# AES256 GCM

# encryption and decryption

CryptPy - Python Message Encrypter/Decrypter.

## Overview

CryptPy is a program written in Python to encrypt any messages and also decrypt the messages.

## Installing

You will first need to have Python installed and to make sure that you have added Python to PATH.

Extract the folder to somewhere easily accessible.

You should have a folder containing the following files:

1. crytpPy.py
2. keys.png
3. requirements.txt

Once you have a command prompt open to the correct path, you should type the following command:

py -m pip install -r requirements.txt

(to run the program successfully, we need a python module called pybase64 installed: running the above command installs it for you)

To run CryptPy, open your CMD and type "Crypt.py", then press enter:

## Code review

The following code imports all the necessary libraries:

from tkinter import \*

from tkinter import messagebox

import base64

import os

### Encrypt function

The above code is for decrypting the message. It: gets the password typed by the user and compares it with 1234: secret key for this program-you can change it in the codeto your liking. If user has not inputed any password it pops a window prompting the user to input the password. If the password inputed is wrong, it pops a window prompting the user that the password is wrong.

If the password is correct, the function takes user input and decodes it into the original message using a base 64 algorithm. It then outputs this message to the user

def encrypt():

    password=code.get()

    if password=="1234":

        screen1=Toplevel(screen)

        screen1.title("encryption")

        screen1.geometry("400x200")

        screen1.configure(bg="#ed3833")

        message=text1.get(1.0,END)

        encode\_message=message.encode("ascii")

        base64\_bytes=base64.b64encode(encode\_message)

        encrypt=base64\_bytes.decode("ascii")

        Label(screen1,text="ENCRYPT",font="arial",fg="white",bg="#ed3833").place(x=10,y=0)

        text2=Text(screen1,font="roboto 10",bg="white",relief=GROOVE,wrap=WORD,bd=0)

        text2.place(x=10,y=40,width=380,height=150)

        text2.insert(END,encrypt)

    elif password =="":

        messagebox.showerror("encryption","Input password")

    else:

        messagebox.showerror("encryption","Inavlid password")

### Decrypt function

def decrypt():

    password=code.get()

    if password==”1234”:

        screen2=Toplevel(screen)

        screen2.title(“decryption”)

        screen2.geometry(“400x200”)

        screen2.configure(bg=”#00bd56”)

        message=text1.get(1.0,END)

        decode\_message=message.encode(“sci”)

        base64\_bytes=base64.b64decode(decode\_message)

        decrypt=base64\_bytes.decode(“sci”)

        Label(screen2,text=”ENCRYPT”,font=”arial”,fg=”white”,bg=”#00bd56”).place(x=10,y=0)

        text2=Text(screen2,font=”roboto 10”,bg=”white”,relief=GROOVE,wrap=WORD,bd=0)

        text2.place(x=10,y=40,width=380,height=150)

        text2.insert(END,decrypt)

    elif password ==””:

        messagebox.showerror(“encryption”,”Input password”)

    else:

        messagebox.showerror(“encryption”,”Inavlid password”)

The above code is for decrypting the message. It: gets the password typed by the user and compares it with 1234: secret key for this program-you can change it in the code to your liking. If user has not input any password it pops a window prompting the user to input the password. If the password input is wrong, it pops a window prompting the user that the password is wrong.

If the password is correct, the function takes user input and decodes it into the original message using a base 64 algorithm. It then outputs this message to the user.

### Main**\_**screen function

def main\_screen():

        global screen

        global code

        global text1

        #screen

        screen=Tk()

        screen.geometry("375x398")

        #icon

        image\_icon=PhotoImage(file="keys.png")

        screen.iconphoto(False,image\_icon)

        screen.title("CryptPy App")

        #reset function

        def reset():

            code.set("")

            text1.delete(1.0,END)

        #text area for text to be encrypted

        Label(text="Enter text for encryption", fg="black",font=("calbri,13")).place(x=10,y=10)

        text1=Text(font="Roboto 20", bg="white", relief=GROOVE,wrap=WORD,bd=0)

        text1.place(x=10,y=50,width=355,height=100)

        #text area for the secret key

        Label(text="Enter secret key for encryption and decryption", fg="black",font=("calbri,13")).place(x=10,y=170)

        code=StringVar()

        Entry(textvariable=code,width=19,bd=0,font=("aril",25),show="\*").place(x=10,y=200)

        #encrypt button

        Button(text="ENCRYPT",height="2",width=23,bg="#ed3833",fg="white",bd=0,command=encrypt).place(x=10,y=250)

        #decrypt button

        Button(text="DECRYPT",height="2",width=23,bg="#00bd56",fg="white",bd=0,command=decrypt).place(x=200,y=250)

        #rest button

        Button(text="RESET",height="2",width=50,bg="#1089ff",fg="white",bd=0, command=reset).place(x=10,y=300)

        screen.mainloop()

main\_screen()

In the above code:

Screen, text, code variables are defined by the global keyword to make them accessible from anywhere within the program. The screen functions create the widow which is used as a graphical user interface for the user to interact with the program.

We have the <screen.tittle> statement which prints the program’s name on top of the screen.

The icon statements create an icon besides the program name. We also have the labels which create the text areas for the user to input the message they want to be encrypted or decrypted, and also a text area to input the secret key.

We also have the code variable which we use to take the user input and store it. The rest function refreshes the program and all the input text is deleted. Note that its deletes everything from the first to the last string.

The button methods create the encrypt, decrypt and reset buttons. Besides the styling, we pass commands to these buttons. Once the buttons are pressed, these commands are executed by calling the functions which are defined earlier n the program.