

Configuration Manual

MSc Research Project
Cyber Security

Dhruvesh Parekh
Student ID: X20182457

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland
MSc Project Submission Sheet
School of Computing



Student Name: Dhruvesh Parekh
Student ID: X20182457
Programme: MS in Cyber Security **Year:** 2021-2022
Module: Research Project
Lecturer: Vikas Sahni
Submission Due Date: 26/04/2022
Project Title: Using AES Encryption to Securely Embed Data in Video Files.
Word Count: 734 **Page Count:** 09

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Dhruvesh Parekh

Date: 26/06/2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Dhruvesh Parekh
Student ID: X20182457

1 Introduction

The configuration manual would discuss the research study, the key methodologies, evaluations, and implementations that occurred. It presents a revolutionary method for concealing data within a digital medium by combining video steganography and the AES algorithm. A Java GUI code is used to allow the user to encrypt and decrypt data over video files. To conduct the encryption, the AES encryption technique takes the plain text and encrypts it using a key size of AES-256 bits. A salt function is used in conjunction with AES encryption to create ciphertext with double security. The video files are divided into image frames, and then the LSB technique is used to hide data in the image frames. The key benefit of using LSB as an embedding technique over other approaches is that it can embed huge amounts of data with minimal distortion.

2 System Configuration

2.1 Hardware Configuration

Hardware	Configuration
Processor	Intel i5
Ram	8Gb
Hard disk	512Gb

2.2 Software Configuration

Software	Configuration
Operating System	Windows 10
Tools	Apache NetBeans 12
Scripting language	Java
Scripting language Version	Java 15

3 Implementation

After installing JDK 15 and NetBeans 12.0, the NetBeans IDE is used to build the application. For this, open NetBeans, File → New Project → Java Project → Provide project name → Finish.

The purpose of this project to build a secure communication between two person using combination of cryptography and steganography.

To run the project:

1. Open NetBeans IDE
2. File → Open → Project Name (Video Steganography)

Once the project is imported and executed the home page of the application opens as shown in figure 1.



Figure 1: Main Page

As shown in figure 2 the code snippet is meant for encryption of file.

```
byte[] buffer = new byte[262144];
int bufferSize=buffer.length;
int keySize=key.length();
progressOfFilesTextField.setText(progressOfFilesTextField.getText()+" 0%\n");
while(fileReader.available()>0)
{
    int bytesCopied=fileReader.read(buffer);
    for(int i=0,keyCounter=0; i<bufferSize; i++, keyCounter%=keySize )
    {
        buffer[i]+=key.toCharArray()[keyCounter];
    }

    fileWriter.write(buffer, 0, bytesCopied);
    long fileLength=file.length();
    percentageOfFileCopied+= ((double)bytesCopied/fileLength)*100;
    showProgressOnprogressOfFilesTextField(progressOfFilesTextField,
        percentageOfFileCopied, bytesCopied, fileLength);

    showProgressOnProgressBarAndProgressPercent(progressBar,
        progressPercentLabel, bytesCopied, totalSizeOfAllFiles);

    System.out.println("aes file length= "+tempo.length());
}
```

Figure 2: Encryption Process

Figure 3 shows the file content after the encryption of file.

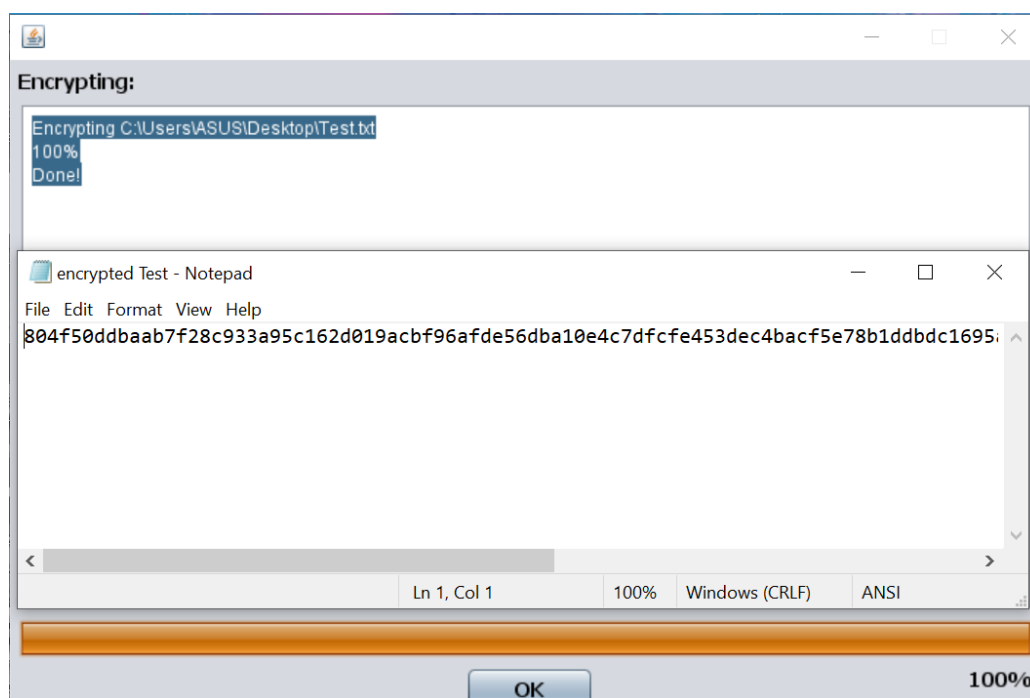


Figure 3: Encrypted file

As shown in figure 4 the code snippet is meant for decryption of file.

```
byte[] buffer = new byte[262144];
int bufferSize = buffer.length;
int keySize = key.length();
progressOfFilesTextField.setText(progressOfFilesTextField.getText()+" 0%\n");
for(int i=0; i<128; i++)
{
    if(fileReader.available(>0)
    {
        fileReader.read();
    }
}
while(fileReader.available(>0)
{
    int bytesCopied=fileReader.read(buffer);
    for(int i=0,keyCounter=0; i<bufferSize; i++, keyCounter%=keySize )
    {
        buffer[i]-=key.toCharArray()[keyCounter];
    }

    fileWriter.write(buffer, 0, bytesCopied);
    long fileLength=file.length();
    percentageOfFileCopied+= (((double)bytesCopied/fileLength)*100);
    showProgressOnprogressOfFilesTextField(progressOfFilesTextField, percentageOfFileCopied,
        bytesCopied, fileLength);

    showProgressOnProgressBarAndProgressPercent (progressBar, progressPercent,
        bytesCopied, totalSizeOfAllFiles);
    System.out.println("aes file length= "+tempo.length());
}
```

Figure 4: Decryption process

Figure 5 shows the file content after the decryption of file.

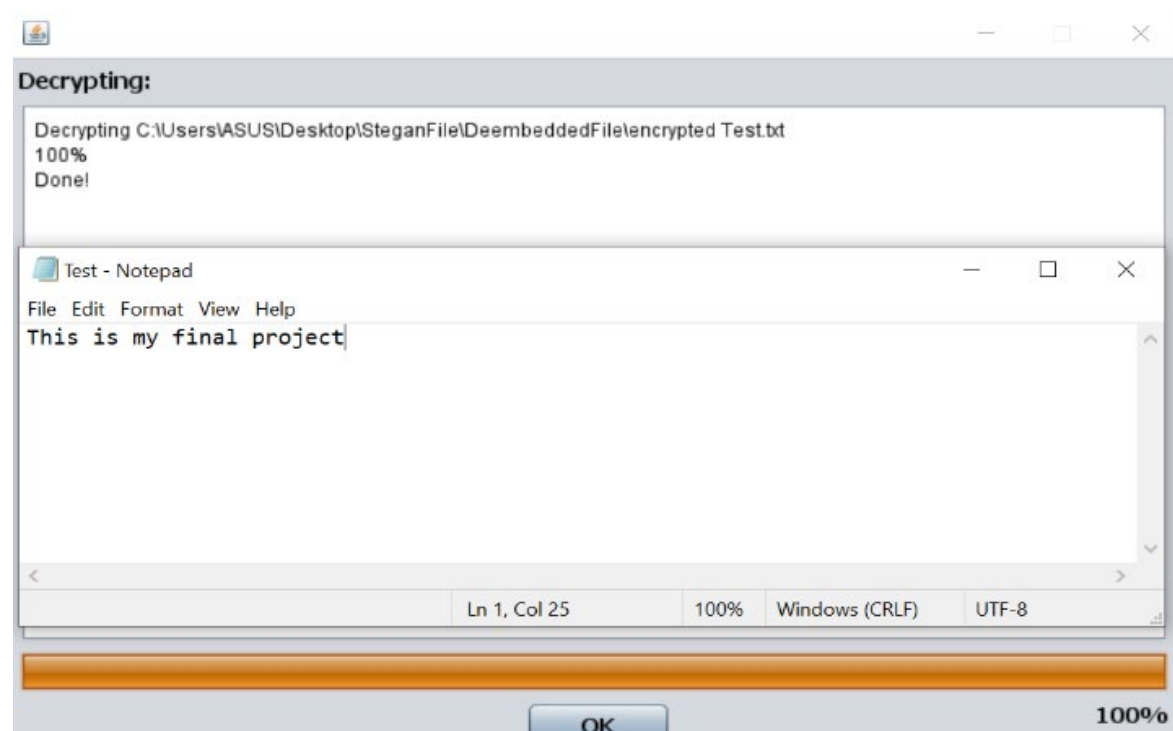


Figure 5: Decrypted file

As presented in figure 6 & 8 is code snippet of Stegnography process.

```
public String emb(String s, String sl)
{
    try{
        File file = new File(s);
        File file1 = new File(sl);
        String des = System.getProperty("user.home")+"/Desktop/SteganFile/";
        File fi=new File(des, "EmbeddedFile");
        fi.mkdirs();
        File tempo = new File(fi , file.getName());
        if(tempo.exists()){
            tempo.delete();
        }
        FileInputStream fileinputstream = new FileInputStream(s);
        FileOutputStream fileoutputstream = new FileOutputStream(tempo);
        byte abyte0[] = new byte[8];
        int i;
        int k;
        for(k = 0; (i = fileinputstream.read(abyte0, 0, 8)) > 0; k = i)
            fileoutputstream.write(abyte0, 0, i);
        fileinputstream.close();
        for(int l = 1; l <= 8 - k; l++)
            fileoutputstream.write(65);

        fileoutputstream.write("DATAFILE".getBytes(), 0, 8);
        System.out.println("File name===="+file1.getName());
        StringBuilder stringbuffer = new StringBuilder(file1.getName());
        stringbuffer.setLength(50);
        fileoutputstream.write(stringbuffer.toString().getBytes(), 0, 50);
        fileinputstream = new FileInputStream(sl);
        int j;
        while((j = fileinputstream.read(abyte0, 0, 8)) > 0)
            fileoutputstream.write(abyte0, 0, j);
        fileinputstream.close();
        fileoutputstream.close();
        embfilename = tempo.toString();
    }
    catch(IOException e){
        embfilename="";
    }
    return embfilename;
}
```

Figure 6: Embedding process

Below figure shows embedding the encrypted file into a video file

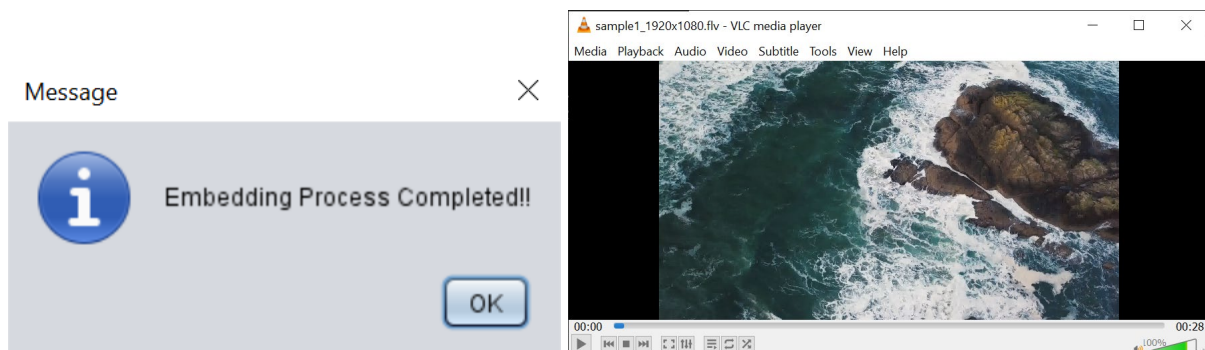


Figure 7: Embedding file

```

public String demb(String s)
{
    boolean flag;
    String demfile = "";
    try{
        File file = new File(s);
        String outpath=s.substring(0, s.lastIndexOf("\\")+1);
        FileInputStream fileinputstream = new FileInputStream(s);
        char c = '\b';
        byte abyte0[] = new byte[c];
        String s1 = "";
        int i;
        while((i = fileinputstream.read(abyte0, 0, c)) > 0) {
            s1 = new String(abyte0);
            if(s1.equals("DATAFILE"))
                break; }
        if(!s1.equals("DATAFILE")){
            flag=false;
            fileinputstream.close();
            return demfile; }
        abyte0 = new byte[50];
        fileinputstream.read(abyte0, 0, 50);
        s1 = new String(abyte0);
        String s2 = s1.trim();
        String fpath = s2;
        System.out.println("fpath-----"+fpath);
        String des = System.getProperty("user.home")+"/Desktop/SteganFile/";
        File fi=new File(des, "DeembeddedFile");
        fi.mkdirs();
        File filedem = new File(fi,fpath);
        if(filedem.exists()){
            filedem.delete(); }
        filedem.toString();
        FileOutputStream fileoutputstream = new FileOutputStream(filedem);
        c = '\u5000';
        abyte0 = new byte[c];
        while((i = fileinputstream.read(abyte0, 0, c)) > 0)
            fileoutputstream.write(abyte0, 0, i);
        fileinputstream.close();
        fileoutputstream.close();
        demfile=fi.toString()+File.separator+fpath;
    }
}

```

Figure 8: De-Embedding process

Below process is executed for de-embedding a video file.

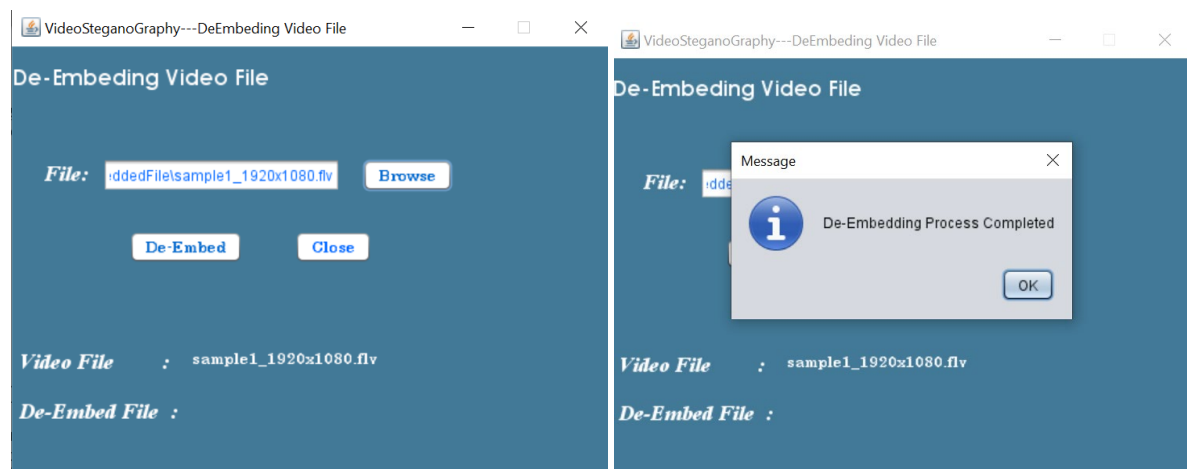


Figure 9: De-Embedding file

4 References

- [1] Upasana, "How To Install Java 12 On Windows 10," edureka.co, 03 07 2019. [Online]. Available: <https://www.edureka.co/blog/install-java-on-windows/>. [Accessed 15 04 2022].
- [2] R. Saive, "How to Install NetBeans IDE 12 in Debian, Ubuntu and Linux Mint," tecmint, 20 06 2020. [Online]. Available: <https://www.tecmint.com/install-netbeans-ide-in-ubuntu-debian-linux-mint/>. [Accessed 19 04 2022].
- [3] P. Krill, "JDK 12: The new features in Java 12," infoworld, 19 03 2019. [Online]. Available: <https://www.infoworld.com/article/3301197/jdk-12-the-new-features-in-java-12.html>. [Accessed 02 04 2022].