**CHAPTER 9**

**APPENDICES**

**A.1 Sample Coding**

**Cryptor**

package video;

import java.io.\*;

import javax.crypto.spec.\*;

import java.util.\*;

import javax.crypto.\*;

import java.security.spec.\*;

import java.math.\*;

import java.security.\*;

public class Cryptor{

static {

Security.addProvider(new com.sun.crypto.provider.SunJCE());

}

private Cipher cipher;

private PBEParameterSpec paramSpec;

private PBEKeySpec keySpec;

private KeyGenerator keygen;

private SecretKey secretKey;

public String algorithm;

public String password;

private int encryptTechnique;

private byte[] key; // for Triple DES

IvParameterSpec IvParameters = new IvParameterSpec(

new byte[] { 12, 34, 56, 78, 90, 87, 65, 43 });

// Salt

byte[] salt = {(byte)0xc7,(byte)0x73,(byte)0x21,(byte)0x8c,(byte)0x7e,(byte)0xc8,(byte)0xee,(byte)0x99};

// Iteration count

int count = 20;

String outputFileName;

public Cryptor(){

setAlgorithm(1);

}

public void setAlgorithm(int choice){

encryptTechnique = choice;

switch(choice){

case 1: algorithm = new String("PBEWithMD5AndDES"); //DES

break;

case 2: algorithm = new String("DESede/CBC/PKCS5Padding"); //Triple DES

break;

default: algorithm = new String("PBEWithMD5AndDES"); //DES

break;

}

}

private void InitiateCipher(int mode){

try{

if(encryptTechnique!=2){ // if not triple des

paramSpec = new PBEParameterSpec(salt, count);

keySpec = new PBEKeySpec(password.toCharArray());

secretKey = SecretKeyFactory.getInstance(algorithm).generateSecret(keySpec);

}

else{

key = toKey(password);

// Create a DESede key spec from the key

DESedeKeySpec spec = new DESedeKeySpec( key );

// Get the secret key factor for generating DESede keys

SecretKeyFactory keyFactory = SecretKeyFactory.getInstance("DESede");

secretKey = keyFactory.generateSecret(spec);

}

cipher = Cipher.getInstance(algorithm);

if(encryptTechnique==2)

cipher.init(mode, secretKey, IvParameters);

else

cipher.init(mode, secretKey, paramSpec);

}

catch(Throwable t) {t.printStackTrace();}

}

private String getExtension(String fileName){

return fileName.substring(fileName.lastIndexOf("."));

}

int encrypt(String inputFileName, String password){

this.password = password;

try{

// JOptionPane.showMessageDialog(null,"algorithm: "+algorithm);

InitiateCipher(Cipher.ENCRYPT\_MODE);

FileInputStream fin = new FileInputStream(inputFileName);

BufferedInputStream in = new BufferedInputStream(fin);

outputFileName = new String("C:\\Windows\\Temp\\tempe") + getExtension(inputFileName);

// String outputFileName = new String("tempe") + getExtension(inputFileName);

FileOutputStream fout = new FileOutputStream(outputFileName);

CipherOutputStream out = new CipherOutputStream(fout, cipher);

int buffer;

while ((buffer = in.read()) != -1)

out.write(buffer);

out.close();

in.close();

return 0;

}

catch(Exception ex){

return 1;

}

}

int decrypt(String inputFileName, String dstPathName, String password){

this.password = password;

try{

InitiateCipher(Cipher.DECRYPT\_MODE);

FileInputStream fin = new FileInputStream(inputFileName);

CipherInputStream in = new CipherInputStream(fin, cipher);

outputFileName = new String(dstPathName + "\\tempd") + getExtension(inputFileName);

// String outputFileName = new String("tempd") + getExtension(inputFileName);

FileOutputStream fout = new FileOutputStream(outputFileName);

BufferedOutputStream out = new BufferedOutputStream(fout);

int buffer;

while ((buffer = in.read()) != -1)

out.write(buffer);

in.close();

out.close();

return 0;

}catch(Exception ex){

return 1;

}

}

public String getOutputFileName(){

return outputFileName;

}

public byte[] toKey(String password){

int pwdlen = password.length();

int i=pwdlen;

while(i < 25){

password = password + password;

i = password.length();

}

byte[] encryptKey = password.getBytes();

return encryptKey;

}

}

**Decryptor**

package video;

import java.io.\*;

import java.math.BigInteger;

public class Decryptor {

private BigInteger d;

private BigInteger n;

private FileOutputStream fout;

private DataInputStream fin;

private String outputFileName;

private String getExtension(String fileName){

return fileName.substring(fileName.lastIndexOf("."));

}

public String getOutputFileName(){

return outputFileName;

}

public void setD(String dVal) {

d = new BigInteger(dVal);

}

public void setN(String nVal) {

n = new BigInteger(nVal);

}

public boolean setSrc(String finStr) {

try {

fin = new DataInputStream(new FileInputStream(finStr));

} catch (FileNotFoundException ex) {

return false;

} catch (IOException ex) {

return false;

}

return true;

}

public boolean setDst(String foutStr) {

try {

fout = new FileOutputStream(foutStr);

} catch (FileNotFoundException ex) {

return false;

}

return true;

}

public boolean decrypt(String src, String dVal, String nVal) {

setD(dVal);

setN(nVal);

setSrc( src );

outputFileName = new String("C:\\Windows\\Temp\\tempd") + getExtension(src);

setDst( outputFileName );

try {

BigInteger input;

BigInteger output;

int data;

while (fin.available()>0) {

data = fin.readShort();

input = new BigInteger(String.valueOf(data));

output = input.modPow(d, n);

fout.write(output.intValue());

}

fout.close();

} catch(EOFException ex){

} catch (IOException ex) {

}

return true;

}

}

**Encryptor**

package video;

import java.io.\*;

import java.math.BigInteger;

public class Encryptor {

private BigInteger n;

private BigInteger d;

private BigInteger e;

private FileInputStream fin;

private DataOutputStream fout;

private String outputFileName;

public Encryptor() {

}

void setE(String eVal) {

e = new BigInteger(eVal);

}

BigInteger getE() {

return e;

}

void setN(String nVal){

n = new BigInteger(nVal);

}

boolean setSrc(String finStr) {

try {

File f = new File(finStr);

if(!f.exists()) // file does not exists

return false;

fin = new FileInputStream(f);

} catch (FileNotFoundException ex) {

return false;

}

return true;

}

boolean setDst(String foutStr) {

try {

fout = new DataOutputStream(new FileOutputStream(foutStr));

} catch (FileNotFoundException ex) {

return false;

} catch (IOException ex) {

return false;

}

return true;

}

private String getExtension(String fileName){

return fileName.substring(fileName.lastIndexOf("."));

}

public String getOutputFileName(){

return outputFileName;

}

int encrypt(String src, String eVal, String nVal) {

BigInteger data;

BigInteger cipher;

setE(eVal);

setN(nVal);

setSrc( src );

//JOptionPane.showMessageDialog(null,src);

outputFileName = new String("C:\\Windows\\Temp\\tempe") + getExtension(src);

setDst( outputFileName );

try {

while (fin.available() > 0) {

int input;

input = fin.read();

data = new BigInteger(String.valueOf(input));

cipher = data.modPow(e, n);

fout.writeShort(cipher.intValue()) ;

}

fin.close();

fout.close();

} catch (IOException ex) {

return 0;

}

return 1;

}

}

**Hide**

package video;

import javax.swing.\*;

public class Hide{

HidePanel parent;

String msg;

String secretFileName;

String coverFileName;

String dstFileName;

String password;

String eVal;

String nVal;

int encryptTechnique;

// Splash splashScreen;

boolean isConverted = false;

boolean isHideFile = true;

int coverType=3;

int BMP = 1, MEDIA= 2, OTHER=3;

//ConvertToBMP c;

public static native int hideInBMP(String secretFile, String coverFile, String dstFile, String Password);

public static native int hideInMEDIA(String secretFile, String coverFile, String dstFile, String password);

public static native int hideInOTHER(String secretFile, String coverFile, String dstFile, String password);

private Object FileParser;

public Hide(HidePanel parent){

this.parent = parent;

}

public void hideUsingRSA(String msg, String secretFileName, String coverFileName, String dstFileName, String eVal, String nVal){

this.msg = msg;

this.secretFileName = secretFileName;

this.coverFileName = coverFileName;

this.dstFileName = dstFileName;

this.eVal = eVal;

this.nVal = nVal;

parent.encryptTechnique = 3;

if(!verifyData())

return;

// splashScreen = new Splash(2); // Processing.jpg

// Encrypt the File

Encryptor cryptor = new Encryptor();

if(cryptor.encrypt(this.secretFileName, eVal, nVal) == 0){

JOptionPane.showMessageDialog(parent,"Invalid Secret File",

"Error", JOptionPane.ERROR\_MESSAGE);

parent.secretField.setText("");

return;

}

// New Encrypted File

this.secretFileName = cryptor.getOutputFileName();

// create password for Stego

password = nVal;

// Now Hide the Data

hide();

}

public void hideUsingPassword(String msg, String secretFileName, String coverFileName, String dstFileName, String password, int encryptTechnique){

this.msg = msg;

this.secretFileName = secretFileName;

this.coverFileName = coverFileName;

this.dstFileName = dstFileName;

this.password = password;

this.encryptTechnique = encryptTechnique;

if(!verifyData())

return;

Splash splashScreen = new Splash(2); // Processing.jpg

try{

Thread.sleep(200);

}catch(Exception ex){

}

// Encrypt the File

Cryptor cryptor = new Cryptor();

cryptor.setAlgorithm( encryptTechnique );

if(cryptor.encrypt(this.secretFileName, password) == 1){

JOptionPane.showMessageDialog(parent,"Invalid Secret File",

"Error", JOptionPane.ERROR\_MESSAGE);

parent.secretField.setText("");

return;

}

// New Encrypted File

this.secretFileName = cryptor.getOutputFileName();

// Now Hide the Data

hide();

}

// It verifies given data and covert Images to BMP

private boolean verifyData(){

if(!secretFileName.equals("") && !msg.equals("")){

JOptionPane.showMessageDialog(parent,"You are trying to hide both Secret File and Secret Message"+" \nStegoStick is hiding Secret File only",

"Warning",

JOptionPane.WARNING\_MESSAGE);

}

if(secretFileName.equals("")){

// isHideFile = false;

if(msg.equals("")){

JOptionPane.showMessageDialog(parent,"Secret File Name or Message not Entered",

"Error", JOptionPane.ERROR\_MESSAGE);

return false;

}

// Convert message to file.XXXX

// secretFileName = FileParser.toFile(msg);

}

if(coverFileName.equals("")){

JOptionPane.showMessageDialog(parent,"Cover File Name not Entered",

"Error", JOptionPane.ERROR\_MESSAGE);

return false;

}

if(dstFileName.equals("")){

JOptionPane.showMessageDialog(parent,"Destination File Path not Entered",

"Error", JOptionPane.ERROR\_MESSAGE);

return false;

}

// if(parent.encryptTechnique != SelectionPanel.RSA){

if(password.equals("")){

JOptionPane.showMessageDialog(parent,"password not Entered",

"Error", JOptionPane.ERROR\_MESSAGE);

return false;

// }

}

else{

if(eVal.equals("")){

JOptionPane.showMessageDialog(parent,"E Value not Entered",

"Error", JOptionPane.ERROR\_MESSAGE);

return false;

}

if(nVal.equals("")){

JOptionPane.showMessageDialog(parent,"N Value not Entered",

"Error", JOptionPane.ERROR\_MESSAGE);

return false;

}

}

// If image is not BMP image convert to temporary BMP image for Hiding

try{

if((coverFileName.substring(coverFileName.indexOf('.')).equalsIgnoreCase(".jpg")) ||

(coverFileName.substring(coverFileName.indexOf('.')).equalsIgnoreCase(".jpeg")) ||

(coverFileName.substring(coverFileName.indexOf('.')).equalsIgnoreCase(".gif")) ||

coverFileName.substring(coverFileName.indexOf('.')).equalsIgnoreCase(".bmp")){

// if(!c.convertToBMP(coverFileName)){

// JOptionPane.showMessageDialog(parent,"Invalid Cover File",

// "Error", JOptionPane.ERROR\_MESSAGE);

// parent.coverField.setText("");

// return false;

// }

isConverted = true;

coverFileName = new String("C:\\WINDOWS\\Temp\\temp.avi");

coverType = BMP;

}

else if(coverFileName.substring(coverFileName.indexOf('.')).equalsIgnoreCase(".wav") ||

coverFileName.substring(coverFileName.indexOf('.')).equalsIgnoreCase(".avi") ||

coverFileName.substring(coverFileName.indexOf('.')).equalsIgnoreCase(".mpg") ||

coverFileName.substring(coverFileName.indexOf('.')).equalsIgnoreCase(".mpeg") )

coverType = MEDIA;

else

coverType = OTHER;

}catch(StringIndexOutOfBoundsException ex){

JOptionPane.showMessageDialog(parent, "Invalid Cover File","Error",

JOptionPane.ERROR\_MESSAGE);

parent.coverField.setText("");

return false;

}

return true;

}

private void hide(){

int result = 0;

String outFileName="";

// JOptionPane.showMessageDialog(null,"Hiding "+secretFileName);

try{

System.loadLibrary("StegBMP");

System.loadLibrary("StegMEDIA");

System.loadLibrary("StegOTHER");

}catch(Exception ex){

JOptionPane.showMessageDialog(null,"Required DLLs Not Found\n"+ex.getCause(),"Error Loading Libraries", JOptionPane.ERROR\_MESSAGE);

}

if(coverType == BMP) // Hide file -- 1

result = hideInBMP(secretFileName, coverFileName, dstFileName, password);

else if(coverType == MEDIA)

result = hideInMEDIA(secretFileName, coverFileName, dstFileName, password);

else

result = hideInOTHER(secretFileName, coverFileName, dstFileName, password);

switch(result){

case 0 : // Unknown Error during hiding

JOptionPane.showMessageDialog(parent, "Unable to hide due to unknown Error",

"Error", JOptionPane.ERROR\_MESSAGE);

break;

case 1 : // Invalid secret File Name

JOptionPane.showMessageDialog(parent, "Invalid Secret File Name",

"Error", JOptionPane.ERROR\_MESSAGE);

parent.secretField.setText("");

break;

case 2 : // Invalid cover File Name

JOptionPane.showMessageDialog(parent, "Invalid Cover File Name",

"Error", JOptionPane.ERROR\_MESSAGE);

parent.coverField.setText("");

break;

case 3 : // Capacity of cover file is less than secret file

JOptionPane.showMessageDialog(parent, "Secret File exceeds capacity of Cover File",

"Error", JOptionPane.ERROR\_MESSAGE);

parent.clearFields();

break;

case 4 : // success

if(coverType == BMP)

outFileName = "steg.bmp";

else

outFileName = "steg"+coverFileName.substring(coverFileName.indexOf('.'));

JOptionPane.showMessageDialog(parent, "Secret File is successfully hidden into cover file with resultant file "+outFileName ,

"Hiding Successful", JOptionPane.INFORMATION\_MESSAGE);

parent.clearFields();

break;

case 5 : // Invalid Destination File Name

JOptionPane.showMessageDialog(parent, "Invalid Destination File Path",

"Error", JOptionPane.INFORMATION\_MESSAGE);

parent.dstField.setText("");

break;

}

// if(isConverted){ // if other images converted to temporary BMP image

// c.deleteTempImage();

}

//splashScreen.dispose(); }

**HidePanel**

package video;

import javax.swing.\*;

public class HidePanel extends JPanel{

protected JPanel secretFilePanel;

protected JTextField secretField;

protected JButton secretFileBrowseButton;

protected JButton hideButton;

protected JButton backButton;

protected JScrollPane messageScrollPane;

protected JEditorPane messagePane;

protected JPanel messagePanel;

protected JTabbedPane secretPane;

protected JButton clearButton;

protected JPasswordField pwdField;

protected JLabel pwdLabel;

protected JTextField dstField;

protected JButton dstBrowseButton;

protected JPanel DestinationPanel;

protected JButton coverFileBrowseButton;

protected JTextField coverField;

protected JPanel CoverFilePanel;

protected JLabel eLabel;

protected JLabel nLabel;

protected JTextField eField;

protected JTextField nField;

protected int encryptTechnique;

public void clearFields(){

secretField.setText("");

messagePane.setText("");

coverField.setText("");

dstField.setText("");

if(encryptTechnique == 3){

eField.setText("");

nField.setText("");

}

else

pwdField.setText("");

}

}

**Main Program**

package video;

public class Main {

public static void main(String[] args) {

}

}

**Splash**

package video;

import java.awt.Dimension;

import java.awt.Toolkit;

import java.awt.Container;

import java.awt.Color;

import java.awt.BorderLayout;

import javax.swing.JLabel;

import javax.swing.JWindow;

import javax.swing.ImageIcon;

import javax.swing.border.LineBorder;

public class Splash extends JWindow

{

private Dimension d = Toolkit.getDefaultToolkit().getScreenSize();

String image;

public Splash(int img)

{

Container c = getContentPane();

JLabel lImage;

if(img==1)

lImage = new JLabel(new ImageIcon("logo.jpg"));

else if(img==2)

lImage = new JLabel(" Processing.... ");

else

lImage = new JLabel(" Sending.... ");

Color borderColor = new Color(0, 0, 0);

lImage.setBorder(new LineBorder(borderColor, 1));

c.add (lImage, BorderLayout.CENTER);

pack();

setSize(getSize().width, getSize().height);

setLocation(d.width / 2 - getWidth() / 2, d.height / 2 - getHeight() / 2);

setVisible (true);

}

public void close()

{

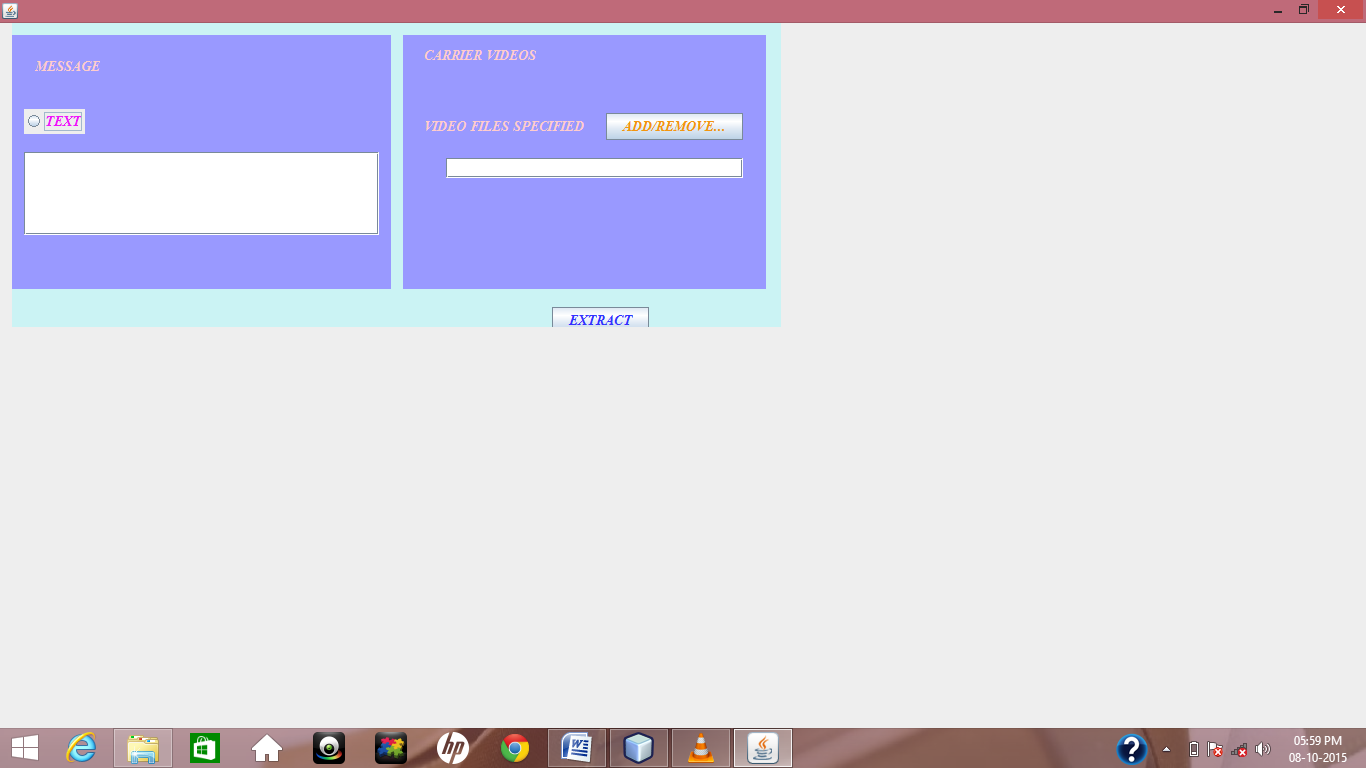
setVisible(false);

dispose();

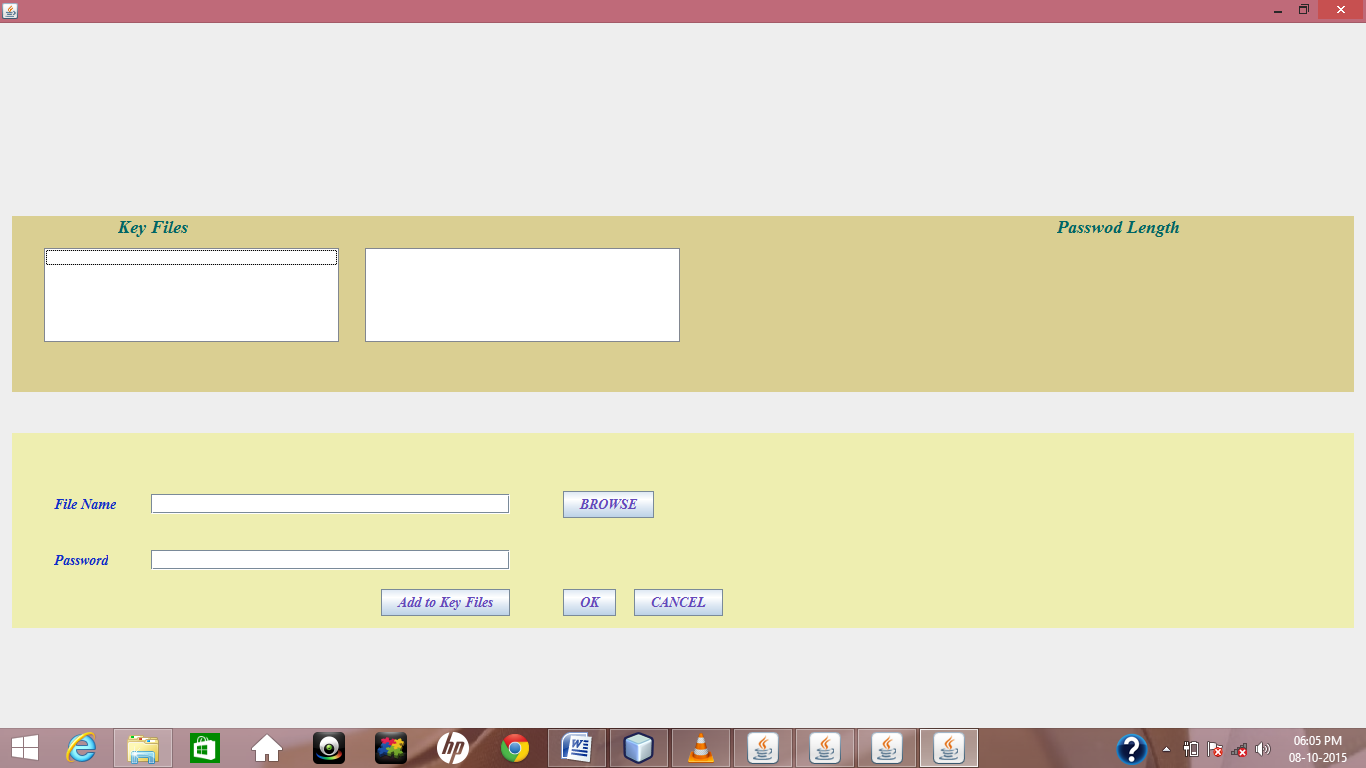
}

}

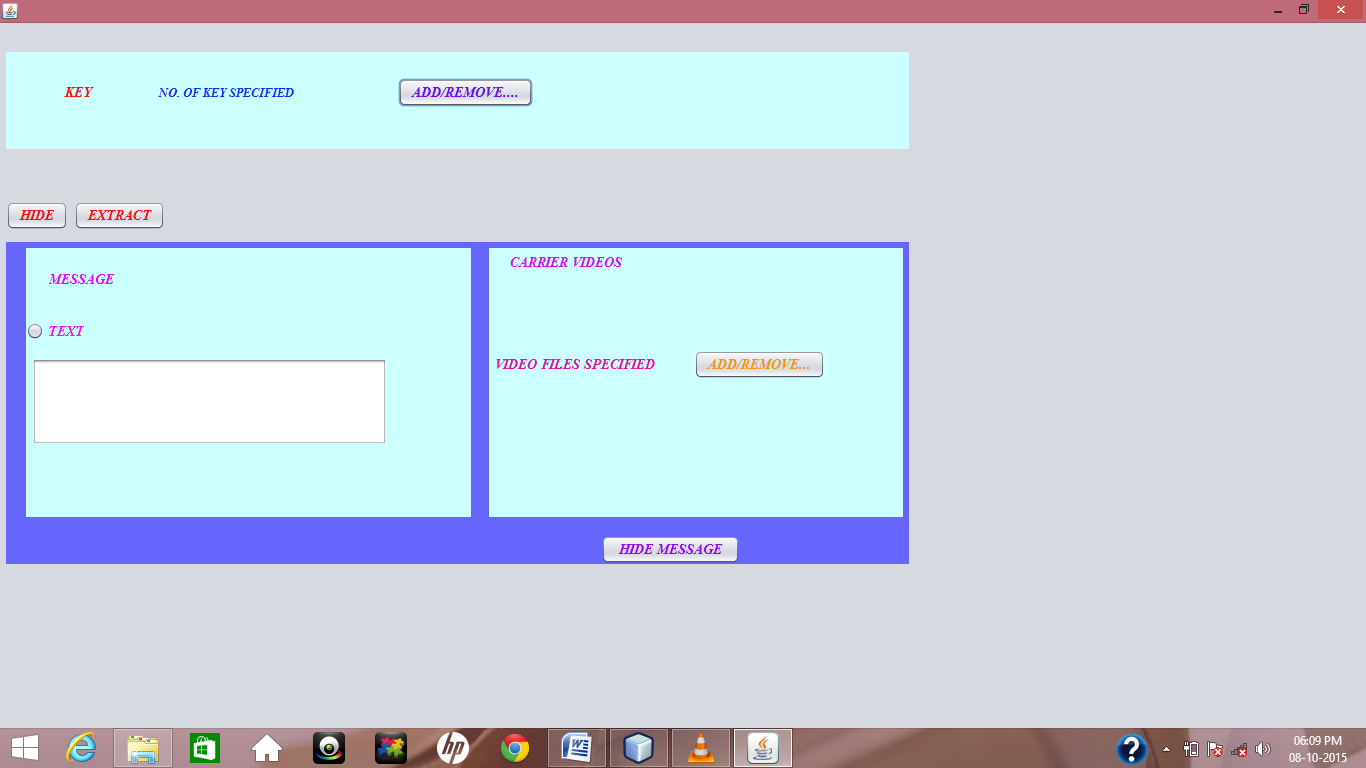
**A.2 Screen shot**

****

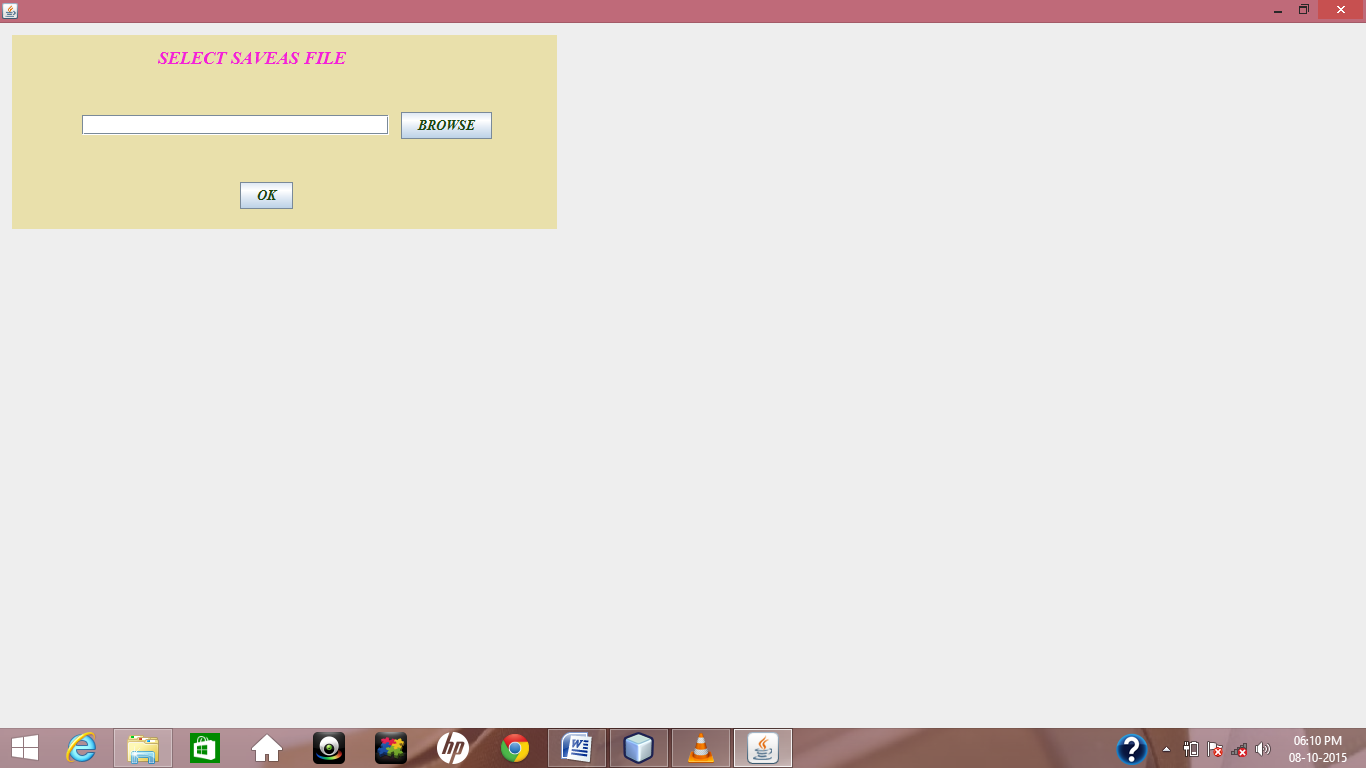
**9.1 Extract**

****

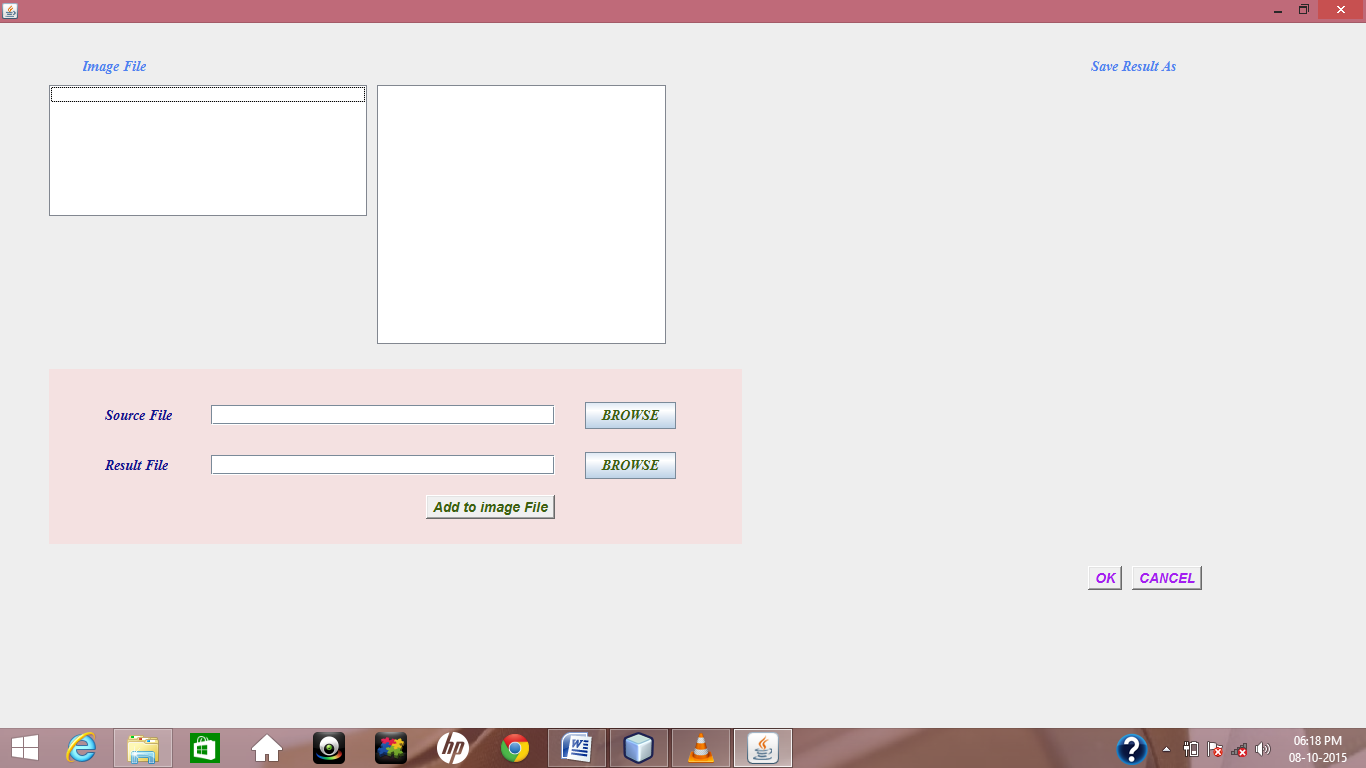
**9.2 Key Field**

****

**9.3 Input**

****

**9.4 Save File**

****

**9.5 Image File**