# encoding=utf-8

# Copyright HXBer

from tkinter import \*

import tkinter.messagebox

import base64

import rsa

from pyDes import des, CBC, PAD\_PKCS5

import binascii

import random

import string

#可自行设定RSAkey

'''

with open('./rsa\_public\_key.pem', 'r') as f:

pubkey = rsa.PublicKey.load\_pkcs1(f.read().encode('utf-8'))

with open('./rsa\_private\_key.pem', 'r') as f:

privkey = rsa.PrivateKey.load\_pkcs1(f.read().encode('utf-8'))

'''

#初始化生成DESkey、DESiv、RSAkey

(pubkey, privkey) = rsa.newkeys(1024)

Des\_Key = ''.join(random.sample(string.ascii\_letters + string.digits, 8))

Des\_IV = ''.join(random.sample(string.ascii\_letters + string.digits, 8))

print(Des\_Key)

print(Des\_IV)

def des\_encrypt(): #进行DES加密

inp2.delete('1.0', 'end')

plain = inp1.get('1.0', 'end')[:-1]

print(plain)

plain = plain.encode('utf-8')

k = des(Des\_Key, CBC, Des\_IV, pad=None, padmode=PAD\_PKCS5)

en = k.encrypt(plain, padmode=PAD\_PKCS5)

cipher = binascii.b2a\_hex(en)

inp2.insert(END, cipher)

print (cipher)

def rsa\_sign(): #进行RSA数字签名

inp3.delete('1.0', 'end')

cipher = inp1.get('1.0', 'end')[:-1] #获取明文

crypto = rsa.sign(cipher.encode('utf-8'), privkey, 'SHA-1')

crypto = base64.b64encode(crypto)

print(crypto)

inp3.insert(END, crypto)

def rsa\_encrypt(): #对DESkey进行RSA加密

inp5.delete('1.0', 'end')

plain = inp4.get('1.0', 'end')[:-1] #获取DESkey

plain = plain.encode('utf-8')

print(plain)

val\_list = []

for i in range(0, len(plain), 117): #进行RSA分段解密

tpl = plain[i:i + 117]

val = rsa.encrypt(tpl, privkey)

val\_list.append(val)

cipher = b''.join(val\_list)

cipher = base64.b64encode(cipher)

inp5.insert(END, cipher)

print(cipher)

def transl(): #传输数据

inp7.delete('1.0', 'end')

content = inp2.get('1.0', 'end')[:-1]

inp7.insert(END, content)

inp8.delete('1.0', 'end')

content = inp3.get('1.0', 'end')[:-1]

inp8.insert(END, content)

inp10.delete('1.0', 'end')

content = inp5.get('1.0', 'end')[:-1]

inp10.insert(END, content)

def rsa\_decrypt(): #对enkey进行RSA解密获得DESkey

inp9.delete('1.0', 'end')

cipher = base64.b64decode(inp10.get('1.0', 'end')[:-1])

val\_list = []

for i in range(0, len(cipher), 128): #进行RSA分段解密

tpl = cipher[i:i + 128]

val = rsa.decrypt(tpl, privkey)

val\_list.append(val)

plain = b''.join(val\_list)

plain = plain.decode('utf-8')

inp9.insert(END, plain)

print(plain)

def des\_decrypt(): #对密文用DESkey进行DES解密

inp6.delete('1.0', 'end')

cipher = inp7.get('1.0', 'end')[:-1] #获取RSA解密后的DESley

k = des(Des\_Key, CBC, Des\_IV, pad=None, padmode=PAD\_PKCS5)

plain = k.decrypt(binascii.a2b\_hex(cipher), padmode=PAD\_PKCS5) #进行DES解密

plain = plain.decode('utf-8')

inp6.insert(END, plain)

print(plain)

def rsa\_verify(): #校验

indata = inp6.get('1.0', 'end')[:-1] #获取解密后得到的明文

indata = indata.encode('utf-8')

signature = base64.b64decode(inp8.get('1.0', 'end')[:-1]) #获取签名

print("indata", indata)

print("signature", signature)

try:

rsa.verify(indata, signature, pubkey) #进行校验

tkinter.messagebox.showinfo('Result', 'correct')

except rsa.VerificationError:

tkinter.messagebox.showinfo('Result', 'incorrect')

raise ('Verification failed.')

root = tkinter.Tk()

root.geometry('1600x900')

root.title('RSA')

#设定框图

Label(root, text="plain", anchor=NW).grid(row=0, column=0)

Label(root, text="cipher", anchor=NE).grid(row=2, column=0)

Label(root, text="sign", anchor=NE).grid(row=4, column=0)

Label(root, text="Skey", anchor=NE).grid(row=0, column=1)

Label(root, text="enSkey", anchor=NE).grid(row=2, column=1)

Label(root, text="decode", anchor=NE).grid(row=0, column=2)

Label(root, text="cipher", anchor=NE).grid(row=2, column=2)

Label(root, text="sign", anchor=NE).grid(row=4, column=2)

Label(root, text="Skey", anchor=NE).grid(row=0, column=3)

Label(root, text="enSkey", anchor=NE).grid(row=2, column=3)

Label(root, text="publicKey", anchor=NE).grid(row=0, column=5)

Label(root, text="privateKey", anchor=NE).grid(row=2, column=5)

Label(root, text="desKey", anchor=NE).grid(row=4, column=5)

Label(root, text="desiv", anchor=NE).grid(row=6, column=5)

inp1 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp2 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp3 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp1.grid(row=1, column=0)

inp2.grid(row=3, column=0)

inp3.grid(row=5, column=0)

inp4 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp5 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp4.grid(row=1, column=1)

inp5.grid(row=3, column=1)

inp6 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp7 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp8 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp6.grid(row=1, column=2)

inp7.grid(row=3, column=2)

inp8.grid(row=5, column=2)

inp9 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp10 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp9.grid(row=1, column=3)

inp10.grid(row=3, column=3)

inp11 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp12 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp13 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp14 = Text(root, width=40, height=9, relief="solid", borderwidth=1)

inp11.grid(row=1, column=5)

inp12.grid(row=3, column=5)

inp13.grid(row=5, column=5)

inp14.grid(row=7, column=5)

#设定按钮

btn1 = Button(root, width=6, text='Encode', command=des\_encrypt)

btn2 = Button(root, width=5, text='Sign', command=rsa\_sign)

btn3 = Button(root, width=9, text='EncodeSkey', command=rsa\_encrypt)

btn4 = Button(root, width=5, text='Transl', command=transl)

btn5 = Button(root, width=9, text='DecodeSkey', command=rsa\_decrypt)

btn6 = Button(root, width=6, text='Decode', command=des\_decrypt)

btn7 = Button(root, width=5, text='Verify', command=rsa\_verify)

btn1.place(relx=0.08, rely=0.9)

btn2.place(relx=0.18, rely=0.9)

btn3.place(relx=0.28, rely=0.9)

btn4.place(relx=0.38, rely=0.9)

btn5.place(relx=0.48, rely=0.9)

btn6.place(relx=0.58, rely=0.9)

btn7.place(relx=0.68, rely=0.9)

inp4.insert(END, Des\_Key)

inp11.insert(END, pubkey)

inp12.insert(END, privkey)

inp13.insert(END, Des\_Key)

inp14.insert(END, Des\_IV)

mainloop()

数字信封的思路就是利用对称加密的快捷性与安全性以及非对称加密的安全与验证完整性来实施。对明文进行DES加密后，对DESkey进行RSA加密，并进行签名。传输后接受方需要对enDESkey进行解密，获得DESkey后对密文进行解密。