Report: Project 4 Part 2 – Facebook Application API in Scala with Security

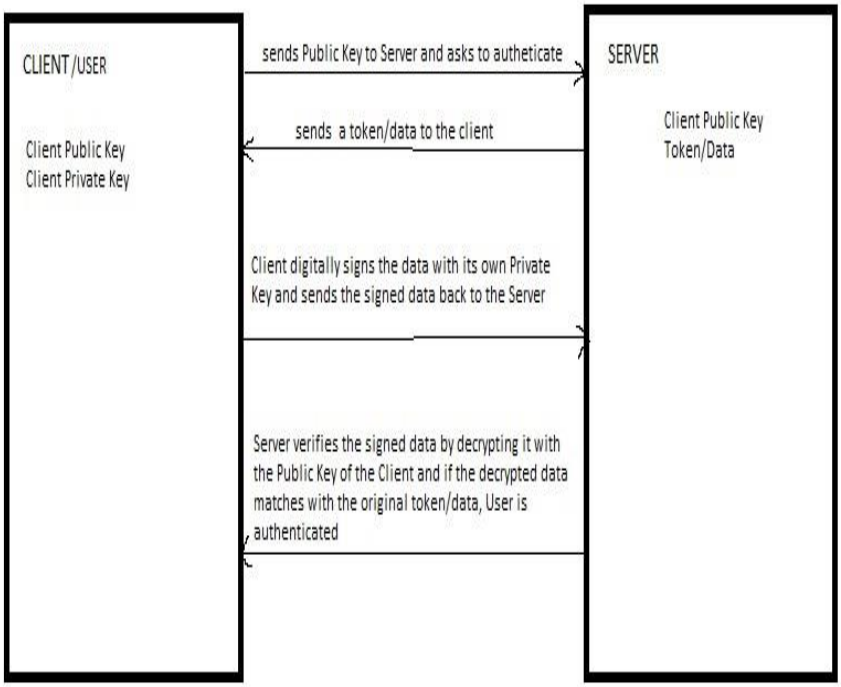
Students: Chiranjib Sur (UFID : 2531 4396)

Pramit Dutta (UFID : 7513 2433)

Course: COP 5615: Distributed Operating Systems Principles

User Authentication:

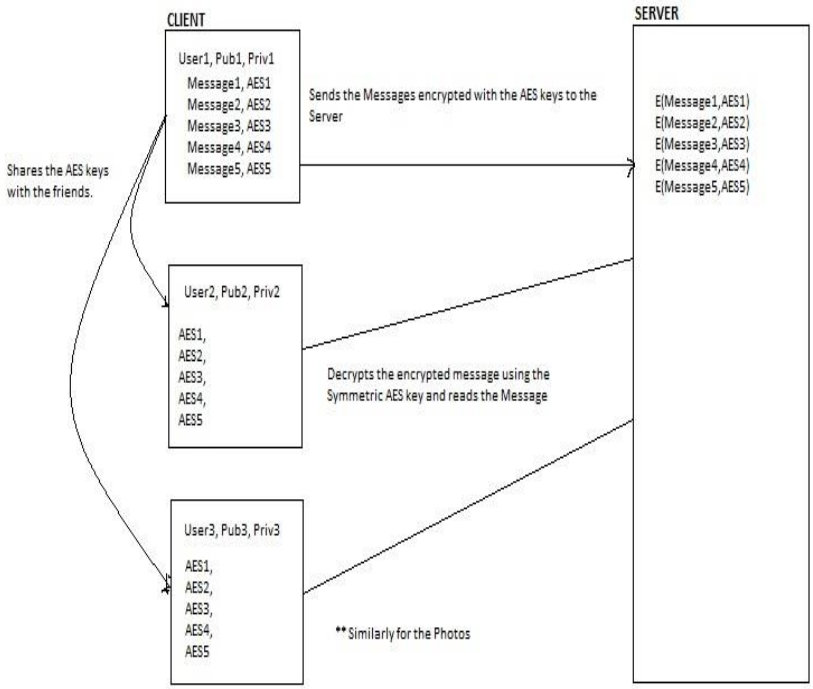
Every User has a RSA key pair (Private and Public). When a client wants to be authorized, the client sends its public Key to the Server. The Server sends a data to the Client. The client digitally signs the data with its own Private Key and sends this signature back to the Server. The Server then decrypts the Signature with the Client's public key and then verifies it with the original data sent. It they both matches a Client is authenticated else not.



Client Authentication using Digital Signature

Secure Posts/Message/Photo by User:

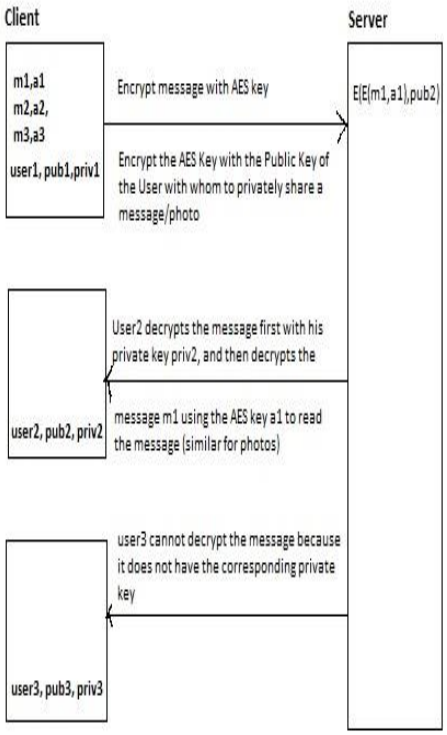
User messages or posts are encrypted by a Symmetric AES key and stored on the Server. Only those in possession of the AES key can see that particular message/photo by decrypting it. We are maintaining a hash map of the message/photo and its encrypting AES key. Whenever a client User wants to see his or her own message or photo, a User searches the hash map for the AES key of the message and then decrypts it. The hash map is also shared among the friends of the User, so that his friends can also read his messages or see the photos.



Posts/Messages/Photos - Encryption and Decryption using Symmetric Keys

One–to-One message or photo sharing (**Private**):

The message or photo is encrypted with an AES key at first and then this encrypted message is encrypted with the public key of the particular friend(s) with whom the post/message has to be shared. This ensures that the AES key can be decrypted only by the user who has the private key of the RSA key-pair with whose public key the AES key was encrypted. Once he decrypts the AES key he is then able to decrypt the post/photo and see it.



Secure Message Passing between two Users

Hashing of Public Keys to prevent attack:

The Client stores a hash value of the public key after hashing it with a hash function SHA-256. Now suppose some attacker changes the public key on the server. The client every time hashes the public key on the server with SHA-256 and matches it with the hash value it maintains. If they do not match the client declares a security breach in terms of public key swapping.

Paths/Directives used on the project are:

Profile Module

Get

http://localhost:8080/myFacebook/hello - for testing the **Profile** domain (myFacebook)

http://localhost:8080/myFacebook/all - for displaying all Profiles

http://localhost:8080/myFacebook/profileDetails - for getting the profile details of a particular user

Post

http://localhost:8080/myFacebook/sessionStart – creating session for the current user
[parameter("idg".as[String])]

http://localhost:8080/myFacebook/addUser – adding a user profile to the server
[parameters("id".as[String], "fname".as[String], "lname".as[String], "birth".as[String], "locationFrom".as[String], "locationIn".as[String])]

http://localhost:8080/myFacebook/updateUser – updating the user profile in the server
[parameters("id".as[String], "fname".as[String], "lname".as[String], "birth".as[String], "locationFrom".as[String], "locationIn".as[String])]

http://localhost:8080/myFacebook/deleteUser – deleting the user profile from the server
[parameters("id".as[String], "fname".as[String], "lname".as[String], "birth".as[String], "locationFrom".as[String], "locationIn".as[String])]

Post/Message Module

Get

http://localhost:8080/myFacebookM/helloM - for testing the **Post/Message** domain (myFacebookM)

http://localhost:8080/myFacebookM/allM - for displaying all Posts/Messages of all Users

http://localhost:8080/myFacebookM/myM - for displaying posts/messages of a particular User.

http://localhost:8080/myFacebookM/myMwall - for displaying User and its Friend's Message on the Timeline

Post

http://localhost:8080/myFacebookM/addM - Adding Messages by a particular User to the server [parameters("messageId".as[String], "userId".as[String], "messageFeed".as[String], "location".as[String])]

http://localhost:8080/myFacebookM/deleteM – Deleting Messages by a particular User from the server [parameters("messageId".as[String], "userId".as[String])]

Page Module

Get

http://localhost:8080/myFacebookPa/helloPa - for testing the **Page** domain (myFacebookPa)

http://localhost:8080/myFacebookPa/allPa - for displaying all the pages available

http://localhost:8080/myFacebookPa/allPaU - for displaying all pages and the users of those pages

http://localhost:8080/myFacebookPa/MyPages – for displaying the pages liked or used by a particular User

Post

http://localhost:8080/myFacebookPa/addPage – for adding pages to the Server [parameters("pageId".as[String], "msg".as[String])]

http://localhost:8080/myFacebookPa/addPageUser – for adding a page to a User [parameters("pageId".as[String], "userId".as[String])]

http://localhost:8080/ myFacebookPa/deletePageUser – for deleting a page from a User [parameters("pageId".as[String], "userId".as[String])]

Friend List Module

Get

http://localhost:8080/myFacebookF/helloF - for testing the **Friend List** domain (myFacebookF).

http://localhost:8080/myFacebookF/allF - for testing purpose to see the friendship between users.

http://localhost:8080/myFacebookF/myF - for displaying the friend list of a particular User.

Post

http://localhost:8080/myFacebookF/addF - for adding a Friend to a User [parameters("friendId".as[String], "user1Id".as[String], "user2Id".as[String])]

http://localhost:8080/myFacebookF/deleteF - for deleting a Friend from a User [parameters("user1Id".as[String], "user2Id".as[String])]

Photo & Album Module

[**Here we have assumed that adding any photo creates a default album if the User does not specifically create an album and that every photo is associated with an album**]

Get

http://localhost:8080/myFacebookPh/helloPh - for testing the **Photo and Album** domain (myFacebookPh)

http://localhost:8080/myFacebookPh/allPh - for testing all photos that are available

http://localhost:8080/myFacebookPh/myPh - for displaying photos of a particular user .

http://localhost:8080/myFacebookPh/myAl - for displaying albums of a particular user

http://localhost:8080/myFacebookPh/AlbumDetails - for displaying the album details of a particular user

Post

http://localhost:8080/myFacebookPh/addPh - for adding photos by a particular user [parameters("photoId".as[String], "albumId".as[String], "userId".as[String])]

http://localhost:8080/myFacebookPh/deletePh - for deleting photos by a particular user [parameters("photoId".as[String], "userId".as[String])]

http://localhost:8080/myFacebookPh/deleteAl – for deleting an album of a particular user [parameters("albumId".as[String], "userId".as[String])]

We have simulated Users based on their activity rates and assigned them friends, posts, photos and albums according to their activity rates.
High Users – 100% of feeds/photos/friends available
Medium User - 60% of feeds/photos/friends available
Low User – 40% of feeds/photos/friends available
Infrequent User – 20%% of feeds/photos/friends available