AES Encryption/Decryption Utility Report

Operating System:

- Windows 10

Programming Language and Version:

- Python 3.11.1

Overview:

This project provides an AES encryption and decryption utility that supports the following operations:

- Key generation
- Encryption
- Decryption

AES encryption is performed using the Cipher Block Chaining (CBC) mode with a 256-bit key.

Directory Structure:

The project is organized as follows:
bash
Copy code

```
./aes_m13349332/
│
├── src/
│   ├── aes.py
│
├── data/
│   ├── plaintext.txt
│   ├── key.txt
│   ├── iv.txt
│   ├── ciphertext.txt
│   └── result.txt
│
└── report.pdf
```

Compilation & Execution Instructions:

Since the code is written in Python, there is no need for compilation. Directly run the Python script. Here are the steps:

Navigate to the directory containing aes.py using terminal or command prompt.

Run the script using one of the following commands based on the desired operation:

- For key generation:

```
python aes.py keygen ../data/key.txt
```

- For encryption:

```
python aes.py enc ../data/key.txt ../data/plaintext.txt ../data/ciphertext.txt
```

- For decryption:

```
python aes.py dec ../data/key.txt ../data/ciphertext.txt ../data/result.txt
```

-

Functionality:

1. aes_keygen:

- Purpose: Generates a random 256-bit key for AES.
- Inputs: None.
- Outputs: Prints the generated key to the terminal and writes it to ../data/key.txt in hexadecimal format.

2. aes_enc:

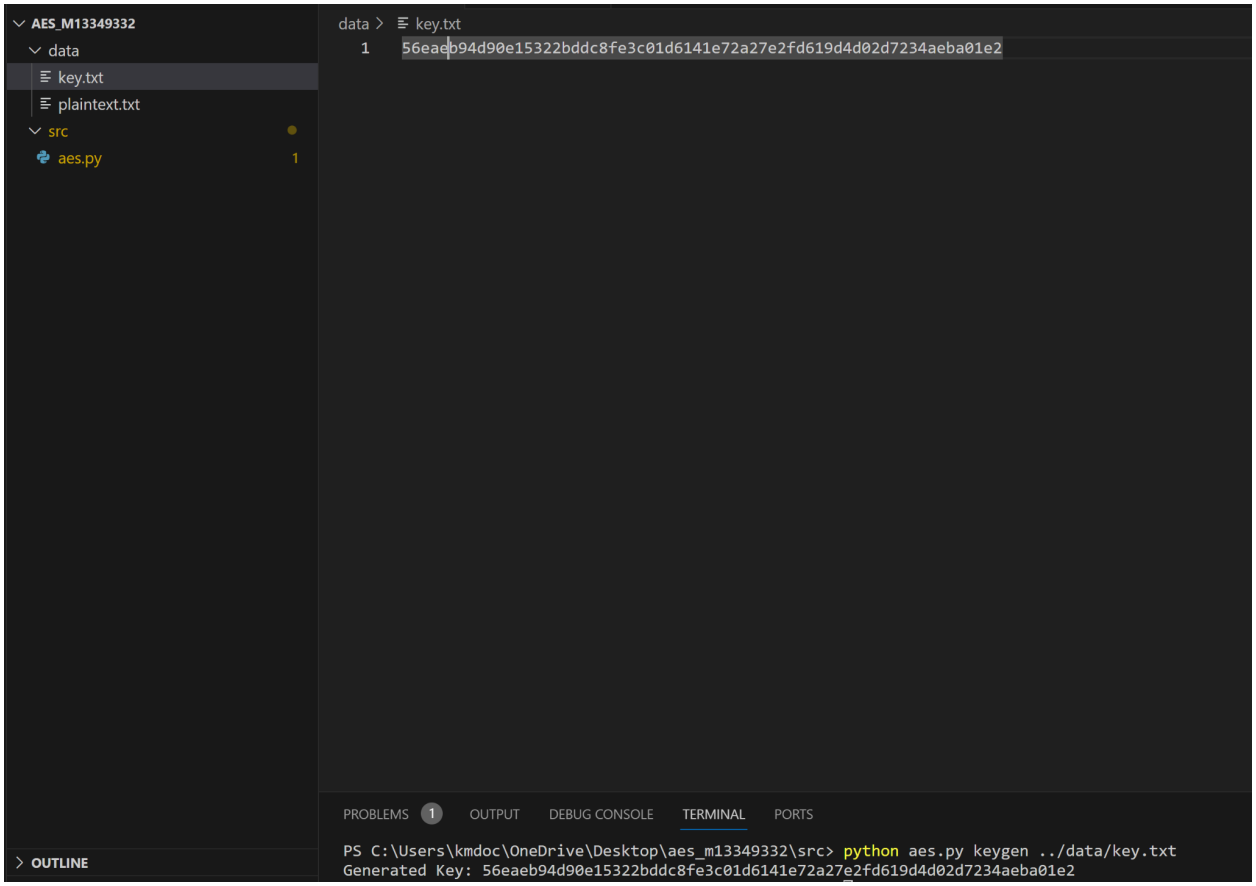
- Purpose: Encrypts a given plaintext using AES in CBC mode.
- Inputs:
  - Path to the secret key.
  - Path to the plaintext.
  - Path to save the generated ciphertext.
  - Path to save the generated IV.
- Outputs: Encrypted ciphertext written to ciphertext_path and IV written to iv_path.

3. aes_dec:

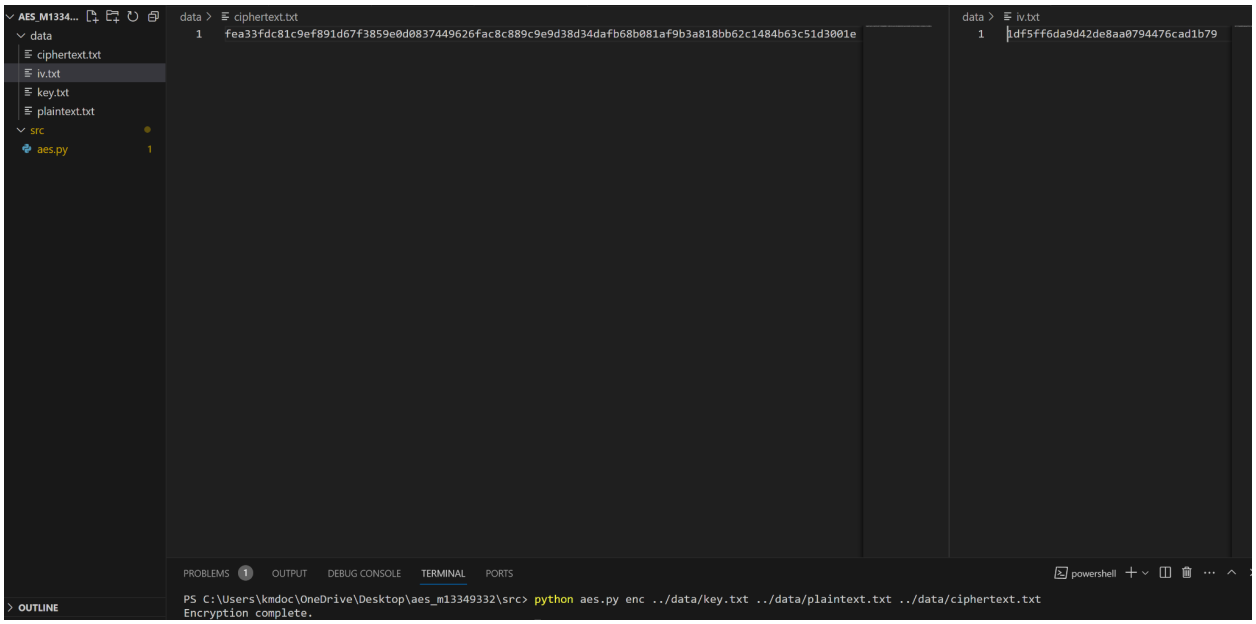
- Purpose: Decrypts a given ciphertext using AES in CBC mode.
- Inputs:
  - Path to the secret key.
  - Path to the IV.
  - Path to the ciphertext.
  - Path to save the decrypted plaintext.
- Outputs: Decrypted plaintext written to result_path.

Screenshots:

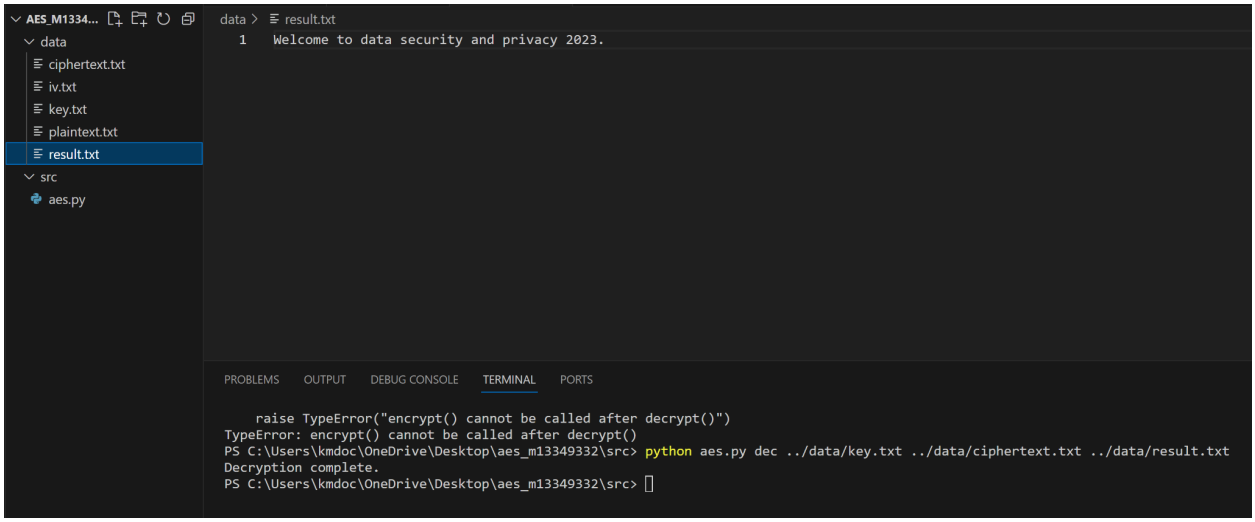
Key gen:


AES_M13349332
data
key.txt
plaintext.txt
src
aes.py
data > key.txt
1 56eaeb94d90e15322bddc8fe3c01d6141e72a27e2fd619d4d02d7234aeba01e2
OUTLINE
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS C:\Users\kmdoc\OneDrive\Desktop\aes_m13349332\src> python aes.py keygen ../data/key.txt
Generated Key: 56eaeb94d90e15322bddc8fe3c01d6141e72a27e2fd619d4d02d7234aeba01e2

Enc:


data > ciphertext.txt
1 fea33fdc81c9ef891d67f3859e0d0837449626fac8c889c9e9d38d34dafb68b081af9b3a818bb62c1484b63c51d3001e
data > iv.txt
1 1df5ff6da9d42de8aa0794476cad1b79
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS C:\Users\kmdoc\OneDrive\Desktop\aes_m13349332\src> python aes.py enc ../data/key.txt ../data/plaintext.txt ../data/ciphertext.txt
Encryption complete.

Dec:


AES_M1334...
data
ciphertext.txt
iv.txt
key.txt
plaintext.txt
result.txt
src
aes.py
data > result.txt
1 Welcome to data security and privacy 2023.
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
raise TypeError("encrypt() cannot be called after decrypt()")
TypeError: encrypt() cannot be called after decrypt()
PS C:\Users\kmdoc\OneDrive\Desktop\aes_m13349332\src> python aes.py dec ../data/key.txt ../data/ciphertext.txt ../data/result.txt
Decryption complete.
PS C:\Users\kmdoc\OneDrive\Desktop\aes_m13349332\src>