NSc Assignment-3 Report

About RSA:

1) **Overview:**

   RSA (Rivest-Shamir-Adleman) is an asymmetric cryptographic technique used for secure online data transfer. It uses two keys for each user: a public key for encrypting messages and a private key for decrypting them. RSA's security relies on the challenge of factoring large numbers.

   Working of RSA Algorithm involves

   Key Generation (in 2)
   Encryption & Decryption (in 3)

2) **Key Generation**

- Select p and q as two very large prime numbers(the larger the number, higher the security)
- Calculate n = p * q
- Calculate Φ= (p- 1) * (q- 1)
- Choose e such that 1 <= e < Φ and gcd(e, Φ) = 1
- d =e^{-1} mod Φ
- Public key pair = {e, n}, Private key pair = {d, n}

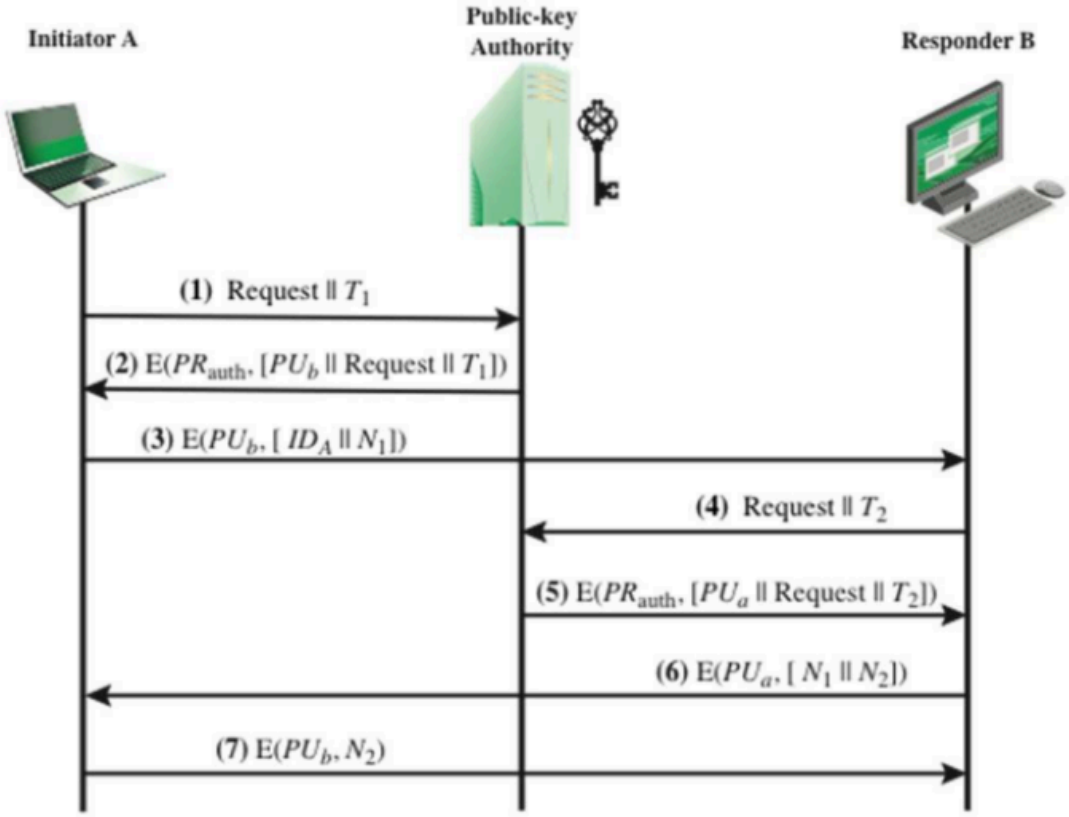
3) **Encryption and Decryption**

- Calculate and return ciphertext = pow(plaintext, e, n) or C = M^e mod n
- Calculate and return plaintext = pow(ciphertext, d, n) or M = C^d mod n

About Public Key Distribution Authority (PDKA)

A PKDA (Public Key Distribution Authority) system securely shares public keys among clients. Clients sign up to get the PKDA's public key and use it to request others' keys. The PKDA acts as a middleman for encrypted message exchange. Public keys are stored openly, and clients can make public and private keys. To send a message, a client asks PKDA for the recipient's public key. The message is encrypted before sending if the key is obtained. This allows secure communication without sharing keys

directly. PKDA ensures secrecy and integrity of communication by managing key distribution. The underlying encryption technique we are using for encryption is RSA.



Assumptions:

- Clients already (somehow) know the public key of the distribution authority, PKDA.
- Clients already know their own [private-key, public-key], but do not have the public-keys of other clients.
- PKDA has the public keys of all the clients.

System and working

There are two components in this system

- Two client programs (client1.py and client2.py)
- PKDA server (pkda.py)

Rsa is implemented in rsa.py and used in other files.

The system ensures secure communication between clients. Clients access the PKDA server's public key to register and request others' keys. A communication protocol based on RSA encrypts and decrypts messages using public and private keys. Additionally, a handshake is used to establish and verify secure connections between clients.

PKDA output :

```
PKDA server is listening for connections...
New connection from ('127.0.0.1', 53764)
Received data: b'{"client_id": "client_2", "type_of_req": "Register", "public_key": [667, 1147]}'
New connection from ('127.0.0.1', 53765)
Received data: b'{"client_id": "client_1", "type_of_req": "Register", "public_key": [533, 667]}'
New connection from ('127.0.0.1', 53766)
Received data: b'{"type_of_req": "Request_public_key", "client_id": "client_1", "other_client_id": "client_2", "cur_time": "13:02:29"}'
[667, 1147]
New connection from ('127.0.0.1', 53768)
Received data: b'{"type_of_req": "Request_public_key", "client_id": "client_2", "other_client_id": "client_1", "cur_time": "13:02:29"}'
[533, 667]
```

Client_1 output :

```
Sending request to register to PKDA
Received PKDA_public key and client has been register
[169, 221]
Sending request to get public key of client_2 to PKDA
Received public key from PKDA: {'pu_arg1': [184, 184, 55], 'pu_arg2': [49, 49, 52, 55], 'cur_time': [49, 51, 214, 139
0, 214, 50, 96]}

client_1 Got public key of (667, 1147) at Time: 13:02:29
Sending hankshape request
Received reply to handshake request from client2: {'type_of_req': [372, 652, 286, 282, 324, 583, 116, 517, 583, 133,
, 371, 361, 115, 133, 503, 513, 652, 583, 114, 652, 635, 262, 652, 115, 116], 'client_id': [592, 282, 395, 652, 371,
, 588, 311], 'Nonce_1': [255, 433, 28, 546, 301, 457, 433, 433, 315, 503, 301, 311, 255, 592, 361, 255, 301, 592, 546
61, 361, 592, 606, 542, 546, 457, 479, 457, 606, 503, 301, 28], 'Nonce_2': [315, 301, 503, 315, 28, 433, 606, 592, 60
361, 479, 361, 255, 457, 546, 479, 433, 479, 28, 301, 315, 542, 479, 457, 433, 255, 457, 592, 606, 193, 361, 28]}
Sending confirmation of handshake to client_2

Sending Hi message: Hi_1
Received Got_1 message from client_2
Sending Hi message: Hi_2
Received Got_2 message from client_2
Sending Hi message: Hi_3
Received Got_3 message from client_2
```

Client-2 output

```
Sending request to register to PKDA
Received PKDA_public key
Waiting for client1 to send handshake request
Received handshake connection from ('127.0.0.1', 53767)
Sending request to get public key of client_1 to PKDA
Received public key from PKDA: client 1
client_2 Got public key of (533, 667) at Time: 13:02:29
(533, 667)
Received handshake request from client_1
Sending reply to handshake request
Received Confirmation from client_1
Received handshake connection from ('127.0.0.1', 53769)
Received message: Hi_1
Sending response to Hi message
Received message: Hi_2
Sending response to Hi message
Received message: Hi_3
Sending response to Hi message
```