SYSTEM DESIGN DOCUMENT

Coin Flipping Game using Stellar

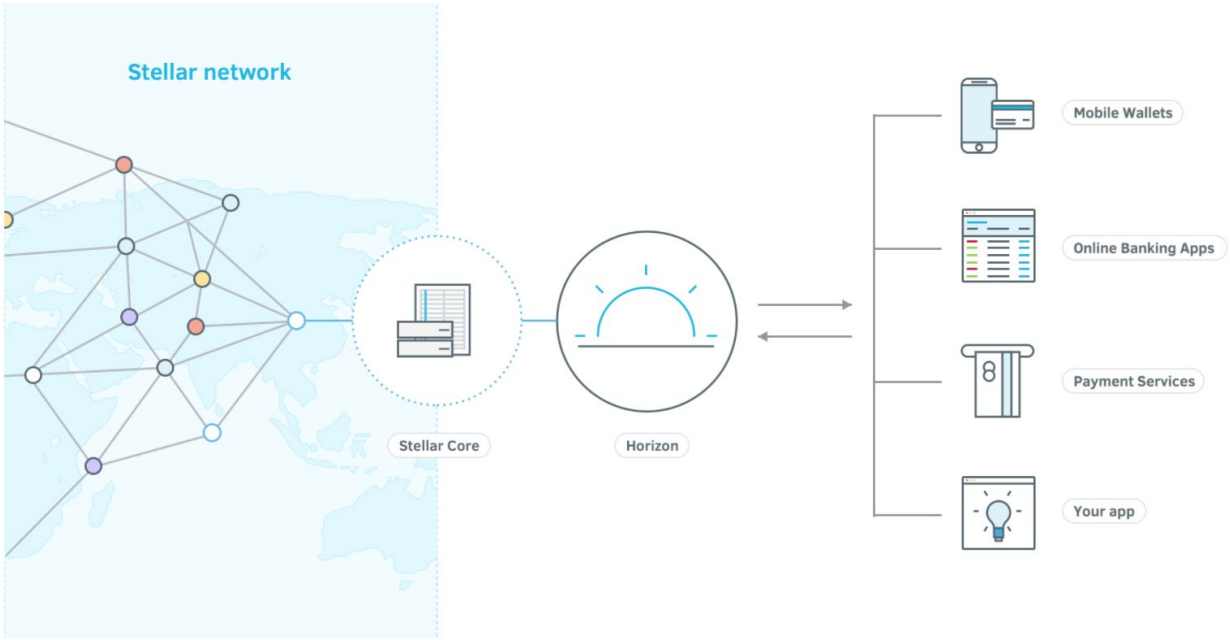
Shikhar Vats

1 INTRODUCTION

1.1 Purpose and Scope

1.2 Project Executive Summary

1.2.1 System Overview



Simple Architecture of Stellar Network

Our application system consists of two basic modules the client and the server:

It includes the following features:

Client side:

- The command line User Interface
- Creating a user Account
- Funding the account
- Making a transaction
- Checking the history of the transactions made

Server side:

- Transfer the lumens to the winner
- Host the coin-flip game
- Hold 95% of the bids and send the rest

Data associated with each transaction:

- Amount to be transferred
- Public ID of the target

Data associated with each new game:

- Random number
- Guess Number
- Amount for bet

Implementation:

- CS-Communication: Java using Socket Programming
- Client: Horizon API for connection with Stellar environment

The Interface file for the project

2 SYSTEM ARCHITECTURE

The system architecture has three main components:

1. The client side: Participant.java
2. The server side: Banker.java
3. The creating account side: CreateAccount.java

The server and the servant part of the application operate out of the same directory, while the client operates from a different directory.

1. Please explain/prove whether this protocol actually works or not, in term of fairness, and what are the possible attacks (cheating) among the three parties. In your implementation, how have you solved them?

I think the protocol works, as long as the guess values and the seeds are properly encrypted before sending them. As if an attacker can see the seed, he can find the guess of the other participant. The attacker can pose as any the banker, or the participants. All he needs is a man in the middle attack, and ARP spoofing to get the packets. I have use MD5 cryptography in the implementation, but even it has its own problems, as it is susceptible to collision attacks. As a solution to this we can use SHA256.

3 HUMAN-MACHINE INTERFACE

The interface for the Distributed Auction program is completely command-line based.

3.1 Inputs

The program asks the user for an input after displaying a menu.

For the client:

The input can be any number ranging from 1-5.

1. Create a Stellar user account
2. Fund the user account
3. Make a transaction
4. Check the history of payment
5. Exit

For the client who wants to play the game:

The input is:

1. Enter private Key
2. Enter Public Key
3. Enter Bet Amount
4. Enter Seed value
5. Exit/Retry the account

3.2 Outputs

Based on the input, the program returns the relevant output from the server side.

4 Steps to run the program

```
javac -cp ".:./lib/stellar-sdk.jar:./lib/commons-codec-1.11.jar" Banker.java
Participant.java
java -cp ".:./lib/stellar-sdk.jar:./lib/commons-codec-1.11.jar" Banker

java -cp ".:./lib/stellar-sdk.jar:./lib/commons-codec-1.11.jar" Participant

javac -cp ".:./lib/stellar-sdk.jar:./lib/commons-codec-1.11.jar" CreateAccount.java

java -cp ".:./lib/stellar-sdk.jar:./lib/commons-codec-1.11.jar" CreateAccount
```