DEPARTMENT OF COMPUTING

COMP1011 Individual Project

Block Chain System

Name:
CHEN Junru

Student ID:
21105092d

Contents

1 Problem Statement

In this project, I need to implement a simple block-chain system in C++. The system need to be able to add new blocks after the block-chain, to verify the integrity of its own data, to store a certain amount of data, and to support multiple ways of query and access.

What kind of data structure to use to store the data, how to read and write file, and how to design the block-chain system's structure are the major problems.

2 Objectives of the program

This simple block chain system is designed for teaching and demonstration purposes.

It contains block hash operations, block-chain integrity verification operations. Users can create accounts, trade with other users, get information about all transactions on the block-chain, and add new blocks to the block-chain. It also applies the Proof-of-Work (PoW) and the RSA algorithms. Users can observe a simple virtual currency system through it.

This report describes the implementation and functionality of the block-chain system developed in C++. This project is a part of the PolyU COMP 1011 Programming Fundamentals.

3 Program Design

3.1 Interaction Design

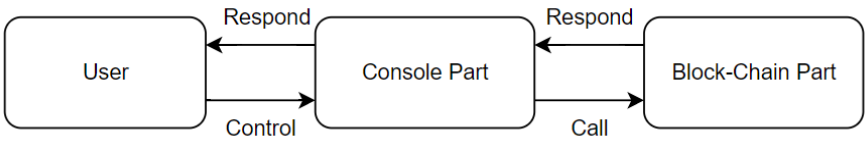

Figure 1: Overall interaction mode

The program is mainly composed of two parts: the console part and the block-chain part. Users interact with console, and the console will call the function in block-chain class to implement the users' instructions.

3.2 Transaction Pool

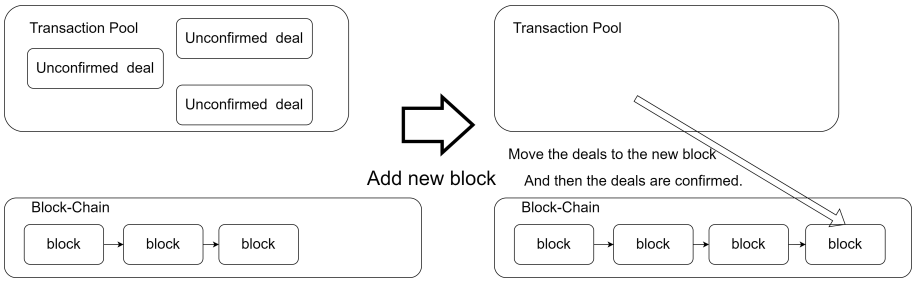

Figure 2: Deals been confirmed and moved into the new block

In the program, user can start a new deal like registering a new user or making a transfer to others. Every new deal will be placed into the transaction pool at first, and all the deals in transaction pool are not confirmed by the system. At this time, Newly created users are not able to send or receive money, and newly initiated transfers do not affect the balance of either party.

Then, if there is a user passed the Proof-of-Work (PoW) and add a new block into the block-chain, the deals in transaction pool will be confirmed and stored in that new block.

3.3 Authentication

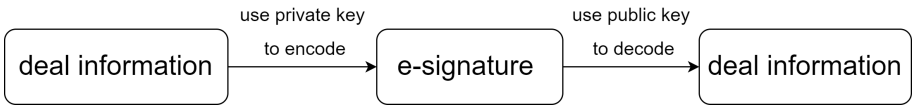

Figure 3: Overall process

In order to prevent others from forging deal information, the deal information will generates a corresponding electronic signature through initiator's private key. The electronic signature can be decrypted by the initiator's public key, and the transaction information will be submitted to the transaction pool only when the decrypted electronic signature and the transaction information are identical.

4 Structure of the program

4.1 Console Part

The console part covers the entire user menu. The user menu consist of parent menu and son menu. Parent menu is used for choose the son menu. Son menu is a while loop, so that the program will return to the parent menu only when the user decides to return to the parent menu.

Console part will call the block-chain class's function to implement the user's instructions.

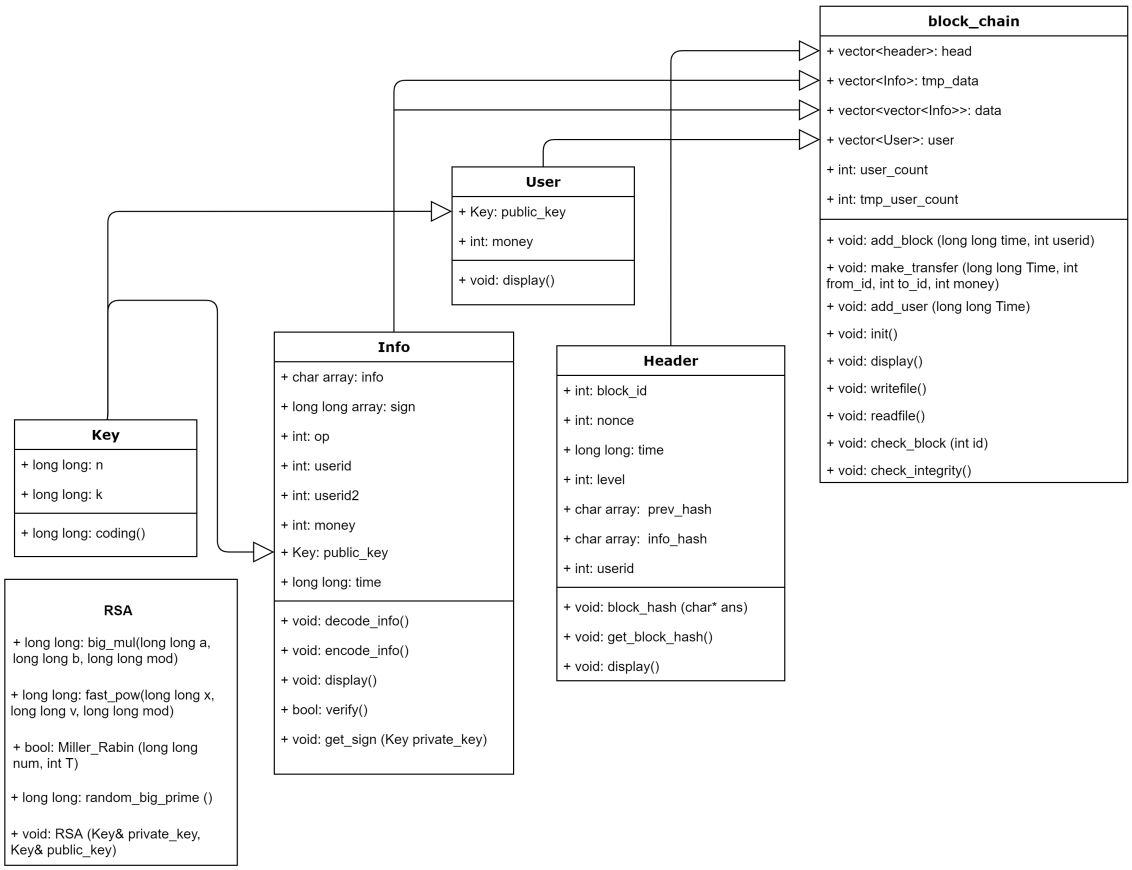

Figure 4: Overall Structure

There are five class in total, the Block-chain class, the Header class, the Info class, the User class and the Key class. Block-Chain class is the core of the block-chain part, and it combines three major class, the Header class, the Info class and the User class. Key class will appear in User class and Info class, as the deal information needs Key class variables private and public keys for encryption and verification.

4.2 Block-Chain class

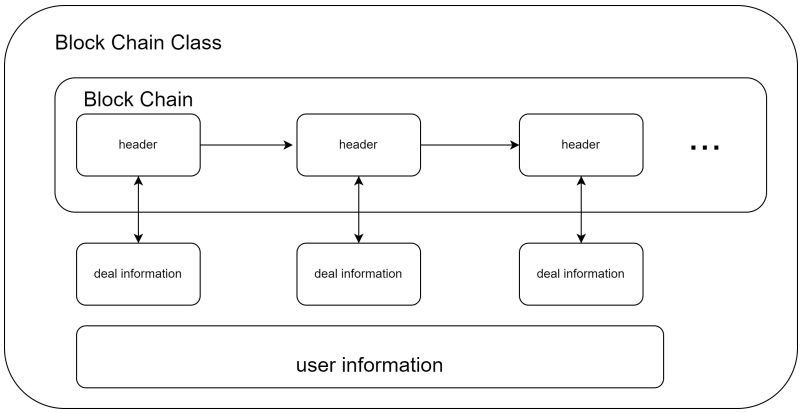

Figure 5: Block-Chain's Design

Block Chain class is actually composed of three parts, header, deal information, and user information. And in fact, the block of block-chain is only the header part. The deal information does not stored on the block-chain. The deal information is hashed and its hash value is stored in the header to ensure its integrity. The advantage of this is that the size of each block on the block-chain is fixed. And in practical applications, users often only need to keep a light chain of header. When the user needs to visit the deal information of one node, then load the node's detailed information.

User information is summarized from the deal information and used to check the operation's validation when making a transfer.

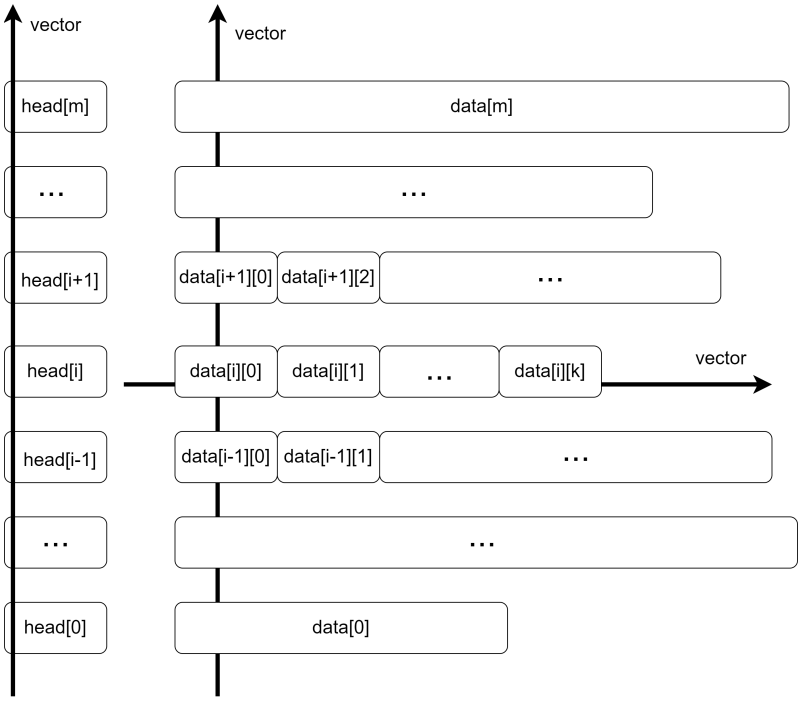

Figure 6: Block-Chain's data storage

As this program only deals with a single block-chain, the entire block-chain is stored via a c++ vector. The block header is stored in a separate vector "head", while the deal information is stored in the vector "data". "head" and "data" correspond to each other. head[i] is the header of block i on the block chain (block 0 is not considered), while data[i] is the deal information of block i on the block chain.

At the same time, there is more than one deal information stored on one block, so "data" is a vector of vectors, and data[i] is the vector of deal information stored on the block i.

4.3 Info class

Info class mainly involves two parts, the char array "info" and the electronic signature "sign" part, and the deal information part. These two parts can be converted to each other, so that the deal information can be derived in case the char

array "info" is known, and the char array "info" can be derived in case the deal information is known. Function "encode()" is to transfer the deal information into char array "info", while the function "decode()" is to the contrary. The char array "info" is mainly used for writing to files.

The electronic signature "sign" must be generated with the private key of the user who initiated the transaction, and the public key of the user who initiated the transaction is required to decrypt the electronic signature. "get_sign()" is the function that calculate the e-signature.

The format of char array "info" is:

```
/*
"A" + ID(10 digits) + PK(40 digits) + T(20 digits)
Register new user ID with public key PK at UNIX time T.
"T" + ID1(10 digits) + PK(40 digits) + T(20 digits) + ID2(10 digits) + M(10 digits)
User ID with public key PK transfer M coins to user ID2 at UNIX time T.
*/
```

4.4 Header Class

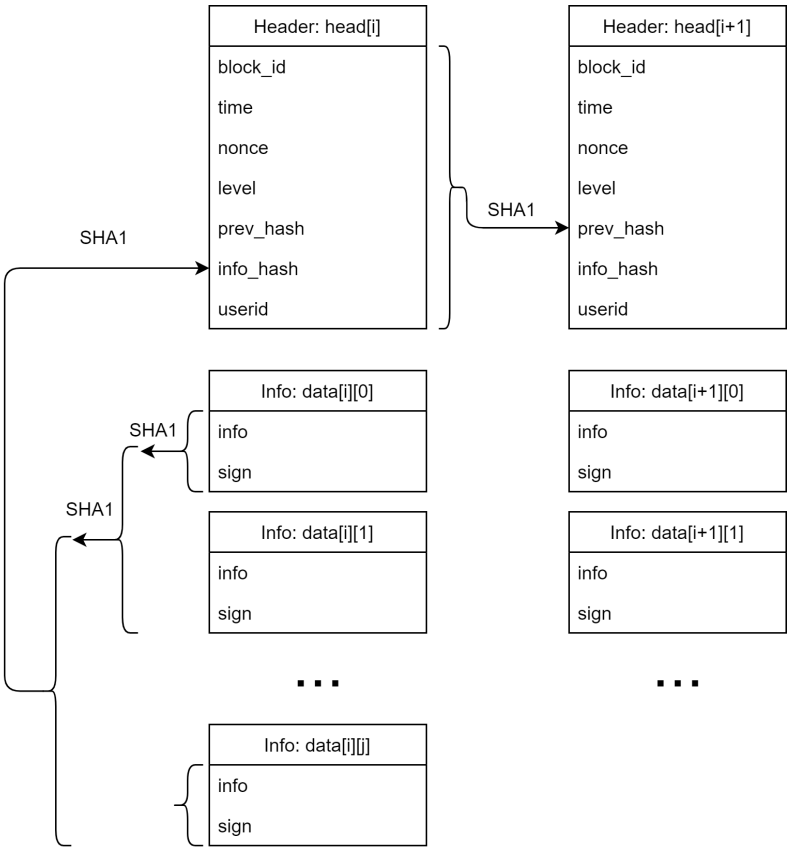

Figure 7: Header

"prev_hash" is the hash value of the previous header, "info_hash" is the hash value of the current block's deal information. Header also contains the user to which the header belongs and the time when the block was added.

Nonce is used for Proof-of-Work(PoW), the "level" means that how many digits in the front of the block's hash value should be 0. In the code, it is 2, means that the first 2 hexadecimal digits should be 0. User need to randomly generate the nonce to let the hash value reach to the standard.

4.5 RSA

The RSA consists of six functions, "big_mul()", "fast_pow()", "Miller_Rabin()", "random_big_prime()", "exgcd()", and "RSA()".

"big_mul()" is used for solving the problem that the overflow of the multiplication of two big long long type integer. "fast_pow()" is used to compute the integer powers. "Miller_Rabin" is a algorithm used for testing whether a big number is a prime.

When the program generates two big prime p_1, p_2 and let $n = p_1 \times p_2$. It can easily calculate the Euler function $\Phi(n)$ and it's equal to $(p_1 - 1) \times (p_2 - 1)$.

Then, the program will use Extended Euclid algorithm, "exgcd()", to get a pair of numbers a, b that $a \times b \equiv 1(mod\ \Phi(n))$.

According to Euler's theorem, any value p that are mutually prime with n, there is $p^{\Phi(n)} \equiv 1(mod\ n)$. So that $(p^a)^b \equiv p^{\Phi(n)+1} \equiv p(mod\ n)$. So that the program can encrypt the p into $p^a(mod\ n)$ with one key (n, a), and then decrypt $p^a(mod\ n)$ into p with another key (n, b). The program will then assign these two keys to a user as a private key and a public key respectively.

5 User Manual

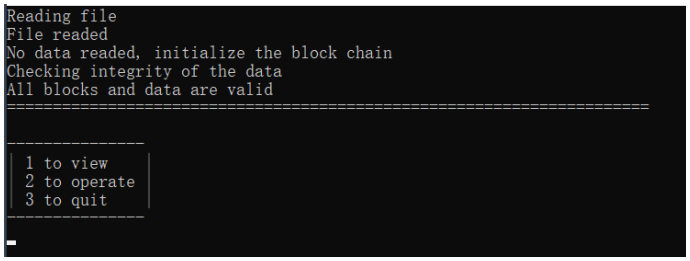

Figure 8: Beginning

At the beginning, we can see this parent menu. Input 1 we can enter the "view" menu. Input 2 we can enter the "operator" menu. Input 3 the program will exit and save.

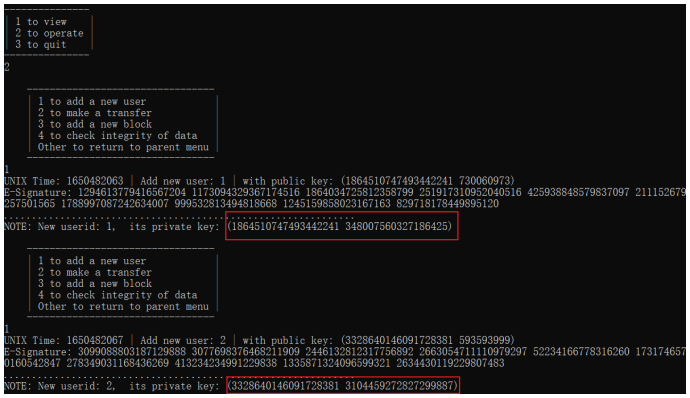

Figure 9: Create new users

At first, we input 2 to enter the "operator" menu, and input 1 twice to register two new users. Please remember the private key of the users. Now, these two users is in transaction pool, and haven't been confirmed.

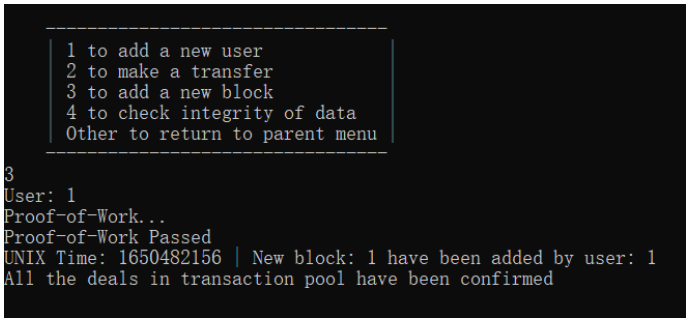

Figure 10: Add new block

Now we input 3 to add a new block, and input 1 to let this block belong to user 1. Then, all the deals in transaction pool have been confirmed and recorded in this block.

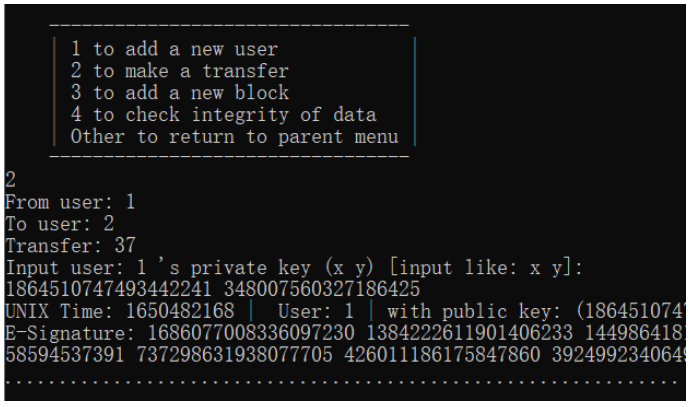

Figure 11: Make new transfer

Now we make a 31 coins transfer from user 1 to user 2, and input the user 1's private key for the validation of the transfer. The deal will be move to transaction pool, and wait for new block to confirm it.

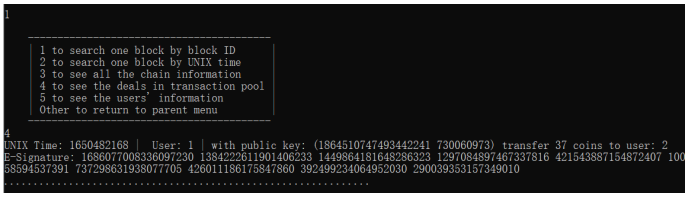

Figure 12: Transaction Pool

We can see that the transfer is in transaction pool.

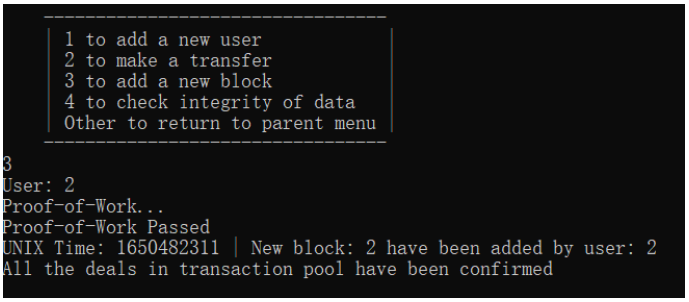

Figure 13: Add new block

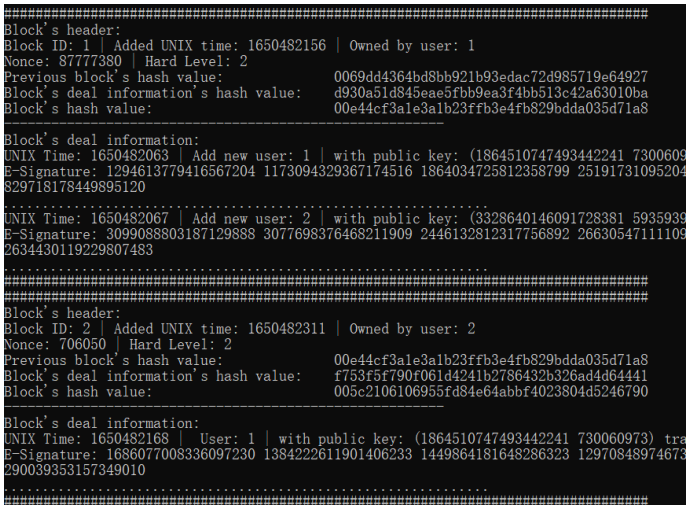

Figure 14: the block information

After adding a new block, the deal is confirmed by the new block. And we can see the block information by using "view" menu's function.