به نام خدا

دانشکده مهندسی کامپیوتر

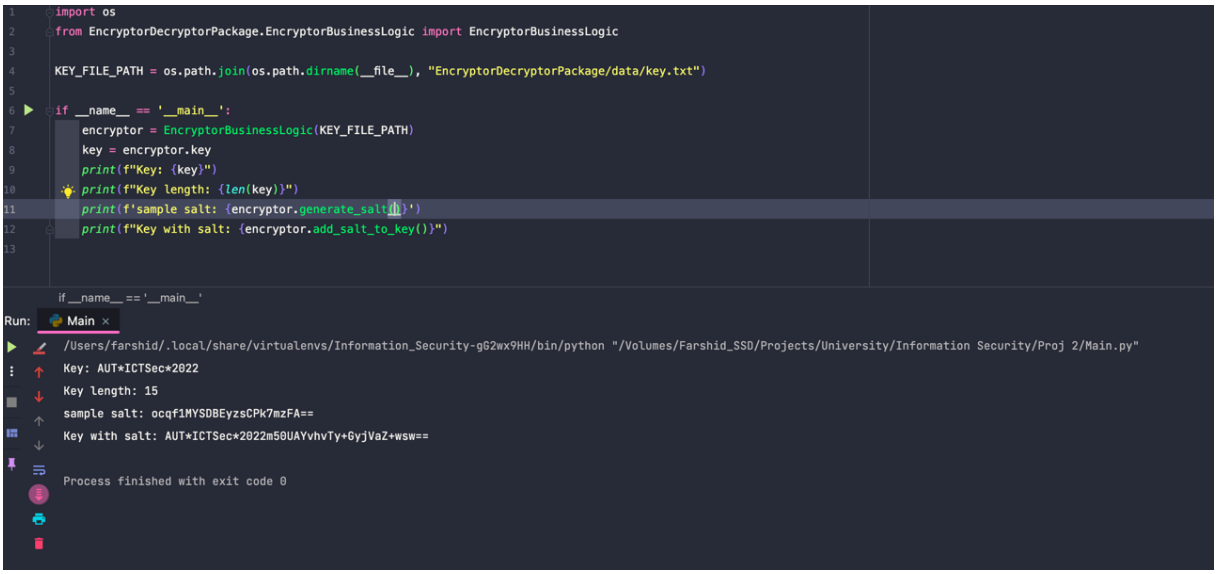
گزارش تمرین عملی دوم

درس: مبانی امنیت اطلاعات

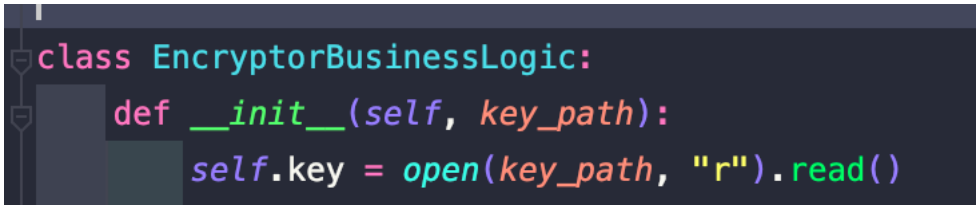
دانشجو: فرشید نوشی – ۹۸۳۱۰۶۸

گزارش

با توجه به نیازمندی مطرح شده در این بخش با توجه به کلید مطرح شده در بخش ۸ تمرین، مقدار کلید در یک فایل متنی به عنوان key.txt ذخیره شد و سپس با استفاده از کتابخانه os همانند نیازمندی مطرح شده یک میزان salt با استفاده از تابع urandom(16) و انکودینگ در base64 انجام شد. خروجی نیز در تصویر زیر آمده است.



در بخش دوم نیز به علت نیازمندی موجود در مورد عدم hard code کردن کلید در برنامه در فایل متنی گفته شده در بالا گذاشته شد. و در تصویر پایین نیز همانگونه که مشاهده میشود مقدار کلید از روی فایل خوانده میشود.



در بخش سوم نیز با استفاده از کتابخانه pbkdf2 و binascii فعالیت‌های خواسته شده انجام شدند و طول کلید به ۲۵۶ بیت (۳۲ بایت) رسانده شد و به اپراتور نیز در دمو برنامه در خروجی کنسول نشان داده شد.

```python
def change_key_size(self, key_size):
    """
    Change the key size to the given key size

    :param key_size: the key size to change to (in bits)
    :return: the new key
    """
    self.key = pbkdf2.PBKDF2(self.key, self.key).read(key_size // 8)
    return self.key

def show_hex_key(self):
    """
    Show the key in hexadecimal
    :return: the key in hexadecimal
    """
    return binascii.hexlify(self.key)
```

```
Key: AUT*ICTSec*2022
Key length: 15
sample salt: /ezZaQMNleO4SbHZqDlKUw==
Key with salt: AUT*ICTSec*2022GNp1PMo6BW7tiTK3H7udQA==
----------------------------------------------------------------------------------------------------
Key with salt length in bytes: 39
Key with salt and key size of 256 bits: b'=\x83\xa4\xd34\xf3\x0f|\x8b\xc9\x16\x8a@3\xc3\xed\xb1\xf0\xb7s\xf8\x81\xc0hC\x94\xc8\xaa\x1f}\x95\xa3'
Key with salt and key size of 256 bits length in bytes: 32
Key with salt and key size of 256 bits in hexadecimal: b'3d83a4d334f30f7c8bc9168a4033c3edb1f0b773f881c0684394c8aa1f7d95a3'
Key with salt and key size of 256 bits in hexadecimal length: 64
```

با استفاده از کتابخانه secrets ما یک initial vector نیز ساختیم که به شرح زیر در خروجی دمو برنامه نشان داده شده است و در aes برای ctr نیز نشان داده میشود. در تصویر البته برای زیبایی تصاویر، طول بردار برابر ۱۶ گذاشته شده است. در خروجی متنی کنسول که از برنامه گرفته شده است برای تصحیح، طول این بردار برابر ۱۲۸ گذاشته شده است.

```python
    self.initial_vector = secrets.token_bytes(size)
    return self.initial_vector
```

```python
print(f"Key with salt and key size of 256 bits in hexadecimal length in bytes: {len(encryptor.show_hex_key())}")
separate_lines_in_terminal()
print(f"Initial vector for ctr mode: {encryptor.generate_initial_vector_for_ctr_mode()}")
print(f"Initial vector for ctr mode length in bytes: {len(encryptor.generate_initial_vector_for_ctr_mode())}")
```

```
Run: Main
Key length: 15
sample salt: 01967rySu3QHOMPrdYEorg==
Key with salt: AUT*ICTSec*2022ui9ILSMPa+AR8b5kzM1ebA==
----------------------------------------------------------------------------------------------------
Key with salt length in bytes: 39
Key with salt and key size of 256 bits: b'\x1d\xdc\x12\xab$\xa9\x9e\x0f\xf3\xa6\x08f/\xb1\xc9\xd8=\xd8\xc0\xd3\x13\x8f^\xc6d\xd1%\xeb\xf9\xe4\xa8='
Key with salt and key size of 256 bits length in bytes: 32
Key with salt and key size of 256 bits in hexadecimal: b'1ddc12ab24a99e0ff3a608662fb1c9d83dd8c0d3138f5ec664d125ebf9e4a83d'
Key with salt and key size of 256 bits in hexadecimal length in bytes: 64
----------------------------------------------------------------------------------------------------
Initial vector for ctr mode: b'\xf4[&\xa9\x80\x18\x14\xbbcNH\x1e\xbc7*\xc8'
Initial vector for ctr mode length in bytes: 16

Process finished with exit code 0
```

در بخش بعدی متن برای رمز گذاری از روی فایل خوانده میشود و عمل رمزگذاری با استفاده از pyaes و دادن ورودی‌های لازم به ان انجام میشود و خروجی نیز در فایل ذخیره میشود. فایلهای ciphertext.enc, ciphertext.txt مربوط به متن کدگذاری شده یکی به صورتی بیتی ذخیره شده و دیگری به صورت متنی. فایل‌های date.iv, data.key نیز مربوط به initial vector, key هستند که در پوشه program در داخل پوشه داده‌ها ذخیره میشوند تا در هنگام decryption بتوانیم اطلاعاتی که با آن‌ها رمزگذاری کرده‌ایم را داشته باشیم.

```
aes = pyaes.AESModeOfOperationCTR(self.key, pyaes.Counter(int_initial_vector))
return aes.encrypt(plaintext)
```

```
Encrypting...
Plaintext: 9831068
Initial vector: 290136451790434880687579864387990349002
Key: b'\x16\x1bw\x88\xe3\x05i20on\xc2\xab\x82\x88\x1f\xc5\xe0\xf5\xe5\x83^\x91\x9e\x1aE\xa0\x9e\xe1q\xea\xad'
Ciphertext: b'\xbc`\xcb\xcdZ$S'
Encrypted text: b'\xbc`\xcb\xcdZ$S'
Encrypted text length in bytes: 7
----------------------------------------------------------------------------------------------------
saving key to file: Saved data to file
----------------------------------------------------------------------------------------------------
```

همانطور که در تصویر بالا مشاهده میشود مقدار plaintext برابر شماره دانشجویی میباشد که از فایل متنی خوانده شده است. نام فایل متنی در دموی تمرین text.txt میباشد. بعد از این عملیات در قسمت بعدی برای رمزگشایی از متن کدگذاری شده وارد عمل شدیم و اطلاعات از روی فایل‌های متنی خوانده شدند، در ادامه نیز با دادن ورودی های مناسب به تابع برای decrypt این کار را انجام دادیم و تصویر زیر خروجی نمونه کار است.

```
Decrypting...
Ciphertext: b'\xbc`\xcb\xcdZ$S'
Initial vector: 290136451790434880687579864387990349002
Key: b'\x16\x1bw\x88\xe3\x05i20on\xc2\xab\x82\x88\x1f\xc5\xe0\xf5\xe5\x83^\x91\x9e\x1aE\xa0\x9e\xe1q\xea\xad'
Plaintext: b'9831068'
Decrypted text: b'9831068'
Decrypted text length in bytes: 7
```

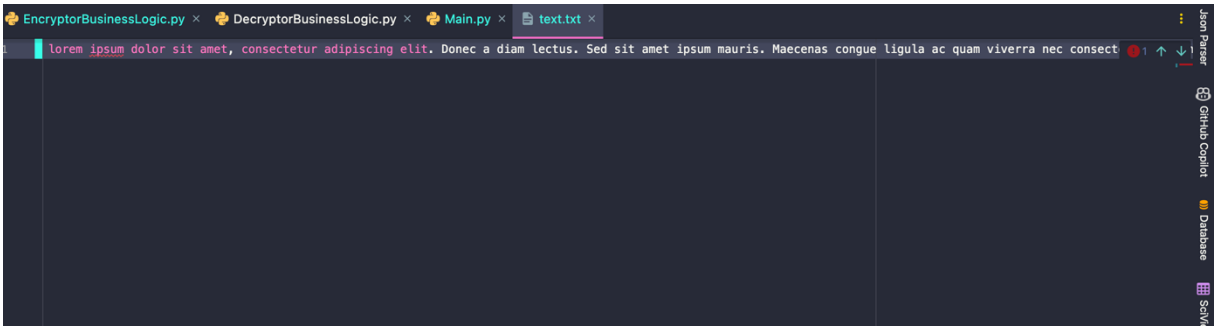
خروجی قسمت باز کردن کد در بخش آخر در فایل decrypted.dec نیز نوشته میشود تا از آن استفاده بشود.

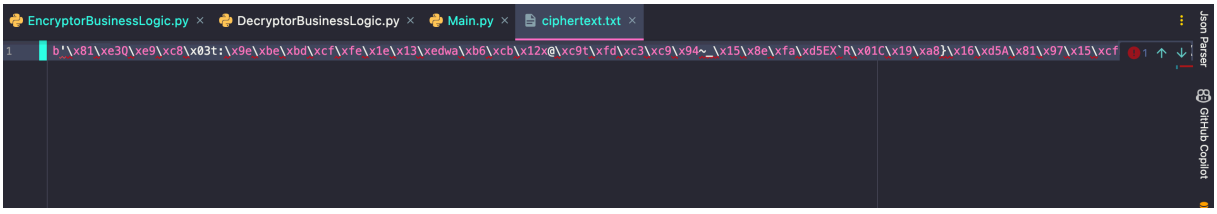
در پایان بخش دمو البته تمامی فایل‌های ساخته شده برای این بخش پاک میشوند اما در بخش کنسول برنامه که برنامه اجرا میشود باید درخواست توسط کاربر برای پاک کردن آن‌ها داده بشود. در زیر تصویر کامل دموی تمرین آورده شده است که در ان بخش به بخش سعی شده است که مراحل آورده بشوند.

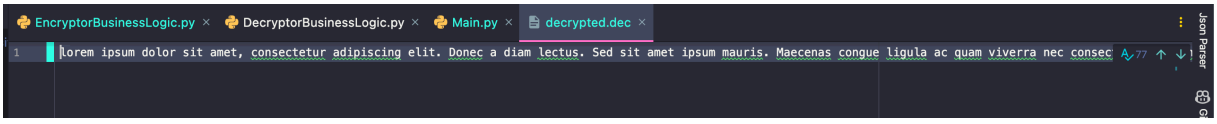
```
 ____
|  _ \  ___ _ __ ___   ___
| | | |/ _ \ '_ ` _ \ / _ \
| |_| |  __/ | | | | | (_) |
|____/ \___|_| |_| |_|\___/


Key: AUT*ICTSec*2022
Key length: 15
sample salt: Dgwqvqek0WX8zsj6pxObAg==
Key with salt: AUT*ICTSec*2022CLp2Q14nUWgpt7y3jQVdVg==
-----------------------------------------------------------------------------------------------
Key with salt length in bytes: 39
Key with salt and key size of 256 bits: b'\x16\x1bw\x88\xe3\x05i20on\xc2\xab\x82\x88\x1f\xc5\xe0\xf5\xe5\x83^\x91\x9e\x1aE\xa0\x9e\xe1q\xea\xad'
Key with salt and key size of 256 bits length in bytes: 32
Key with salt and key size of 256 bits in hexadecimal: b'161b7788e3056932306f6ec2ab82881fc5e0f5e5835e919e1a45a09ee171eaad'
Key with salt and key size of 256 bits in hexadecimal length in bytes: 64
-----------------------------------------------------------------------------------------------
Initial vector for ctr mode: b'dE\xcc#/d\x1bq\x0c\x11\xaa\xe0\xe0\x0f(\x1c'
Initial vector for ctr mode length in bytes: 16
-----------------------------------------------------------------------------------------------
Encrypting...
Plaintext: 9831068
Initial vector: 290136451790434880687579864387990349002
Key: b'\x16\x1bw\x88\xe3\x05i20on\xc2\xab\x82\x88\x1f\xc5\xe0\xf5\xe5\x83^\x91\x9e\x1aE\xa0\x9e\xe1q\xea\xad'
Ciphertext: b'\xbc`\xcb\xcdZ$S'
Encrypted text: b'\xbc`\xcb\xcdZ$S'
Encrypted text length in bytes: 7
-----------------------------------------------------------------------------------------------
saving key to file: Saved data to file
-----------------------------------------------------------------------------------------------
Decrypting...
Ciphertext: b'\xbc`\xcb\xcdZ$S'
Initial vector: 290136451790434880687579864387990349002
Key: b'\x16\x1bw\x88\xe3\x05i20on\xc2\xab\x82\x88\x1f\xc5\xe0\xf5\xe5\x83^\x91\x9e\x1aE\xa0\x9e\xe1q\xea\xad'
Plaintext: b'9831068'
Decrypted text: b'9831068'
Decrypted text length in bytes: 7
-----------------------------------------------------------------------------------------------
Deleting demo files...
Demo files deleted.
```

در ادامه نیز سعی بر توسعه یک ابزار تحت کنسول شد، در این ابزار نیز کلید ها در تصویر زیر معنی آن ها آورده شده است که e, d, r, q برای خروج، پاک سازی فایل‌ها، decryption و encryption می‌باشند.

در ادامه نیز یک‌بار برنامه با استفاده از متنی دلخواه اجرا شد و خروجی نمونه آن در زیر آورده شده است که تصویر اول متن ورودی، دومی متن کدشده و سومی بازگردانده متن کدشده هستند:







برای تمرین دو فایل businesslogic توسعه داده شدند که یکی برای کدگذاری و دیگری برای برگرداندن کدها هستند. هر دو فایل doc دارند تا تمامی متدهایی که دارند قابل خواندن و توسعه باشند.

EncryptorBusinessLogic.py × DecryptorBusinessLogic.py × Main.py ×

```python
class EncryptorBusinessLogic:
    def __init__(self, key_path):
        self.key = open(key_path, "r").read()
        self.initial_vector = None

    @staticmethod
    def generate_salt():
        """
        Generate a salt
        :return: the salt
        """
        salt = os.urandom(16)
        return str(base64.b64encode(salt))[2:-1]

    def add_salt_to_key(self):
        """
        Add a salt to the key and return the new key
        :return: the new key
        """
        salt = self.generate_salt()
        self.key = self.key + salt
        return self.key

    def change_key_size(self, key_size):
        """
        Change the key size to the given key size

        :param key_size: the key size to change to (in bits)
        :return: the new key
        """
        self.key = pbkdf2.PBKDF2(self.key, self.key).read(key_size // 8)
        return self.key

    def show_hex_key(self):
        """
        Show the key in hexadecimal
        :return: the key in hexadecimal
        """
        return binascii.hexlify(self.key)

    def generate_initial_vector_for_ctr_mode(self, size=128):
        """
        Generate an initial vector for ctr mode encryption
```

EncryptorBusinessLogic.py | DecryptorBusinessLogic.py | Main.py

```python
class DecryptorBusinessLogic:
    def __init__(self, program_data_path):
        self.program_data_path = program_data_path
        self.key = open(program_data_path + ".key", "rb").read()
        self.initial_vector = open(program_data_path + ".iv", "rb").read()

    def __decrypt(self, ciphertext):
        """
        Decrypt the given ciphertext
        :param ciphertext: the ciphertext to decrypt
        :return: the plaintext
        """
        int_initial_vector = int.from_bytes(self.initial_vector, byteorder="big")
        print("Initial vector: " + str(int_initial_vector))
        print("Key: " + str(self.key))
        aes = pyaes.AESModeOfOperationCTR(self.key, pyaes.Counter(int_initial_vector))
        return aes.decrypt(ciphertext)

    def decrypt(self, ciphertext_path):
        """
        Decrypt the given ciphertext file
        :param ciphertext_path: the ciphertext file to decrypt
        :return: the plaintext
        """
        ciphertext = open(ciphertext_path, "rb").read()
        print("Decrypting...")
        print("Ciphertext: " + str(ciphertext))
        plaintext = self.__decrypt(ciphertext)
        print("Plaintext: " + str(plaintext))
        write_path = os.path.join(os.path.dirname(ciphertext_path), "decrypted")
        self.write_to_file(plaintext, write_path + ".dec")
        # self.write_to_file(plaintext, write_path + ".txt", "wb")
        return plaintext

    @staticmethod
    def write_to_file(data, file_path, mode="wb"):
        """
        Write the given data to the given file path
        :param data: the data to write in bytes
        :param file_path: the file path to write to
        :param mode: the mode to write in
        """
        with open(file_path, mode) as file:
```