Programming Assignment-4

Abhinav Ujjawal (2021120) & Nikhil Suri (2021268)

Assumption:

- Clients already (somehow) know their own [private-key, public-key] but do not have their certificates or that of others,
- The format of the certificate and the hashing function are publicly available.
- Clients know the public key of the certification authority. (Handled by gRPC)
- CA has the public keys of all the clients. (Handled by gRPC)

Project no. 0: RSA Public-key Certification Authority

CA issues and verifies certificates. It digitally signs the certificates, ensuring the validity of the public key. It is also involved in sharing these certificates with clients. CA encrypts the certificate using its private key. This ensures the authenticity of the certificate. A sample certificate for A is of the form:

$(ID_A, PU_A, T_A, DUR_A, INFO_{CA}) || ENC_{PR\text{-}CA}(Hash(ID_A, PU_A, T_A, DUR_A, INFO_{CA}))$

- ID_A is the user ID
- PU_A is the public key of A
- T_A is the time of issuance of the certificate
- DUR_A is the duration for which the certificate is valid
- $INFO_{CA}$ is information about certification authority
- ENC is the encryption algorithm used by CA
- PR_{CA} is the public key of the certification authority

Encryption and decryption

We will use RSA for encryption and decryption. We generate p and q prime numbers. Using them, we can get $n = p*q$. And $\phi = (p-1)*(q-1)$.
These further allow us to generate numbers d and e. Choose e such that:

$$1 < e < \phi \text{ and } gcd(e,\phi) = 1$$

Next, use these keys for encryption and decryption so that:

$$C = M^e \pmod n$$

$$M = C^d \pmod n$$

(Here, M is a plaintext such that $M<n$ and C is the ciphertext)

Encryption:

- RSA encryption involves generating public and private keys.
- In the code, the RequestCertificate method in the CertificateAuthority class signs certificate data using the CA's private key.
- This is then appended to the original message and sent to the client.
- Clients use the public key to encrypt data before sending it to the other client.

Decryption:

- Decryption in the code involves verifying signatures and decrypting messages on the client’s side,
- Upon receiving a certificate, clients verify its authenticity using the CA's public key.

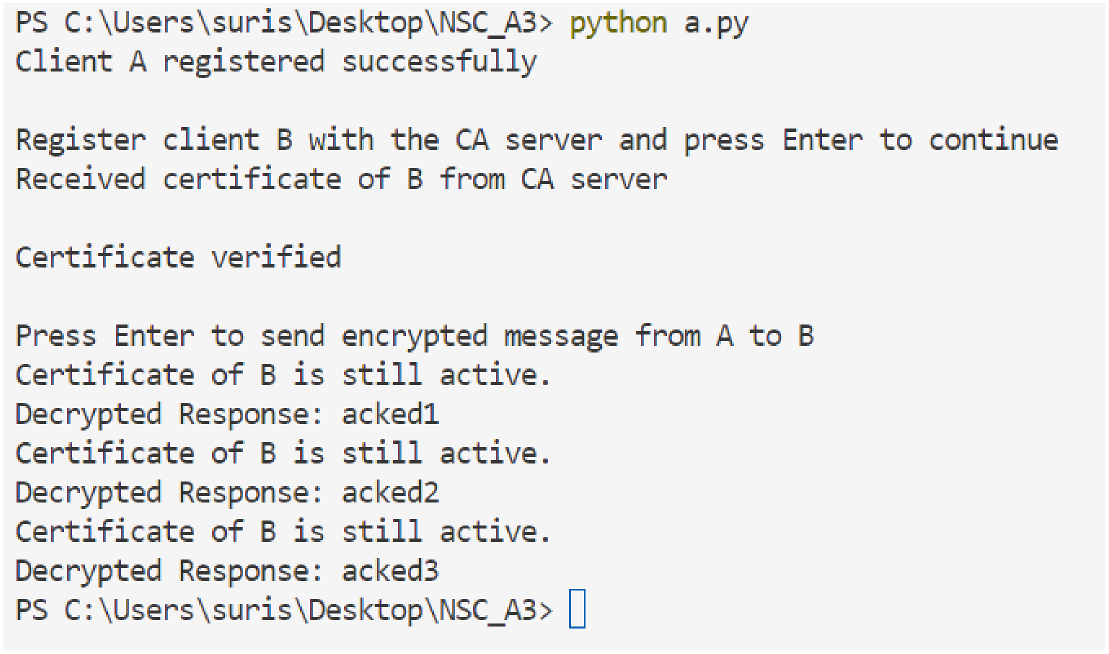
- If the signature is valid, decryption is successful, and the client can trust the received certificate for secure communication.
- Clients also send messages to each other while encrypting and decrypting them back and forth.

Constraints

- Specially chosen n, and blocks of data of size k bits
  s.t. $2^k < n \leq 2^{k+1}$

This constraint is **inherent** in the RSA algorithm. This is satisfied by default in our implementation.

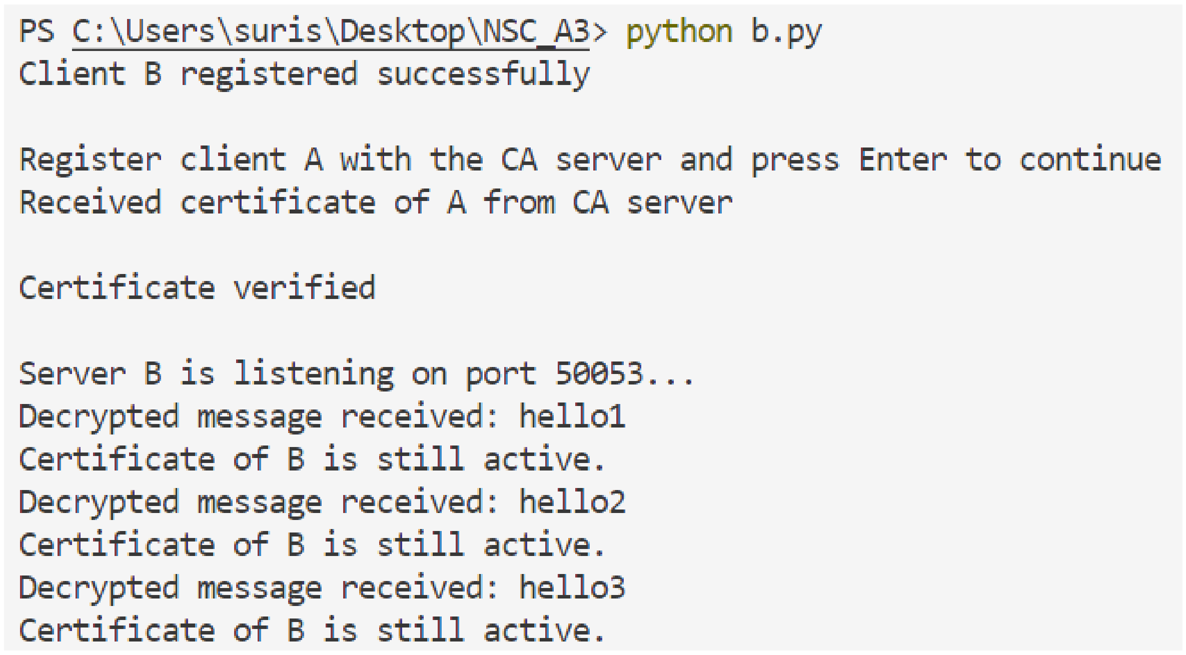
Sample Inputs & Outputs

1. 

```
PS C:\Users\suris\Desktop\NSC_A3> python a.py
Client A registered successfully

Register client B with the CA server and press Enter to continue
Received certificate of B from CA server

Certificate verified

Press Enter to send encrypted message from A to B
Certificate of B is still active.
Decrypted Response: acked1
Certificate of B is still active.
Decrypted Response: acked2
Certificate of B is still active.
Decrypted Response: acked3
PS C:\Users\suris\Desktop\NSC_A3>
```

Client A

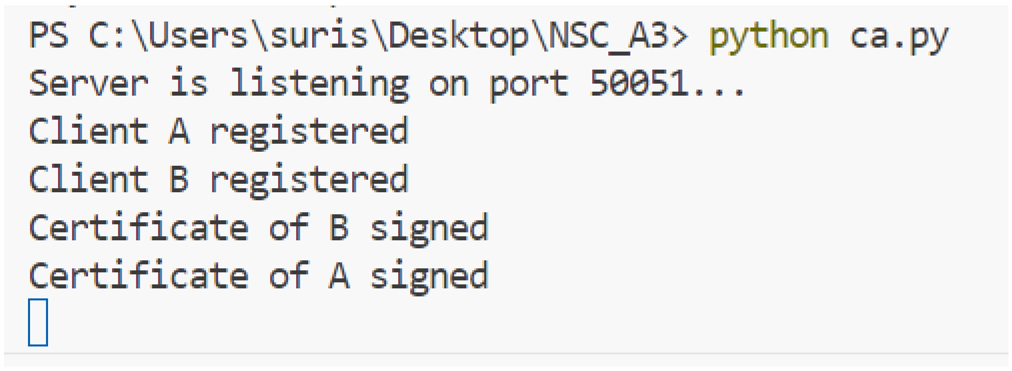
2.



```
PS C:\Users\suris\Desktop\NSC_A3> python b.py
Client B registered successfully

Register client A with the CA server and press Enter to continue
Received certificate of A from CA server

Certificate verified

Server B is listening on port 50053...
Decrypted message received: hello1
Certificate of B is still active.
Decrypted message received: hello2
Certificate of B is still active.
Decrypted message received: hello3
Certificate of B is still active.
```

Client B

3.



```
PS C:\Users\suris\Desktop\NSC_A3> python ca.py
Server is listening on port 50051...
Client A registered
Client B registered
Certificate of B signed
Certificate of A signed
```

RSA Public-key Certification Authority