CSCI301 Contemporary Topics in Security

Report For Assignment 1

enct.py

A close-up of a text

Description automatically generated

These are the resources that I use in this python file. One of the requirements is using the pycryptodome package. Pycryptodome provides public key encryption like RSA and I use PKCS1\_OAEP as the version of RSA for encryption and decryption. I also have import ‘os’ module in enct.py. It offers a means of interacting with the operating system. It provides several functions to carry out tasks that are dependent on the operating system, like modifying paths, reading from or writing to the file system, and communicating with the underlying system.

A computer screen shot of a code

Description automatically generated

The generate\_key\_pair function is a part of the code responsible for generating an RSA key pair. In summary, this function allows to easily generate RSA key pairs and save them to files, providing a convenient way to manage cryptographic keys for encryption and decryption processes.

A computer screen shot of a computer code

Description automatically generated

The choose\_public\_key function is intended to ask the user to select from "firstpublic.pem" and "secpublic.pem," which are the two public keys. In conclusion, this function makes sure the user chooses a legitimate option (either "1" or "2") for the public key, and it continuously prompts the user until a legitimate selection is made.

A screenshot of a computer program

Description automatically generated

With asymmetric (RSA) and symmetric (AES) encryption, the encrypt\_message function is intended to encrypt a message. To put it briefly, the function uses symmetric encryption (AES) to encrypt the message and asymmetric encryption (RSA) to secure the symmetric key. Both forms of encryption benefit by this two-layered strategy.

A computer screen shot of a code

Description automatically generated

In the main code, it produces two pairs of RSA keys (public and private) for every "first" and "second" prefix, asks the user to select a public key, and then uses the selected public key to encrypt three text files, resulting in binary encrypted files that match.

Expected Outcomes

A computer screen shot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

dect.py

A screenshot of a computer

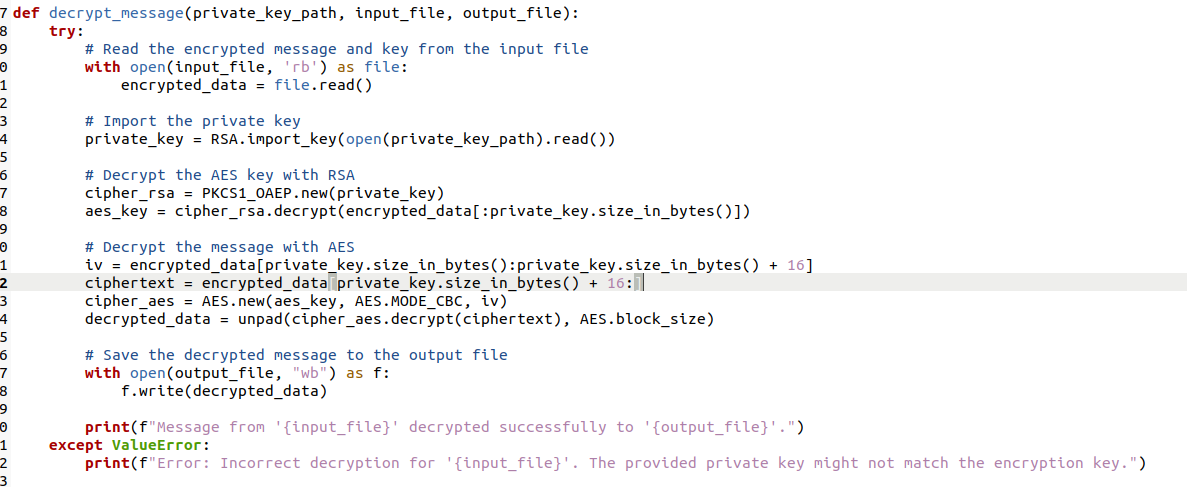
Description automatically generated

These are the resources that I use in this python file. All are similar to the resources in enct.py except the third row. In enct.py is from Crypto.Util.Padding import pad. The unpad function in the Crypto.Util.Padding module is used to remove padding from a block of data. Padding is often added to ensure that the length of the data is a multiple of the block size, which is a common requirement for block ciphers like AES.

A white background with red text

Description automatically generated

The choose\_private\_key function is intended to ask the user to select from "firstprivate.pem" and "secprivate.pem," which are the two private keys. In conclusion, this function makes sure the user chooses a legitimate option (either "1" or "2") for the private key, and it continuously prompts the user until a legitimate selection is made.



The decrypt\_message function is intended to decrypt a message that has been encrypted using a combination of RSA and AES encryption algorithms. All possible mistakes that may arise during the decryption process are handled by an exception block, apart from ValueError. An error message stating that the supplied private key might not match the encryption key is printed in the event of a ValueError (for example, because of improper decoding).

A screenshot of a computer code

Description automatically generated

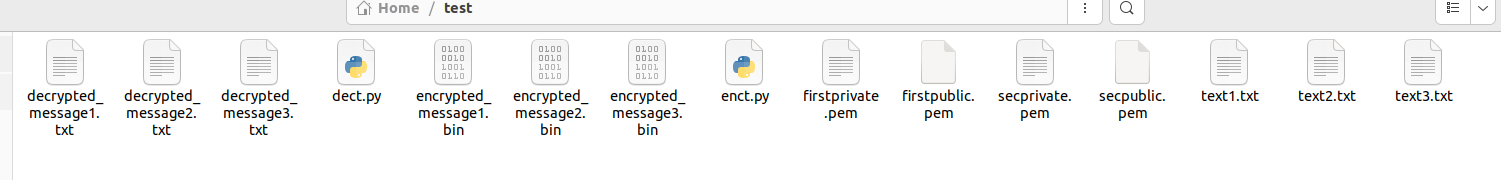
In the main code, it includes employing a selected private key to decrypt messages. Using the choose\_private\_key() function, the script asks the user to pick a private key (either "firstprivate.pem" or "secprivate.pem"). The path to the chosen private key file is then determined by the user's selection.

The script goes into a three-iteration loop (for i in the range of 1 to 4) after receiving the private key path. The script uses the selected private key to decrypt a corresponding encrypted message file ("encrypted\_message{i}.bin") at the start of each cycle. Next, an output file ("decrypted\_message{i}.txt") is written with the encrypted content. To summarise, this code segment saves the decrypted content to distinct output files and automatically decrypts three encrypted messages using a user-selected private key.

Expected output

A screenshot of a computer screen

Description automatically generated



A computer screen shot of white text

Description automatically generated