COP5615: DOSP Project 4 Part-2 Report

Name: Pranath Reddy Kumbam
UFID: 8512-0977

Problem Statement

The first part of the fourth course project involved the development of a Twitter clone with support for various functionalities and a client tester/simulator. The Twitter engine developed in the first part of the project is paired up with WebSockets in the second part of the project in order to provide a complete Twitter-like functionality. Additionally, the second part of the project also involves the implementation of a public key-based authentication method.

language: Erlang

Implementation Details

As suggested in the problem statement, the Twitter engine now uses a websocket interface. Part-1 of my project used the gen_tcp interface for server-client communication, and for part-2, I am using the websocket interface with JSON support. I have used **Cowboy** for implementing the websockets, and the JSON data for messaging is parsed using the **MochiWeb** framework. For the bonus part, I have also implemented a public key-based authentication method using **RSA-2048**. The project uses Erlang-make, and as mentioned earlier, the dependencies include Cowboy and MochiWeb. The components of the source code include the application, the supervisor, the handler, and the module for bonus part implementation. The application has the dispatcher, and the handler contains the primary code of the server that contains all the functions for each service and handles all the requests from the client. The init function first establishes a websocket handshake between the server and the client, and then the websocket_init function sends a confirmation to the client after the handshake has been made. The websocket_handle function receives the messages from the client and handles the specific requests. Here we parse the JSON data that the client has sent us, and then we perform the action by calling its respective function. The server can handle several types of requests, such as user registration, login, tweets, subscriptions, queries, and so on. For client-side operations, we can submit requests to the server in the form of JSON messages after connecting to the websocket, and I have used the **Postman** platform for this. Last but not least, I have also implemented a module for the bonus part that is a public key-based authentication method based on RSA 2048.

Execution Details

To start the server, we need to run "*make run"* in the project directory. Once the server has started, we can connect to the websocket on port 8080 and send JSON requests.

Server:

%% directory: project4_ws/
- make run

Client:

- localhost:8080

Bonus:
To Generate RSA keys:
%% directory: project4_ws/_rel/project4_ws_release/
openssl genrsa -out rsa.pem 2048

Functionalities

Once the client has connected to the server, we can send a request for a specific service in the form of a JSON message and a detailed example guide is presented below.

Client JSON Commands Guide

Register: {"action":"register", "username":"user1", "password":"password123"}
Login: {"action":"login", "username":"user1", "password":"password123"}
Tweet: {"action":"tweet", "username":"user1", "tweet":"hello #hashtag"}
Retweet: {"action":"retweet", "username":"user1"}
Subscribe: {"action":"subscribe", "username":"user2", "subscribe_to":"user1"}
Hashtag Query: {"action":"hashQuery", "username":"user1", "query":"#hashtag"}
Sub Query: {"action":"subQuery", "username":"user1", "query":"SubTweets", "keyword":"hello"}
Mention Query: {"action":"mentionQuery", "username":"user1", "query":"Mentions"}
Logoff: {"action":"logoff", "username":"user1", "password":"password123"}
Bonus: {"action":"bonus", "username":"user1", "path":"rsa.pem"}

Submission Details

The submission zip file contains the project folder (project4_ws). The contents of the source (src) folder are as follows:

project4_ws_app.erl - *application*
project4_ws_sup.erl - *supervisor*
pubKeyAuth.erl - *module for bonus implementation*
server_handler.erl - *handler code containing the primary functions*

Video Submission

YouTube Link: *https://youtu.be/U_rz8cT55Z0*
Drive Link: *https://drive.google.com/file/d/1r3NoF6rQijTG2IDDGuHFEitByv2gSQjr/view?usp=sharing*