RSA-based Public-key Certification Authority (CA)

Mohd Shakir(2020524)

Rahul Kumar (2020110)

Assignment3 - Project0 - Report

The provided code ‘main.py’ implements a basic system for managing RSA encryption, decryption, and certificate generation/authentication. It simulates a scenario where clients communicate securely with each other through a Certificate Authority (CA).

Our code consists of several functions organized into some logical groups:

RSA Encryption and Decryption has been done by below :

- **`rsa_encrypt`**: Encrypts a plaintext using RSA encryption.
- **`rsa_decrypt`**: Decrypts a ciphertext using RSA decryption.

Hashing done by below :

- **`sha256`**: Computes the SHA-256 hash of a given input text.

RSA Key Generation done by below :

- **`generate_rsa_keypair`**: Generates RSA public-private key pairs.

Certificate Authority (CA) Functions below:

- **`create_certificate_authority`**: Creates a new Certificate Authority with public and private keys.
- **`set_public_keys`**: Sets public keys for clients.
- **`get_certificate`**: Retrieves a certificate from the CA.
- **`make_certificate`**: Generates and stores a certificate for a client.

Client Functions below :

- **`create_client`**: Creates a new client with public and private keys.
- **`get_request`**: Gets a certificate request from a client.
- **`get_my_cert`**: Stores the client's own certificate.
- **`add_cert`**: Adds a certificate to a client's store.
- **`check_cert`**: Checks if a client has a specific certificate.
- **`get_cert`**: Retrieves a certificate from a client's store.

- **`verify_cert`**: Verifies the validity of a certificate.
- **`show_certificates`**: Displays a client's certificates.
- **`encrypt_my_messages`**: Encrypts messages for sending.
- **`decrypt_my_messages`**: Decrypts received messages.
- **`certificate_validity`**: Checks the validity of a client's certificate.

Main Execution Function :

- **`execute`**: Implements the main functionality of the system, including client interactions.It executed main

Time Complexity Analysis

Time complexity of the code varies based on the operations performed. Here's a brief overview:

- RSA Encryption and Decryption:
  - Time complexity for encryption/decryption depends on the key size (**`n`**) and the exponent (**`e` or `d`**). Generally, it's O(log n) where n is the modulus.
- Hashing:
  - The time complexity of SHA-256 hashing is O(n), where n is the size of the input.
- RSA Key Generation:
  - Generating RSA key pairs involves prime number generation and modular arithmetic operations, which can have varying time complexities depending on the algorithm used. Generally, it's considered to be moderately expensive.
- Certificate Operations:
  - Certificate generation and verification involve RSA encryption/decryption and hashing, which contribute to the overall time complexity.
- Client Interactions:
  - The time complexity of client interactions depends on the operations performed, such as requesting certificates, checking validity, and sending/receiving messages.

Below libraries we have imported :

- **`hashlib`**: for SHA-256 hashing.
- **`random`**: for generating random numbers.
- **`math.gcd`**: for calculating the greatest common divisor.

Considerations or Assumption

- The code implements basic RSA encryption for secure communication.
- Certificate generation and verification ensure the authenticity of clients and messages.
- SHA-256 hashing adds integrity to the certificates.
- Random number generation is used for key generation, enhancing security.

Conclusion

The provided code offers a foundation for building a secure communication system using RSA encryption and certificate authentication. It provides functionalities for managing clients, generating certificates, and ensuring secure message exchange. Further enhancements could include additional security measures, optimizations for performance, and a more robust user interface.

Screenshot of the working

```
1. Create new user
2. User Login
Enter: 2
ID: 1
1. Request CA for certificate
2. Check available certificates
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
6. Check messages
7. Logout
Enter: 3
Certificate issued
1. Request CA for certificate
2. Check available certificates
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
6. Check messages
7. Logout
Enter: 7
1. Create new user
2. User Login
Enter: 2
ID: 2
1. Request CA for certificate
2. Check available certificates
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
6. Check messages
7. Logout
Enter: 3
Certificate issued
```

```
Enter ID of user: 1
Certificate of user 1 received
Certificate of user 1 verified
1. Request CA for certificate
2. Check available certificates
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
6. Check messages
7. Logout
Enter: 5
Enter ID of user: 1
Enter message: sdfsd
1. Request CA for certificate
2. Check available certificates
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
6. Check messages
7. Logout
Enter: 7
1. Create new user
2. User Login
Enter: 2
ID: 1
1. Request CA for certificate
2. Check available certificates
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
6. Check messages
7. Logout
Enter: 1
Enter ID of user: 2
Certificate of user 2 received
Certificate of user 2 verified
1. Request CA for certificate
2. Check available certificates
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
```

```
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
6. Check messages
7. Logout
Enter: 7
1. Create new user
2. User Login
Enter: 2
ID: 2
1. Request CA for certificate
2. Check available certificates
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
6. Check messages
7. Logout
Enter: 5
Enter ID of user: 1
Enter message: sdfds
1. Request CA for certificate
2. Check available certificates
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
6. Check messages
7. Logout
Enter: 6
Message from User 1: Acknowledging messages from User 2
1. Request CA for certificate
2. Check available certificates
3. Make/Update my certificate
4. Check my certificate validity
5. Send message
6. Check messages
7. Logout
Enter:
```