College of Science and Mathematics

Department of Computer Science

Montclair State University

Masters Project

Secured Data Encryption Application for Various File Formats Using Public Images

By:

Senthilraj Rajendran

Montclair State University

Advisor:

Dr. Stefan A Robila

Department of Computer Science

Montclair State University

Montclair, NJ 07043

TABLE OF INDEX

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENT

First, I would like to use this opportunity to express my gratitude to my advisor Dr. Stefan A Robila for providing this opportunity and support to complete this project.

I was motivated by Professor Dr. Stefan Robila with this topic to proceed further with his great encouragement. I am sure with one thing that, it would not be possible, if Dr. Stefan A Robila was not with me to advise and guide me throughout this semester.

I would like to thank Dr. Constantine Coutras and Dr. George Antoniou for serving in my MS project committee and spending their precious time to review the project.

I am eternally grateful and thankful to all my professors who supported me in completing this project.

Finally, I would like to thank all my faculties, peers and my wife for their huge support, even though they all have tight schedule in their work, during my entire Master's program.

ABSTRACT

The main goal of this project was to design an application that provides secured data encryption for text and media files such as excel, pdf, video, audio, ppt, word, image and text files using modern steganography techniques. Such tool is useful in securing peer to peer communication as well as in ensuring the protection of private individual data.

Since currently published steganographic techniques can be used to hide data in files of any format, the tool developed also allows for embedding data in various data files, although the focus was image files.

Steganography is described as the technique of concealing any type of file within another file such as file, video or image. Even though it has been done many times successfully for text format files, this idea is brought up mainly for media files such as video, audio and image files. In this project, I will use any file converted as a single image file along with secret key protection which provides higher level security.

A second goal of this project was to allow me to improve my skills as software developer and integrator, and thus acquire valuable pre-professional experience. Steganography based data hiding tools are currently readily available. However, developing one from ground up is challenging and requires the use of various software environments, thus providing an excellent opportunity to demonstrate one's programming and system integration tools.

1. INTRODUCTION

1.1. Cryptography

Cryptography is a method of transmitting data into unreadable or unusable form so the data is secured in such a way that they can be used only by authorized people. Cryptography is also called the study of secret writing concerned with converting data from plaintext into ciphertext and vice versa. The process of converting data from plaintext into cipher text is called enciphering or encryption. The reverse process is called Deciphering or Decryption.

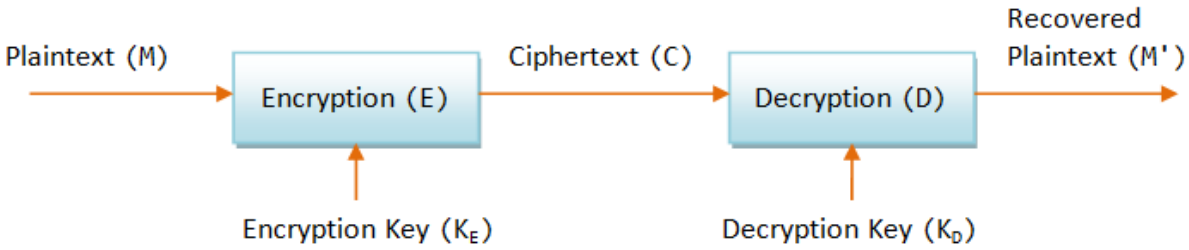


Figure 1 - Process of Cryptography [1]

Various encryption algorithms are used in different applications based on the security level, cost, design and implementation. Cryptography algorithm is used as a mathematical function in encoding and decoding processes. Security level of encryption or decryption is completely depending on the secret key and the algorithm used for encryption [2].

1.2. Steganography

Digital steganography, a modern data hiding technique, plays a vital role in securing data and communication. Steganography is the process of hiding data with in another data that could be in any format such as text, video, audio or image file. Steganography is originally derived from Greek words ‘Steganos’ and ‘graphos’ which mean covered and written respectively. Stenography is an enhanced method of data hiding that encodes and decodes the data for a secured data transmission [3]. In Steganography, hidden data is invisible so nobody can suspect that data is hidden or existing within the data.

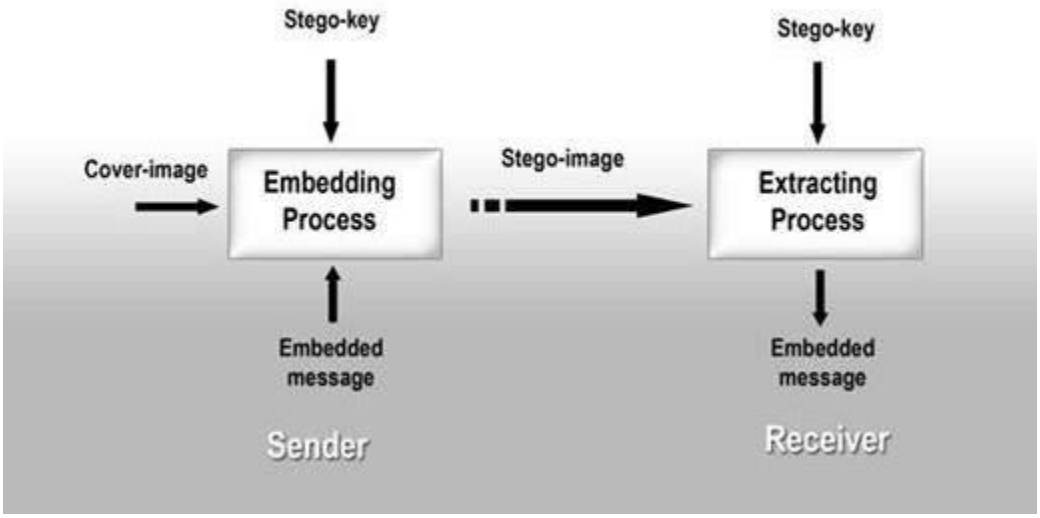


Figure 2 - Framework of Steganography model [4]

While many data protection techniques exist, steganography has the added advantage of “hiding information in plain sight” i.e. of presenting an approach in which the secured data is not easily discoverable, being embedded in a host data. Depending on the host file format various steganographic techniques have been introduced ranging from simple bit modification (changes in the Least Significant Bit) to more complex ones (wavelets based techniques) [5]. The choice of technique has an impact on the size of the data that can be embedded as well as on the ease of the discoverability of the fact that data is embedded at all (i.e. steganalysis). To strengthen the steganography step, encryption is often used before the data is embedded.

In order to achieve ease of process and more security, image files are preferred when compared to video, audio or text files for hiding data in steganography. However, the quality and pixels of the image may be affected due to added data that are hidden in it [6]. This might affect secured transmission of data and lead to vulnerability of data loss.

The paper is organized as follows.

Chapter 1 provides a brief introduction about cryptography and steganography.

Chapter 2 explains the scope of the project that includes project goal and proposed work.

Chapter 3 describes Rijndael algorithm and LSB encoding.

Chapter 4 provides details on how the system was implemented.

Chapter 5 explains about system design that includes input and output system design and system process.

Chapter 6 shows various UML diagrams like use case, class, sequence and component diagrams.

Chapter 7 provides a brief summary on the testing processes used.

Chapter 8 explains system requirements which include hardware and software requirements.

Chapter 9 provides the conclusion.

Chapter 10 provides proposed future enhancements.

Chapter 11 provides brief information about tools, database and software used in the project.

Chapter 12 provides all the references made for this project.

Chapter 13 shows screen shots of various phases.

Chapter 14 contains the source code.

2. SCOPE OF THE PROJECT

2.1. Project Goal

In our current world, everyone keeps data such as pictures, text, excel, word, video and audio files in their devices. So it is quite common that people share, store and transfer the data files among them. Sharing files among multiple devices is very common and often. However, threats and attacks on data are also common nowadays.

The main goal of this project is to come up with a technique of embedding the data in the image file. At the same time, if the data that is hidden in the image file is encrypted, the security level will be increased to a satisfactory level [7]. Even the hidden data is discovered by the person trying to get the original data, it is possible only to have the encrypted data with no way of being able to decrypt it. There are some free online applications are available like Hide in Picture, wbStego and mp3stego for data hiding. However, these tools didn't win any awards for maintaining the good looks of images [8].

For this project, I have decided to develop a data embedding tool using a steganography technique called LSB encoding. The data file is embedded in an image file called cover file using a secret key. Data file will be encrypted using the AES encryption algorithms also called Rijndael algorithm. As a reverse process, the data file will be extracted from the cover file in the encrypted form. Then the encrypted data file will be decrypted using the same algorithm.

2.2. **Proposed Work**

The data file which is in any format such as text, excel, ppt, video, audio, image or word document will be encrypted first and the encrypted data file will be embedded in the image file using 8 digit secret key which is either generated automatically or by user. After embedding the data files, the image file will still look like a simple image file; however it contains the embedded data in it. Once these steps are done the embedding process will be over.

The retrieval part will be done in the reverse order. In the retrieval part the embedded data file will be extracted from the image file using the secret key. After extracting the data file which is still in encrypted form, it needs to be decrypted. Retrieval process will also be done in the similar way how the data embedding was done. More importantly, capacity and quality of the image file will also be maintained same and the data files will be recovered without any loss.

C# .NET is used for coding in the Windows environment that provides high level security for this kind of applications.

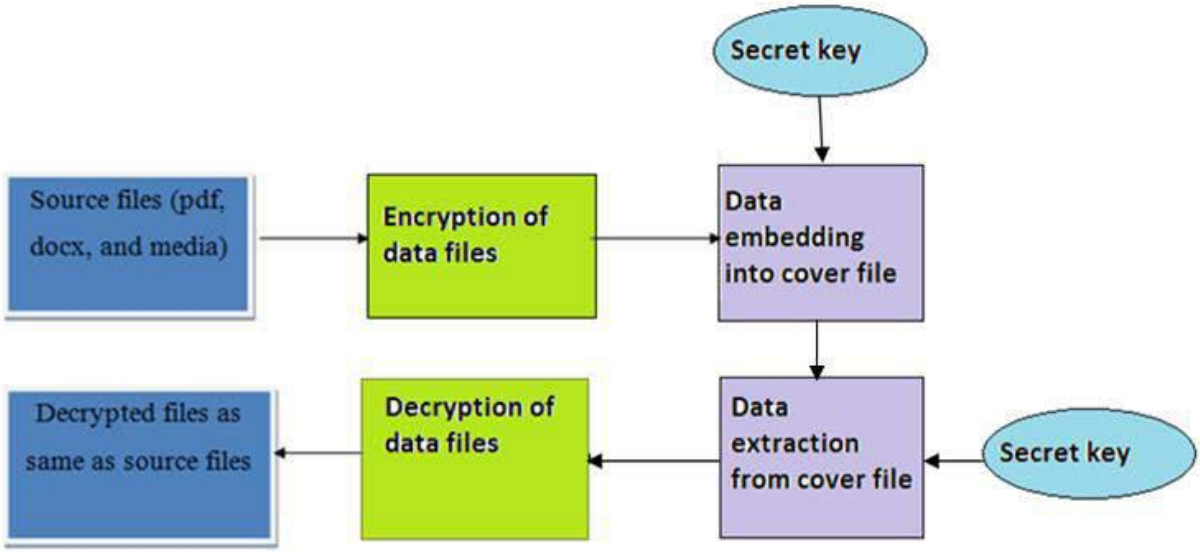


Figure 3 - Block Diagram of the Proposed System

3. BACKGROUND

3.1. Rijndael Algorithm

Rijndael is the AES algorithm used in the cryptography. It was developed by Joan Daemen and Vincent Rijmen. It is an iterated block cipher which means data encryption or decryption of a set of data is performed by the iteration or cycle [8]. The Rijndael algorithm uses a method of generating sub keys from the original key. These sub keys generated from the original key are then used in the various iterations. This algorithm was developed on the basis of three criteria listed below.

- Secured from common attacks.
- Simple design.
- Compatibility of code on various platforms.

Rijndael was assessed on the basis of cost effectiveness, security and implementation characteristics. Apparently, for any algorithm, these three factors are very important. Even though it was focused primarily on security, the choice of Rijndael algorithm was based on its simple design and implementation characteristics [8]. Rijndael algorithm has the combination of efficiency, flexibility, ease of implementation and security.

As Rijndael algorithm is an iterated block cipher, it accepts 8 bit byte arrays as input. These arrays make data blocks in which a set of data is considered for encryption or decryption for a particular iteration. The input plaintext is

mapped to state bytes and the cipher key used in the process is also 8 bit byte array.

3.1.1. Key and Block Size

Rijndael algorithm is capable of operating with different size of data blocks/sets and keys. It has both the key and block sizes of 128, 192 and 256 bits. DES keys have maximum of 56 bits length which can provide 1016 * 7.2 keys approximately [9]. So, there can be 1021 times more AES 128-bit keys generated when compared to 56-bit DES keys.

3.1.2. Key Expansion and Key Schedule

Key expansion is performed to make sub keys from the original key. AES key and the block size will be considered as input in key expansion. More importantly, the key expansion methods for 128,192 and 256 bit keys are slightly varying from each other [10]. The key expansion can be performed in the sequence of 4 steps. In step 1, 8-bit circular rotation takes on a 32-bit key. In this operation last 8 bit is rotated to the extreme left. In step 2, rcon operation is performed in which the user supplied value in the Galios field is simply exponentiated by 2. Galios field is a mathematical construct used for arithmetic refinement operations and there are a limited number of integers used in this field [11]. Rijndael's Galois field allows only 8-bit number in it, so the results of all arithmetic operations are 8-bit numbers. In step 3, Rijndael's S-box operation takes place. S-box is created by finding multiplicative inverse of a number provided in Rijndael's Galois field. In step 4, key schedule routine takes place in which 32-bit output is obtained. Then

the 32-bit input is copied over the 32-bit output and the last 8-bit of the output is rotated to the left [12].

After completing the key expansion phase, a sub key is generated by deriving a round key by another round key. The round key length is equal to the length of the data block/set multiplied by number of iterations plus 1 [13]. This is how the round keys or sub keys are generated from the expanded original key. The expanded key is not mentioned anywhere directly in this method as it will lead to attacks on the key generation method.

3.1.3. Enciphering with Rijndael

Since Rijndael algorithm is an interactive block cipher, it performs encipher or decipher of data in a sequential way. It always performs encryption and decryption by mixing sub keys with data blocks. This step is performed internally as part of security purpose to protect data against cryptanalysis. Adding a round key by itself is another important step in Rijndael enciphering. When a data block is ready for enciphering the input bits are arranged in the order of plaintext length. For 128-bit plaintext length, 44 matrix of bytes is made by arranging the input bits; for 192-bit plaintext length, 46 matrix of bytes is made by arranging the input bits; for 256-bit plaintext length, 48 matrix of bytes is made by arranging the input bits [14]. Then the cipher keys are also arranged in the same way. The numbers 4, 6 and 8 are block width which is denoted asN_b.

The Figure 4 shows that how ciphering is performed in 3 rounds. First round is called initial round in which the input plain text matrix is XORed with a predefined key. The process of XORing the input plain text matrix is called add-round key.

$$\mathrm{B} = \mathrm{A} \oplus \mathrm{K} \tag{1}$$

Where: B =Output byte matrix
A= Input byte matrix
K= Key byte matrix

The second round is the Standard Round in which the operations listed below are performed.

- Sub Bytes: It uses 2 predefined lookup tables to perform a simple byte substitution in this operation. One table is for encoding and another one is for decoding.

- Shift Row: Each row of the input matrix is shifted and rotated in a predefined way. The bytes will be shifted and rotated repeatedly according the value of block width N_b [14].

- Mix-Column: The columns of the matrix are mixed using a matrix multiplication of the plaintext.

The third round is called the Final Round in which all the above operations are performed except Mix Column operation.

All these steps are done in a reverse order to decrypt the data and Mix Column operation is called Inverse Mix-Column or InvMix-Column in decryption process.

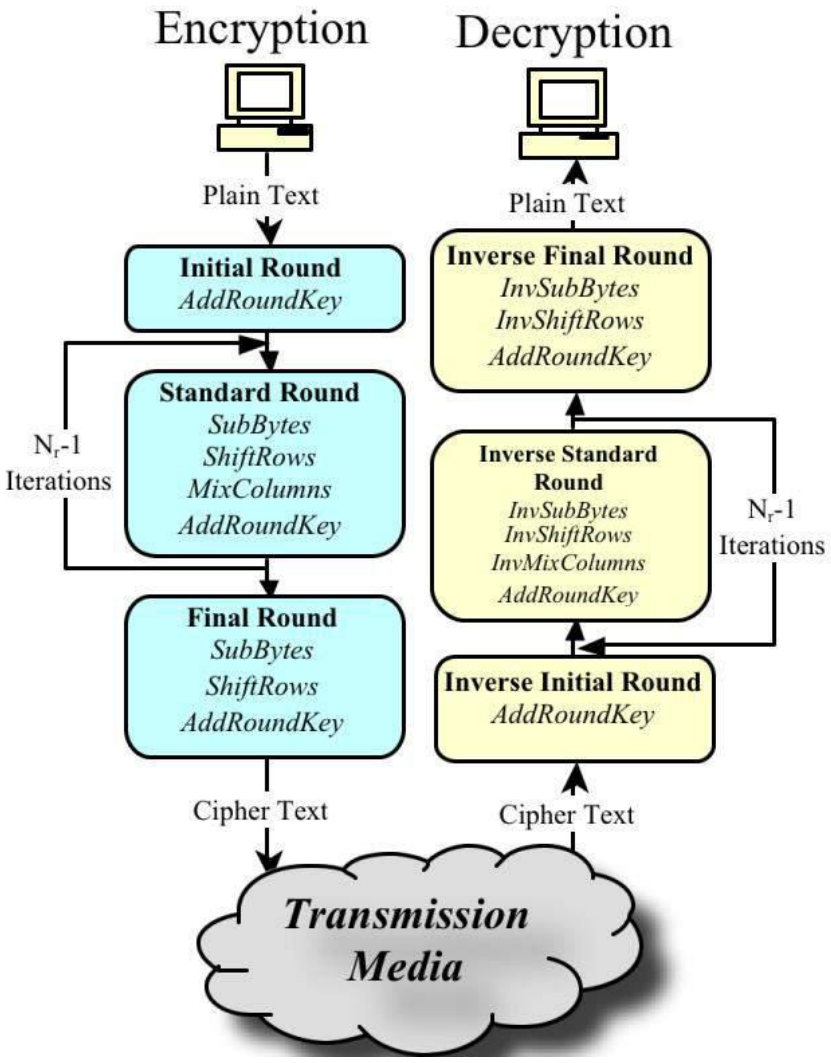


Figure 4 - Flow Diagram of Rijndael Algorithm [14]

3.1.4. Advantages of Rijndael Algorithm

- It is simple to design.
- It is easy to implement as it can be used with various keys and block sizes.
- Cipher's security can be evaluated easily as it has rich algebraic structure.
- It needs very low memory.
- It can easily defend timing and power attacks.

3.2. Low-Bit Encoding

Low-bit encoding is the one of the widely used techniques to embed data into other data. Low bit encoding adopts Least Significant Bit (LSB) technique in which the least significant bit of each sampling point by a coded binary string is replaced by the data bits to be embedded. A large amount of data can be embedded in an image file [15]. The channel capacity for this transmission is 1 kb per second (kbps) per 1 kilohertz (kHz). The LSBs are identified in a cover medium and those bits are replaced by the data bits that are to be embedded. Low bit encoding is robust as it reduces the data rate which could result in the requirement of a host of higher magnitude, often by one to two orders of magnitude. In practice, this method is useful only in closed, digital-to-digital environments.

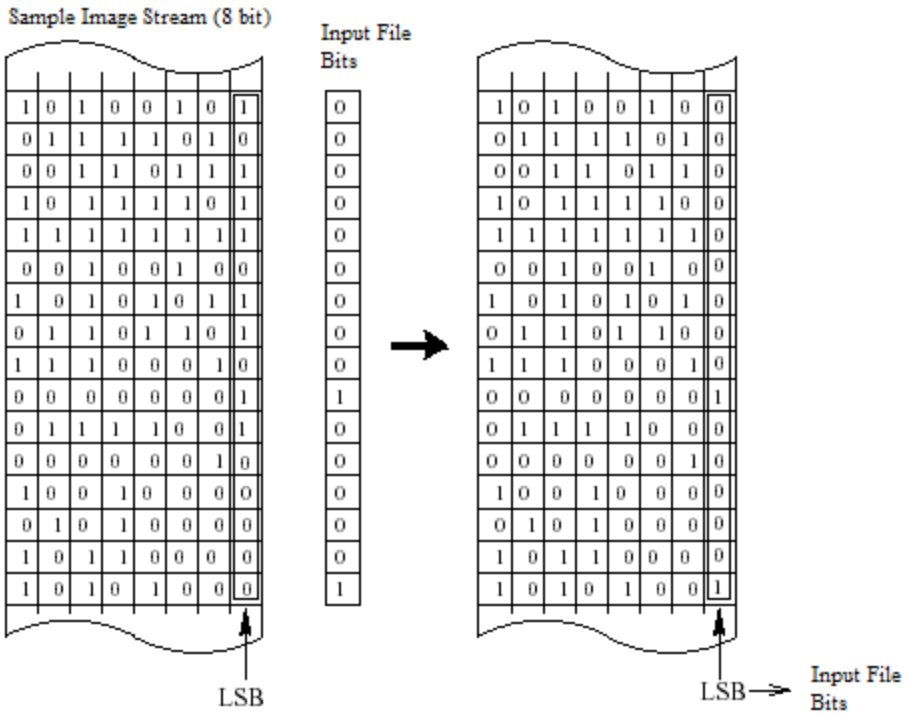


Figure 5 - LSB Embedding Technique [15]

3.2.1. LSB Embedding

Least Significant Bit (LSB) Embedding is a common steganographic technique which is employed to hide the data into a variety of cover file. However, most of the applications are using images as cover media in which LSB embedding is used for hiding the data into image file. LSB embedding is also called LSB encoding. LSB embedding is known for its simple design and efficiency [16]. However, a large amount of data can be embedded using this steganography technique. The quality of the image file needs to be maintained same throughout the embedding process. The Figure below shows the process of LSB embedding.

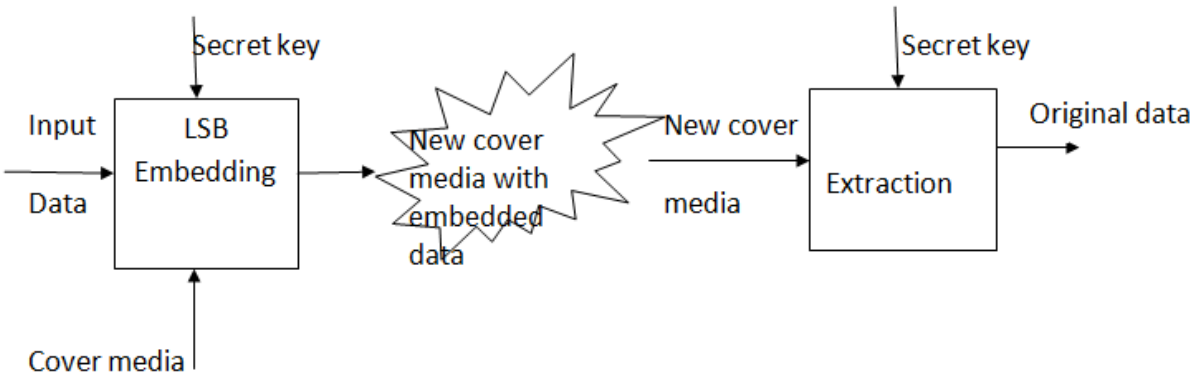


Figure 6 - LSB Embedding process

3.2.2. LSB embedding with Image file

LSB embedding is used with variety of cove medium such as audio, , image etc… Among all the steganography techniques image steganography is a widely used technique due to it simple design and ease of implementation. Image files are more appropriate for cover medium than any other files.

Image steganography with LSB encoding is performed in four steps. In step 1, the image is converted to matrix values and these values will be stored in a text file [17]. When the image file chosen by the user as a cover medium, it is then read and converted into grey image. Initially, the image will be in RGB form which is an opposite form of grey image. Each pixel of an image has intensity values that are stored in the text file. These intensity values are used at the time of image to matrix conversion. The Figure below shows a sample image file used one of the cover medium in this project.

Figure 7 - Sample Cover File

In step 2, after converting the image into matrix form, data embedding process begins. The new cover medium is received after the embedding process and it will be in unreadable format. The data is embedded into the intensity value of cover medium.

In step 3, the intensity values of the image are converted back to the image. Now, the converted image contains data embedded in it. The new image file is also called stegano-image [18].

In step 4, extraction process is performed in which the embedded data will be extracted from the cover medium. In this process 8 bit message byte is

declared and a pixel starting from address 0 is read. Then the LSB is extracted and i^{th} bit of the message byte is replaced by the extracted LSB. Here the value of i is from 1 to 8. Once all the 8 bits are replaced, then it means 1 byte of the message is extracted [15]. Then it starts extracting the next byte of the message. This cycle continues until all the message bytes are extracted.

3.2.3. Choosing the cover medium

Cover medium and data are 2 main factors in steganography. Selecting an appropriate cover medium is very important as it affects efficiency and time duration of embedding process if it is chosen wrongly. Images are more appropriate cover medium than any other media [19]. One of the important things in choosing an image as a cover is that it should have a large variety of colors so that it won't be easy to detect.

4. System Implementation

This section explains how the system implementation was done.

4.1. Initiating Visual Studio

Using visual studio, as an IDE for C#, coding part was completed to create pages, back-end connections and processes for the project. In .NET Framework, discussed briefly in software overview section, there are various applications such as Windows, console, domain, asp.net web form applications available to design several projects on the basis of user's requirements [20]. As we need Windows application for our system, the Windows form application is selected among various applications as shown in Figure 8.

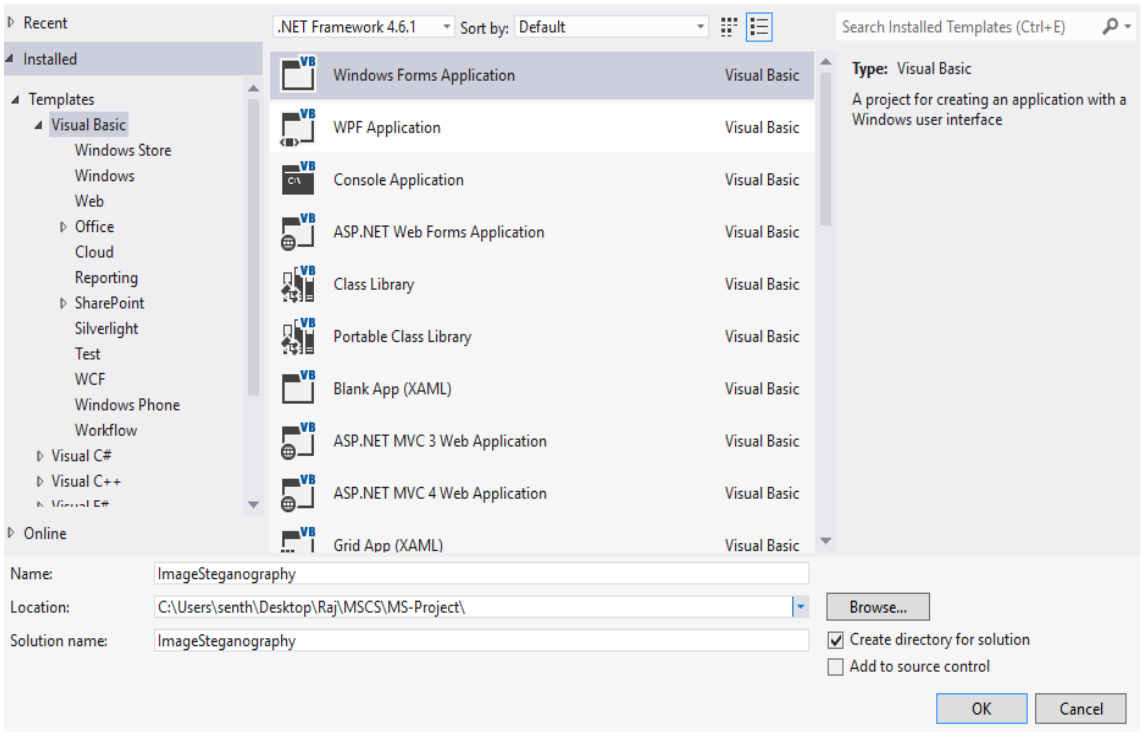


Figure 8 - Solution and Project creation

4.2. Solution and Project Creation

The solution and project are created for the proposed system with the name of 'ImageSteganography' as shown below. As mentioned above the required folders and files are created in the path 'C:\Users\SenthilRaj\Desktop\Raj\MSCS\MS-PRJ\ImageSteganography'.

Selecting an application for designing a project, user is then allowed to create the projects. A solution needs to be created under which one or more projects can be created. After creating the solution and project all the required files, directories and other supporting files for the project are created in the above mentioned workspace.

4.3. Solution Components of the Project

There are several components available for a project such as forms, class files, resources, properties, references etc. Next step is to add forms in which codes are written. In addition to the script file, a form consists of supporting files called designer and resx files. A designer file contains the code for front end applications like page view. The resx files are also called resource files which are used to add, remove or edit the resources to the project. Resources can be string, object data or images used for background etc… A resx file can be created by using "System.Resources.ResXResourceWriter" class.

Class file is another important component of a project in which user defined classes are created. Class file name indicates the name of the class and they

contain methods and instances used for the project. There are 2 user defined class files are used for this project.

- clsEmbed: This ClsEmbed class file contains the methods used for secret key generation and details of secret key length generated at the time of embedding process.

- clsEmbedExtract: This class file contains the methods used for operations on pixel values of image and other data files. It also defines the maximum pixel length of data file in order to be embedded in the image file.

4.4. Application Phases

Image steganography for various data files is implemented 5 phases as discussed below.

4.4.1. Initial phase

In the initial phase, it displays the main window of the process which allows users to start with encypting the data. User can continue with embedding data, extracting data and finally decrypting the data. ‘Main.cs’ is the name of the form used for this phase which also consists of designer and resx files named ‘Main.designer.cs’ and ‘Main.resx’. Designer file imports all the required system classes which could be used as supporting classes for the

main window and resx file stores the background image and strings used for the main window.

4.4.2. Encryption Phase

Data encryption phase allows user to encrypt the data which is to be embedded in the image file. User can choose the data file and get it encrypted in this phase. 'Encryption.cs' is the name of the form used for this phase. The designer and resx files are named as 'Encryption.designer.cs' and 'Encryption.resx' for this phase. Encrypted data file will be named as "file name"_enc and stored in the same location where the actual data file exists.

4.4.3. Embedding Phase

Data embedding phase allows user to embed the data which was encrypted in the previous phase. Users can choose the cover medium which hides the encrypted data in it. Image chosen as cover medium is displayed on this page. Then user can generate 8 digit secret key or use the system generated secret key which is stored in the database. After generating the secret key user can select the data file which is in the encrypted form. Finally, encrypted data file is embedded in the image file by clicking "Embed File" button displayed on the page. Once data file is embedded in the image file, the image file won't show any data in it. It will not display anything if it gets opened without proper manner by hackers. A new cover file is created with the name of "Image file name"_new in the same

location where the actual cover/image file exists. ‘Embedding.cs’ is the name of the form used for this phase. The designer and resx files are named as ‘Embedding.designer.cs’ and ‘Embedding.resx’ for this phase.

4.4.4. Extraction Phase

Reverse process begin in extraction phase in which data file is extracted from the new cover file that hides encrypted data in it. User can choose the new cover file created in embedding phase to begin extraction. To extract the data file user needs to provide the 8 digit secret key generated at the time of data embedding. Entering the correct secret key user can click on “Extract” button to start the extraction process. The data file in encrypted form is extracted from the image file in the same location. ‘Extraction.cs’ is the name of the form used for this phase. The designer and resx files are named as ‘Extraction.designer.cs’ and ‘Extraction.resx’ for this phase.

4.4.5. Decryption Phase

Decryption phase is the last phase in which the extracted data in encrypted form is decrypted. It allows user to select the file to start decryption process. Decrypted data file is stored with the name of “Data file name”+_enc+_dec. ‘Decryption.cs’ is the name of the form used for this phase. The designer and resx files are named as ‘Decryption.designer.cs’ and ‘Decryption.resx’ for this phase.

5. SYSTEM DESIGN

5.1. Input Design of the System

The user input gets converted into system based format to achieve effective data compression. Quality of system output is strongly based on the quality and format of the input. This is why each user needs to get authorized before selecting the data files as input.

Input Design is the process of changing the user based format into computer based format. If the input quality is good then the output quality will always be good and vice versa. The format and criteria used for validating the input data could be determined by the input system. It is required to include the impacts and quality of the input during the analysis.

As required, in this project, the input of the system is determined in a way how errors get rectified as little as possible. Input design of a system makes it reliable and accurate output.

5.2. Output Design of the System

The system output is as much important as the input given to the system and it is also the direct source of data or information to the users. Output design needs to be done in a good and proper manner. A system output is right if each element of output is designed properly and this ensures that people don't find difficulties in using the system. Specific outputs could be identified while designing the output as it is mandatory to fulfill the requirements.

Consistency of the output is very important and it is often verified. As we know, output determines if the system is success or not. The system usage could be determined by the appearance and user friendliness of the system. Successful output of a system is ultimate goal for many users as they use the output for evaluating the system.

5.3. System Process

The system processes the data encryption and then hiding it in an image file called cover file, extracting the embedded data from the cover file and store the data files in the destination.

6. UML Diagrams

6.1. **Use Case Diagram**

Use case diagram represents, graphical depiction, the communication process among the elements such as actors (users) and use cases of the system. Use case diagram is used in identifying, analyzing, and arranging the requirements in an understandable manner.

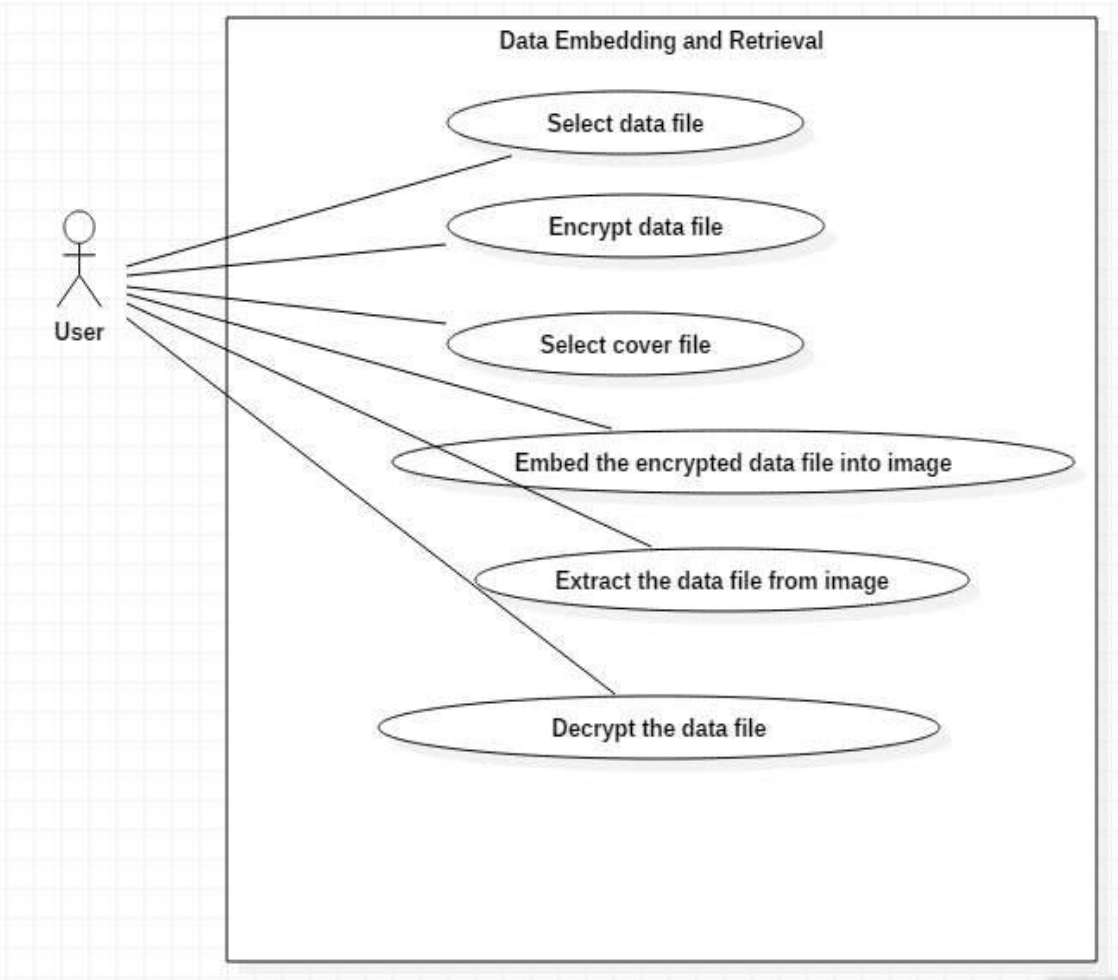


Figure 9 - Use Case Diagram

6.2. Class Diagram

Class diagram is used to represent the structure of a system and it is one of the structure static diagrams. The components of a class diagram are classes, attributes, operations and association among the objects or components. A class has attributes and operations/methods.

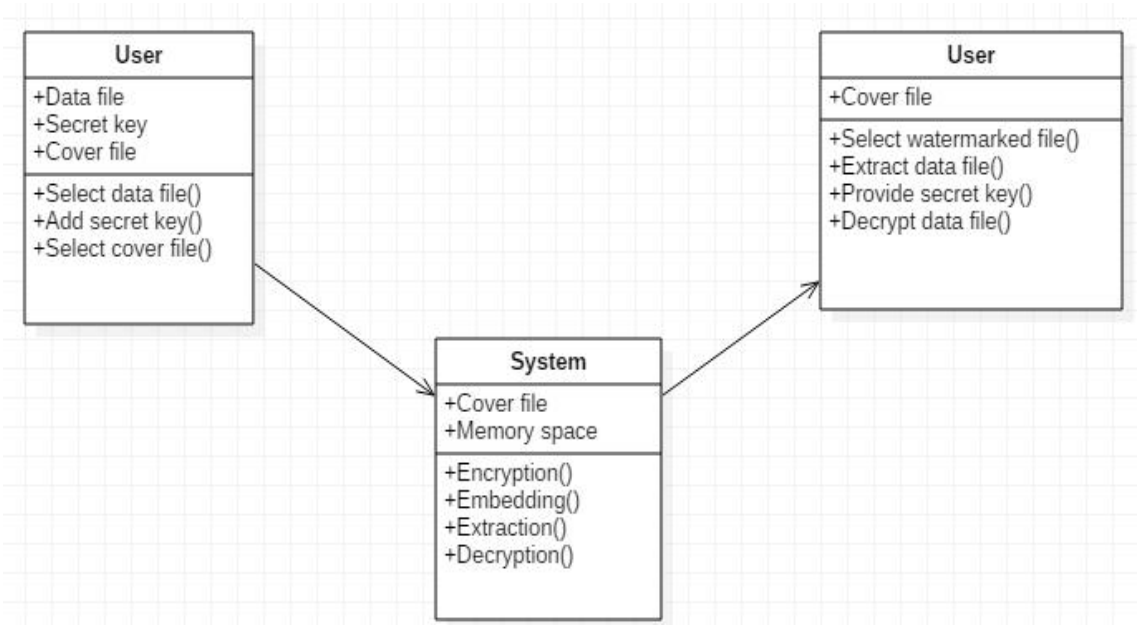


Figure 10 - Class Diagram

6.3. Sequence Diagram

Sequence diagram shows the interaction and communication from one object to another object. This interaction or communication between objects could be arranged in a sequential form. The sequence diagram for the system is shown below.

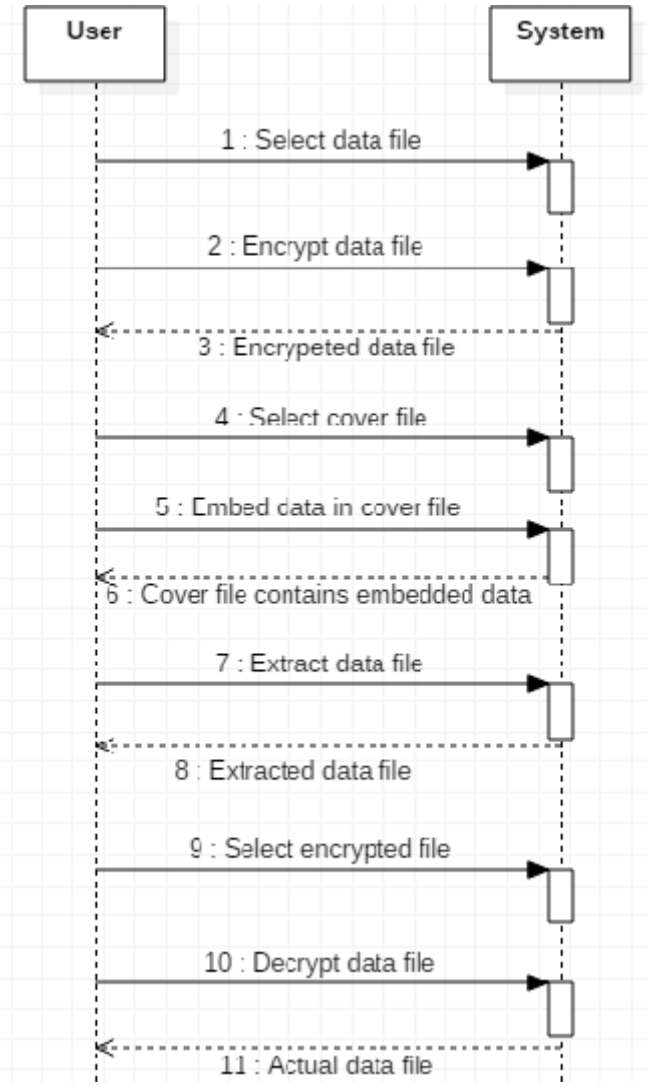


Figure 11 - Sequence Diagram

6.4. Component Diagram

Component diagram, unlike other UML diagrams, doesn't represent the functionalities of the system. It is used to describe the main components and

the dependencies among them in the system. The component diagram used for this system is shown below.

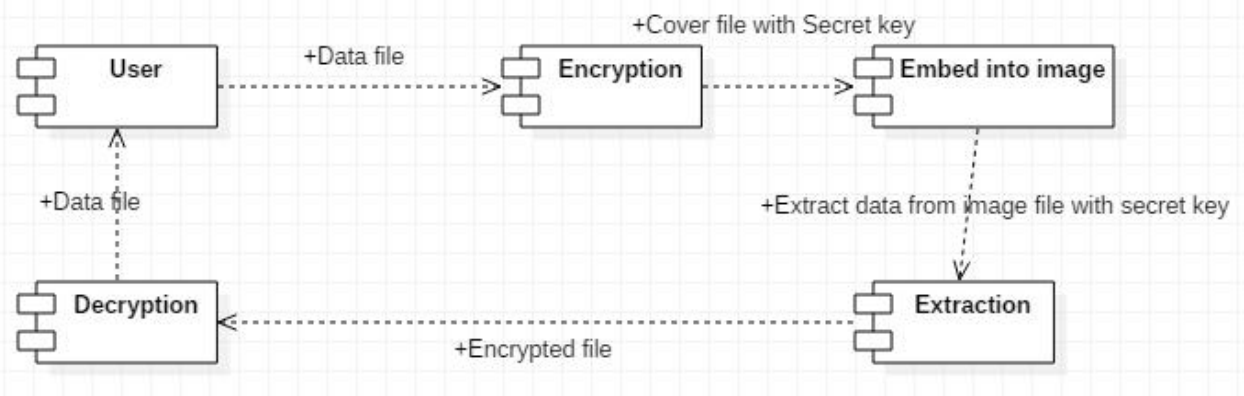


Figure 12 - Component Diagram

7. TESTING PROCESS

As mentioned earlier, the complete tasks are split into 5 different phases. They are tested at 2 levels called unit testing and system integration testing.

7.1. Unit Testing

Unit testing is done at the phase level to make sure that each and individual phase is working fine. Test scenarios are defined and results are captured at each and every phase. Unit testing is performed to ensure the data files are compatible to the pixel level, able to open the files, parameter values are passed correctly etc.

7.2. System Integration Testing

After the phase level testing (Unit Testing), the system integration testing will be done in which all the modules get integrated and tested in a single shot. System integration testing is performed to ensure the program structure is correct and interface among the modules is properly done.

8. SYSTEM REQUIREMENTS

8.1. Hardware Requirements

S. No	Hardware	Desk
1	Hard Disk	40 GB
2	RAM	512 MB
3	Processor	Pentium IV 2.16 GHz
4	Processor bits	32/64 bits

Table 1 – List of Hardware

8.2. Software Requirements

S. No	Software	Description
1	Language	C# .NET
2	OS	Windows 10
3	Database	SQL server 2016

Table 2 - List of Software

9. CONCLUSION

Cryptography has been used to achieve more data security in data embedding process. Data security could be significantly improved during data embedding process as data is encrypted using one of the AES algorithms called Rijndael algorithm. Data hiding technique uses less computational level of complexity in both data hiding and data extraction.

Data file will be encrypted before embedding it in to cover file. Quality of the cover image may be affected. Even though if the embedded image is accessed by unknown person or hackers, it is possible for the hackers to find the presence of data compressed in the image. But, it is never possible for them to extract the data. Secret key is used in data embedding process. Image is converted to matrix form using its intensity values. Data is embedded into intensity values of cover medium. As a reverse process, intensity values of the image are converted back to the image from matrix form and then extraction process is performed.

10. FUTURE ENHANCEMENT

As part enhancing this system, it could be developed with any type of cover files, not only the image file used to embed the data in the system. It can be an audio, video or any other file. Along with LB encoding data embedding technique, there can be additional techniques adopted to make data protection. This application provides data embedding without any loss of data as they are encrypted. Along with LB encoding, data embedding can be achieved with invisible data that are compressed in the image file. When data file is embedded in a cover file, it is more important that the quality of the cover file must be maintained and distortion of the file should be very low.

Also, image file or cover file should be retrieved as same as it was before data embedding in it. So, once the embedded data get extracted from the cover file then the originality of the cover file should be restored to its actual condition. Due to high bandwidth that occurs when using multi files as cover file makes delay in data transmission. This needs to be avoided in future work, so the data transmission can be done more easily and quickly, that makes the system robust and faster than the previous systems available.

11. APPENDICES

11.1. APPENDIX I – Software Description

This section provides brief information about tools, database and software used in the project.

11.1.1. .NET Framework

.NET Framework is a set of components used to create.NET platform. This .NET platform is known as .NET Framework which is type safe environment. .NET Framework handles various processes like code execution, memory allocation and permission for the resources. This.NET Framework consists of 2 major components called Common Language Runtime (CLR) and Class Libraries. .NET Framework has cross-language compatibility which allows codes written in one programming language referred in another. For example, VB.NET classes could be referred and derived from C# classes.

.NET Framework is used to handle the applications like Web applications, Web services and Windows applications. These applications are also called mufti-platform applications. .NET Framework is capable of handling various languages like C, C++, C#, COBOL, VB, Java Script, etc... .NET Framework could be accessed by all these programming languages to write, compile, debug and run the codes [21].

There are many components used in.NET Framework like Common Language Runtime, .Net Framework class library, common language

specification, common type system, Windows forms, asp.NET and asp.net ajax, ado.NET, Windows workflow foundation etc.

Once the applications are developed in.NET Framework, they will then be imported and executed. So, .NET Framework is the platform made by collection of the components listed above. .NET Framework is a type safe and managed environment for implementing and executing applications. .NET Framework is compatible with cross language concept. Cross language concept is nothing but logic that allows applications written in one language could be referred by another language. These languages must be supported by.NET Framework [21]. For example, a DLL file written in C++ can be referred by C#, like how a method in class can be accessed by another class. The above listed components are categorized in 2 major types. They are Common Language Runtime (CLR) and Class Libraries.

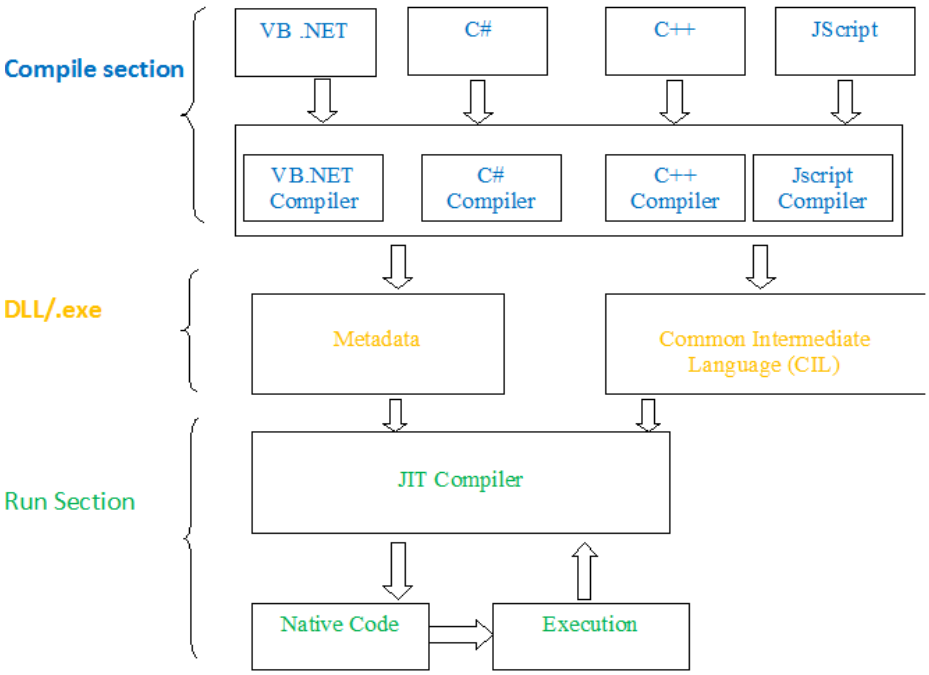


Figure 13 - Compile and execution of .NET Framework [22]

11.1.2. C#

C# is an Object Oriented Programming language and it is a product from Microsoft. C# is designed for Common Language Infrastructure (CLI) that can execute the code on various platforms. As we have seen above, .NET Framework is the platform that runs C# programs and it is used with many .NET applications. C# codes could be compiled and run on various Integrated Development Environment (IDE) such as Visual Studio (VS), Visual C# Express (VCE) and Visual Web Developer. Among all these 3 IDEs Visual C# Express (VCE) and Visual Web Developer could be downloaded from Microsoft website for free. In this project, Visual Studio 2012 has been used as IDE.

C# source codes could be written using normal editors like notepad, edit plus etc. The codes written in editors could be compiled in command line compiler which belongs to .NET Framework. .NET Framework could be supported by various operating systems like Windows, LINUX/UNIX and Mac. Mono is one of the .NET Framework versions that are an open source, containing in-built C# compiler. Mono could be run on various operating systems such as iOS, Linux/UNIX, Android, BSD, Windows, OS X, Solaris, and UNIX [23].

C# has several classes, methods and types like Java programming language and it has oops concepts like inheritance, polymorphism, encapsulation, package, types and arrays. Iterate methods are used to allow types to iterate with their elements. C# introduces several language extensions,

including Generics, Anonymous Methods, Iterators, Partial Types, and Nullable Types.

11.1.3. SQL Server 2016

SQL server is a successful Relational Database Management System (RDBMS) from Microsoft. SQL Server has been published in multiple editions. They are majorly categorized as principle and specialized editions. SQL Server is platform independent and GUI and command based. SQL Server could be implemented in client-server model. So, it could be installed in both server and workstation [24].

SQL has 4 main architectures that are general, Memory, log file and data file architectures. In theses architectures, there are many components used. They are client, query, logical units, protocols, server, relational engine, query parser and compiler, query optimizer, execution plan, query executor, storage engine, SQL operating system and checkpoint process etc.

11.2. APPENDIX II – SQL Table and Structure

This section provides the SQL Server tables used to store the file details and secret keys. Table is named as “File_Details” and it is created under the database named as “ImageSteganography”.

11.2.1. Structure of File_Details Table

Attribute Name	Data Type	Null/Not Null
File_Id	Integer	Null
Cover_File_Name	Varchar (5000)	Null
Cover_File_Length	Bigint	Not Null
Secret_Key	Varchar (5000)	Not Null
Data_File_Path	Varchar (5000)	Null
Data_File_Length	Bigint	Null

Table 4 - Structure of File_Details Table

```
CREATE TABLE [dbo].[File_Details](
                    [File_Id] [int] IDENTITY(1,1) NOT NULL,
                    [Cover_File_Name] [varchar](5000) NULL,
                    [Cover_File_Length] [bigint] NULL,
                    [Secret_Key] [varchar](5000) NULL,
                    [Data_File_Path] [varchar](5000) NULL,
                    [Data_File_length] [bigint] NULL,
) ON [PRIMARY];
```

12. REFERENCES

[1] Hock-Chuan Chua., "***HTTP Security with SSL***", Retrieved January 28, 09, 2017 http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/http_ssl.html

[2] "***Introduction to Cryptography***", Retrieved November 26, 2016, from https://gpgtools.tenderapp.com/kb/how-to/introduction-to-cryptography

[3] Suresh Gawande. Rakhi., "***A Review on Steganography Methods***", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering., vol. 2, issue. 10, October 2013.

[4] Tahernezhadi Mansour., "***Special Topics in Electrical Engineering***", Retrieved January 28, 09, 2017, from http://studentweb.niu.edu/9/~z172699/Description.html

[5] Kumar Sushil., Muttoo S.K., "***A Comparative Study of Image Steganography in Wavelet Domain***", International Journal of Computer Science and Mobile Computing, vol. 2, pp. 91 – 101, February 2013.

[6] Grover Nidhi ., Mohapatra A. K., "***Digital Image Authentication Model Based on Edge Adaptive Steganography***", IEEE Advanced Computing, Networking and Security (ADCONS), December 2013.

[7] Ullagaddi, Vishwanath., "***Development of Data Encryption Algorithms for Secure Communication***", thesis submitted in MS program, 2012.

[8] Trapani, Gina., "***Geek to Live: Hide data in files with easy steganography tools***", Retrieved October, 09, 2016, from http://lifehacker.com/230915/geek-to-live--hide-data-in-files-with-easy-steganography-tools

[10] McLoone M., McCanny J. "***High Performance Single-Chip FPGA Rijndael Algorithm Implementations***"., Lecture Notes in Computer Science, vol 2162, 2001.

[11] Trenholme Sam., "***AES' Galois field***" Retrieved January 27, 09, 2017, from http://www.samiam.org/galois.html

[12] "***RIJNDAEL Advanced Encryption Standard***", Retrieved January 26, 2017, from http://www.cs.mcgill.ca/~kaleigh/computers/crypto_rijndael.html

[13] Daemen Joan., Rijmen Vincent***., "The Design of Rijndael: AES - The Advanced Encryption Standard",*** Springer-Verlag Berlin Heidelberg***.,*** *Vol. 1, pp. 101-122, 2002****.***

[14] Majzoub Sohaib., Diab Hassan., ***"Reconfigurable Systems for Cryptography and Multimedia Applications, Data Acquisition Applications",*** InTech, DOI: 10.5772/30186, 23 August, 2012.

[15] Roy Sangita., Singh Avinash., Parida Jyotirmayee., Sairam Ashok., "***Audio Steganography Using LSB Encoding Technique with Increased Capacity and Bit Error Rate Optimization***", Vol. 1, pp. 372-375, October 2012.

[16] Neeta Deshpande ., Snehal Kamalapur ., Jacobs Daisy ., "***Implementation of LSB Steganography and Its Evaluation for Various Bits***", Digital Information Management, 2006 1st International Conference., 6 December, 2006.

[17] Champakamala B.S, Padmini K, Radhika D. K., "***Least Significant Bit algorithm for image steganography***", International Journal of Advance Computer Technology, Vol. 3, pp. 34-38, 2012.

[18] Olanrewaju R.F., Khalifa Othman., Binti Husna., "***Increasing the Hiding Capacity of Low-Bit Encoding Audio Steganography Using a Novel Embedding Technique***", World Applied Sciences Journal 20., 2012.

[19] Miller Aaron., "***Least Significant Bit Embeddings: Implementation and Detection***", Retrieved January 25, 09, 2017 from http://www.aaronmiller.in/thesis/

[20] "***Overview of the .NET Framework***", Retrieved November 29, 2016, from https://msdn.microsoft.com/en-us/library/zw4w595w(v=vs.110).aspx

[21] Dajbych, V,. "***Overview of the .NET Framework***", Retrieved November 23, 2016, from http://www.codeproject.com/Articles/680100/Overview-of-the-NET-Framework

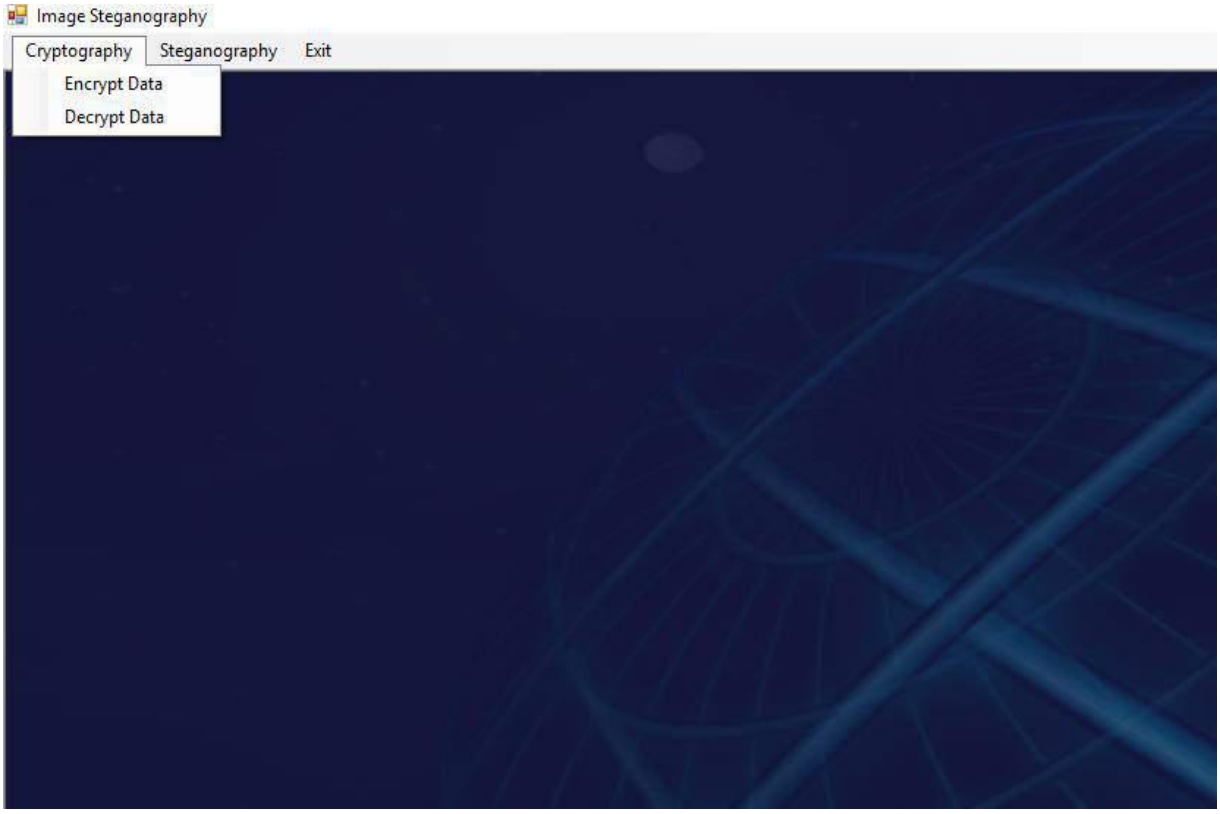
[22] Tsvetomir Y. Todorov,. "***Understanding .NET Just-In-Time Compilation***", Retrieved November 26, 2016, from http://www.telerik.com/blogs/understanding-net-just-in-time-compilation

[23] "***C# Tutorial***", Retrieved November 26, 2016, from http://www.tutorialspoint.com/csharp/

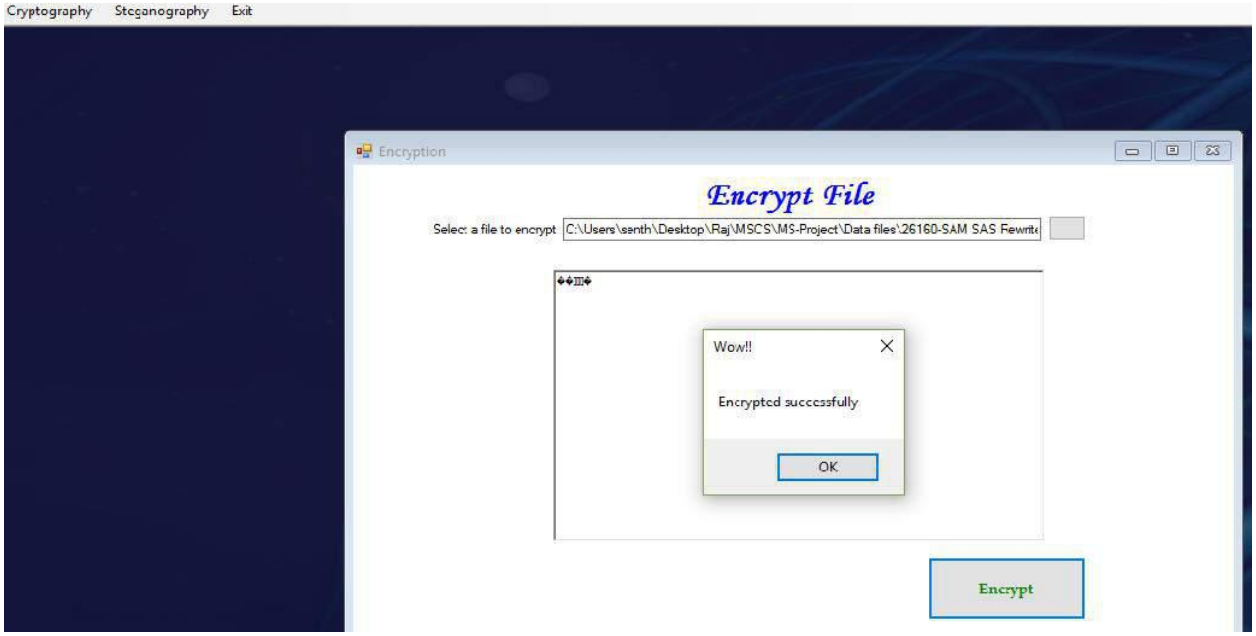
[24] "***SQL Server Technical Documentation***", Retrieved November 26, 2016, from https://msdn.microsoft.com/en-us/library/ms130214.aspx

13. SCREEN SHOTS

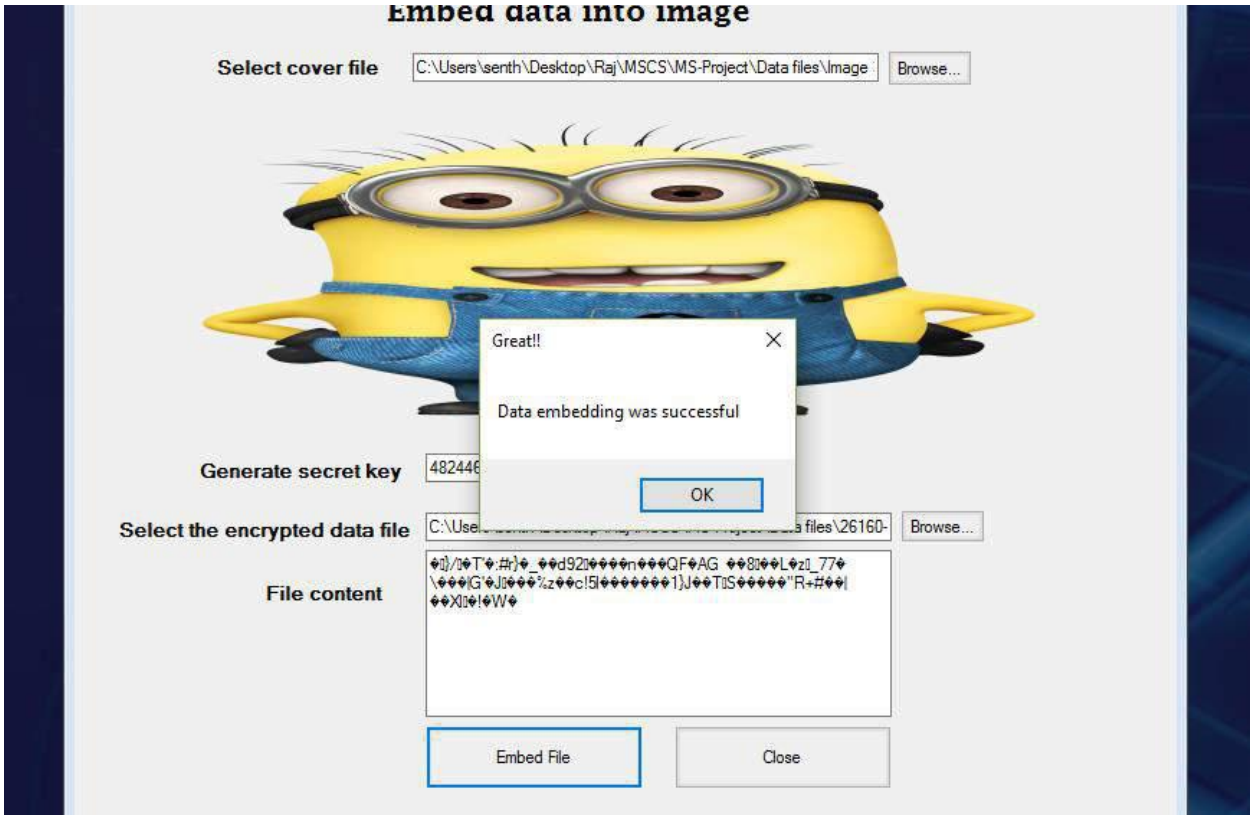
Main Window



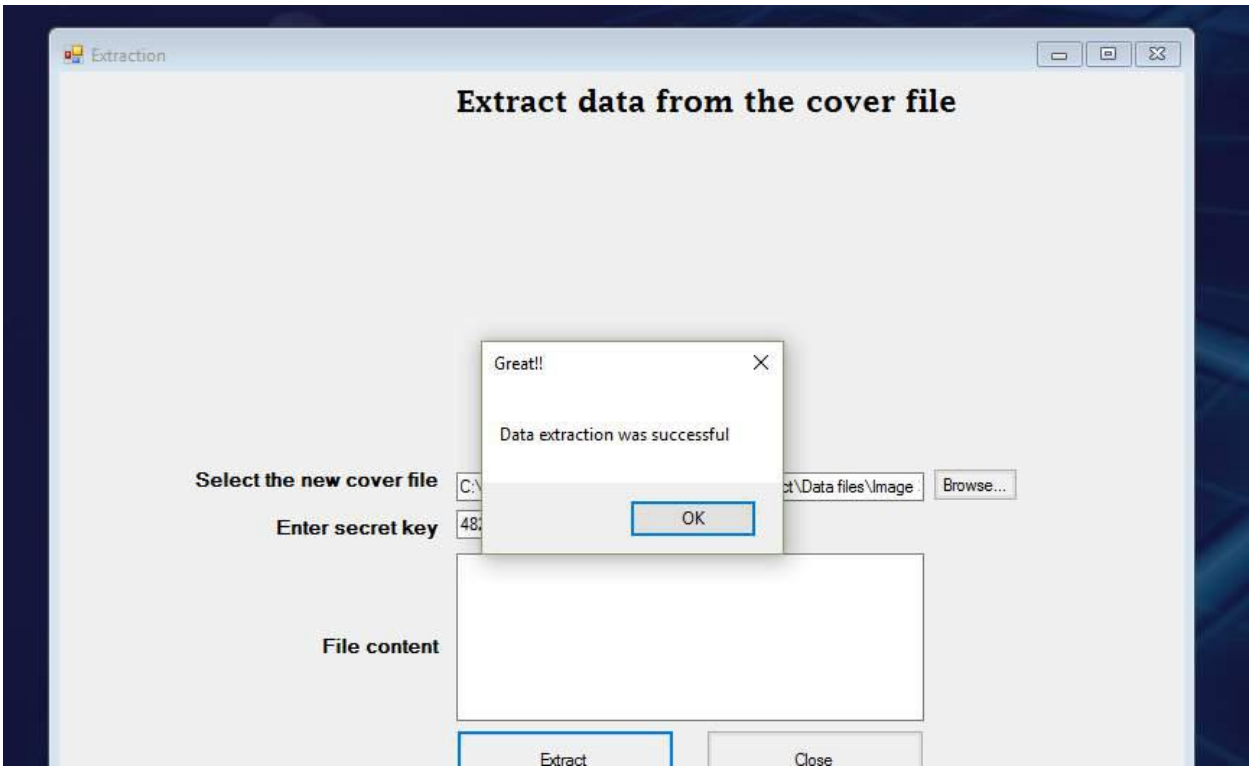
Encrypt Data Window



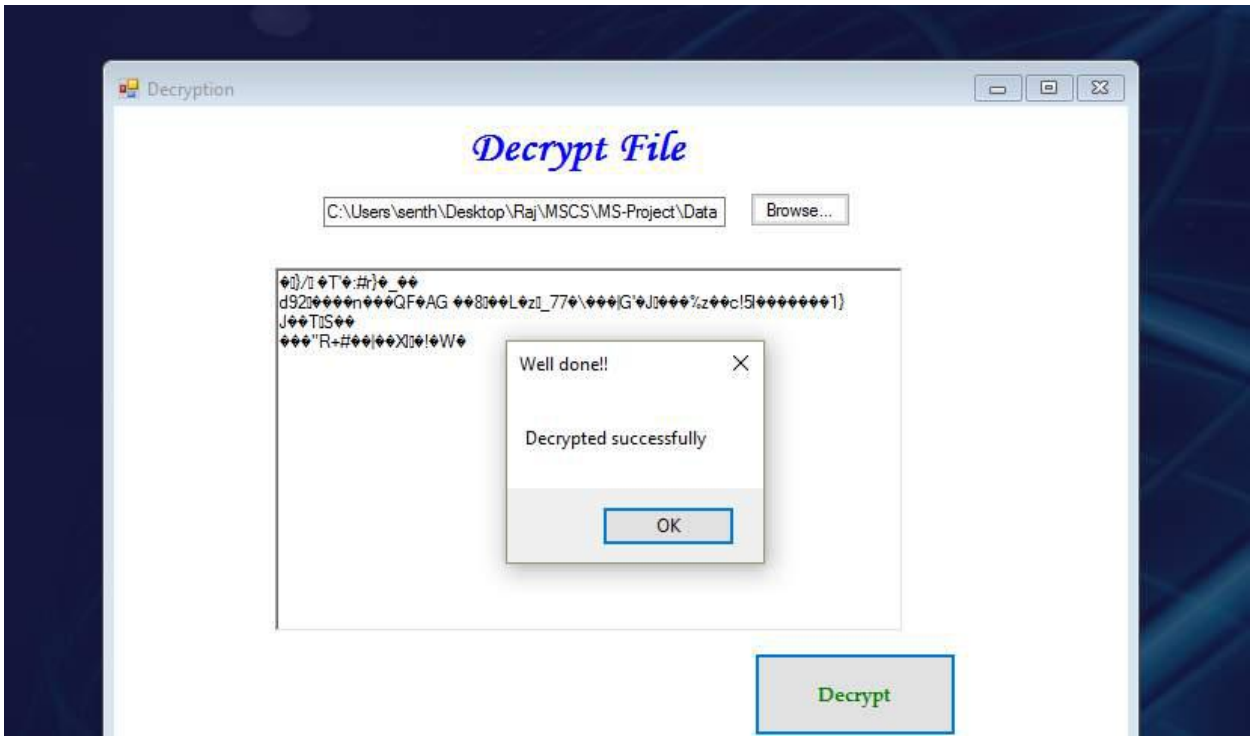
Embed Data Window



Extract Data Window



Decrypt Data Window



Files Processed