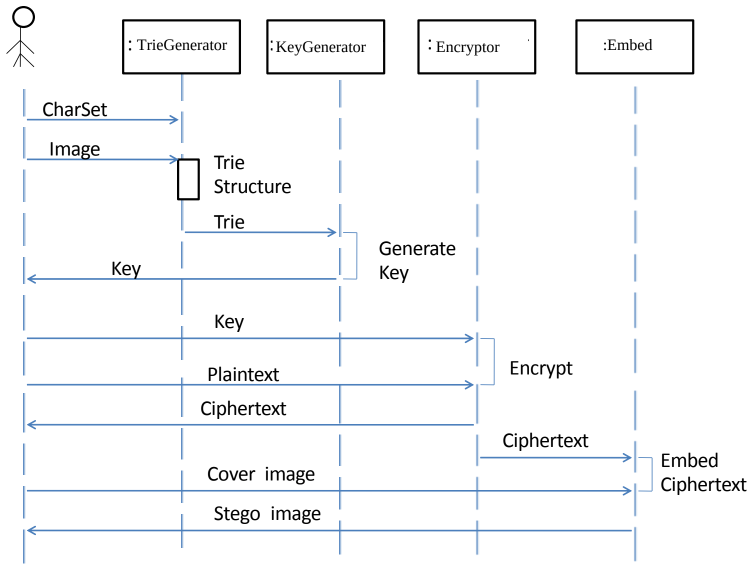


FIGURE 5.4: Sequence Diagram for Encryption

Sequence Diagram for Decryption:

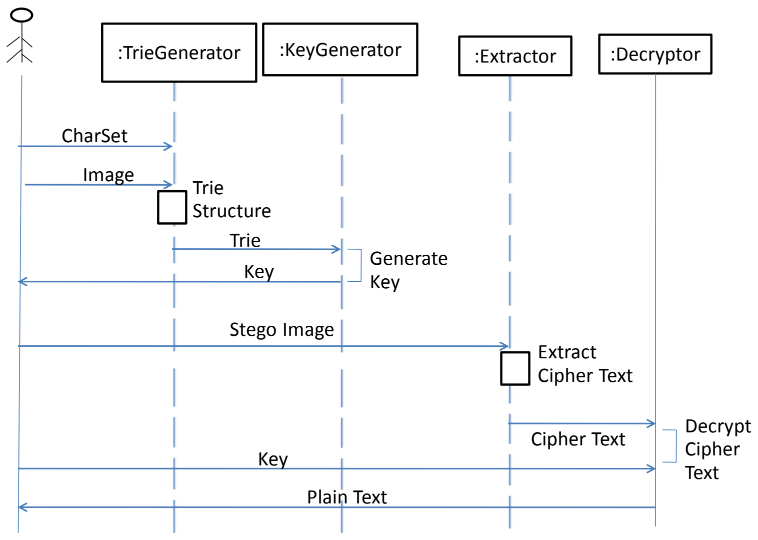


FIGURE 5.5: Sequence Diagram for Decryption

5.4.3 Class Diagram

5.4.3.1 Contents

Class diagram commonly contain the following things:

- Classes
- Interfaces
- Collaborations
- Dependency, generalization, and association relationships.

5.4.3.2 Definition and Common Uses

A class diagram is a diagram that shows a set of classes, interfaces and their relationships. Graphically, a class diagram is a collection of vertices and arcs. A class diagram will shares the same common properties as do all other diagrams[5]. A class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modeling Language (UML). In this context, a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity. Class diagrams are useful in all forms of object-oriented programming (OOP). The concept is several years old but has been refined as OOP modeling paradigms have evolved. In a class diagram, the classes are arranged in groups that share common characteristics. A class diagram resembles a flowchart in which classes are portrayed as boxes, each box having three rectangles inside. The top rectangle contains the name of the class; the middle rectangle contains the attributes of the class; the lower rectangle contains the methods, also called operations, of the class. Lines, which may have arrows at one or both ends, connect the boxes. These lines define the relationships, also called associations, between the classes.

- Class: A definition of objects that share given structural or behavioral characteristics.
- Attribute: A typed value attached to each instance of a classifier.
- Operation: A method or function that can be performed by instances of a classifier

5.4.3.3 Class Diagram:

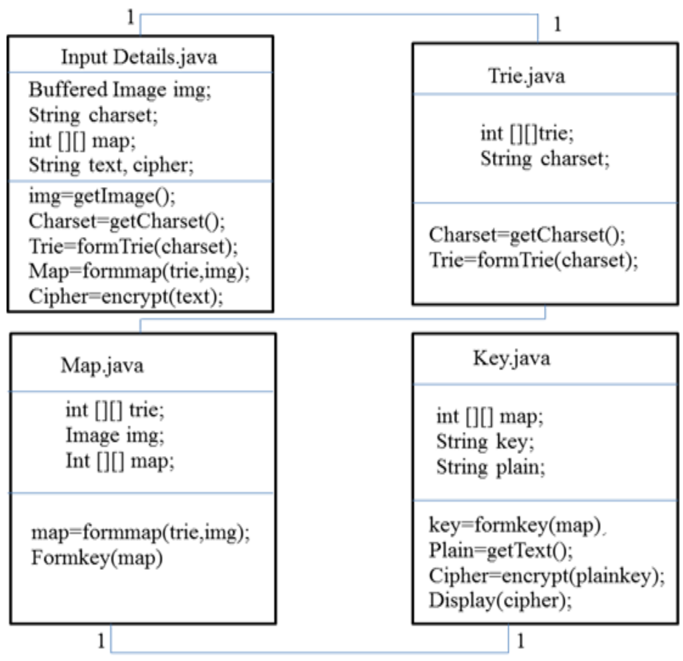


FIGURE 5.6: Class Diagram

5.4.4 Deployment Diagram

subsectionDefinition A deployment diagram shows the configuration of run time processing nodes and the components that live on them. Deployment diagram address the static deployment view of architecture[5]. They are related to component diagrams in that a node typically encloses one or more components .

5.4.4.1 Nodes and Components

The UML provides a graphical representation of node. This canonical notation permits you to visualize a node apart from any specific hardware. Using stereotype this notation to represents specific kinds of processors and devices.

A *node* is a physical element that exists at run time and represents a computational resource, generally having at least some memory, and often processing capability. Graphically, a node is rendered as a cube. Every node must have a name that distinguishes it from other nodes. A name is a textual string. Components are things that participate in the execution of a system; nodes are things that execute components. Components represent the physical packaging of otherwise logical elements; nodes represent the physical development of components and components that things are executed by nodes. The UML can often use stereotypes to specify new kinds of nodes that you can use to represent specific kinds of processors and devices. A *Processor* is a node that has processing capability, meaning that it can be executed by component. A*device* is a node that has no processing capability and general, represents something that interfaces to real world.

5.4.4.2 Deployment Diagram

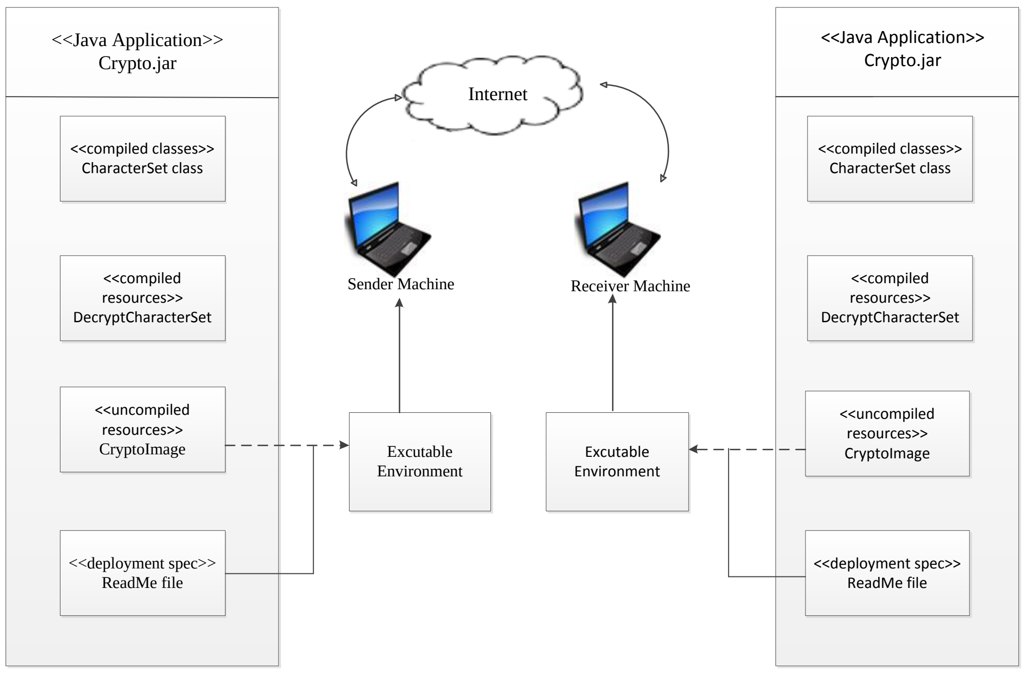


FIGURE 5.7: deployment diagram

Chapter 6

CODING

6.1 Introduction of tools and Installation

6.1.1 NetBeans IDE 7.3.1

6.1.1.1 Installation:

The Java SE Development Kit (JDK) 6 Update 26 (or later) or JDK 7 Update 10 (or later) is required to install the NetBeans IDE.

To install the software:

1. After the download completes, run the installer.
   - For Windows, the installer executable file has the .exe extension. Double-click the installer file to run it.
   - For Solaris and Linux platforms, the installer file has the .sh extension. For these platforms, you need to make the installer files executable by using the following command: chmod +x <installer-file-name>
2. If you downloaded the All or Java EE bundle, you can customize your installation. Perform the following steps at the Welcome page of the installation wizard:
   (a) Click Customize.
   (b) In the Customize Installation dialog box, make your selections.
   (c) Click OK.
3. At the Welcome page of the installation wizard, click Next.

4. At the License agreement page, review the license agreement, click the acceptance check box, and click Next.

5. At the JUnit License Agreement page, decide if you want to install JUnit and click the appropriate option, click Next.

6. At the NetBeans IDE installation page, do the following:

   (a) Accept the default installation directory for the NetBeans IDE or specify another directory.

   Note: The installation directory must be empty and the user profile you are using to run the installer must have read/write permissions for this directory.

   (b) Accept the default JDK installation to use with the NetBeans IDE or select a different installation from the drop-down list. If the installation wizard did not find a compatible JDK installation to use with the NetBeans IDE, your JDK is not installed in the default location. In this case, specify the path to an installed JDK and click Next, or cancel the current installation. After installing the required JDK version you can restart the installation.

   Note: If the JDK version is older than the recommended JDK 7 Update 10, download and install the latest JDK update from Java SE Downloads page and restart the NetBeans IDE installer. You can run the NetBeans IDE on JDK 6 as an alternative.

7. If the GlassFish Server Open Source Edition 4.0 installation page opens, accept the default installation directory or specify another installation location.

8. If you are installing Apache Tomcat, on its installation page, accept the default installation directory or specify another installation location. Click Next.

9. At the Summary page, do the following:

   (a) Verify that the list of components to be installed is correct.

   (b) Select the Check for Updates check box if you want to check the Update Center for possible updates and have the JUnit library installed during the installation (provided you accepted the license in step 5.)

   (c) Verify that you have adequate space on your system for the installation.

10. Click Install to begin the installation.

11. At the Setup Complete page, provide anonymous usage data if desired, and click Finish.

Generating Trie structure:

```
String date=txtdate.getText();
String month=txtmonth.getText();

String total=date+month;

//remove duplicates
 String rowdata="";
char[] arr=total.toCharArray();

for(int i=0;i<arr.length;i++)
{

    if(i==0)
    {
     rowdata+=arr[0];
    }
    else
    {
        boolean flag=false;

        for(int j=0;j<rowdata.length();j++)
        {
            if(arr[i]==rowdata.charAt(j))
            {
                flag=true;
            }
        }

        if(flag==false)
        {
           rowdata+=arr[i];
        }
    }
```

```
     }


        System.out.println(rowdata);

        //create matrix of rowdata*26

         subrows=rowdata.length();


   char[]alpha={'a','b','c','d','e','f','g','h','i','j','k','l','m','n',
        int cols=26;
      trie=new int[subrows][cols];

           for(int i=0;i<subrows;i++)
        {
              outer:     for(int j=0;j<cols;j++)
              {

                  for(int k=0;k<26;k++)
                  {
                  if(alpha[k]==rowdata.charAt(i))
                  {
                        trie[i][k]=1;

                        break outer;
                  }

              }
         }


        }

               System.out.println();
               for(int i=0;i<subrows;i++)
                {
                 for(int j=0;j<cols;j++)
                {
```

```
                System.out.print(trie[i][j]);

            }
           System.out.println();
         }
```

Divide the image into row blocks based on the number of nodes in the above structure and columns into 26 blocks representing the alphabets.

```
                int cols=width/26;  //size of column block
                int rows=height;

                int divrows=rows/subrows; //size of row block



                // get the first element of trie.

                int maprow=0;
                int mapcol=0;
                int p=0,q=0;
                for(int i=0;i<subrows;i++)
                {

                    for(int j=0;j<26;j++)
                    {

                        if(trie[i][j]==1)
                        {

                                //Store the location of colno

                            maprow=i;
                            mapcol=j;
                            break;
                        }
                    }

                        // first row 20th column
                    float temp[][]=new
float[divrows][cols];

                    int imgrowno=0;

                    if(maprow==0)
                    {
                    imgrowno=maprow;
                    }
                    else
                        imgrowno=maprow*divrows;

                    for(int
m=imgrowno;m<imgrowno+divrows;m++)
                    {

                        for(int
n=(mapcol*cols);n<(mapcol*cols)+cols;n++)
                        {



                            int clr=  img.getRGB(n,m);
//colno,rowno
                            int  r = (clr & 0x00ff0000)
>> 16; //notation for red
                            int  g = (clr & 0x0000ff00)
>> 8; //notation for green
                            int  b = clr &
0x000000ff;      //notation for blue
                           //  System.out.println("row
"+m+"col   "+n);
                            //

                            temp[p][q]=
Math.round(0.2989 * r + 0.5870 * g + 0.1140 * b)
;//intensity of image coods[m][n];
```

```
                                             }

                                        }
```

Decryption

```
private void
   jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
    DecryptImage is=new DecryptImage(labelkey.getText());
    is.setVisible(true);
    is.setSize(800,800);

}

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info
   : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

   javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

   java.util.logging.Logger.getLogger(DecryptCharacterSet.class.
getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

   java.util.logging.Logger.getLogger(DecryptCharacterSet.class.
getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
```

```
    java.util.logging.Logger.getLogger(DecryptCharacterSet.class.
getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException
    ex) {

    java.util.logging.Logger.getLogger(DecryptCharacterSet.class.
getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {

            public void run() {
                new DecryptCharacterSet().setVisible(true);
            }
        });
    }
```

Chapter 7

TESTING

7.1 What is Software Testing

Software testing is the process of analyzing or operating software for the purpose of finding bugs. Testing can be described as a process used for reveling defects in software, and for establishing that the software has attained a specified degree of quality with respect to selected attribute.The fundamental objective of testing is to find defects ,as early as possible and get them fixed.

Software Testing Process

- Test Planning high level plans which list test objectives, test approach, measurement criteria along with test schedule and resources.
- Test Design create test cases, identify test cases for automation(if applicable),prioritize test cases and finalize test iterations.
- Test Implementation Create test scripts using automated testing tools.
- Test Execution Execute the test cases on the test environment and test reports.
- Test analysis Use test and project metrics to calculate key indicators. The data usually will be obtained from your defect tracking system.
- Postmortem reviews Discuss lessons learnt and identify strategies which will prevent such problems in future.

7.2 Test methods

7.2.1 Black box testing

It is also called as functional testing, it is the process of giving the input to the system and checking the output of the system.Without bothering about the system that how the system generates the output. It is also called as Behavior testing.

- Approach to testing where the program is considered as a âĂIJBlack BoxâĂİ.
- Testing based solely on analysis of requirements user specification, user documentation etc.
- The test cases are based on the specifications.
- Black box testing techniques apply to all levels of testing.
- Test planning and design can begin early in the software process.
- Tests are done from a users point of view.

7.2.2 White Box Testing

White box testing or structural testing considers facets like programming style, control method, source language, database design. A test for remote monitoring routine can be an example of structural test. This type of testing helps to uncover defects at structural level. The tests go below the top or functional layer to uncover the defects.

- Testing that takes into account internal structure and flow of a system or component.
- The testing is based on code structure or the algorithm.
- White box testing assumes that the procedural design and code is known to the tester.
- Obliviously test design can be done only after coding is complete.
- White box tests are inherently finite.

7.3 Test cases and test data

- Test data are inputs that have been devised to test the system.
- Test cases are inputs and output specification plus a statement of the function under test.
- Test data can be generated automatically or real.

Table 7.1: Test Cases For Encryption and Decryption

Sr.No.	**TC_ID**	**Objectives**	**Prerequisites**	**Steps to be followed**	**Expected Result**	**Remark**
1	Run project.	Run project	1. Click on ⟨ Next ⟩ button.	Next window displayed.	Next window displayed.	Pass
2	Select option	To perform Encryption user must be click on Encryption button	1. Click on ⟨ Encryption ⟩ button.	Encryption window will display.	Encryption process window displayed.	Pass
3	Unique key.	Unique key creation	1. Enter Day and Month.	Unique key created.	Unique key for Trie structure created.	Pass
4	Trie structure	Trie structure creation	1. Click on ⟨ Create Trie ⟩ button.	Trie structure created.	Trie structur created.	Pass
5	Key Genration	Public key for encryption	1. Click on ⟨ Browse ⟩ button and select image. 2. Click on ⟨ Generate Key ⟩ button.	Public key	Key for encryption genrated.	Pass
6	Encryption.	Encryption of plaintext.	1. Enter plaintext for encryption.	Transliterated text will be displayed.	Ciphertext genrated.	Pass

Continued on next page

Table 7.1 –âĂŘ–âĂŘ continued from previous page

Sr.No.	**TC_ID**	**Objectives**	**Prerequisites**	**Steps to be followed**	**Expected Result**	**Remark**
			2. Click on ⟨ Encrypt ⟩ button.			
7	Stego image.	Wrapping of message	1. Click on ⟨ Embed in CoverImage ⟩ button. 2. Click on ⟨ Browse ⟩ button and select image 3. Click on ⟨ EMBED ⟩ button.	Message wrapped in selected image.	Stego image generated.	Pass
8	Select option	To perform Decryption user must be click on Decryption button	1. Click on ⟨ Decryption ⟩ button.	Decryption window will display.	Decryption process window displayed.	Pass
9	Unique key	Unique key creation	1. Enter Day and Month.	Unique key created.	Unique key for Trie structure created.	Pass
10	Trie structure	Trie structure creation	1. Click on ⟨ Create Trie ⟩ button.	Trie structure created.	Trie structure created	Pass

Continued on next page

Table 7.1 –âĂŘ–âĂŘ continued from previous page

Sr.No.	TC_ID	Objectives	Prerequisites	Steps to be followed	Expected Result	Remark
11	Key Genra-tion	Public key for de-cryption	1. Click on ⟨ Browse ⟩ button and select image. 2. Click on ⟨ Gen-erate Key ⟩ but-ton.	Public key .	Key for decryption gen-rated.	
12	Decryption	Decrypt cipher-text	1. Click on ⟨ Ex-tract from Cover-Image ⟩ button. 2. Click on ⟨ Browse ⟩ button and select stego image. 3. Click on ⟨ Ex-tract and Decrypt ⟩ button.	Ciphertext decrypted.	Plaintext generated.	Pass

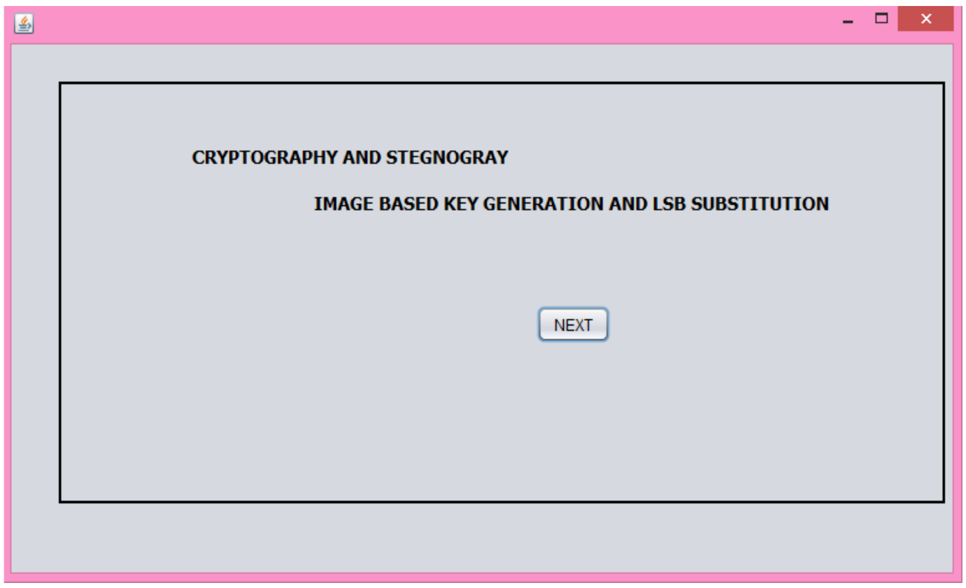
Chapter 8

Deployment

8.1 SNAPSHOTS

8.1.1 Start Window:

When we click on crypto.jar file, application gets started and we get the below display.



CRYPTOGRAPHY AND STEGNOGRAY

IMAGE BASED KEY GENERATION AND LSB SUBSTITUTION

FIGURE 8.1: Start Window

8.1.2 Select The Option:

We have to select any one the option from this window. For Encryption choose the ENCRYPTION, for decryption choose the DECRYPTION option.

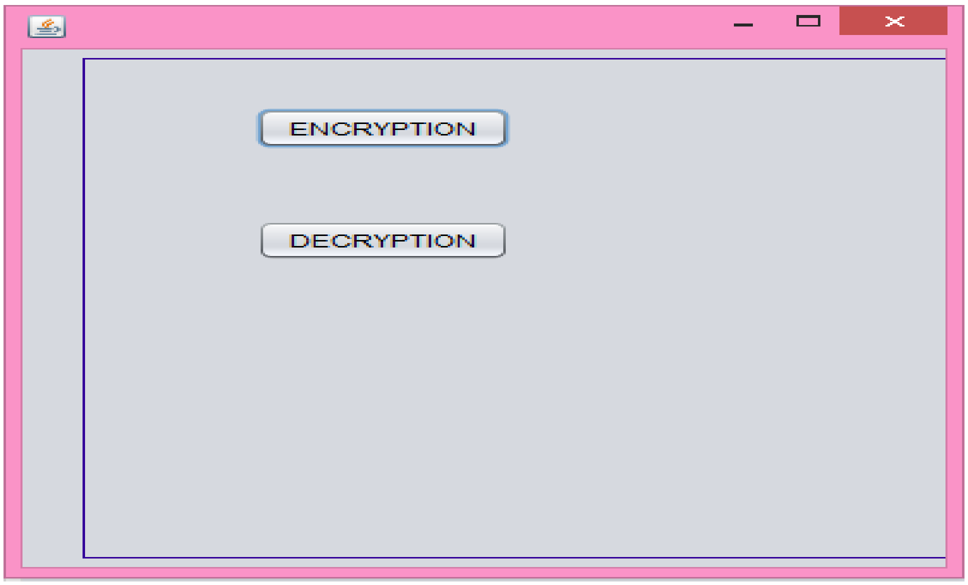
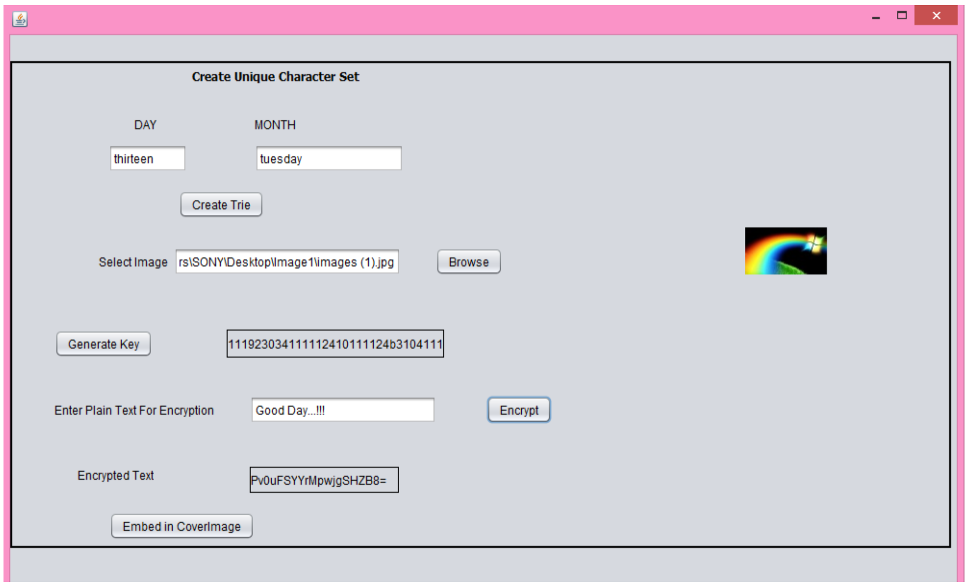


FIGURE 8.2: Select the option as a Encryption

8.1.3 Encryption:

For Creating Unique charater set we have to give Day and Month. After clicking on Create Trie ,Trie structure will be generated. Then Browse the image for generating key. Enter the plaintext for encryption.By clicking on Embed in coverimage, Encrypted text will be wrapped in the image.



Create Unique Character Set

DAY: thirteen
MONTH: tuesday

Select Image: rs\SONY\Desktop\Image1\images (1).jpg

1119230341111124101111124b3104111

Enter Plain Text For Encryption: Good Day...!!!

Encrypted Text: Pv0uFSYYrMpwjgSHZB8=

FIGURE 8.3: Generate Key and Encryption of a Plaintext

8.1.4 Generate Stego image:

After clicking on Embed in coverimage,this window will open. To Generate the Stego image,Browse the image and click on Embed.

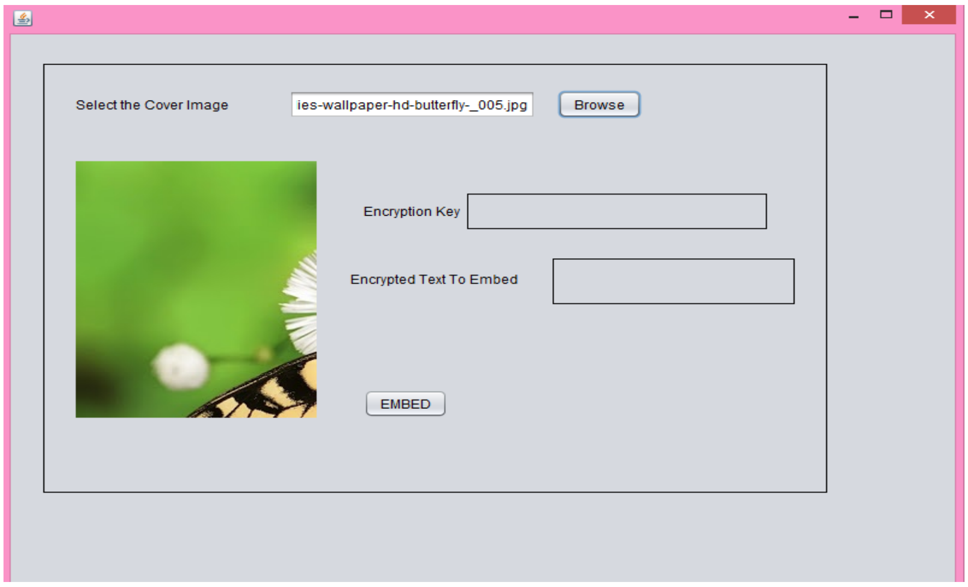


FIGURE 8.4: Generate Stego image

8.1.5 Select The Option:

For Decryption of a text,select the Decryption option.

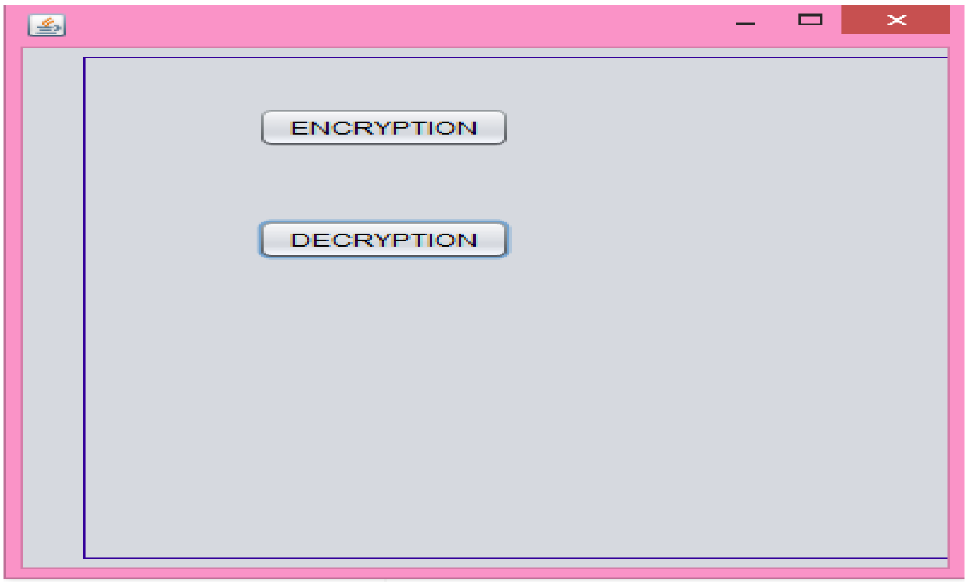
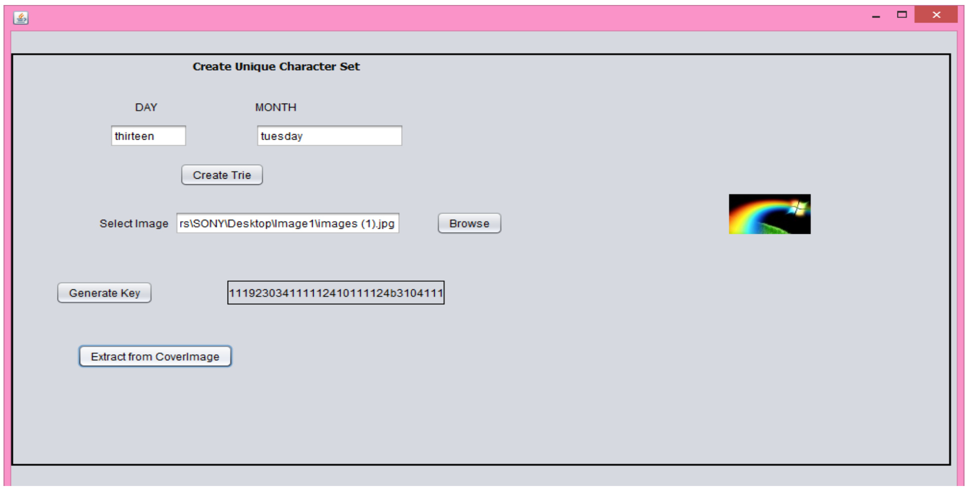


FIGURE 8.5: Select The Option as a Decryption

8.1.6 Generate Key:

For Creating Unique charater set we have to give Day and Month which is same as which we given for Encryption. After clicking on Create Trie ,Trie structure will be generated. Then Browse the image for generating key. Then click on Extract from Coverimage.


Create Unique Character Set

DAY: thirteen

MONTH: tuesday

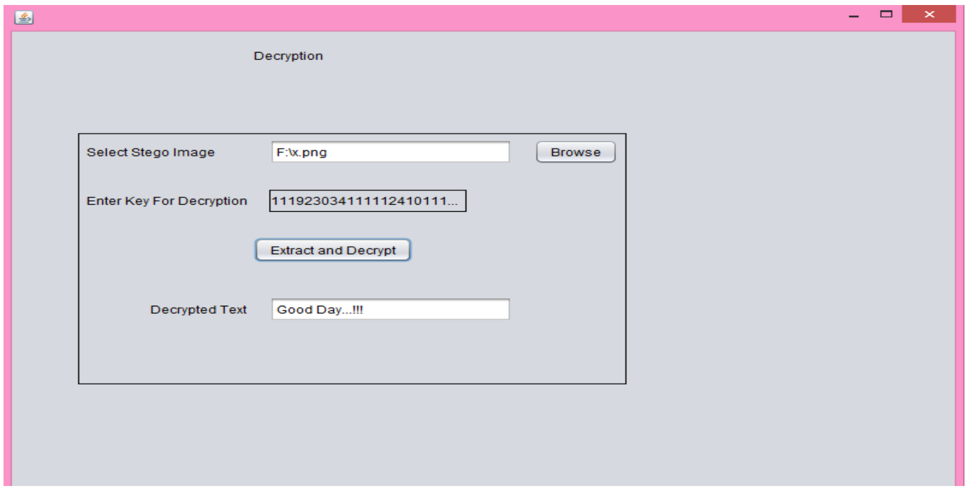
Select Image: rs\SONY\Desktop\Image1\images (1).jpg

Generate Key: 1119230341111124101111124b3104111

FIGURE 8.6: Generate Key for Decryption

8.1.7 Decryption:

For Decryption we have to select the Stego Image. The Decrypted text will be generated after clicking on the Extract and Decrypt button.


Decryption

Select Stego Image: F:\x.png

Enter Key For Decryption: 1119230341111112410111...

Decrypted Text: Good Day...!!!

FIGURE 8.7: Extraction and Decryption

Chapter 9

CONCLUSION

9.1 Conclusion

The experimental results conclude that we have proposed a new algorithm for key generation technique that is the AES algorithm. Thus the AES algorithm increases the key complexity without affecting the performance of the system up to a maximum level. The unique feature of the algorithm is that the key generated is dynamic in nature.AES algorithm is more secured than traditional and simple steganography.

The process has an advantage of key generation. To generate key, we used unique character set and image. This process is more secured for key generation, where it is directly based on the image properties. So, the key generation technique using AES algorithm is very complex and hard to crack it.

Bibliography

[1] William Stallings. *Cryptography and Network Security, 3rd Ed, Wiley*. 1995.

[2] Leela G.H Krishnamurthy G.N, Dr. V. Ramaswamy and Ashalatha M.E. âĂIJperformance enhancement of blowfish and cast 128 algorithms and security analysis of improved blowfish algorithm using avalanche effectâĂİ. *IJCSNS 244 International Journal of Computer Science and Network Security*, 8(3), 3, March 2008.

[3] B Santhi, KS Ravichandran, AP Arun, and L Chakkarapani. A novel cryptographic key generation method using image features. *Research Journal of Information Technology*, 4(2):88–92, 2012.

[4] G Manikandan, R Manikandan, P Rajendiran, G Krishnan, and G SundarGanesh. An integrated block and stream cipher approach for key enhancement. *Journal of Theoretical and applied information Technology*, 28(2):83–87, 2011.

[5] Google. Uml standard diagrams, âĂIJlearning uml in easy stepsâĂİ, 2014. URL http://www.tutorialspoint.com/uml/uml_diagrams.html./.