السلام عليكم و رحمة الله و بركاته معكم الاخ iladz

اليوم اريد ان اشارك معكم طريقة كسر برامج التنفيذية التي تم إنشاؤها عن طريق لغة بايثون

المطورة للبرنامج اخت من ليبيا أرسلت لي البرنامج من اجل التحقق من قوة الحماية ان غير قابلة للكسر

لهذا انا لن استطيع مشاركت الملف التنفيذي للبرنامج ساكتفي بوضع صور للكود تحقق من سيريال

البرنامج تنفيذي تم انشائه عن طريق الادات pyinstaller و تمت حمايته ب pyarmor حتى لا يتم اعادته للغة بيثون بسهولة

ليتم تحويل الملف تنفيذي الى ملفات pyc نستعمل الادات pyinstxtractor

رابط الاداة

https://github.com/extremecoders-re/pyinstxtractor

وبهذا سنتحصل على مجلد يحوي ملفات pyc وملفات pyd ....

سنهتم بالملف ذي يحمل اسم البرنامج و امتداده pyc هذا الملف هو الذي يحوي السكريبت المبرمج الذي سنقوم بتحليله واذا كان

يستعمل ملفات اخرى ايضا مكن تحليلها لتتبع خوارزيمة المستعملة

توجد ادات تحول ملفات pyc الى ملف py السكريبت المبرمج لاكن في حلتنا لن تعمل الادات لاستعمال المبرج ادات pyarmor

ادات التي تقوم بتحويل pyc الى سريبت بيثون هي pycdc

رابط الاداة

https://github.com/zrax/pycdc

لكن يوجد اداة مرفقة مع pycdc وهي الاداة pycdas التي تقوم بتحويل pyc الى لغة بايثون بايت كود (نوع من لغات الاسمبلي)

يجب الاطلاع مصادر لتعلم هذه اللغة بايثون بايت كود حتى تتمكن من فهم كود الناتج عن هذه الأداة

عندما نجرب اي مفتاح نجد رسالة الخطا بالغة العربية ان سيريال غير صحيح

عندما نبحث عن الرسالة في الملف الذي يحوي بايثون بايت كود نجدها

```
            34      LOAD_CONST                  1: 'نجاح'
            36      LOAD_CONST                  2: 'تم تفعيل التطبيق بنجاح!'
            38      CALL                        2
            46      RETURN_VALUE
[Code]
    File Name: 4Tik.py
    Object Name: <lambda>
    Qualified Name: TiktokOptimizerApp._run_activation_process.<locals>.<lambda>
    Arg Count: 0
    Pos Only Arg Count: 0
    KW Only Arg Count: 0
    Stack Size: 4
    Flags: 0x00000013 (CO_OPTIMIZED | CO_NEWLOCALS | CO_NESTED)
    [Names]
        'messagebox'
        'showerror'
    [Locals+Names]
    [Constants]
        None
        'خطأ'
        'مفتاح التفعيل غير صحيح أو غير مطابق لهذا الجهاز.'
    [Disassembly]
        0       RESUME                      0
        2       LOAD_GLOBAL                 0: messagebox
        12      LOAD_ATTR                   2: showerror
        32      PUSH_NULL
        34      LOAD_CONST                  1: 'خطأ'
        36      LOAD_CONST                  2: 'مفتاح التفعيل غير صحيح أو غير مطابق لهذا الجهاز.'
        38      CALL                        2
        46      RETURN_VALUE
[Code]
    File Name: 4Tik.py
    Object Name: <lambda>
    Qualified Name: TiktokOptimizerApp._run_activation_process.<locals>.<lambda>
    Arg Count: 1
    Pos Only Arg Count: 0
```

واسم لكلاس التي تحوي هذه الميثود حسب الكود هو

```
            354     LOAD_ATTR                   23: start
            374     CALL                        0
            382     POP_TOP
            384     RETURN_CONST                0: None
[Code]
    File Name: 4Tik.py
    Object Name: _run_activation_process
    Qualified Name: TiktokOptimizerApp._run_activation_process
    Arg Count: 2
    Pos Only Arg Count: 0
    KW Only Arg Count: 0
    Stack Size: 7
    Flags: 0x00000003 (CO_OPTIMIZED | CO_NEWLOCALS)
    [Names]
        '_verify_license_key'
        'current_hardware_id'
        'datetime'
        'now'
        'strftime'
        'base64'
        'b64encode'
        'json'
        'dumps'
        'encode'
        'open'
        'ACTIVATION_FILE'
        'write'
        'is_activated'
        'remaining_days'
        'after'
        'initialize_main_app_gui'
        'Exception'
        'hasattr'
    [Locals+Names]
        'self'
        'input_key'
```

هذه الميثود يتم استدعائها من العديد من الأماكن في البرنامج مكان لتحقق من المفتاح و مكان لي تحقق من ملف التسجيل بعد التسجيل الان سنهتم بالمكان الاول و كيف يقوم البرنامج باخذ سيريال من المستخدم و تمريره لهذه الميثود و حتى نتحقق اذا كان يتم تغيره اولا

بعد البحث نجد مكان الاول للاستدعاء هذه الدالة

```
readme.txt | details.txt | unpacked1.txt | Decryptor.py | details.txt | readme.txt | 4 lik.asm

164     LOAD_ATTR                13: configure
184     LOAD_CONST               3: 'disabled'
186     LOAD_CONST               4: ('state',)
188     CALL_KW                  1
190     POP_TOP
192     LOAD_FAST                0: self
194     LOAD_ATTR                0: activation_entry
214     LOAD_ATTR                13: configure
234     LOAD_CONST               3: 'disabled'
236     LOAD_CONST               4: ('state',)
238     CALL_KW                  1
240     POP_TOP
242     LOAD_FAST                0: self
244     LOAD_ATTR                14: activation_label_status
264     LOAD_ATTR                13: configure
284     LOAD_CONST               5: 'جاري التحقق من مفتاح التفعيل...'
286     LOAD_CONST               6: ('text',)
288     CALL_KW                  1
290     POP_TOP
292     LOAD_GLOBAL              16: threading
302     LOAD_ATTR                18: Thread
322     PUSH_NULL
324     LOAD_FAST                0: self
326     LOAD_ATTR                20: _run_activation_process
346     LOAD_FAST                1: input_key
348     BUILD_TUPLE              1
350     LOAD_CONST               7: ('target', 'args')
352     CALL_KW                  2
354     LOAD_ATTR                23: start
374     CALL                     0
382     POP_TOP
384     RETURN_CONST             0: None
[Code]
    File Name: 4Tik.py
```

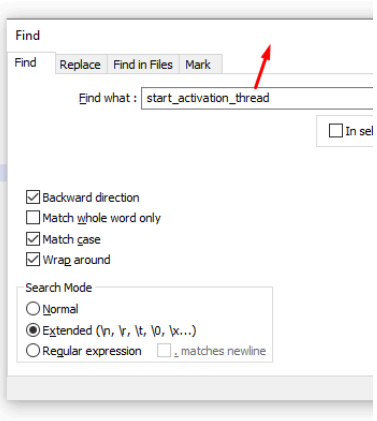
وعند النظر للأعلى نجد جملة جاري التحقق التي تظهر عند الضغط على زر تفعيل

بعد النظر للأعلى نجد ان اسم الدالة start_activation_thread

يبدو ان هذه الميثود هي التي يتم استدعائها عند الضغط على زر تفعيل نتحقق من ذلك عن طريق البحث عنها

وبالفعل هي الدالة التي يتم استدعائها عند الضغط على زر تفعيل

```
914     STORE_ATTR               12: activation_label_status
924     LOAD_DEREF               0: self
926     LOAD_ATTR                24: activation_label_status
946     LOAD_ATTR                13: pack
966     CALL                     0
974     POP_TOP
976     LOAD_GLOBAL              8: customtkinter
986     LOAD_ATTR                18: CTkButton
1006    PUSH_NULL
1008    LOAD_FAST                1: main_frame
1010    LOAD_CONST               47: 'تفعيـــل'
1012    LOAD_DEREF               0: self
1014    LOAD_ATTR                26: start_activation_thread
1034    LOAD_CONST               48: 40
1036    LOAD_CONST               49: ('Segoe UI', 16, 'bold')
1038    LOAD_CONST               50: ('text', 'command', 'height', 'font')
1040    CALL_KW                  5
1042    LOAD_DEREF               0: self
1044    STORE_ATTR               14: activation_button
1054    LOAD_DEREF               0: self
1056    LOAD_ATTR                28: activation_button
1076    LOAD_ATTR                13: pack
1096    LOAD_CONST               8: 20
1098    LOAD_CONST               21: 5
1100    LOAD_CONST               8: 20
1102    LOAD_CONST               51: ('pady', 'ipady', 'ipadx')
1104    CALL_KW                  3
1106    POP_TOP
```



لنذهب لدالة start_activation_thread و نبدأ بقراءة الكود من الاول الى الاخر

```
    )
[Disassembly]
    0       RESUME                  0
    2       LOAD_FAST               0: self
    4       LOAD_ATTR               0: activation_entry
    24      LOAD_ATTR               3: get
    44      CALL                    0
    52      LOAD_ATTR               5: strip
    72      CALL                    0
    80      STORE_FAST              1: input_key
    82      LOAD_FAST               1: input_key
    84      TO_BOOL
    92      POP_JUMP_IF_TRUE        23 (to 140)
    96      LOAD_GLOBAL             6: messagebox # في حالة لم يدخل مفتاح التفعيل
    106     LOAD_ATTR               8: showwarning
    126     PUSH_NULL
    128     LOAD_CONST              1: 'تحذير'
    130     LOAD_CONST              2: 'يرجى إدخال مفتاح التفعيل.'
    132     CALL                    2
    140     RETURN_VALUE
    142     LOAD_FAST               0: self
    144     LOAD_ATTR               10: activation_button
    164     LOAD_ATTR               13: configure
    184     LOAD_CONST              3: 'disabled'
    186     LOAD_CONST              4: ('state',)
    188     CALL_KW                 1
    190     POP_TOP
    192     LOAD_FAST               0: self
    194     LOAD_ATTR               0: activation_entry
    214     LOAD_ATTR               13: configure
    234     LOAD_CONST              3: 'disabled'
    236     LOAD_CONST              4: ('state',)
    238     CALL_KW                 1
    240     POP_TOP
    242     LOAD_FAST               0: self
    244     LOAD_ATTR               14: activation_label_status
    264     LOAD_ATTR               13: configure
```

هذه اسم نافذة الادخال
التي يقوم المستخدم بادخال السيريال فيها

للحصول على النص

يتم تخزينه في متغير الحلي
input_key

لتحويل السترنق الى bool

سيتم تنفيذ القفزة
اذا كان المستخدم
ادخل المفتاح
و سترينق ليست فارغة

سيتم تنفيذ هذا الكود في حال المستخدم
قام بادخال المفتاح

و بعدها نجد انه يتم استدعاء دالة التحقق من سيريال و اضافت input_key ك argument لدالة

```
    192     LOAD_FAST               0: self
    194     LOAD_ATTR               0: activation_entry
    214     LOAD_ATTR               13: configure
    234     LOAD_CONST              3: 'disabled'
    236     LOAD_CONST              4: ('state',)
    238     CALL_KW                 1
    240     POP_TOP
    242     LOAD_FAST               0: self
    244     LOAD_ATTR               14: activation_label_status
    264     LOAD_ATTR               13: configure
    284     LOAD_CONST              5: 'جاري التحقق من مفتاح التفعيل...'
    286     LOAD_CONST              6: ('text',)
    288     CALL_KW                 1
    290     POP_TOP
    292     LOAD_GLOBAL             16: threading
    302     LOAD_ATTR               18: Thread
    322     PUSH_NULL
    324     LOAD_FAST               0: self
    326     LOAD_ATTR               20: _run_activation_process
    346     LOAD_FAST               1: input_key
    348     BUILD_TUPLE             1
    350     LOAD_CONST              7: ('target', 'args')
    352     CALL_KW                 2
    354     LOAD_ATTR               23: start
    374     CALL                    0
    382     POP_TOP
    384     RETURN_CONST            0: None
[Code]
    File Name: 4Tik.py
    Object Name: _run_activation_process
    Qualified Name: TiktokOptimizerApp._run_activation_process
```

دالة تحقق من سيريال

يتم تمرير input_key
ك برماتر لدالة

سنذهب لدالة التحقق و نلقي نظرة

سنجد دوال داخلية 5 دوال

منهم دالة تظهر للمستخدم رسالة نجاح التفعيل و دالة تظهر رسالة فشل التفعيل و دالة تظهر فشل كتابة ملف التفعيل ...

بعدها ياتي الكود رئيسي لدالة

نرى ان يتم استدعاء دالة التحقق من سيريال _verify_license_key و لتي تقوم بارجاع عدد الأيام الممنوحة لاستخدام البرنامج

```
[Disassembly]
    0       MAKE_CELL                0: self
    2       RESUME                   0
    4       NOP
    6       LOAD_DEREF               0: self
    8       LOAD_ATTR                1: _verify_license_key
   28       LOAD_DEREF               0: self
   30       LOAD_ATTR                2: current_hardware_id
   50       LOAD_FAST                1: input_key
   52       CALL                     2
   60       STORE_FAST               2: duration_days        <-- تخزين عدد الايام في متغير المحلي
   62       LOAD_FAST                2: duration_days
   64       TO_BOOL
   72       POP_JUMP_IF_FALSE        211 (to 496)
   76       LOAD_DEREF               0: self
   78       LOAD_ATTR                2: current_hardware_id
   98       LOAD_FAST                1: input_key
  100       LOAD_GLOBAL              4: datetime
  110       LOAD_ATTR                6: now
  130       PUSH_NULL
  132       CALL                     0
  140       LOAD_ATTR                9: strftime
  160       LOAD_CONST               1: '%Y-%m-%d'
  162       CALL                     1
  170       LOAD_CONST               2: ('hwid', 'key', 'activation_date')
  172       BUILD_CONST_KEY_MAP      3
  174       STORE_FAST               3: activation_data
  176       LOAD_GLOBAL              10: base64
  186       LOAD_ATTR                12: b64encode
  206       PUSH_NULL
  208       LOAD_GLOBAL              14: json
  218       LOAD_ATTR                16: dumps
  238       PUSH_NULL
  240       LOAD_FAST                3: activation_data
```

و بعدها يقوم المبرمج بتحويل معلومات عن تفعيل برنامج الى json ثم يقوم بتشفيرها بستعمال خوارزمية base64 و يتم حفظها

في ملف التفعيل حتى يقوم في كل مرة يتم فتح البرنامج فيها اذا كان مسجل او لا

الان سنقوم بإلقاء نظرة على الدالة التحقق _verify_license_key التي تستقبل برمترين current_hardware_id و input_key

```
    1
[Disassembly]
    0       RESUME                  0
    2       LOAD_FAST               2: license_key
    4       LOAD_ATTR               1: split
    24      LOAD_CONST              1: '-'
    26      CALL                    1
    34      STORE_FAST              3: parts
    36      LOAD_GLOBAL             3: NULL + len
    46      LOAD_FAST               3: parts
    48      CALL                    1
    56      LOAD_CONST              2: 5
    58      COMPARE_OP              119 (!=)
    62      POP_JUMP_IF_FALSE       1 (to 66)
    66      RETURN_CONST            0: None
    68      LOAD_FAST               3: parts
    70      UNPACK_SEQUENCE         5
    74      STORE_FAST_STORE_FAST   69: key_prefix, key_hwid
    76      STORE_FAST_STORE_FAST   103: key_random, key_hash
    78      STORE_FAST              8: key_duration
    80      LOAD_FAST               4: key_prefix
    82      LOAD_CONST              3: '4TK'
    84      COMPARE_OP              88 (==)
    88      POP_JUMP_IF_FALSE       45 (to 180)
    92      LOAD_GLOBAL             3: NULL + len
    102     LOAD_FAST               5: key_hwid
    104     CALL                    1
    112     LOAD_CONST              4: 16
    114     COMPARE_OP              88 (==)
    118     POP_JUMP_IF_FALSE       30 (to 180)
    122     LOAD_GLOBAL             3: NULL + len
    132     LOAD_FAST               6: key_random
    134     CALL                    1
    142     LOAD_CONST              5: 8
    144     COMPARE_OP              88 (==)
    148     POP_JUMP_IF_FALSE       15 (to 180)
```

المفتاح

دالة split لفصل string
بستعمال '-'

تخزين السلاسل الناتجة في متغير المحلي

دالة len
التي تستعمل في حساب عدد العناصر

اذاكان عدد سلال ليس 5 يتم اعادة اقيمة0
دليل على فشل عملية تسجيل

تخزين السلاسل 5 ناتجة
في 5 متغيرات محلية

يجب ان تكون سلسلة الاولى
'4TK'

يجب ان يكون طوله 16

يجب ان يكون طوله 8

```
    144     COMPARE_OP              88 (==)
    148     POP_JUMP_IF_FALSE       15 (to 180)
    152     LOAD_GLOBAL             3: NULL + len
    162     LOAD_FAST               7: key_hash
    164     CALL                    1
    172     LOAD_CONST              5: 8
    174     COMPARE_OP              88 (==)
    178     POP_JUMP_IF_TRUE        1 (to 182)
    182     RETURN_CONST            0: None
    184     LOAD_FAST_LOAD_FAST     81: key_hwid, hardware_id
    186     COMPARE_OP              119 (!=)
    190     POP_JUMP_IF_FALSE       1 (to 194)
    194     RETURN_CONST            0: None
    196     LOAD_FAST               1: hardware_id
    198     FORMAT_SIMPLE
    200     LOAD_CONST              1: '-'
    202     LOAD_FAST               6: key_random
    204     LOAD_ATTR               5: lower
    224     CALL                    0
    232     FORMAT_SIMPLE
    234     LOAD_CONST              6: '-Manal4TiktokSecretPhrase-'
    236     LOAD_GLOBAL             6: SECRET_SALT
    246     FORMAT_SIMPLE
    248     BUILD_STRING            5
    250     STORE_FAST              9: data_to_hash
    252     LOAD_GLOBAL             8: hashlib
    262     LOAD_ATTR               10: sha256
    282     PUSH_NULL
    284     LOAD_FAST               9: data_to_hash
    286     LOAD_ATTR               13: encode
    306     LOAD_CONST              7: 'utf-8'
    308     CALL                    1
    316     CALL                    1
    324     LOAD_ATTR               15: hexdigest
    344     CALL                    0
    352     STORE_FAST              10: re_generated_hash
    354     LOAD_FAST_LOAD_FAST     122: key_hash, re_generated_hash
```

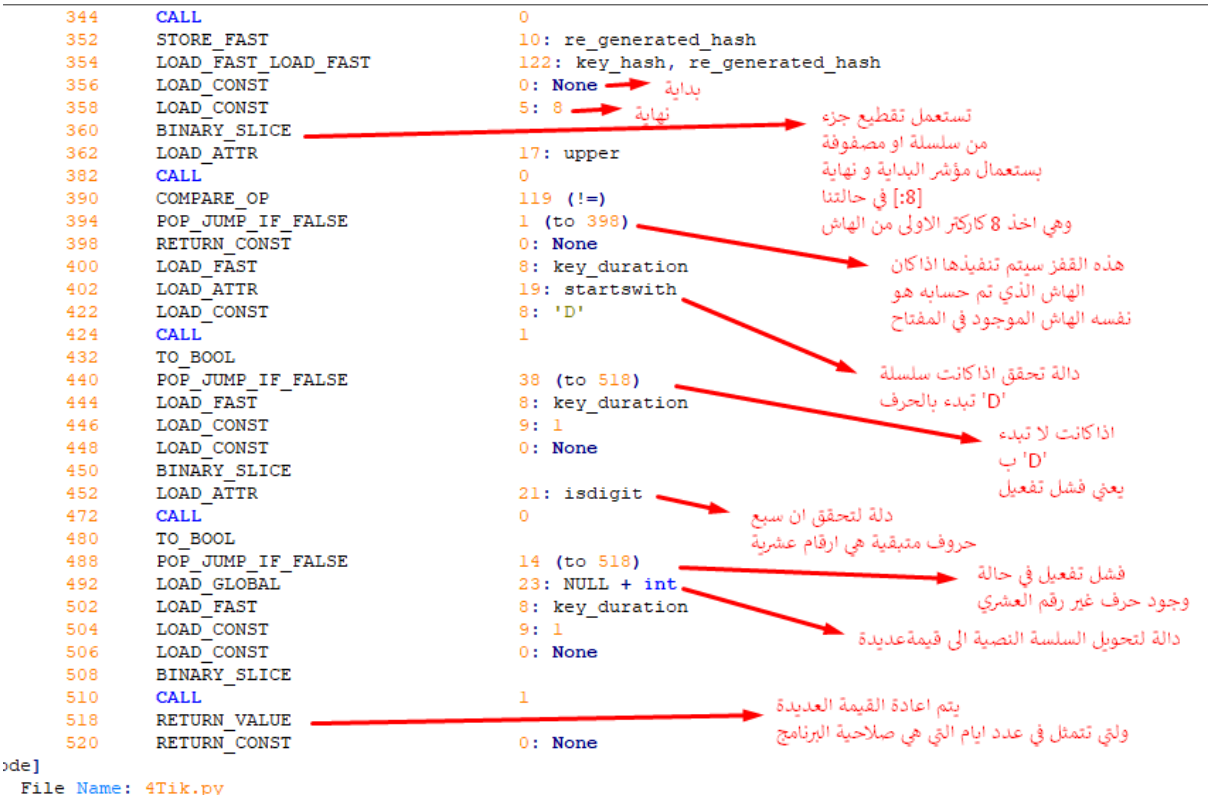
يجب ان يكون طوله 8

يجب ان يكون
hwid
المدخل عن طريق المفتاح نفسه
الذي تم حسابه عن طريق البرنامج

هنا يتم دمج
5 سلاسل
و حفظها في متغير المحلي
data_to_hash

استعمال خوارزمية
sha256
لحساب الهاش

تخزين الهاش في متغير محلي



بعد شرح الموجود في الصور

يجب ان يكون

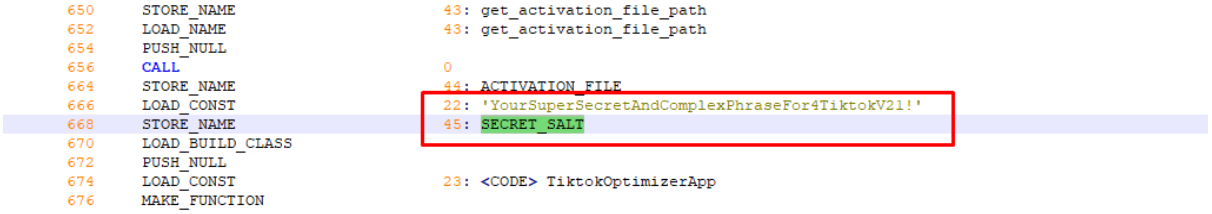
key_prefix = '4TK'

data_to_hash =
<hardwar_id>+'-'+<key_random>+'-Manal4TiktokSecretPhase-'+SECRET_SALT

key_hash = sha256(data_to_hash)

key_duration = D<number of days>

وبعد البحث نجد قيمة SECRET_SALT



SECRET_SALT = 'YourSuperSecretAndComplexPhraseFor4TiktokV21!'

يوجد عيبين في هذا النظام

الاول ان key_duration ليست مجدرجة في حساب الهاش و هذا يتيح لمستخدم تغيره كما يشاء و يبقى المفتاح صحيح

اما الثاني فهو ان صاحبة البرنامج تريد ان يكون للمفتاح صلاحية و نسيت ان تظيف تاريخ صدور المفتاح الى المفتاح

هكذا البرنامج يستطيع معرفة اذا كان المفتاح منتهي صلاحية ام لا

هنا يستطيع الزبون عند انقضاء مدة تفعيل البرنامج استعمال المفتاح القديم و سيعمل معه المفتاح بلا اي مشاكل

الحل ان تستحدث key_date تضع فيها تاريخ اليوم على قيمة timestamp

و تقوم بادراج key_date و key_duration في حساب الهاش

يعني ان يكون

data_to_hash = <hardwar_id>+'-'+<key_random>+'-Manal4TiktokSecretPhase-'+SECRET_SALT+key_duration +key_date

و هكذا المستخدم لن يستطيع تغيرهم كما يشاء و يستعمل البرنامج بالمجان

هذا كود البيثون الذي كتبته لي يقوم بتوليد المفاتيح keygen

```python
import hashlib
import random
import string

def generate_random_string(length):
    """
    Generates a random alphanumeric string of a given length.
    """
    characters = string.ascii_letters + string.digits  # All letters (upper and lower) and digits
    random_string = ''.join(random.choice(characters) for i in range(length))
    return random_string

def calculate_sha256_hash(input_string):
    """
    Calculates the SHA-256 hash of a given string.

    Args:
        input_string (str): The string to be hashed.
```

```
    Returns:
        str: The hexadecimal representation of the SHA-256 hash.
    """
    # Create a new SHA-256 hash object
    sha256_hash_object = hashlib.sha256()

    # Update the hash object with the bytes-like object (encoded
string)
    # It's crucial to encode the string to bytes, as hash functions
operate on bytes.
    sha256_hash_object.update(input_string.encode('utf-8'))

    # Return the hexadecimal digest of the hash
    return sha256_hash_object.hexdigest()

key_prefix = '4TK'
SECRET_SALT1 = '-Manal4TiktokSecretPhrase-'
SECRET_SALT2 = 'YourSuperSecretAndComplexPhraseFor4TiktokV21!'

key_random = generate_random_string(8)

print(f"key random string: {key_random}")

serial = input("Please enter your serial: ")

key_random_lower = key_random.lower()
data_to_hash = serial + '-' + key_random_lower + SECRET_SALT1 +
SECRET_SALT2

print(f"data_to_hash: {data_to_hash}")

re_generated_hash = calculate_sha256_hash(data_to_hash)

print(f"re_generated_hash: {re_generated_hash}")

key_hash = re_generated_hash[:8].upper()

print(f"key_hash: {key_hash}")

days = input("Please enter number of days: ")
num_days = int(days)
padded_string = str(num_days).zfill(7)
```

```
key_duration = 'D'+padded_string

print(f"key_duration: {key_duration}")



serial2  = key_prefix + '-' + serial + '-' + key_random + '-' +
key_hash + '-' + key_duration

print(f"serial: {serial2}")
```

اذا اعجبك درس اليوم اترك رد او اعجاب على منتدى www.at4re.net

^_^