ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH

MẬT MÃ VÀ AN NINH MẠNG

Bài tập lớn 1

ỨNG DỤNG MÃ HÓA ĐƠN GIẢN

GVHD: Nguyễn Hữu Hiếu
SV: Dương Tấn Sang 1512777
Lê Bá Anh Tuấn 1414384
Ngô Thiên Tín ...

TP. HỒ CHÍ MINH, THÁNG 4/2018

Mục lục

1 Các nội dung được trình bày trong báo cáo. 2

2 Giới thiệu tổng quan về công việc đã làm. 2

3 Nội dung công việc đã làm 2

4 Phân tích và tự đánh giá 9

5 Hướng dẫn sử dụng phần mềm 9
5.1 Mã hóa bằng giải thuật AES 9
5.2 Mã hóa bằng giải thuật DES3 10
5.3 Mã hóa bằng giải thuật RSA 11
5.4 Giải mã bằng giải thuật AES 12
5.5 Giải mã bằng giải thuật DES3 13
5.6 Giải mã bằng giải thuật RSA 14

6 Hướng phát triển 15

7 Đánh giá hoạt động các thành viên trong nhóm 16

1 Các nội dung được trình bày trong báo cáo.

- Tìm hiểu các giải thuật mã hóa: Đối xứng(DES,AES), Bất đối xứng(RSA).
- Lý do chọn các giải thuật mã hóa và cơ sở lý thuyết.
- Tập tin mà chương trình hỗ trợ mã hóa/ giải mã.
- Ngôn ngữ được sử dụng để hoàn thành.
- Hướng dẫn sử dụng phần mềm.
- Nhiệm vụ của các thành viên trong nhóm.

2 Giới thiệu tổng quan về công việc đã làm.

- Công việc thực hiện dựa trên các giải thuật đã học, bằng cách sử dụng ngôn ngữ python để hoàn thành giao diện cho người dùng cùng với việc nhúng các mã vào giao diện.
- Trao đổi và thảo luận để trình bày báo cáo. Báo cáo được soạn thảo bằng LaTex.
- Phạm vi đề tài trong những kiến thức đã học, sinh viên có thể sử dụng bất cứ ngôn ngữ nào để hoàn thiện bài tập lớn.
- Không có sự giới hạn cho đề tài, thỏa mãn sức sáng tạo của sinh viên.

3 Nội dung công việc đã làm

- Tìm hiểu về các giải thuật mã hóa đối xứng và bất đối xứng
  - Những khái niệm cơ bản về mã hóa:
    - Văn bản gốc (plaintext) là văn bản ban đầu có nội dung có thể đọc được và cần được bảo vệ.
    - Văn bản mã hóa (ciphertext) là văn bản sau khi mã hóa, nội dung không thể đọc được.
    - Mã hóa (encryption) là quá trình chuyển văn bản rõ thành văn bản mã hóa.
    - Giải mã (decryption) là quá trình đưa văn bản mã hóa về lại văn bản gốc ban đầu
    - Hệ thống mã hóa (cryptosystem): Cryptosystem = encryption + decryption algorithms
    - Khóa (key) được sử dụng trong quá trình mã hóa và giải mã.
    - Hệ thống mã hóa đối xứng (Symmetric cryptosystem) là hệ thống mã hóa sử dụng một khóa bí mật chia sẻ (sharedsecret-key) cho cả hai quá trình mã hóa và giải mã
    - Hệ thống mã hóa bất đối xứng (Asymmetric cryptosystem) là hệ thống mã hóa sử dụng một khóa công khai (public key) và một khóa bí mật (private key) cho quá trình mã hóa và giải mã.
      - Hệ thống mã hóa bất đối xứng còn được gọi là hệ thống mã hóa khóa công khai (public-key cryptosystem)

- Các kỹ thuật mã hóa
  - Các kỹ thuật mã hóa đối xứng thông dụng: DES, Triple DES, AES
  - DES: Data Encryption Standard
    - NBS (National Bureau of Standards) – bây giờ là NIST (National Institute of Standards and Technology) (Mỹ) chọn DES làm tiêu chuẩn mã hóa vào năm 1977.
    - Mỗi thông điệp (message) được chia thành những khối (block) 64 bits
    - Khóa có 56 bits
    - Có thể bị tấn công bằng giải thuật vét cạn khóa (Brute-force or exhaustive key search)
  - AES: Advanced Encryption Standard
    - Tháng 10/2000, NIST đã chọn AES làm tiêu chuẩn mã hóa thay thế DES
    - AES còn gọi là Rijndael, tên đặt theo hai nhà mật mã học thiết kế ra giải thuật là Daemen và Rijmen
    - Rijndael là giải thuật mã hóa theo khối. Tuy nhiên, khác với DES, Rijndael có thể làm việc với dữ liệu và khóa có độ dài block là 128, 192 hoặc 256 bit.
  - Kỹ thuật mã hóa bất đối xứng phổ biến: RSA
    - RSA: tên được đặt theo tên 3 nhà phát minh ra giải thuật Rivest, Shamir và Adleman
    - Thuật toán sử dụng 2 khóa có quan hệ toán học với nhau: khóa công khai và khóa bí mật
    - Khóa công khai được công bố rộng rãi cho mọi người và được dùng để mã hóa
    - Khóa bí mật dùng để giải mã.
- So sánh mã hóa đối xứng và bất đối xứng
  - Kỹ thuật mã hóa đối xứng có tốc độ mã hóa và giải mã nhanh hơn so với kỹ thuật mã hóa bất đối xứng.
  - Kỹ thuật mã hóa bất đối xứng an toàn hơn so với kỹ thuật mã hóa đối xứng
  - Trong thực tế, ta sử dụng kết hợp cả hai kỹ thuật (hybrid scheme) mã hóa đối xứng và bất đối xứng.
    - Kỹ thuật mã hóa bất đối xứng: thích hợp mã hóa những dữ liệu nhỏ và yêu cầu bảo mật cao. Dùng để Mã hóa khóa bí mật
    - Kỹ thuật mã hóa đối xứng: thích hợp mã hóa những dữ liệu lớn và yêu cầu bảo mật không cao lắm. Dùng để Mã hóa dữ liệu

- Thảo luận và đưa ra hướng giải quyết vấn đề
- Source code giải thuật DES3

```
        from Crypto.Cipher import DES3
from Crypto import Random
from Crypto.Hash import MD5

def pad(s, block_size):
    # Dam bao kich thuoc cua file nhap vao la boi cua AES.block_size
    padding_size = block_size - len(s) % block_size
    return s + b'\0' * padding_size,padding_size
```

```
def encrypt_des3(message,key_in):
    hashFunc = MD5.new()
    hashFunc.update(bytes(key_in,'utf-8'))
    key = hashFunc.digest()

    iv = Random.new().read(DES3.block_size)
    cipher = DES3.new(key,DES3.MODE_CFB,iv)

    padded_mess, padding_size = pad(message,DES3.block_size)

    return iv + cipher.encrypt(padded_mess) + bytes([padding_size])

def decrypt_des3(ciphertext,key_in):

    hashFunc = MD5.new()
    hashFunc.update(bytes(key_in,'utf-8'))
    key = hashFunc.digest()

    iv = ciphertext[:DES3.block_size]
    cipher = DES3.new(key,DES3.MODE_CFB,iv)

    message = cipher.decrypt(ciphertext[DES3.block_size:-1])
    # *-1 de dung cho cau lenh ke tiep
    padding_size = ciphertext[-1]*(-1)
    # Ban chat cau nay la tu dau den vi tri -padding_size, padding_size da duoc
        *-1 o cau lenh truoc
    if padding_size == 0:
        return message
    else:
        return message[:padding_size]
```

- Source code giải thuật AES

```
        from Crypto.Cipher import AES
from Crypto import Random
from Crypto.Hash import MD5

def pad(s, block_size):
    # Dam bao kich thuoc cua file nhap vao la boi cua AES.block_size
    padding_size = block_size - len(s) % block_size
    return s + b'\0' * padding_size,padding_size

def encrypt_aes(message,key_in):

    hashFunc = MD5.new()
    hashFunc.update(bytes(key_in,'utf-8'))
    key = hashFunc.digest()

    # khoi tao aes
    padded_mess, padding_size = pad(message,AES.block_size)
    iv = Random.new().read(AES.block_size)
    cipher = AES.new(key,AES.MODE_CFB,iv)
```

```
    # padding_size ho tro viec loai bo phan bu duoc them vao khi pad 1 file
    #print(len(iv + cipher.encrypt(padded_mess) + bytes([padding_size])))

    return iv + cipher.encrypt(padded_mess) + bytes([padding_size])

def decrypt_aes(ciphertext,key_in):

    hashFunc = MD5.new()
    hashFunc.update(bytes(key_in,'utf-8'))
    key = hashFunc.digest()

    iv = ciphertext[:AES.block_size]
    cipher = AES.new(key,AES.MODE_CFB,iv)

    # Vi tri -1 la padding_size
    message = cipher.decrypt(ciphertext[AES.block_size:-1])
    padding_size = ciphertext[-1] * (-1)
    if padding_size == 0:
        return message
    else:
        return message[:padding_size]
```

- Source code giải thuật RSA

```
        from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_v1_5 as rsa
from Crypto import Random
import file.file_handle as file

def generate_key(folder2save_key):
    key = RSA.generate(2048)
    file.check_and_create_folder(folder2save_key)
    with open(folder2save_key + "\public_key.pem","wb") as public:
        with open(folder2save_key + "\private_key.pem","wb") as private:
            public.write(key.publickey().exportKey())
            private.write(key.exportKey())
            public.close()
            private.close()

def encrypt_rsa(plaintext, filepath_key):
    publickey = RSA.importKey(open(filepath_key).read())
    cipher = rsa.new(publickey)
    return cipher.encrypt(plaintext)

def decrypt_rsa(ciphertext,filepath_key):
    privatekey = RSA.importKey(open(filepath_key).read())
    sentinel = Random.new().read(20)
    cipher = rsa.new(privatekey)
    return cipher.decrypt(ciphertext,sentinel)
    #
    #
    #
```

- Chương trình chính

```
          import os
import sys
from PyQt4 import uic
from PyQt4.QtGui import *
from PyQt4.QtCore import *
import algorithms_rsa as rsa
import file.file_handle as file_handle

AES = 0
DES3 = 1

qtCreatorFile = "main_ui.ui" # Enter file here.
Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)

class MyApp(QMainWindow, Ui_MainWindow):
    def __init__(self):
        QMainWindow.__init__(self)
        Ui_MainWindow.__init__(self)
        self.setupUi(self)
        #ENCRYPT TAB
        self.enc_address_origin_button.clicked.connect(lambda:
            self.getFileNametoTextBox(self.enc_address_origin_text))
        self.enc_address_key_button.clicked.connect(lambda:
            self.getFileNametoTextBox(self.enc_address_key_text))
        self.enc_address_save_button.clicked.connect(lambda:
            self.getFileNametoTextBox(self.enc_address_save_text,True))
        self.encryptButton.clicked.connect(self.encryptFile)
        self.enc_gen_key_button.clicked.connect(self.generateRSAKey)

        #DECRYPT TAB
        self.dec_address_enc_button.clicked.connect(lambda:
            self.getFileNametoTextBox(self.dec_address_enc_text))
        self.dec_address_key_button.clicked.connect(lambda:
            self.getFileNametoTextBox(self.dec_address_key_text))
        self.dec_address_save_button.clicked.connect(lambda:
            self.getFileNametoTextBox(self.dec_address_save_text,True))
        self.decryptButton.clicked.connect(self.decryptFile)


    def decryptFile(self):
        #Kiem tra thong tin duong dan da duoc dien day du
        if self.dec_address_enc_text.text() and self.dec_address_key_text.text()
            and \
            self.dec_address_save_text.text():

            #Kiem tra 1 trong so cac radio option duoc chon
            #
            if self.dec_radio_des.isChecked() or self.dec_radio_aes.isChecked() or
                self.dec_radio_rsa.isChecked():
```

```
                if self.dec_radio_des.isChecked():
                    success =
                        file_handle.decrypt_file(self.dec_address_enc_text.text(),
                    self.dec_address_save_text.text(),self.dec_address_key_text.text(),DES3)
                elif self.dec_radio_aes.isChecked():
                    success =
                        file_handle.decrypt_file(self.dec_address_enc_text.text(),
                    self.dec_address_save_text.text(),
                    self.dec_address_key_text.text(),AES)
                else:

                    success =
                        file_handle.decrypt_file_rsa(self.dec_address_enc_text.text(),
                    self.dec_address_key_text.text(),self.dec_address_save_text.text())

                if success:
                    self.showdialog("Decrypt file success, hash value is correct.
                        This is origin file")
                else:
                    self.showdialog("Decrypt file success, hash value is incorrect.
                        This is not origin file")
            else:
                self.showdialog("No algorithms is chosen")
        else:
            self.showdialog("Missing some file/folder path")

    def encryptFile(self):
        #Kiem tra thong tin duong dan da duoc dien day du
        if self.enc_address_origin_text.text() and self.enc_address_key_text.text()
            and \
            self.enc_address_save_text.text():

            #Kiem tra 1 trong so cac radio option duoc chon
            #
            if self.enc_radio_des.isChecked() or self.enc_radio_aes.isChecked() or
                self.enc_radio_rsa.isChecked():
                if self.enc_radio_des.isChecked():
                    file_handle.encrypt_file(self.enc_address_origin_text.text(),
                    self.enc_address_save_text.text(),self.enc_address_key_text.text(),DES3)
                    toBigFile = False
                elif self.enc_radio_aes.isChecked():
                    file_handle.encrypt_file(self.enc_address_origin_text.text(),
                    self.enc_address_save_text.text(),self.enc_address_key_text.text(),AES)
                    toBigFile = False
                else:
                    try:
                        file_handle.encrypt_file_rsa(self.enc_address_origin_text.text(),
                        self.enc_address_key_text.text(),self.enc_address_save_text.text())
                    except ValueError:
                        toBigFile = True
                if toBigFile:
                    self.showdialog("RSA Algorithms: This size of currnet file is to
                        big")
```

```
                else:
                    self.showdialog("Encrypt file success")

            else:
                self.showdialog("No algorithms is chosen")
        else:
            self.showdialog("Missing some file/folder path")

    def generateRSAKey(self):
        str = self.getFolderName()
        rsa.generate_key(str)
        self.showdialog("Key have been save at ..." + str)

    def showdialog(self,message):
        dialog = QDialog()
        button = QPushButton("OK")
        text = QLabel(message)
        text.setAlignment(Qt.AlignCenter)
        text.setWordWrap(True)

        vbox = QVBoxLayout()
        vbox.addWidget(text)
        vbox.addStretch()
        vbox.addWidget(button)

        dialog.setLayout(vbox)
        dialog.setMaximumSize(250,150)
        dialog.setMinimumSize(250,150)
        dialog.setWindowTitle("Notification")
        dialog.setWindowModality(2)

        button.clicked.connect(dialog.reject)

        dialog.exec_()

    def getFileNametoTextBox(self,textBox,isFolder = False):
        dialog = QFileDialog()
        if isFolder:
            dialog.setFileMode(QFileDialog.Directory)
        else:
            dialog.setFileMode(QFileDialog.AnyFile)

        filenames = []
        if dialog.exec():
            filenames = dialog.selectedFiles()
        if (filenames):
            textBox.setText(filenames[0])

    def getFolderName(self):
        return str(QFileDialog.getExistingDirectory(self, "Select Directory"))

if __name__ == "__main__":
    app = QApplication(sys.argv)
```

```
window = MyApp()
window.show()
sys.exit(app.exec_())
```

- Chạy thử demo
- Kiểm tra lỗi và khắc phục
- Hoàn thành báo cáo bằng LaTex

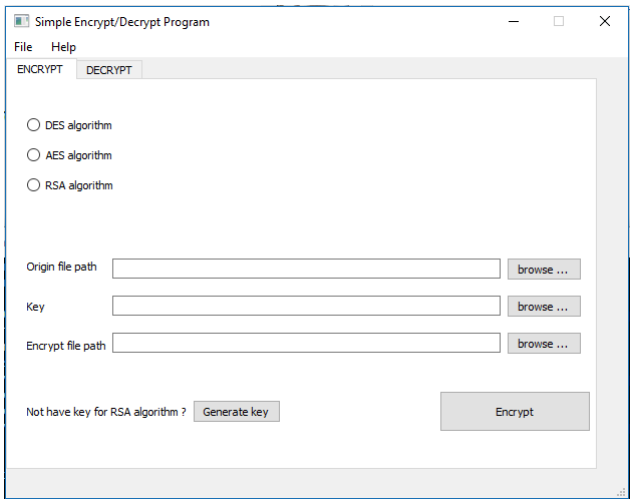
4 Phân tích và tự đánh giá

- Kết quả đạt được
  - Học tập thêm được ngôn ngữ python
  - Nâng cao khả năng sử dụng LaTex
  - Nâng cao tinh thần làm việc nhóm
  - Nâng cao cách tổ chức và quản lý công việc
  - Hiểu rõ hơn các kiến thức về mật mã
- Mặt hạn chế
  - Chưa thêm được nhiều tính năng mới vượt qua đề bài yêu cầu
  - Cách tổ chức vẫn còn đơn giản

5 Hướng dẫn sử dụng phần mềm

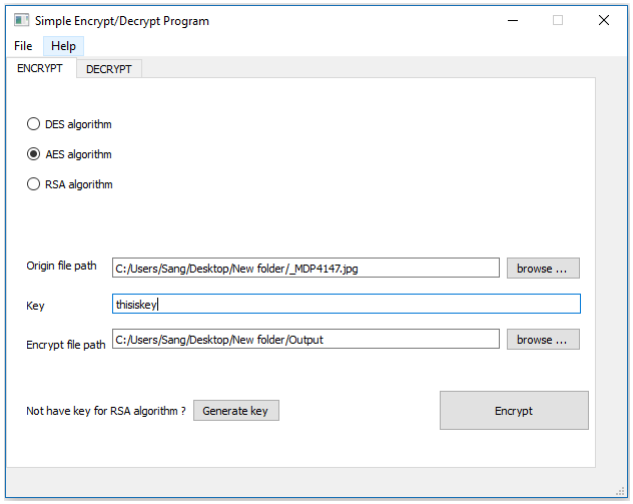
5.1 Mã hóa bằng giải thuật AES

- Mở ứng dụng ta có được giao diện làm việc



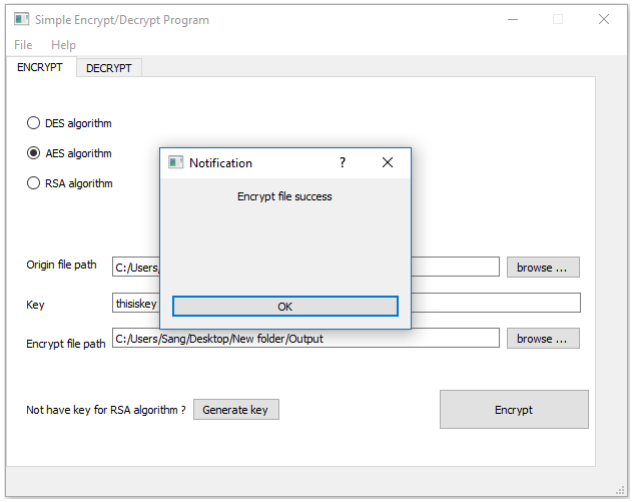
Hình 1: Màn hình khởi động của ứng dụng

- Lựa chọn giải thuật AES và chọn đường dẫn file



Hình 2: Nhập đường dẫn tệp, khóa và địa chỉ lưu tệp đã mã hóa cho giải thuật AES

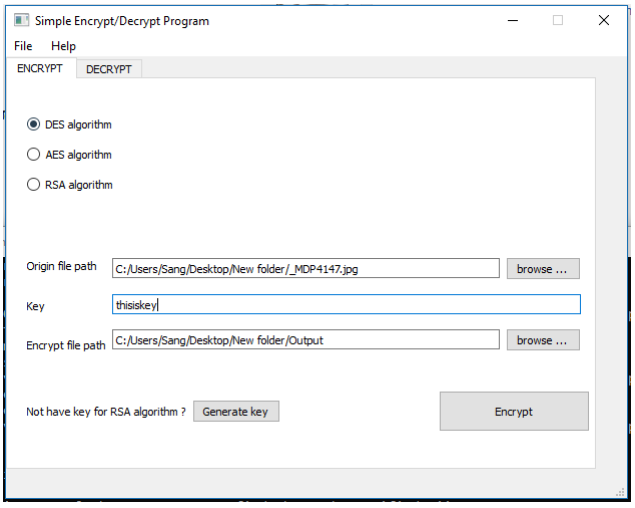
- Nhấn Encrypt để mã hóa file



Hình 3: Mã hóa tệp thành công

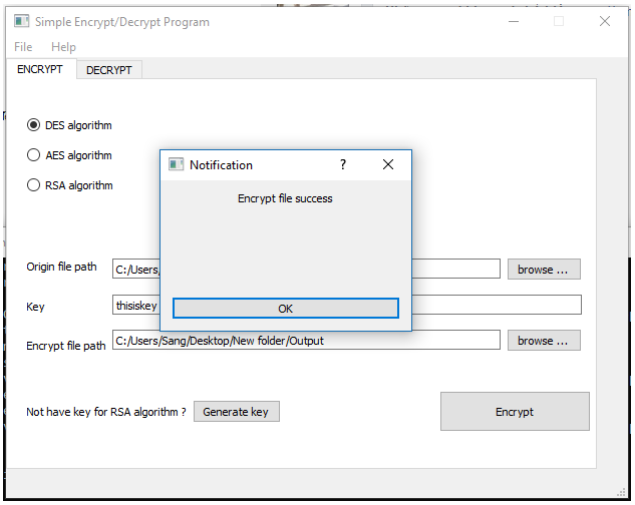
5.2 Mã hóa bằng giải thuật DES3

- Lựa chọn giải thuật DES và chọn đường dẫn file



Hình 4: Nhập đường dẫn tệp, khóa và địa chỉ lưu tệp đã mã hóa cho giải thuật DES3

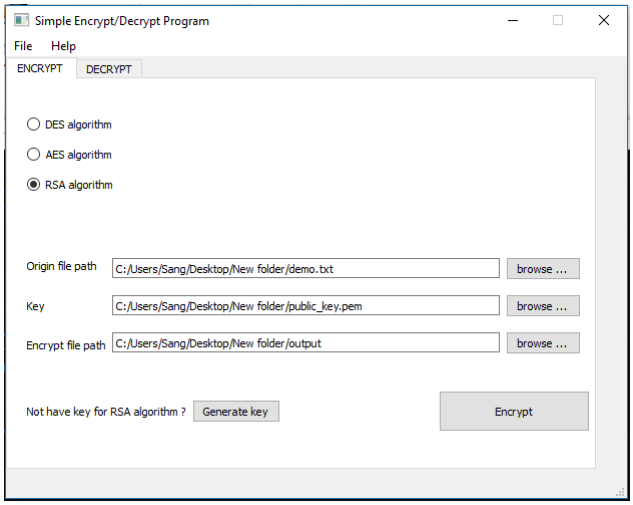
- Nhấn Encrypt để mã hóa file



Hình 5: Mã hóa tệp thành công

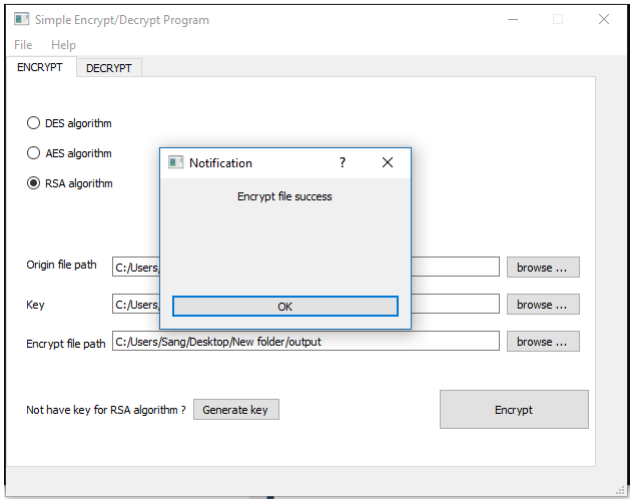
5.3 Mã hóa bằng giải thuật RSA

- Chọn đường dẫn file cần mã hóa



Hình 6: Nhập đường dẫn tệp, đường dẫn khóa, đường dẫn của thư mục lưu tệp đã mã hóa

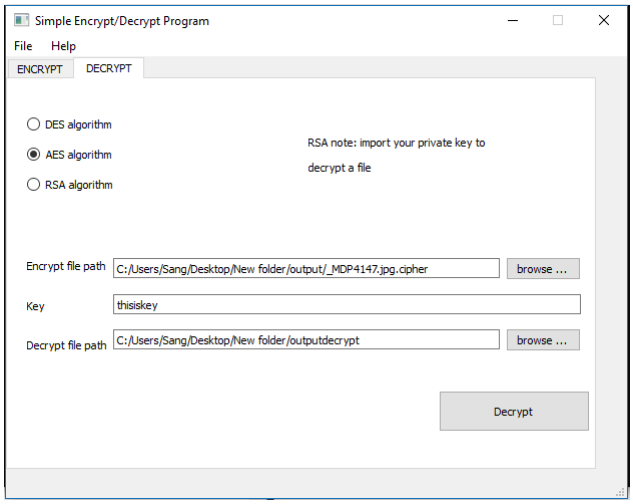
- Chọn ENCRYPT



Hình 7: Mã hóa tệp thành công

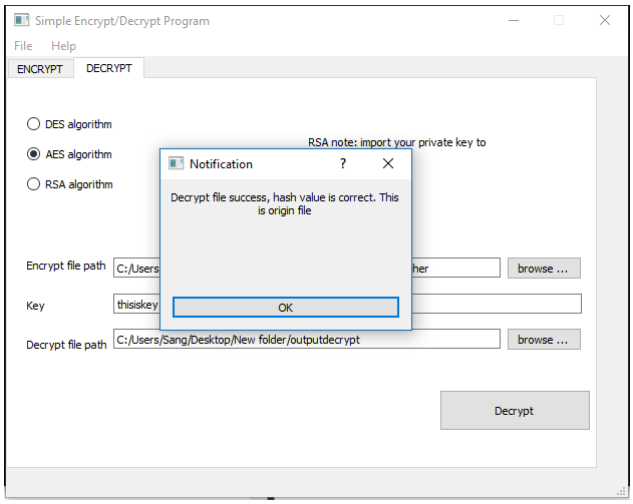
5.4 Giải mã bằng giải thuật AES

- Lựa chọn giải thuật AES và chọn đường dẫn file



Hình 8: Chọn đường dẫn của tệp cần giải mã, khóa, thư mục lưu

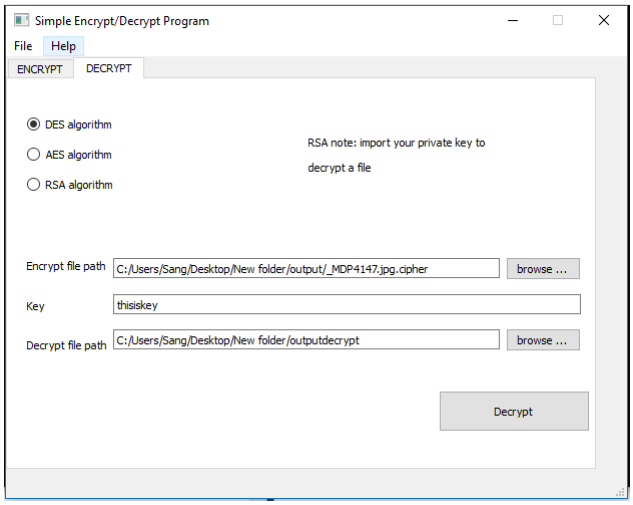
- Nhấn Decrypt để giải mã file



Hình 9: Kết quả kiểm tra hàm hash của tệp vừa được giải mã

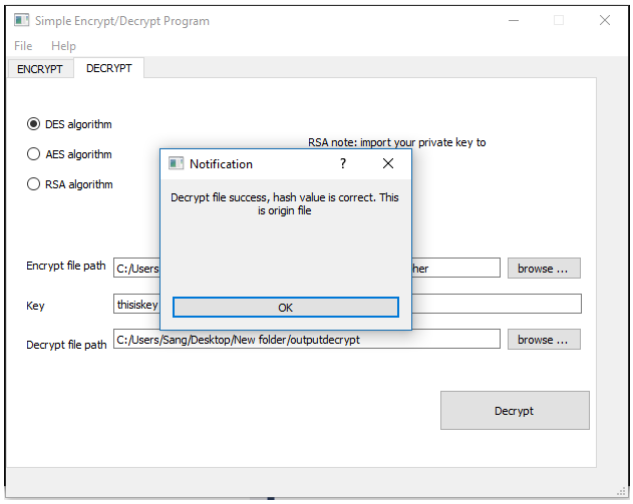
5.5 Giải mã bằng giải thuật DES3

- Lựa chọn giải thuật DES và chọn đường dẫn file



Hình 10: Chọn đường dẫn của tệp cần giải mã, khóa, thư mục lưu của giải thuật DES3

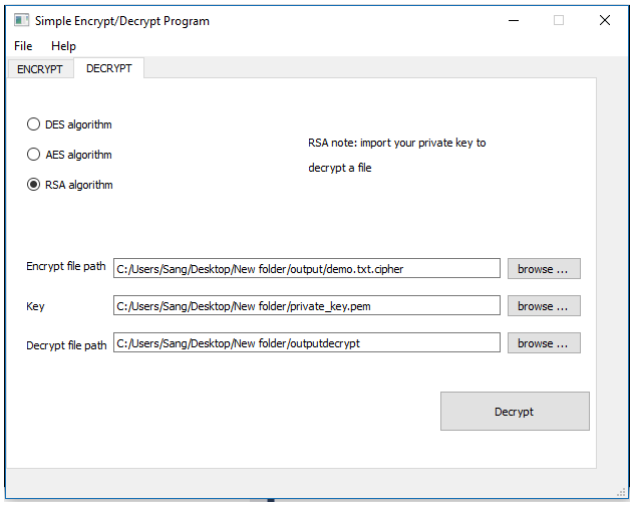
- Nhấn Decrypt để giải mã file



Hình 11: Hiển thị kết quả giải mã, kết quả của hàm hash

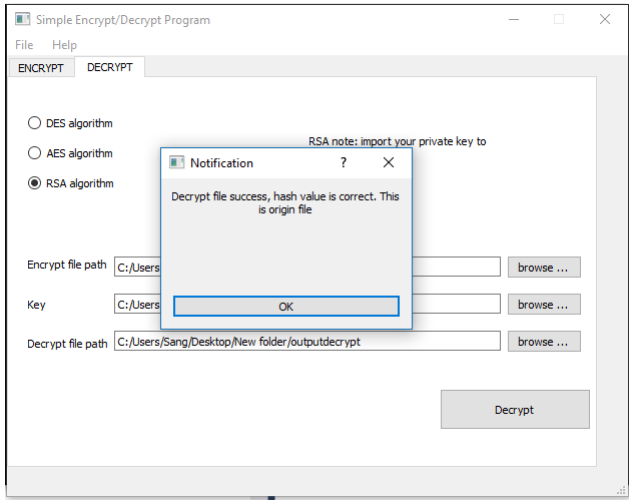
5.6 Giải mã bằng giải thuật RSA

- Lựa chọn giải thuật RSA và chọn đường dẫn file



Hình 12: Chọn đường dẫn của tệp cần giải mã, đường dẫn khóa, địa chỉ lưu

- Nhấn Decrypt để giải mã file



Hình 13: Kết quả giải mã và kiểm tra của hám hash

6 Hướng phát triển

- Củng cố phần giao diện
- Xây dựng thanh tiến trình khi đang mã hóa
- Xây dựng một gói cài đặt ứng dụng, để người dùng dễ dàng cài đặt

7 Đánh giá hoạt động các thành viên trong nhóm

- Dương Tấn Sang: 33,33 %
- Ngô Thiên Tín: 33,33 %
- Lê Bá Anh Tuấn: 33,33 %

Tài liệu

[1] PyCryptodome "**https://www.pycryptodome.org/en/latest**",

[2] PyQt4 "**http://pyqt.sourceforge.net/Docs/PyQt4/**",