



الجمهورية العربية السورية
جامعة تشرين
كلية الهندسة الميكانيكية والكهربائية
هندسة الاتصالات والإلكترونيات
السنة الخامسة

"تشفير وفك تشفير الرسائل بلغة البايثون باستخدام مكتبي Base64 و Tkinter"

Encode and Decode messages in Python with Tkinter and Base64 "
"libraries

إشراف الدكتور:

مهند عيسى

إعداد:

سلافة عارف عيسى

2395

هديل عبد المجيد محمد

2824

من طلاب السنة الخامسة هندسة الاتصالات والإلكترونيات

ملخص:

سنقوم في هذا البحث بكتابة برنامج بلغة بايثون يقوم بتشفير وفك تشفير الرسائل باستخدام مفتاح مشترك وباستخدام مكتبة Tkinter والتي هي مكتبة واجهة المستخدم الرسومية القياسية لبائثون حيث يوفر Python عند دمجها مع Tkinter طريقة سريعة وسهلة لإنشاء تطبيقات واجهة المستخدم الرسومية. ومكتبة Base64 التي توفر توابع لتشفير البيانات الثنائية وتحويلها إلى محارف ASCII أو فك تشفير محارف ال ASCII وتحويلها إلى بيانات ثنائية.

Abstract:

In this paper, we will write a python program that encrypts and decrypts messages using a shared key and using the Tkinter library, which is the standard GUI library for python . python when combined with Tkinter provides a quick and easy way to create GUI applications.

The base64 library provides methods for encoding binary data into ASCII characters, or for decoding ASCII characters into binary data

• مقدمة:

في هذه الأيام يرتبط كل نشاط بشري ارتباطاً وثيقاً بأنظمة الحوسبة حيث يتم استخدام هذه الأنظمة في كل تطبيق في مجال الرعاية الصحية والتعليم والخدمات المصرفية والبرمجيات والتسويق. لكن قد نتساءل كيف تقوم المؤسسات بتأمين معلوماتها؟ وكيف يتم الحفاظ على سرية معاملتنا المصرفية؟ الجواب على كل ذلك هو "التشفير".

فالتشفير هو بشكل عام فن أو تقنية إنشاء رسائل مشفرة باستخدام مفاتيح سرية حيث لا يمكن فك تشفيرها ومعرفة هذه الرسائل إلا من قبل الشخص الذي توجه إليه أو من يملك المفتاح.

في الحوسبة، تُعرف البيانات غير المشفرة بالنص الشائع أو الصريح، وتسمى البيانات المشفرة أيضاً بالنص المشفر.

تسمى الصيغ المستخدمة لتشفير وفك تشفير الرسائل خوارزميات التشفير.

لكي تكون هذه الخوارزميات فعالة، يتضمن التشفير متغيراً كجزء من الخوارزمية. المتغير، الذي يسمى المفتاح، هو ما يجعل إخراج الشيفرة فريداً.

عندما يتم اعتراض رسالة مشفرة من قبل كيان غير مصرح به ، يتعين على المتطفل تخمين هوية المرسل المستخدم لتشفير الرسالة ، وكذلك المفاتيح التي تم استخدامها كمتغيرات. الوقت الذي يستغرقه تخمين هذه المعلومات هو ما يجعل التشفير أداة أمان قيّمة.

• كيف يعمل التشفير؟ وما هي أنواعه؟

في بداية عملية التشفير ، يجب أن يقرر المرسل ما هو التشفير الذي سيخفي معنى الرسالة وأفضل متغير لاستخدامه كمفتاح لجعل الرسالة المشفرة فريدة من نوعها. تنقسم أنواع الخوارزميات الأكثر استخدامًا إلى فئتين: **متماثل** و**غير متماثل**.

تستخدم الخوارزميات **المتماثلة**، والتي يشار إليها أيضًا باسم **تشفير المفتاح السري**، مفتاحًا واحدًا. يشار في بعض الأحيان إلى المفتاح باعتباره سرًا مشتركًا لأن المرسل أو نظام الحوسبة الذي يقوم بالتشفير يجب أن يشارك المفتاح السري مع جميع الكيانات المخولة بفك تشفير الرسالة. تشفير المفتاح المتماثل عادة ما يكون أسرع بكثير من التشفير غير المتماثل. أكثر خوارزميات التشفير المتماثلة استخدامًا هو معيار التشفير المتقدم (AES) ، الذي صمم لحماية المعلومات المصنفة من قبل الحكومة.

تستخدم الخوارزميات **غير المتماثلة**، والمعروفة أيضًا باسم **تشفير المفتاح العام**، مفتاحين مختلفين - ولكن مرتبطين منطقيًا - هما **المفتاح العام** و**المفتاح الخاص**. غالبًا ما يستخدم هذا النوع من التشفير أعداد أولية كبيرة لإنشاء المفاتيح لأنه من الصعب حساب الأعداد الأولية والتصميم العكسي للتشفير. تعد خوارزمية تشفير RSA (Rivest-Shamir-Adleman) حاليًا خوارزمية المفتاح العام الأكثر استخدامًا. باستخدام RSA ، يمكن استخدام المفتاح العام لتشفير رسالة والخاص لفك تشفيرها.

اليوم، تستخدم العديد من عمليات التشفير خوارزمية متماثلة لتشفير البيانات وخوارزمية غير متماثلة لتبادل المفتاح السري بشكل آمن.

● أهمية التشفير:

الغرض الأساسي من التشفير هو حماية سرية البيانات الرقمية المخزنة على أنظمة الكمبيوتر أو المنقولة عبر الإنترنت أو أي شبكة كمبيوتر أخرى حيث يلعب التشفير دورًا مهمًا في تأمين العديد من أنواع أصول تكنولوجيا المعلومات. يوفر ما يلي:

- السرية : تشفير محتوى الرسالة.
- المصادقة: التحقق من أصل الرسالة.
- التكاملية: تثبت أن محتويات الرسالة لم تتغير منذ إرسالها.

● كيف يتم استخدامه؟

يشجع استخدام التشفير لحماية البيانات في العبور أثناء إرسالها. في كل مرة يستخدم شخص ما جهاز صراف آلي أو يشتري شيئًا عبر الإنترنت باستخدام هاتف ذكي ، يتم استخدام التشفير لحماية المعلومات التي يتم نقلها. تعتمد الشركات بشكل متزايد على التشفير لحماية التطبيقات والمعلومات الحساسة من تلف السمعة عند حدوث خرق للبيانات.

هناك ثلاثة مكونات رئيسية لأي نظام تشفير: البيانات ومحرك التشفير وإدارة المفاتيح. في تشفير الكمبيوتر المحمول ، يتم تشغيل جميع المكونات الثلاثة أو تخزينها في نفس المكان: على الكمبيوتر المحمول.

ومع ذلك ، في بنى التطبيقات ، عادةً ما يتم تشغيل المكونات الثلاثة أو تخزينها في أماكن منفصلة لتقليل فرصة أن تؤدي التسوية الخاصة بأي مكون منفرد إلى تسوية النظام بأكمله.

• فوائد البحث:

نتحدث في هذا المشروع عن إنشاء برنامج لتشفير وفك تشفير الرسائل والذي له الفوائد التالية:

١. برنامج مفتوح المصدر يمكن التعديل عليه
٢. البساطة والسهولة في التصميم
٣. الأمان وحماية الخصوصية
٤. التوفير المادي لأنه مجاني
٥. السرعة في التشفير وفك التشفير

• طرائق البحث:

قمنا بالبحث على يوتيوب عن المكتبات التي استخدمناها وأهميتها في البايثون بالإضافة الى كيفية تنصيبها واستدعائها والية عملها .
ثم قمنا بكتابة كود برمجي يتضمن انشاء واجهة مستخدم رسومية باستخدام Tkinter بأبعاد ثابتة تحتوي على نصوص ثابتة وحقول ادخال نصية وأزرار.

باستخدام مكتبة base64 قمنا بكتابة تعليمات لتشفير النص المدخل بالمفتاح الذي اخترناه (تشفير متناظر لأننا استخدمنا مفتاح واحد للتشفير وفك التشفير) حيث نلاحظ أنه في حال استخدمنا مفتاح مختلف عند فك تشفير النص المشفر سيعطينا نص مختلف عن النص الصريح الذي شفرناه.

● فيما يلي الكود الموافق:

```
from tkinter import *
import base64
root = Tk()
root.geometry('500x300')
root.resizable(0,0)
root.title("DataFlair - Message Encode and Decode")
Label(root, text='ENCODE DECODE', font = 'arial 20 bold').pack()
Label(root, text='DataFlair', font = 'arial 20 bold').pack(side
=BOTTOM)
Text = StringVar()
private_key = StringVar()
mode = StringVar()
Result = StringVar()
def Encode(key,message):
    enc=[]
    for i in range(len(message)):
        key_c = key[i % len(key)]
        enc.append(chr((ord(message[i]) + ord(key_c)) % 256))
    return base64.urlsafe_b64encode("".join(enc).encode()).decode()
def Decode(key,message):
    dec=[]
    message = base64.urlsafe_b64decode(message).decode()
    for i in range(len(message)):
        key_c = key[i % len(key)]
        dec.append(chr((256 + ord(message[i]) - ord(key_c)) % 256))
    return "".join(dec)
def Mode():
    if(mode.get() == 'e'):
        Result.set(Encode(private_key.get(), Text.get()))
    elif(mode.get() == 'd'):
        Result.set(Decode(private_key.get(), Text.get()))
    else:
        Result.set('Invalid Mode')
def Exit():
    root.destroy()
def Reset():
    Text.set("")
    private_key.set("")
    mode.set("")
    Result.set("")
Label(root, font= 'arial 12 bold', text='MESSAGE').place(x= 60,y=60)
Entry(root, font = 'arial 10', textvariable = Text, bg = 'ghost
white').place(x=290, y = 60)
Label(root, font = 'arial 12 bold', text = 'KEY').place(x=60, y = 90)
Entry(root, font = 'arial 10', textvariable = private_key , bg
='ghost white').place(x=290, y = 90)
Label(root, font = 'arial 12 bold', text = 'MODE(e-encode, d-
decode)').place(x=60, y = 120)
Entry(root, font = 'arial 10', textvariable = mode , bg= 'ghost
white').place(x=290, y = 120)
Entry(root, font = 'arial 10 bold', textvariable = Result, bg = 'ghost
white').place(x=290, y = 150)
Button(root, font = 'arial 10 bold', text = 'RESULT' ,padx =2,bg
='LightGray' ,command = Mode).place(x=60, y = 150)
```

```
Button(root, font = 'arial 10 bold',text = 'RESET' ,width =6, command
= Reset,bg = 'LimeGreen', padx=2).place(x=80, y = 190)
Button(root, font = 'arial 10 bold',text= 'EXIT' , width = 6, command
= Exit,bg = 'OrangeRed', padx=2, pady=2).place(x=180, y = 190)
root.mainloop()
```

• المناقشة وطرائق البحث:

```
from tkinter import *
import base64
```

• الخطوة الأولى هي استيراد مكتبات Tkinter و Base64

```
root = Tk()
root.geometry('500x300')
root.resizable(0,0)
root.title("DataFlair - Message Encode and Decode")
```

- Tk () تهيئة مكتبة Tkinter تعني إنشاء النافذة
- Geometry () تعيين عرض وارتفاع النافذة
- resizable (0,0) ضبط الحجم الثابت للنافذة
- title () تحديد عنوان النافذة

```
• Label(root, text = 'ENCODE DECODE', font = 'arial 20
bold').pack()
Label(root, text = 'DataFlair', font = 'arial 20
bold').pack(side = BOTTOM)
```

- Label () عنصر واجهة المستخدم (اللائحة) الذي يستخدم لعرض سطر واحد أو أكثر من النص الذي لا يستطيع المستخدمون تعديله
- Root هو الاسم الذي يشير إلى نافذة المستخدم
- Text النص الذي نعرضه على اللائحة
- Font الخط الذي يكتب فيه النص
- Pack تنظم اللائحة ضمن النافذة

```
• Text = StringVar()
private_key = StringVar()
mode = StringVar()
Result = StringVar()
```

- Text متغير يخزن الرسالة لتشفيرها وفك تشفيرها

- **Private key** متغير يخزن المفتاح الخاص بالمستخدم للتشفير وفك التشفير
- **mode** يستخدم لتحديد ما إذا كان الخيار المستخدم هو تشفير أو فك تشفير
- **Result** يستخدم لتخزين النتيجة

```
def Encode(key,message):
    enc=[]
    for i in range(len(message)):
        key_c = key[i % len(key)]
        enc.append(chr((ord(message[i]) +
ord(key_c)) % 256))
    return
base64.urlsafe_b64encode("".join(enc).encode()).decode()
```

- نعرف تابع Encode يحتوي على بارامترين هما الرسالة المراد تشفيرها ومفتاح التشفير
- `enc = []` هي قائمة فارغة
- نقوم بتشغيل حلقة بطول الرسالة
- `key_c = key[i % Len(key)]` باقي قسمة `i` على طول المفتاح والناتج هو فهرس المفتاح ، قيمة المفتاح التي يدل عليها هذا الفهرس تخزن في `key_c`
- `ord()` تابع يأخذ بارامتر من النوع سلسلة من حرف Unicode واحد ويعيد قيمة Unicode الصحيحة
- `Chr()` تابع يأخذ بارامتر من نوع عدد صحيح ويعيد سلسلة
- `Ord(message[i])` تحويل قيمة الرسالة في الفهرس `i` إلى قيمة عدد صحيح
- `Ord(key_c)` تحويل قيمة `key_c` إلى قيمة عدد صحيح
- `Ord(message[i]) + ord(key_c) % 256` يعطي باقي قسمة مجموع `Ord`
- `(message[i])` و `Ord(key_c)` على العدد 256 ويمرر ناتج هذا الباقي إلى التابع `Chr()`
- `Chr()` تابع يحول تلك القيمة الصحيحة إلى سلسلة ويخزنها في القائمة الفارغة `enc`
- `base64.urlsafe_b64encode` تشفير سلسلة
- `join()` طريقة الانضمام: يضم كل عنصر من عناصر `list, string, tuple` بواسطة فاصل سلسلة ويعيد السلسلة المتسلسلة
- `Encode()` يقوم هذا التابع بإرجاع الرسالة المشفرة من السلسلة `utf-8`
- `Decode()` تابع يقوم بفك تشفير السلسلة
- **Return** يعطي نتيجة السلسلة المشفرة


```

• def Decode(key,message):
    dec=[]
    message =
    base64.urlsafe_b64decode(message).decode()
    for i in range(len(message)):
        key_c = key[i % len(key)]
        dec.append(chr((256 + ord(message[i]) -
ord(key_c)) % 256))
    return "".join(dec)

```

- Decode () تابع يحتوي على بارامترين هما الرسالة المراد تشفيرها ومفتاح التشفير
- Dec= [] هي قائمة فارغة
- فك تشفير المحتوى من الدخل وكتابة النتيجة بالثنائي في الخرج
- نقوم بتشغيل حلقة بطول الرسالة
- $(256 + \text{ord}(\text{message}[i]) - \text{ord}(\text{key_c})) \% 256$ يعطي باقي مجموع 256 مع ناتج طرح ل $\text{ord}(\text{message}[i]) - \text{ord}(\text{key_c})$ ثم القسمة على 256 ويمرر الباقي إلى التابع Chr ()
- Chr () تابع يحول تلك القيمة الصحيحة إلى سلسلة ويخزنها في القائمة الفارغة Dec
- return "" . join(dec) إرجاع النتيجة

```

def Mode():
    if(mode.get() == 'e'):
        Result.set(Encode(private_key.get(), Text.get()))
    elif(mode.get() == 'd'):
        Result.set(Decode(private_key.get(), Text.get()))
    else:
        Result.set('Invalid Mode')

```

- إذا كان الوضع الذي حدده المستخدم هو "e" فسيتم استدعاء التابع Encode ()
- إذا تم ضبط الوضع على "d" فسيتم استدعاء التابع Decode ()
- وإلا يطبع 'Invalid Mode'
- قيم يتم تمريرها إلى بارامترات Text.get () و private_key.get ()
- Decode () و Encode ()

```

def Exit():
    root.destroy()

```

- `root. Destroy ()` سيقوم بإنهاء البرنامج عن طريق إيقاف ال `mainloop`

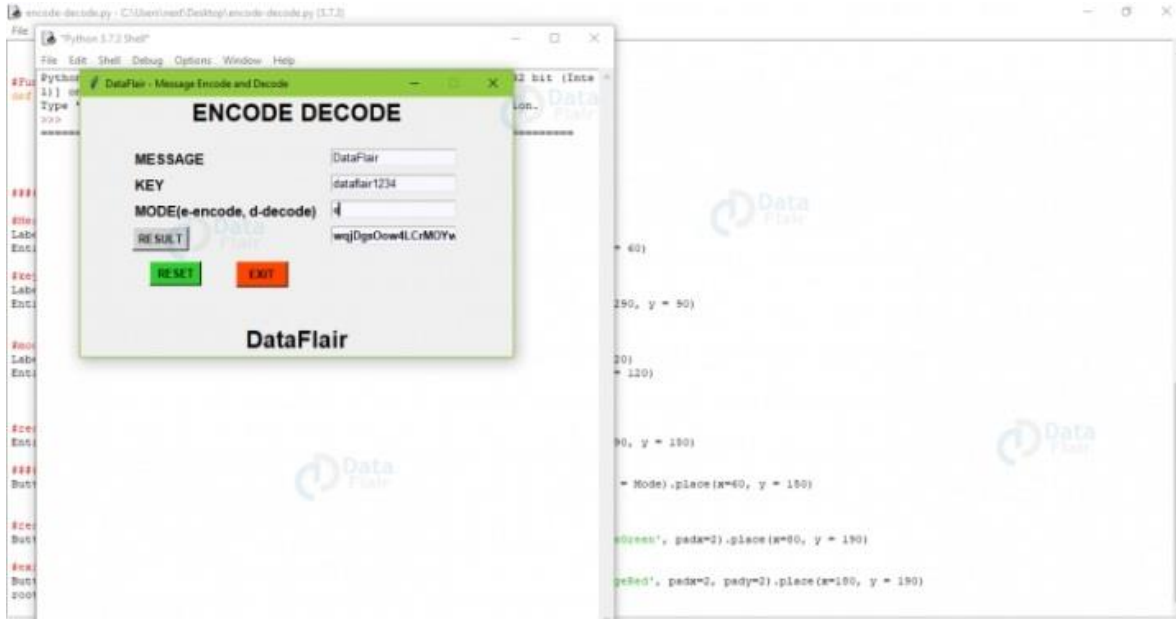
```
def Reset():
    Text.set("")
    private_key.set("")
    mode.set("")
    Result.set("")
```

- يقوم هذا التابع بتعيين جميع المتغيرات على سلسلة فارغة

```
Label(root, font= 'arial 12 bold',
text='MESSAGE').place(x= 60,y=60)
Entry(root, font= 'arial 10', textvariable = Text, bg =
'ghost white').place(x=290, y = 60)
Label(root, font= 'arial 12 bold', text
='KEY').place(x=60, y = 90)
Entry(root, font= 'arial 10', textvariable = private_key
, bg='ghost white').place(x=290, y = 90)
Label(root, font= 'arial 12 bold', text='MODE(e-encode,
d-decode)').place(x=60, y = 120)
Entry(root, font= 'arial 10', textvariable = mode , bg=
'ghost white').place(x=290, y = 120)
Entry(root, font= 'arial 10 bold', textvariable =
Result, bg='ghost white').place(x=290, y = 150)
Button(root, font= 'arial 10 bold', text= 'RESULT'
,padx =2,bg='LightGray' ,command = Mode).place(x=60, y =
150)
Button(root, font= 'arial 10 bold' ,text='RESET' ,width
=6, command = Reset,bg = 'LimeGreen', padx=2).place(x=80,
y = 190)
Button(root, font= 'arial 10 bold',text= 'EXIT' , width
= 6, command = Exit,bg = 'OrangeRed', padx=2,
pady=2).place(x=180, y = 190)
root.mainloop()
```

- `Label ()` عنصر واجهة المستخدم (اللائحة) الذي يستخدم لعرض سطر واحد أو أكثر من النص الذي لا يستطيع المستخدمون تعديله
- `Entry ()` عنصر واجهة المستخدم يستخدم لإنشاء حقل إدخال نصي
- `Button ()` عنصر واجهة المستخدم يستخدم لعرض زر على نافذتنا
- `Root` هو الاسم الذي يشير إلى نافذة المستخدم
- `Text` النص الذي نعرضه على اللائحة
- `Font` الخط الذي يكتب فيه النص
- `width` استخدام عرض الدخول لتعيين عرض مؤشر الإدراج

- **Bg** مجموعات لون الخلفية
- **command** هو الاتصال عند النقر على الزر
- **textvariable** يستخدم لاسترداد النص الحالي إلى عنصر واجهة المستخدم
- **root. Mainloop ()** تابع يتم تنفيذه عندما نريد تشغيل برنامجنا



المراجع:

- <https://data-flair.training/blogs/python-message-encode-decode/>
- <https://youtu.be/0NfDGNcuyAQ>
- <https://arabicprogrammer.com>
- <https://www.krsan4learn.com>