Assignment-4

Keshav Gambhir 2019249
Tanmay Rajore 2019118

Introduction

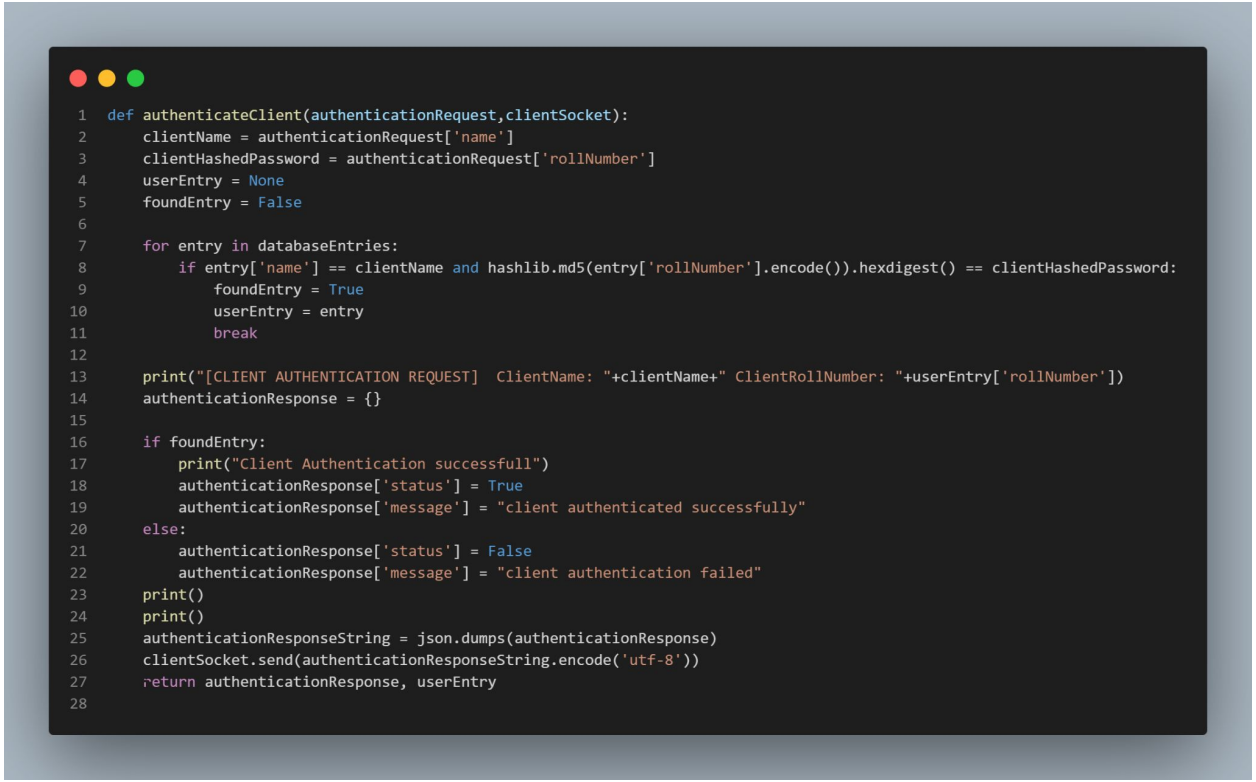
- Created a client server application which served to clients in secured manner.
- The certificate and marksheets are encrypted with passwords/public Key of client
- Full Message integrity is checked before the message is being downloaded

Authentication to the server

- Done using name and roll number of the student
- The design choice suffice because the it is end to end encrypted and not end to middle encrypted communication.
- The roll-number is being shared in hashed format so actual roll number is never shared.

```python
def authenticateWithServer(clientSocket):
    authCredientials = {}
    print("Enter your name")
    authCredientials['name'] = input()
    print("Enter your roll number")
    authCredientials['rollNumber'] =hashlib.md5(input().encode()).hexdigest()

    print("Sending authentication request to server")
    authCredientialsString = json.dumps(authCredientials)
    clientSocket.send(authCredientialsString.encode('utf-8'))
    response = clientSocket.recv(6144).decode('utf-8')
    responseJson = json.loads(response)
    print(responseJson['message'])
    print()
    return responseJson
```



```python
def authenticateClient(authenticationRequest,clientSocket):
    clientName = authenticationRequest['name']
    clientHashedPassword = authenticationRequest['rollNumber']
    userEntry = None
    foundEntry = False

    for entry in databaseEntries:
        if entry['name'] == clientName and hashlib.md5(entry['rollNumber'].encode()).hexdigest() == clientHashedPassword:
            foundEntry = True
            userEntry = entry
            break

    print("[CLIENT AUTHENTICATION REQUEST]  ClientName: "+clientName+" ClientRollNumber: "+userEntry['rollNumber'])
    authenticationResponse = {}

    if foundEntry:
        print("Client Authentication successfull")
        authenticationResponse['status'] = True
        authenticationResponse['message'] = "client authenticated successfully"
    else:
        authenticationResponse['status'] = False
        authenticationResponse['message'] = "client authentication failed"
    print()
    print()
    authenticationResponseString = json.dumps(authenticationResponse)
    clientSocket.send(authenticationResponseString.encode('utf-8'))
    return authenticationResponse, userEntry

```

Encrypting Data

- The received data is encrypted with a password which is currently the DOB of the user
- This password can be replaced by the public key of the user and hence can only be decrypted by the user’s private key
- Design similar to Aadhar card

Message Integrity

- Message Integrity is check using the signed HMAC of Director and Registrar.
- If both of them are valid then only message is considered valid
- Design choice is similar to certificate chaining where root certificate is trusted by everyone and then chaining is supported

Securing Timestamps

- Secure timestamps is obtained using NTP server and system time is not uses

Advantages:

Avoid tampering with the system time

Sharing with other Clients

- Again message integrity is checked similar to certificate chaining.