Reddit Clone REST API and Client Implementation

Team Members:

1. Revathi Patel Marri (ID: 7886-2650)
2. Asna Maheen (ID: 6882-6446)

Introduction

In this report, you will read about the design, development, and implementation of a RESTful API backend for a Reddit-like platform with a client application that connects to it. Through the system, you can exercise every basic function: user registration, subreddit management, posting content, hierarchical commenting, private messaging, and voting. Additionally, it comes with a public-key-based digital signature mechanism for post-verification, an optional bonus feature.

The project aimed to achieve the following:

• Scala REST API engine design with a structure similar to Reddit's.

• Generate a client to interact with the engine.

• Demonstrate robust user interactions through a multi-client simulation.

• Offer secure, verifiable mechanisms for content authenticity.

System Design

1. REST API Architecture

- **Framework:** The server uses the Gorilla Mux router to manage HTTP requests.Data Models:.
- **Data Models:** User, subreddit, post, comment, and message entities are all in-memory storage with mutexes for synchronization
- **Endpoints:**
  - Endpoints:
  - User Registration (/users)
  - Subreddit Management (/subreddits)
  - Post Management (/posts)
  - Comment Management (/comments)

    - Messaging (/messages)
    - Voting (/votes)
    - Get Post by ID with Signature Validation (/posts/{postID})

2. Public Key Infrastructure (PKI) for Content Authenticity

- Users register using a public key RSA-2048 or EC-256.
- Posts involve signing off using the private key to yield a digital signature that verifies the origin of the post.
- This digital signature is verified by the server against the public key stored to ensure the authenticity of the post.

3. Client Application

The client application will interact with the REST API for all operations supported. Major functionalities are:

- **Account Management:** Registering users using unique usernames and public keys.
- **Subreddit Operations:** Creating and joining subreddits plus viewing all subreddits.
- **Posting:** Creating posts with digital signatures.
- **Comments and Replies:** Adding comments and replies to posts.
- **Messaging:** Sending private messages.
- **Voting:** Upvoting or downvoting posts.
- **Content Verification:** Retrieving posts with signature validation

---

Implementation Details

Core Functionalities

1. **User Registration:** Users first come on the platform with a unique username and their public key, automatically generated.
2. **Subreddit Management:** Joining, creating and listing of relevant subreddits are supported at this stage.
3. **Posting and Voting:** All posts that users engaged in are digitally signed and they have the option of voting a post up or down.
4. **Commenting:** Comments and replies to comments can also be made in a parent-child manner..
5. **Messaging:** Users who have registered are enabled to communicate privately with other users
6. **Post Verification:** All posts made are verified for integrity as well as veracity.

Concurrency Handling

- Mutex locks prevent threads from stepping over one another doing a concurrent user action's operations.
- Service seems to be scalable for multiple client at the same time.

Execution Results

The system underwent a multi-client simulation and evaluation. Below are achieved results:

User and Subreddit Management

- Registered users: Revathi, Asna, and Jack.
- Created subreddits: "sub1," "sub2," and "sub3."
- Users joined subreddits, e.g., Revathi joined "sub1."

Posting and Commenting

- Revathi authored a post in "sub1" with the subject "Hey!!" and body saying, "Welcome."
- There were replies and comments which showed a certain degree of hierarchy.

Voting and Messaging

- There was successful upvoting and downvoting of the posts.
- Users communicated through private messages to each other.

Video Demonstration

A video demonstrating the following is provided:

1. Set up and run the server and clients.
2. Handle user registration and interaction with subreddits.
3. Post content and demonstrate signature validation.
4. Handle simultaneous client requests.

Conclusion

This project is a simple yet elegant demonstration of a RESTful platform that imitates how Reddit works. The use of digital certificates improves the integrity and ownership of the submitted content. The system certainly proved to be scalable to multiple clients allowing a strong base for future development such as:

- The embedding of a persistent database.
- Load testing for multiuser simulations on a large scale.

YouTube Link: https://youtu.be/mhAQpn1fhEk?si=Q5gY3_U0K3xbAmZT