# Certification Authority

Implementation in Python

2017MCS2089 - Nichit Bodhak Goel
2017MCS2076 - Shadab Zafar

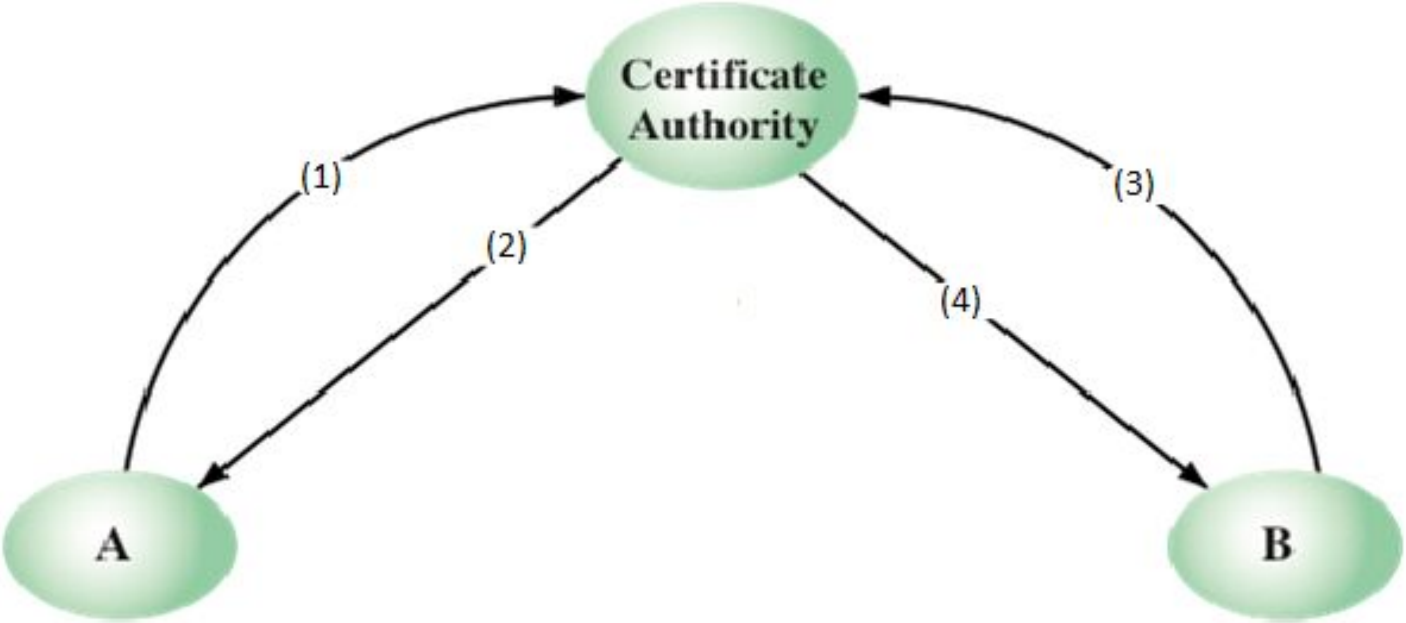
# Introduction

- Certification Authority is a trusted third party which issues the digital certificates to the users which helps them to prove the ownership of their public key to other users.

- These certificates are valid for a certain duration of time after which the users have to request them again.

- A sender will make its public key known via the certificate and the receiver will verify the certificate thereby confirming the validity of the public key.

# Certificate Creation



Obtaining Certificates from CA

# Procedure for Creation of Certificate

**1. Client Request**:- Client will send its identity ($ID_A$) to CA as CA is having the public key for every client.
Client Request :- $ID_A$.

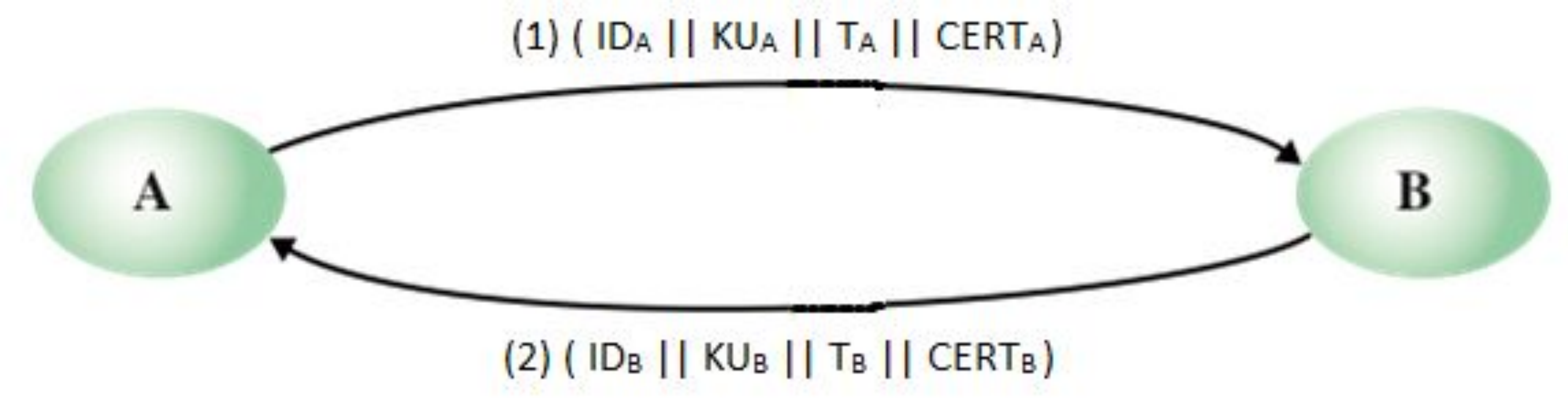
**2. CA Reply**:- CA will send the timestamp ($T_A$), public key ($KU_A$), identity ($ID_A$) and the certificate which will contain the hash of ID of client, public key of client and the timestamp specifying the time of issuance encrypted with the private key of CA i.e.
$CERT_A = ENC_{PRX}( Hash(ID_A, KU_A, T_A) )$.
CA reply :- $ID_A || KU_A || T_A || CERT_A$.

# Communication Between Clients

1. Exchange of certificates

2. Exchange of messages

# Exchange of Certificates


(1) ( $ID_A$ || $KU_A$ || $T_A$ || $CERT_A$ )
A
B
(2) ( $ID_B$ || $KU_B$ || $T_B$ || $CERT_B$ )

**I. Exchanging Certificates**

# Procedure of Exchange of Certificates

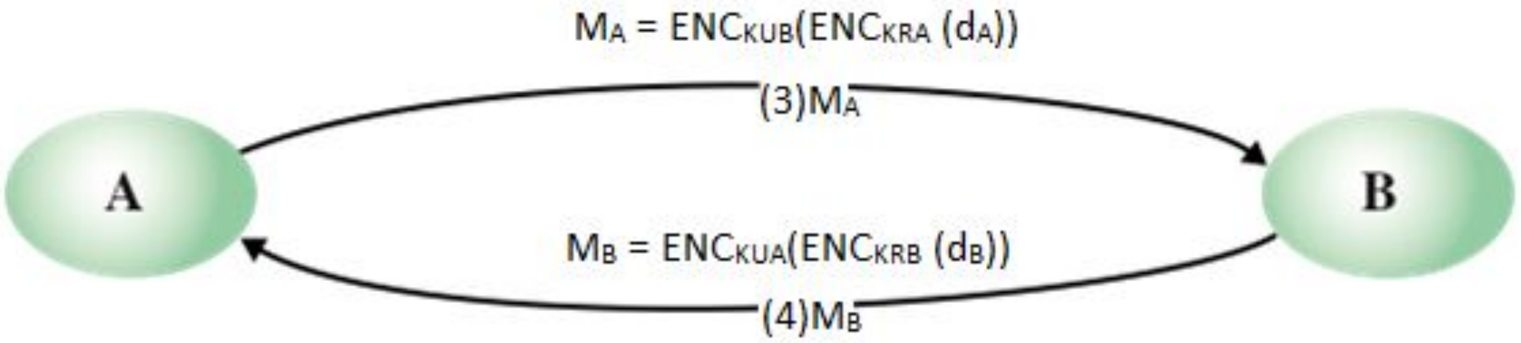
**Sending the Certificate** :- Client A will send its certificate along with its ID and Public key and timestamp to Client B.

$$\text{Client A to B :- } ID_A \,||\, KU_A \,||\, T_A \,||\, CERT_A$$

**Verifying the certificate** :- Now, Client B will verify the certificate of A by taking hash of ($ID_A$ || $KU_A$ || $T_A$ ) and decrypting the $CERT_A$ with the public key of CA ($PU_X$) and checking if they are equal.

$$Hash(ID_A, KU_A, T_A) = DEC_{PUX}(CERT_A)$$

# Exchange of Messages



$M_A = ENC_{KUB}(ENC_{KRA}\ (d_A))$

$M_B = ENC_{KUA}(ENC_{KRB}\ (d_B))$

**II. Exchanging Messages**

# Procedure of Exchange of Messages

**Sending the Message** :- Client A will encrypt the data ($d_A$) first with its own private key ($KR_A$) and after the it will encrypt it with the public key ($KU_B$) of client B. Then, Client A will send this message to Client B.

$$M_A = ENC_{KUB}(ENC_{KRA}\ (d_A)).$$

**Receiving the Message** :- Client B will first decrypt the $M_A$ first with its own private key ($KR_B$) and after that it will decrypt with the public key ($KU_A$) of client A. Then, Client B will be able to read the data.

$$d_A = DEC_{KUA}(DEC_{KRB}\ (M_A)).$$

# Output at Certification Authority

```
> python3 server.py

> Certification authority now listening on port: 7070

> Received request for new certificate: Shadab|835209960655|1000076001443
> Time of issuing:  2018-04-27 16:45:43.658237
> Certificate:
wp0Pw4PDnFpxwrrCtXTDu8OfS1XCoCfCocOfS8Kbw7R2w5pSw4BBGnMOOlzCisOrw63CpMKde8Kk
wrRkbUfDh2TCr8Kiw5ZvNsKnwodWesKAw5hgw5PDkMOgwpBAB0zDisKxwplcVmkHwo1Lw6XDnFfC
iEfDpMOaw6nCog==
```

# Output at Certification Authority

```
>>> New client connected: ('127.0.0.1', 58672)

> Received request for new certificate:  Nichit|927326331365|1000076001443
> Time of issuing:  2018-04-27 16:45:44.739411
> Certificate:
wozCrcKrw7zDpQDDvEjDtMKbNsOkwrByDVtYw5HCoQlew6rDo8KrccOhCUjCgMKKQQw8IcO8CsO
wJw3DlTRdw6YfJ8OjW1DDjMOvwpvCuW7DrCnCicO1wqrClmB2Jn7Cs1LCpn3DtcK/w6pxwoPDk8
KHQAVMwrXDgMO2

>>> Client disconnected:  ('127.0.0.1', 58672)
```

# Output at Client 1

```
> python3 client_1.py
My ID:  Shadab
My Public Key:  (835209960655, 1000076001443)

Sending request for a new certificate to CA

Received certificate from CA.

Shadab|835209960655|1000076001443|2018-04-27
16:45:43.658237|wp0Pw4PDnFpxwrrCtXTDu8OfS1XCoCfCocOfS8Kbw7R2w5pSw4B
BGnMOOlzCisOrw63CpMKde8KkwrRkbUfDh2TCr8Kiw5ZvNsKnwodWesKAw5hgw5PDkM
OgwpBAB0zDisKxwplcVmkHwo1Lw6XDnFfCiEfDpMOaw6nCog==
```

# Output at Client 1

```
>> User ID:  Nichit

>> Public Key:  (927326331365, 1000076001443)
>> Issue Time:  2018-04-27 16:45:44.739411
>> Certificate:
wozCrcKrw7zDpQDDvEjDtMKbNsOkwrByDVtYw5HCoQlew6rDo8KrccOhCUjCgMKKQQw
8IcO8CsOwJw3DlTRdw6YfJ8OjW1DDjMOvwpvCuW7DrCnCicO1wqrClmB2Jn7Cs1LCpn
3DtcK/w6pxwoPDk8KHQAVMwrXDgMO2

> Certificate is valid for given public key.

Received msg from client: Hello, Shadab
```

# Output at Client 2

```
> python3 client_2.py
My ID:  Nichit
My Public Key:  (927326331365, 1000076001443)

Sending request for a new certificate to CA
Received certificate from CA.

Nichit|927326331365|1000076001443|2018-04-27
16:45:44.739411|wozCrcKrw7zDpQDDvEjDtMKbNsOkwrByDVtYw5HCoQlew6rDo8K
rccOhCUjCgMKKQQw8IcO8CsOwJw3DlTRdw6YfJ8OjW1DDjMOvwpvCuW7DrCnCicO1wq
rClmB2Jn7Cs1LCpn3DtcK/w6pxwoPDk8KHQAVMwrXDgMO2

> Sending my public key & certificate
```

# Output at Client 2

```
>> User ID:  Shadab
>> Public Key:  (835209960655, 1000076001443)
>> Issue Time:  2018-04-27 16:45:43.658237
>> Certificate:
wp0Pw4PDnFpxwrrCtXTDu8OfS1XCoCfCocOfS8Kbw7R2w5pSw4BBGnMOOlzCisOrw63
CpMKde8KkwrRkbUfDh2TCr8Kiw5ZvNsKnwodWesKAw5hgw5PDkMOgwpBAB0zDisKxwp
lcVmkHwo1Lw6XDnFfCiEfDpMOaw6nCog==

> Certificate is valid for given public key.

Received msg from client: Hello, Nichit
```

# THANK YOU