CSE350: Assignment-3

Data Encryption Standard

Language:- Python
Operating System:- MacOS

For this assignment, We are presenting an implementation of an RSA-Based Certificate Authority in Python. We have simulated a scenario where two clients request their own certificates from the CA, retrieve the certificates of the other client, and securely exchange messages encrypted using the recipient's public key.

Implementation

We have used the gmpy2 library for large mathematical computations, random for generating the prime numbers for RSA, hashlib for hashing the certificate, time to record the system time, and math for its ceil function in the decryption stage. Apart from the RSA implementation, and the simulation, the code consists of 2 classes.

Certificate Authority

Maintains a record of issued certificates.
Generates and signs certificates.
Provides clients with verified certificates.

Client

Requests certificates from the certificate authority.
Stores and validates certificates received from the certificate authority
Encrypts messages using the recipient's public key and decrypts received messages using the recipient's private key.

Procedure

- Clients generate public and private keys.
- Clients request the certificate authority to create their own certificates using its pre-shared public key (which could have been accomplished by broadcasting it to all devices in the network)
- The Certificate Authority generates and signs the certificate.
- Each client retrieves the certificate of the other client.

- Clients send each other a medley of messages.

Results

```
cert_auth=certificate_authority(duration=1,certificate_authority_id=10001)
client_alice=client("Alice",cert_auth)
client_bob=client("Bob",cert_auth)
print("\nClients request their own certificates\n")
client_alice.get_your_certificate()
client_bob.get_your_certificate()
print("\nClients request the other guy's certificate\n")
cert_alice=client_bob.get_other_certificate("Alice")
cert_bob=client_alice.get_other_certificate("Bob")
pubk_alice=(int(cert_alice[0].split(",")[1]),int(cert_alice[0].split(",")[2]))
pubk_bob=(int(cert_bob[0].split(",")[1]),int(cert_bob[0].split(",")[2]))

print(f"\nClient Bob obtained Alice's public key:",pubk_alice,"\n")
print(f"\nClient Alice obtained Bob's public key:",pubk_bob,"\n")

print("\nClients exchange messages")
messages_from_a = ["Hello1", "Hello2", "Hello3"]
messages_from_b = ["ACK1", "ACK2", "ACK3"]

client_alice.send_msg(client_alice,"Hello1",pubk_bob,client_bob)
client_bob.receive_msg()
client_alice.send_msg(client_alice,"Hello2",pubk_bob,client_bob)
client_bob.receive_msg()
client_alice.send_msg(client_alice,"Hello3",pubk_bob,client_bob)
client_bob.receive_msg()

client_bob.send_msg(client_bob,"ACK1",pubk_alice,client_alice)
client_alice.receive_msg()
client_bob.send_msg(client_bob,"ACK2",pubk_alice,client_alice)
client_alice.receive_msg()
client_bob.send_msg(client_bob,"ACK3",pubk_alice,client_alice)
client_alice.receive_msg()

print("\nProgram Executed Successfully\n")
```

```
Certificate ID matches actual ID
Valid Other-Cert received from CA.
Alice,1203959403884637559317639692777312363682443250547452596825978664265707065495441207806457154954396572547912286036004446294804406161294010518660613570248967444681655042709677612033568211625378146757
No Tampering Detected
Certificate has not expired
Certificate ID matches actual ID
Valid Other-Cert received from CA.
Bob,937584730747736590341317176985952497717686006460027639563312445322878187498444646543917822227988954315208964072545278412686390103911846663866102754973163665436405

Client Bob obtained Alice's public key: (1203959403884637559317639692777312363682443250547452596825978664265707065495441207806457154954396572547912286036004446294804061612940105186606135702489674446816550427096776120335682116253781467578

Client Alice obtained Bob's public key: (937584730747736590341317176985952497717686006460027639563312445322878187498444646543917822227988954315208964072545278412686390103911846663866102754973163665436405

Clients exchange messages
The follwing message is being sent from Alice to Bob
Encrypted Message: 3493470948946572353038761159422668220528775748946326024629406413122816745520488446251641388500385199957794708115411649305236703536756662106121549442720460685188996225506648506599517412
The following message was received by Bob from Alice
Decrypted Message: Hello1
The follwing message is being sent from Alice to Bob
Encrypted Message: 3511450276395534228253812703977370540161941024002230010062487757453446694197758002101821034886476920093901565010681273897413717278676431428011217662462132380395573068332653461637769179
The following message was received by Bob from Alice
Decrypted Message: Hello2
The follwing message is being sent from Alice to Bob
Encrypted Message: 2575941380038737497823307736146570902886620286524485112040489700384085788811888614485501990283422191479843631046888630157244059864947260515118364446554946388394671842175262317350067089
The following message was received by Bob from Alice
Decrypted Message: Hello3
The follwing message is being sent from Bob to Alice
Encrypted Message: 9457070858808451155797996551253673420495656984158730539539567966528496610333668829384576586554456287264917258471074578349875579019584766370138501184621183290278468371165660069355951172
The following message was received by Alice from Bob
Decrypted Message: ACK1
The follwing message is being sent from Bob to Alice
Encrypted Message: 1127571284063642909772456780252003886506909828484467542263067628996396832815618849234240897588795668895051790351407906062096871003686820607543712013650832019531813102379052894943586689
The following message was received by ALice from Bob
Decrypted Message: ACK2
The follwing message is being sent from Bob to Alice
Encrypted Message: 4608950445509384347371105151668298934675981653781190785173937040793749780593868407207670719025215808353527997160914065283287160633120304397691825443832789830443509315790217316737184965
The following message was received by Alice from Bob
Decrypted Message: ACK3

Program Executed Successfully
```