Project Report:
Secure Communication System with PKDA

Abhishek 2020014
Rishabh Jain 2020399

1. Introduction:

The project's goal was to create a secure communication system using Python that uses RSA encryption to encrypt client communications. To securely distribute public keys among clients, a Public Key Distribution Authority(PKDA) was established.

2. Implementation:

a. PKDA (pkda.py):

- **Setup:** The PKDA specifies communication ports and public and private keys when it is initialized.
- **Key Management:** A dictionary contains the client's public keys.
- **Encryption and Decryption Functions:** RSA encryption and decryption are used to provide secure communication.
- **Communication Functions:**
    - send_message() and receive_message(): To receive and send encrypted messages.
    - connect_to_port(): Establishes client relationships.

b. Client 1 (client1.py) and Client 2 (client2.py):

- **Setup:** The public and private keys are used to initialize each client.
- **Key Exchange:** Shares its public key with PKDA so that others can access it.
- **Request Public Key:** Sends requests for another client's public key to the PKDA.
- **Secure Message Exchange:**
    - Nonces are securely generated and exchanged.
    - Uses received public keys to encrypt and decrypt messages.

3. Flow of Execution:

a. PKDA:

1. Receives public keys from clients and stores them.
2. Handles requests from clients for public keys.
3. Facilitates secure exchange of public keys between clients.
4. Encrypts and forwards messages between clients.

```
PS D:\NSC\NSC_ass3_2020014_2020399> python3 pkda.py
Connected to client2
Receiving public key of client 2 -->  (233, 323)
Connected to client1
Receiving public key of client 1 -->  (161, 323)
Receiving Request || T1 -->  ('client2', '00:45:42')
Sending E(PR_auth[PU_b||Request||T1]) -->  ((233, 323), 'client2', '00:45:42')
Receiving Request || T1 -->  ('client1', '00:45:42')
Sending E(PR_auth[PU_a||Request||T2]) -->  ((161, 323), 'client1', '00:45:42')
PS D:\NSC\NSC_ass3_2020014_2020399>
```

b. Client 1:

1. Shares its public key with the PKDA.
2. Sends a request for Client 2's public key.
3. Receives Client 2's public key from the PKDA.
4. Initiates secure communication with Client 2.

```
PS D:\NSC\NSC_ass3_2020014_2020399> python3 client1.py
Enter Prime numbers p and q
17
19
client1_public_key:  (161, 323)
client1_private_key:  (161, 323)
Connected to PKDA
Sharing client1 public key with PKDA --> (161, 323)
Sending Request || T1 -->  ('client2', '00:45:42')
Receiving E(PR_auth[PU_b||Request||T1]) -->  ((233, 323), 'client2', '00:45:4
2')
Connected to client2
Sending E(PU_b,[ID_a||N1]) -->  ('client1', 1170384533)
Receiving E(PU_a[N1 || N2]) -->  (1170384533, 1361931031)
Sending E(PU_b[N2]) -->  1361931031
Enter client 1 message:
Hi 1
Client 2 says:
 Gotit 1
Enter client 1 message:
Hi 2
Client 2 says:
 Gotit 2
Enter client 1 message:
Hi 3
Client 2 says:
 Gotit 3
Enter client 1 message:
```

c. Client 2:

1. Shares its public key with the PKDA.
2. Waits for Client 1 to request its public key.
3. Receives request from Client 1.
4. Sends its public key to Client 1 via the PKDA.
5. Initiates secure communication with Client 1.

```
PS D:\NSC\NSC_ass3_2020014_2020399> python3 client2.py
Enter Prime numbers p and q
17
19
client1_public_key:  (233, 323)
client1_private_key:  (89, 323)
Connected to PKDA
Sharing client2 public key [E(PU_pkda(PUa))] with PKDA --> (233, 323)
Connected to client1
Receiving E(PU_b[ID_a || N1]) from client 1-->  ('client1', 1170384533)
Sending Request || T2 to PKDA -->  ('client1', '00:45:42')
Receiving E(PR_auth[PU_b||Request||T1]) from PKDA -->  ((161, 323), 'client1', '00:45:42')
Sending E(PU_b,[ID_a||N1]) to client 1 -->  (1170384533, 1361931031)
Receiving E(PU_b[N2]) from client 1 -->  1361931031
Client 1 says:
 Hi 1
Client 1 says:
 Hi 2
Client 1 says:
 Hi 3
```

4. Working of Functions:

a. Encryption and Decryption:

- Utilizes RSA algorithm to encrypt and decrypt messages.
- **encrypt(message, public_key):** Encrypts a message using the public key.
- **decrypt(encrypted_message, private_key):** Decrypts an encrypted message using the private key.

b. Communication Functions:

- **send_message(sock, public_key, message):** Sends an encrypted message over a socket.
- **receive_message(sock, private_key):** Receives and decrypts a message from a socket.

5. Conclusion:

By securely distributing public keys via the PKDA and using RSA encryption to encrypt messages, the implemented system ensures secure communication between clients. In the creation of secure communication systems, this project demonstrates how cryptography can be used effectively.

5. References:

Class slides

GeeksforGeeks

Lec-84: RSA Algorithm in Network Security with examples in Hindi rsa algorithm example i…

PUBLIC KEY DISTRIBUTION || PUBLIC KEY INFRASTRUCTURE || DIFFERENT WAYS …