Assignment 2

Meraj Ahmed

2019MCS2565

1 Introduction

We have to implement the following modules:

1. Implementation of CA.
2. Encryption by sender
3. Decryption by receiver

We have implemented different module in different python file. There is one master main.py which first setup CA and public key directory then it sends message and receive message after that. Readme contains the instruction of how to run. Each module can be run independently. Run main.py to execute all modules in sequential.

2 How to run

Run the following in sequence

```
python3 CA.py
python3 sender.py plain vig_key
python3 reciever.py cipher
```

or run the following to execute all 3.

```
python3 main.py
```

3 Implementation of CA

1. CA is implemented in CA.py file.
2. The responsibility of CA is to first generate is own public/private key pair. Then generate user's public/private key pair and then setup a public key directory.

3. **Generation of key pair** We have implemented RSA key generation for this purpose. We have used the gmpy2 library to get the 512 bit primes. Then we check if it is strong prime or not. We generate primes until we get the strong prime. Upon getting strong primes P and Q. We run RSA algorithm to get (e, d, n). Here (e,n) = Public key, (d,n) = private key, n = P*Q.

4. **CA's Key** First CA generate its own public/private key pair using above method. We save these key in ca_private and ca_public file.

5. **Public Key directory** We then setup the public key directory of each users. We assume that there are 2 users. One is sender and other is receiver.
   For each users we generate its public/private key pair using above 3rd point. Private key is saved into in independent file for each users. Public key of all users is stored in the single file pub_dir. Here all public keys are first encrypted with CA's private key before saving in public key directory. Each entry of this file is: (id of user, public key e encrypted with CA's private key, n=pq encrypted with CA's private key). I have assigned id of user sequentially starting from 1.

4 RSA Encryption/ Decryption

1. We have implemented the RSA encryption in RSA_Enc.py file and RSA decryption in RSA_Dec.py file.

2. **RSA Encryption** RSA encryption takes takes message and key. It encrypts message with key. Key may be public key of receiver's or private key of sender's.
   Digital signature is generated when it is encrypted with sender's own private key. Before sending digital signature is encrypted with the receiver's Private key.
   For encryption, first whole message is spitted into block. Then each block is encrypted individually We are using 27 characters. So, B=27. We choose block size(r) according to

$$B^r < n$$

   RSA Decryption Similarly RSA decryption takes cipher and key. It generates decrypted message. First whole cipher is splatted into blocks and each block is decrypted.

5 Vigenere Encryption/ Decryption

1. Vigenere encryption/ decryption is implemented in the Vig.py file

2. **Encryption/ Decryption** For encryption, it takes message and key. Each character is encrypted using key's corresponding character.

$$E[i] = P[i] + K[i\ mod\ m]$$

. Similarly decryption is implemented as reverse of encryption.

$$P[i] = C[i] - K[i\ mod\ m]$$

6 Sender/ Receiver

1. Sender and Receiver is independently implemented in sender.py and receiver.py file respectively.

2. **Sender** Sender sends message to other user. We have assumed that u1 is sender and u2 is the receiver.

   - First sender reads the plain text file from file. Plain text file is named as plain.
   - It read vigenere key from the file vig_key.
   - It read its own private key from file 1_private.
   - It reads receiver's u2 public key from the public key directory file pub_dir. Since each entry in public key directory is encrypted with CA's Private key, so it decrypt the entry for u2 using CA's public key.
   - **Vigenere encryption** Vigenere encryption is applied to message using vigenere key.
   - Now encrypted message is appended with the key length and key in the front of the encrypted message.
   - **Digital Signature** Now the resulting text is RSA encrypted using sender's own private key.
   - Finally the resulting text is encrypted with the receiver's public key. Then the final text is saved into file named cipher. Now receiver Will read the file to get the original message.

3. **Receiver** Receiver u2 will now reads the message from sender u1.

   - First receiver will read the cipher file generated by sender.
   - It read its own private key from file 2_private.
   - It read u1's public key from public key directory. Since it is encrypted with CA's private key,so RSA decryption is applied using CA's public key.
   - First the message is RSA decrypted using receiver's own private key.

- Then it is decrypted using sender's public key.
- Now, receiver can read starting of message to get the key length and key. Key obtained is vignere key.
- **Vigenere decryption** Now it apply vigenere decryption to get the original message back.
- Decrypted vignere key is saved to file decrypted_vignere_key. Decrypted message is saved into decrypted_message file.