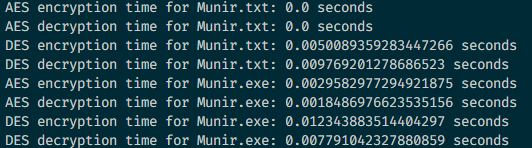
**INFORMATION SECURITY - ASSIGNMENT - 1**

**Munir Abbasi - K20-0244 - BSE-7B**





import time

import os

from Crypto.Cipher import DES, AES

TEXT\_FILE\_PATH = "Munir.txt"

EXE\_FILE\_PATH = "Munir.exe"

**def** des\_encrypt(key, data):

    cipher = DES.new(key, DES.MODE\_ECB)

    padded\_data = data + **b**"\0" \* (8 - len(data) % 8)

    return cipher.encrypt(padded\_data)

**def** des\_decrypt(key, data):

    cipher = DES.new(key, DES.MODE\_ECB)

    return cipher.decrypt(data)

**def** aes\_encrypt(key, data):

    cipher = AES.new(key, AES.MODE\_ECB)

    padded\_data = data + **b**"\0" \* (16 - len(data) % 16)

    return cipher.encrypt(padded\_data)

**def** aes\_decrypt(key, data):

    cipher = AES.new(key, AES.MODE\_ECB)

    return cipher.decrypt(data)

**def** read\_file(file\_path):

    with open(file\_path, "rb") as file:

        data = file.read()

    return data

**def** write\_file(file\_path, data):

    with open(file\_path, "wb") as file:

        file.write(data)

**def** encrypt\_decrypt\_file(file\_path, algorithm, key):

    data = read\_file(file\_path)

    if algorithm == "DES":

        start\_time = time.time()

        encrypted\_data = des\_encrypt(key, data)

        end\_time = time.time()

        print(

**f**"DES encryption time for {file\_path}: {end\_time - start\_time} seconds")

    elif algorithm == "AES":

        start\_time = time.time()

        encrypted\_data = aes\_encrypt(key, data)

        end\_time = time.time()

        print(

**f**"AES encryption time for {file\_path}: {end\_time - start\_time} seconds")

    else:

        raise ValueError("Invalid algorithm")

    write\_file(file\_path + "." + algorithm.lower() + ".enc", encrypted\_data)

    encrypted\_data = read\_file(file\_path + "." + algorithm.lower() + ".enc")

    if algorithm == "DES":

        start\_time = time.time()

        decrypted\_data = des\_decrypt(key, encrypted\_data)

        end\_time = time.time()

        print(

**f**"DES decryption time for {file\_path}: {end\_time - start\_time} seconds")

    elif algorithm == "AES":

        start\_time = time.time()

        decrypted\_data = aes\_decrypt(key, encrypted\_data)

        end\_time = time.time()

        print(

**f**"AES decryption time for {file\_path}: {end\_time - start\_time} seconds")

    else:

        raise ValueError("Invalid algorithm")

    write\_file(file\_path + "." + algorithm.lower() + ".dec", decrypted\_data)

**def** main():

    aes\_key = **b**"1234567890123456"

    des\_key = **b**"abcdefgh"

    for i in range(2):

        if os.path.getsize(TEXT\_FILE\_PATH) < 10000000:

            if i == 0:

                algorithm = "AES"

                key = aes\_key

            else:

                algorithm = "DES"

                key = des\_key

            encrypt\_decrypt\_file(TEXT\_FILE\_PATH, algorithm, key)

        else:

            print("File size is greater than 10MB")

    for i in range(2):

        if os.path.getsize(EXE\_FILE\_PATH) < 10000000:

            if i == 0:

                algorithm = "AES"

                key = aes\_key

            else:

                algorithm = "DES"

                key = des\_key

            encrypt\_decrypt\_file(EXE\_FILE\_PATH, algorithm, key)

        else:

            print("File size is greater than 10MB")

if \_\_name\_\_ == "\_\_main\_\_":

    main()