Private Key Encryption

Using Symmetric Encryption algorithm

1st Rohan Katkam
Master of Computer Science
New Jersey Institution of Technology
UCID: rk944

2nd Sathvika Karri
Master of Computer Science
New Jersey Institution of Technology
UCID: sk3628

1. INTRODUCTION

Throughout data security disciplines, the Secure Data Encryption and Decryption Utility serves as an operational and instructional tool. It offers a comprehensive approach to data security management by combining SHA-256 for secure key creation, Excel systematic logging, and AES-EAX for strong encryption. This tactic not only strengthens the encryption process but also teaches users important lessons about the value of data integrity, encryption, and key management—all of which are critical for safeguarding sensitive information in the modern digital environment.

1. 2. OBJECTIVES:

The primary objectives of the Secure Data Encryption and Decryption Utility are:

- **To provide a robust tool for encrypting and decrypting textual data**, ensuring that sensitive information remains protected against unauthorized access.

- **To educate users on the importance and implementation of encryption**, fostering better understanding and practices around data security.

- **To facilitate traceability and management of encrypted data**, by logging detailed encryption activities, including timestamps and key details, in an Excel file.

1.3. SYSTEM ARCHITECTURE:

- **Key Management Module**:

Manages all aspects of cryptographic key handling, including key generation and hashing. Users can generate a random 256-bit key through a Cryptographically Secure Pseudo-Random Number Generator (CSPRNG) or input their own key, which is hashed using SHA-256 to produce a fixed-size hash value suited for AES-256 encryption.

- **Encryption/Decryption Engine**:

The core component utilizes AES-256 in EAX mode to ensure data confidentiality and integrity. This engine processes plaintext data along with the AES key to produce encrypted data (ciphertext), a nonce for ensuring unique encryption instances, and an authentication tag to verify data integrity upon decryption.

- **Data Handler**:

Responsible for the input and output management of data, including reading plaintext from the user or a file and handling encrypted output. It interfaces directly with the encryption/decryption engine to manage the flow of data efficiently, ensuring that the encryption and decryption processes are seamless.

- **Logging and Audit Trail Module**:

Supports compliance and auditing requirements by logging all encryption and decryption activities. Each log entry includes details such as the date and time of the operation, the base64-encoded key used, the original text, and the resultant ciphertext. These logs are maintained in an Excel file using the openpyxl library, facilitating structured and accessible data recording for audit purposes.

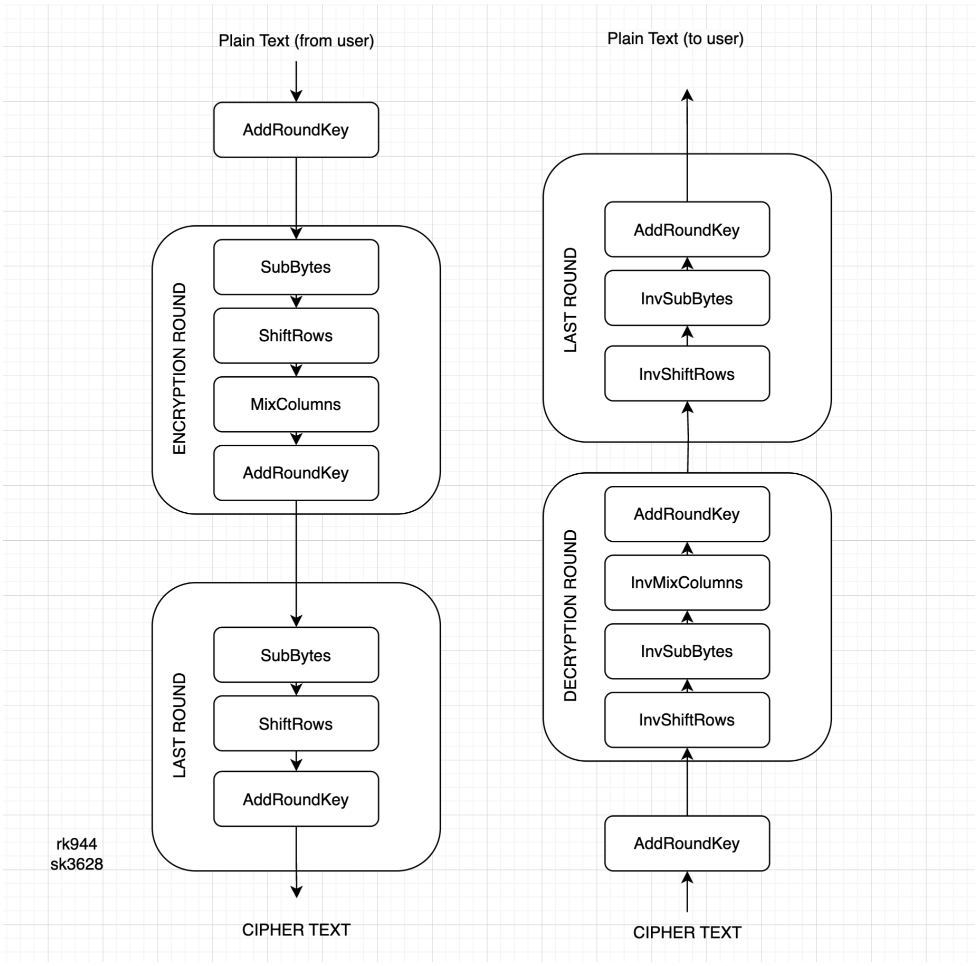


Figure 1: AES Algorithm for encryption and decryption

1.4. DELIVERABLES

These are the following features this project will deliver:

- **A fully functional encryption and decryption utility** that can be used on command line interfaces.
- **A comprehensive user guide** that explains how to use the utility, including how to generate keys, encrypt and decrypt data, and interpret the logs.
- **An Excel log file** that automatically records all encryption and decryption activities performed by the utility, providing a permanent record of data manipulation for security and compliance purposes.

1.5. HARDWARE AND SOFTWARE REQUIREMENTS:

1.5.1. Hardware Requirements:

- Latest Python installation supporting all libraries.

1.5.2. **Software Requirements**:

- **Python**: The utility is developed in Python, leveraging its extensive libraries.
- **PyCryptodome**: This library is used for cryptographic operations including AES encryption.
- **Openpyxl**: This library is employed to handle operations related to Excel file manipulations.
- **Base64 Module**: Used for encoding binary data into ASCII characters, making encrypted data easy to store and transmit.

2. ANALYSIS:

2.1. Project Code:

```
import os
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes
from openpyxl import Workbook, load_workbook
from Crypto.Hash import SHA256
import base64
import logging
from datetime import datetime

# Set up basic configuration for logging
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

def input_private_key():
    """Prompt user to input their own private key and convert it to a suitable AES key."""
    key_input = input("Enter your private key (any string, will be hashed to create a key): ")
    hasher = SHA256.new()
    hasher.update(key_input.encode('utf-8'))
    raw_key = hasher.digest()  # This will be a 32-byte key
    base64_key = base64.b64encode(raw_key).decode('utf-8')
    return raw_key, base64_key

def generate_private_key():
    """Generate a new private key and encode it in base64, ensuring it's 32 bytes (AES-256)."""
    private_key = get_random_bytes(32)  # 256-bit AES key
    encoded_key = base64.b64encode(private_key).decode('utf-8')
    return private_key, encoded_key

def save_data_to_excel(date_time, key, original_text, encrypted_text):
    """Save the data to an Excel file."""
    filename = 'keys.xlsx'
    try:
        wb = load_workbook(filename)
    except FileNotFoundError:
        wb = Workbook()
        ws = wb.active
        ws.append(['Date and Time', 'Key (Base64)', 'Original Text', 'Encrypted Text'])
    else:
        ws = wb.active

    ws.append([date_time, key, original_text, encrypted_text])
    wb.save(filename)
    logging.info("Data saved to keys.xlsx")

def encrypt_data(data, key):
    """Encrypt the data using AES encryption with a given key."""
    cipher = AES.new(key, AES.MODE_EAX)
    ciphertext, tag = cipher.encrypt_and_digest(data.encode('utf-8'))
    return base64.b64encode(ciphertext).decode(), base64.b64encode(cipher.nonce).decode(), base64.b64encode(tag).decode()

def decrypt_data(ciphertext, nonce, tag, key):
    """Decrypt the ciphertext using AES decryption with the given key."""
    try:
        ciphertext = base64.b64decode(ciphertext)
        nonce = base64.b64decode(nonce)
        tag = base64.b64decode(tag)
        cipher = AES.new(key, AES.MODE_EAX, nonce=nonce)
        plaintext = cipher.decrypt_and_verify(ciphertext, tag)
        return plaintext.decode('utf-8')
    except ValueError:
        logging.error("Decryption failed.")
        return "Decryption failed."
```

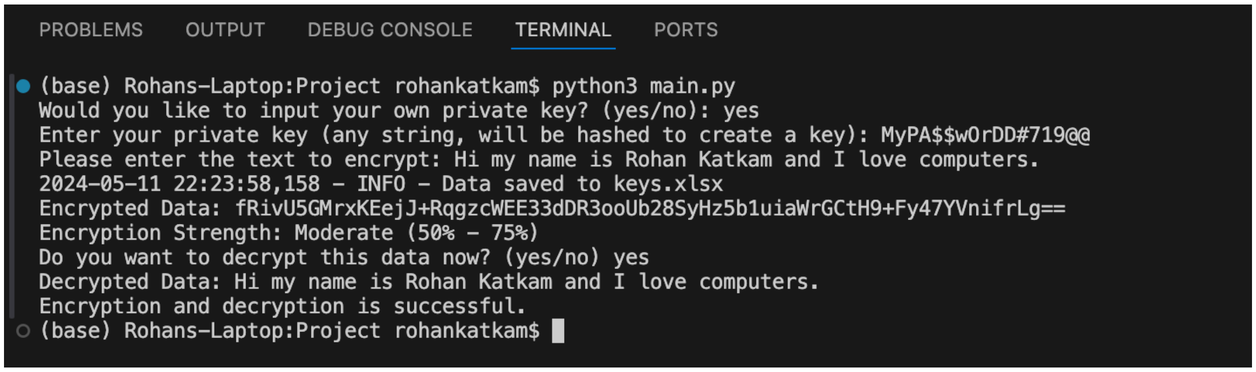
```
def assess_encryption_strength(ciphertext):
    """Assess the strength of the encryption based on ciphertext length."""
    length = len(ciphertext)
    if length < 64:
        return "Weak (less than 50%)"
    elif length < 128:
        return "Moderate (50% - 75%)"
    else:
        return "Strong (greater than 75%)"

def encrypt_or_decrypt_flow():
    """Main flow for encrypting or decrypting based on user input."""
    choice = input("Would you like to input your own private key? (yes/no): ")
    if choice.lower() == 'yes':
        key, encoded_key = input_private_key()
    else:
        key, encoded_key = generate_private_key()

    data = input("Please enter the text to encrypt: ")
    encrypted_data, nonce, tag = encrypt_data(data, key)
    now = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    save_data_to_excel(now, encoded_key, data, encrypted_data)

    print("Encrypted Data:", encrypted_data)
    print("Encryption Strength:", assess_encryption_strength(encrypted_data))

    if input("Do you want to decrypt this data now? (yes/no) ").lower() == 'yes':
        decrypted_data = decrypt_data(encrypted_data, nonce, tag, key)
        print("Decrypted Data:", decrypted_data)
        if decrypted_data == data:
            print("Encryption and decryption is successful.")

if __name__ == '__main__':
    encrypt_or_decrypt_flow()
```
(The code is pasted from VS Code Editor)

2.2. Code Explanation:

a) Users can either hash a passphrase into a 256-bit AES key using SHA-256 or generate a random key, both of which are base64 encoded for security.
b) Utilizes AES in EAX mode to encrypt text, producing a nonce, tag, and ciphertext, and supports decryption that verifies data integrity using the tag.
c) Stores encryption details such as timestamps and keys in an Excel file via openpyxl, aiding in compliance and audit trails.
d) Prompts user decisions for key management and text encryption, offering an optional decryption check to validate the process.

2.3. Screenshots

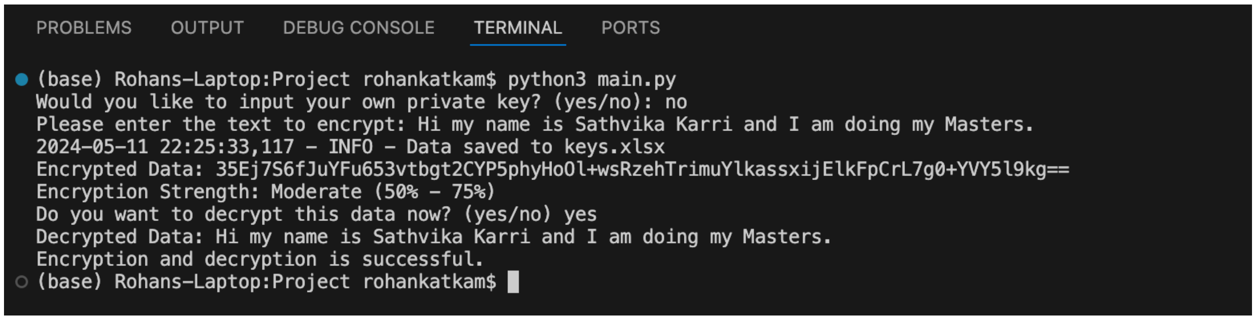
2.3.1 Output where user creates their own private key:



```
(base) Rohans-Laptop:Project rohankatkam$ python3 main.py
Would you like to input your own private key? (yes/no): yes
Enter your private key (any string, will be hashed to create a key): MyPA$$wOrDD#719@@
Please enter the text to encrypt: Hi my name is Rohan Katkam and I love computers.
2024-05-11 22:23:58,158 - INFO - Data saved to keys.xlsx
Encrypted Data: fRivU5GMrxKEejJ+RqgzcWEE33dDR3ooUb28SyHz5b1uiaWrGCtH9+Fy47YVnifrLg==
Encryption Strength: Moderate (50% - 75%)
Do you want to decrypt this data now? (yes/no) yes
Decrypted Data: Hi my name is Rohan Katkam and I love computers.
Encryption and decryption is successful.
(base) Rohans-Laptop:Project rohankatkam$
```

Figure 2: Output with new private key

2.3.2 Output where encryption and decryption happen using the default private key

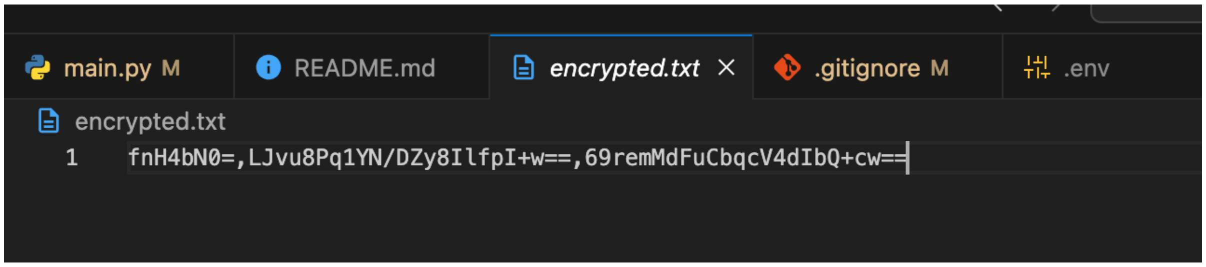


```
(base) Rohans-Laptop:Project rohankatkam$ python3 main.py
Would you like to input your own private key? (yes/no): no
Please enter the text to encrypt: Hi my name is Sathvika Karri and I am doing my Masters.
2024-05-11 22:25:33,117 - INFO - Data saved to keys.xlsx
Encrypted Data: 35Ej7S6fJuYFu653vtbgt2CYP5phyHoOl+wsRzehTrimuYlkassxijElkFpCrL7g0+YVY5l9kg==
Encryption Strength: Moderate (50% - 75%)
Do you want to decrypt this data now? (yes/no) yes
Decrypted Data: Hi my name is Sathvika Karri and I am doing my Masters.
Encryption and decryption is successful.
(base) Rohans-Laptop:Project rohankatkam$
```

Figure 3: Output with default private key

2.3.3. Encryption.txt file:

- This .txt file contains all the encrypted data everytime we run the program.

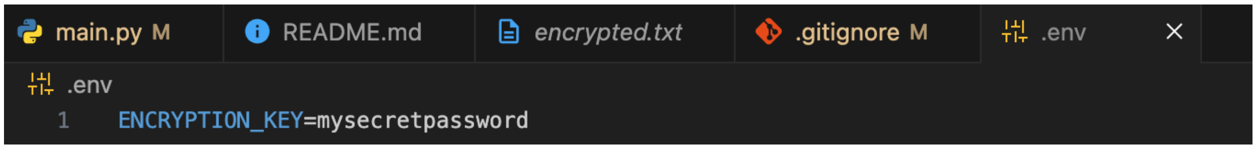


```
fnH4bN0=,LJvu8Pq1YN/DZy8IlfpI+w==,69remMdFuCbqcV4dIbQ+cw==
```

Figure 4: Encryption.txt file

2.3.4 Hidden environment variable:

- This is where we store the default private key when users do not want to create a new private key for encryption and decryption.

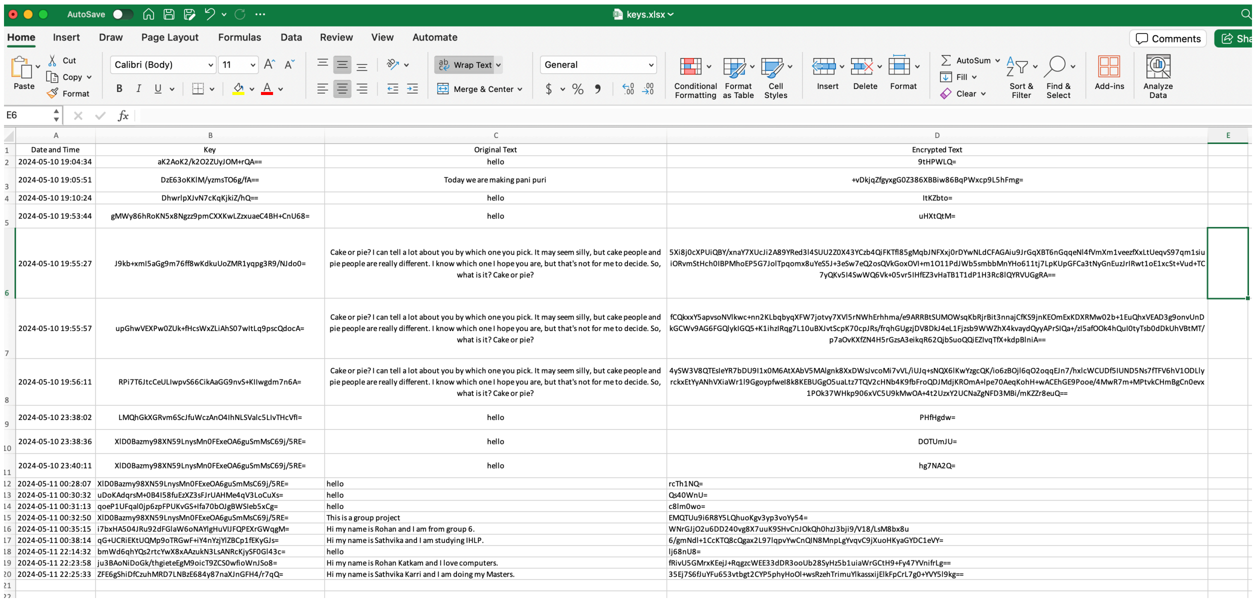


```
ENCRYPTION_KEY=mysecretpassword
```

Figure 5: .env file

2.3.5 Data logging file:

- This file stores all the information we get while running the program.
- The various attributes this file has:
  - **Date and Time**: Records the exact moment encryption was performed.
  - **Key**: Stores the encryption key used, encoded in base64 format for security.
  - **Original Text**: Logs the plaintext before encryption.
  - **Encrypted Text**: Captures the ciphertext after encryption.



Date and Time	Key	Original Text	Encrypted Text
2024-05-10 19:04:34	aK2AoK2/k2O2ZUyJOM+rQA==	hello	9tHPWLQ=
2024-05-10 19:05:51	DzE63oKKlM/yzmsTO6g/fA==	Today we are making pani puri	+vDkjqZfgyxgG0Z386XBBiw86BqPWxcp9L5hFmg=
2024-05-10 19:10:24	DhwrlpXJvN7cKqKjkiZ/hQ==	hello	ItKZbto=
2024-05-10 19:53:44	gMWy86hRoKN5x8Ngzz9pmCXXKwLZzxuaeC4BH+CnU68=	hello	uHXtQtM=
2024-05-10 19:55:27	J9kb+xmI5aGg9m76ff8wKdkuUoZMR1yqpg3R9/NJdo0=	Cake or pie? I can tell a lot about you by which one you pick. It may seem silly, but cake people and pie people are really different. I know which one I hope you are, but that's not for me to decide. So, what is it? Cake or pie?	5Xi8j0cXPUiQBY/xnaY7XUcJi2A89YRed3l4SUU2Z0X43YCzb4QjFKTfl85gMqbJNFXxj0rDYwNLdCFAGAiu9JrGqXBT6nGqqeNl4fVmXm1veezfXxLtUeqvS97qm1siuiORvmStHch0IBPMhoEP5G7JolTpqomx8uYeS5J+3eSw7eQ2osQVkGoxOVl+m1O11PdJWb5smbbMnYHo611tj7LpKUpGFCa3tNyGnEuzJrIRwt1oE1xcSt+Vud+TC7yQKv5I4SwWQ6Vk+05vr5IHfEZ3vHaTB1T1dP1H3Rc8lQYRVUGgRA==
2024-05-10 19:55:57	upGhwVEXPw0ZUk+fHcsWxZLiAhS07wItLq9pscQdocA=	Cake or pie? I can tell a lot about you by which one you pick. It may seem silly, but cake people and pie people are really different. I know which one I hope you are, but that's not for me to decide. So, what is it? Cake or pie?	fCQkxxY5apvsoNVlkwc+nn2KLbqbyqXFW7jotvy7XVl5rNWhErhhma/e9ARRBtSUMOWsqKbRjrBit3nnajCfKS9jnKEOmExKDXRMw02b+1EuQhxVEAD3g9onvUnDkGCWv9AG6FGQlykIGQ5+K1ihzIRqg7L10uBXJvtScpK70cpJRs/frqhGUgzjDV8DkJ4eL1Fjzsb9WWZhX4kvaydQyyAPrSIQa+/zI5afOOk4hQuI0tyTsb0dDkUhVBtMT/p7aOvKXfZN4H5rGzsA3eikqR62QjbSuoQQiEZIvqTfX+kdpBlniA==
2024-05-10 19:56:11	RPi7T6JtcCeULIwpvS66CikAaGG9nvS+KIIwgdm7n6A=	Cake or pie? I can tell a lot about you by which one you pick. It may seem silly, but cake people and pie people are really different. I know which one I hope you are, but that's not for me to decide. So, what is it? Cake or pie?	4ySW3V8QTEsIeYR7bDU9I1x0M6AtXAbV5MAlgnk8XxDWsJvcoMi7vVL/iUJq+sNQX6lKwYzgcQK/io6zBOjl6qO2oqqEJn7/hxlcWCUDf5IUND5Ns7fTFV6hV1ODLlyrckxEtYyANhVXiaWr1l9GgoypfweI8k8KEBUGgO5uaLtz7TQV2cHNb4K9fbFroQDJMdjKROmA+lpe70AeqKohH+wACEhGE9Pooe/4MwR7m+MPtvkCHmBgCn0evx1POk37WHkp906xVC5U9kMwOA+4t2UzxY2UCNaZgNFD3MBi/mKZZr8euQ==
2024-05-10 23:38:02	LMQhGkXGRvm6ScJfuWczAnO4IhNLSValc5LIvTHcVfI=	hello	PHfHgdw=
2024-05-10 23:38:36	XlD0Bazmy98XN59LnysMn0FExeOA6guSmMsC69j/5RE=	hello	DOTUmJU=
2024-05-10 23:40:11	XlD0Bazmy98XN59LnysMn0FExeOA6guSmMsC69j/5RE=	hello	hg7NA2Q=
2024-05-11 00:28:07	XlD0Bazmy98XN59LnysMn0FExeOA6guSmMsC69j/5RE=	hello	rcTh1NQ=
2024-05-11 00:30:32	uDoKAdqrsM+0B4I58fuEzXZ3sFJrUAHMe4qV3LoCuXs=	hello	Qs40WnU=
2024-05-11 00:31:13	qoeP1UFqal0jp6zpFPUKvGS+Ifa70bOJgBWSIeb5xCg=	hello	c8lm0wo=
2024-05-11 00:32:50	XlD0Bazmy98XN59LnysMn0FExeOA6guSmMsC69j/5RE=	This is a group project	EMQTUu9i6R8Y5LQhuoKgv3yp3voYy54=
2024-05-11 00:35:15	i7bxHA504JRu92dFGlaW6oNAYlgHuVIJFQPEXrGWqgM=	Hi my name is Rohan and I am from group 6.	WNrGJjO2u6DD240vg8X7uuK9SHvCnJOkQh0hzJ3bji9/V18/LsM8bx8u
2024-05-11 00:38:14	qG+UCRiEKtUQMp9oTRGwF+iY4nYzjYlZBCp1fEKyGJs=	Hi my name is Sathvika and I am studying IHLP.	6/gmNdl+1CcKTQ8cQgax2L97lqpvYwCnQIN8MnpLgYvqvC9jXuoHKyaGYDC1eVY=
2024-05-11 22:14:32	bmWd6qhYQs2rtcYwX8xAAzukN3LsANRcKjySF0Gl43c=	hello	lj68nU8=
2024-05-11 22:23:58	ju3BAoNiDoGk/thgieteEgM9oicT9ZCS0wfioWnJSo8=	Hi my name is Rohan Katkam and I love computers.	fRivU5GMrxKEejJ+RqgzcWEE33dDR3ooUb28SyHz5b1uiaWrGCtH9+Fy47YVnifrLg==
2024-05-11 22:25:33	ZFE6gShiDfCzuhMRD7LNBzE684y87naXJnGFH4/r7qQ=	Hi my name is Sathvika Karri and I am doing my Masters.	35Ej7S6fJuYFu653vtbgt2CYP5phyHoOl+wsRzehTrimuYlkassxijElkFpCrL7g0+YVY5l9kg==

Figure 6: keys.xlsx file

3. CONCLUSION:

- During this project on the Secure Data Encryption and Decryption Utility, we learned a lot about how to keep data safe, make software easy to use, and keep track of data actions. Using AES-256 in EAX mode showed us how important it is to have encryption that not only locks data away but also checks that it hasn't been tampered with.
- We also saw how crucial it is to keep clear records of data actions for staying compliant with rules and ensuring security, which can be used in many other areas, not just encryption and predicting the strength of the private key. These lessons are especially useful in industries like healthcare and finance where protecting data is critical. This project helped us understand more about securing data and prepared us to tackle future security challenges more effectively.