CSI2108–Cryptographic Concepts

Assignment-2: Part 3 (15%)

Cryptocurrency Simulation (Block Chaining and Block Mining)

Due Date: 27 May, 2019 at 1400 Hours

Task

For this task you will be required to simulate a crypto currency system using Python. Crypto currencies include two important parts namely: block chaining and block mining. A cryptocurrency network is a decentralised system and there is no central authority to verify transactions. Individuals are free to add new transactions which get broadcasted to the entire network. All transactions must be signed and subsequently verified for legitimacy. Only legitimate transactions are included in new blocks and illegal transactions are rejected.

Block miners continuously listen for new legitimate transactions transmitted through the communication network between the participating miners, and combine previous hash values and transaction details to compute a unique nonce value that solves the complex computation problem set by the cryptocurrency miner algorithm. The first miner who successfully computes (mines) the correct nonce, broadcasts the new block (previous hash + transactions + proof of nonce calculation work), leading to an update of all transaction entries for all users of the cryptocurrency network. The first successful miner gets rewarded (in cryptocurrency) for the computation effort put in for calculating the correct nonce.

Including hash of the previous block to new blocks is referred to as ***block chaining***. This ensures transactions are not tampered with, as any changes to transactions will change the hash of dependent blocks. Block mining process includes listening for new transactions and combining the hash of previous blocks and the transaction data, and then finding the nonce that solves the problem of computing the correct nonce quickly. Bitcoin mining or block mining serves to add new blocks as well as release new bitcoins. Furthermore, in a decentralised bitcoin network adding new blocks to the global block chain requires the consensus of more than half the mining power. Refer to this YouTube video to get a better idea of block chaining and crypto currencies. The image below shows a sample block-chaining process.

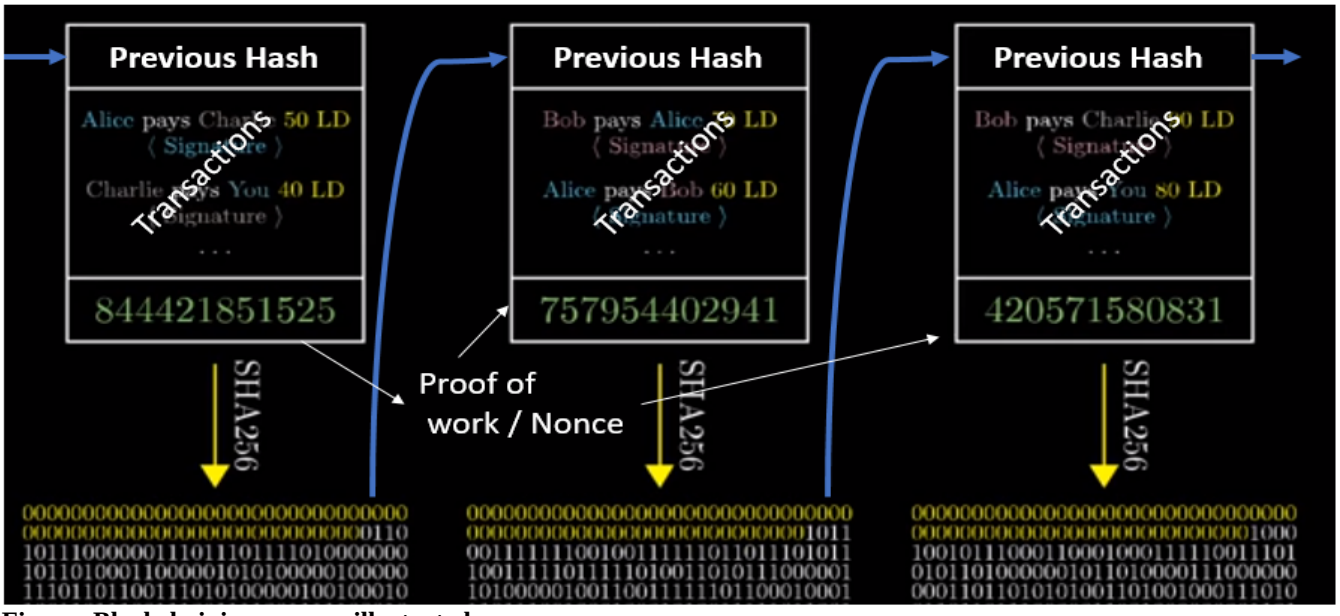

Figure: Blockchaining process illustrated.

This workshop requires you to develop a Python-based cryptocurrency system that has two components:

1. Transaction recording program
2. Block mining program

Transaction recording program

The main role of this program will be to collect information from the users for new transactions (prompting users to enter the below details through the command prompt) and storing this information into a transaction file. The transaction data should have the following information:

- From (username of person sending the cryptocurrency amount)
- To (username of person receiving the cryptocurrency amount)
- Amount
- Timestamp

This program should do the following:

1. Loop continuously to prompt the user if a new transaction needs to be added to the chain,
2. If **yes** prompt for: FROM name, TO name and AMOUNT to send.
3. Store the transaction details to a transaction file in the current directory.
4. Provide option to quit the program during its flow.

Block mining program

This program constantly checks the transaction file for new transactions. Once new transactions are detected, the mining program combines the hash of the previous block and the transaction data and then calculates the correct nonce for the block. The successful SHA-256 hash is added to the blockchain file (BC). The block structure should be as follows:

- Block Index
- Data
- Previous blocks Hash (using SHA256)
- Nonce (calculated using a mining function)

When the mining program is first executed, create the first block with an index 0 and an arbitrary previous hash value. The program should comprise the following steps:

- Create a BlockChain (BC) file in the current directory
- Check the BC file to see if any hashes of previous blocks already exist. If not, create the first block with an index = 0, data = "first block", current time stamp and the SHA-256 hash of "first block". Add the hash of the first block to the BC file.
- When new transaction data is detected, calculate the nonce that produces a SHA-256 (previous block's hash + transaction data + nonce) hash which contains 14 zeros (Why 14 zeroes? Because, it is an arbitrary number that we have chosen for CSI 2108, and it adds to the computational work that needs to be done to find the hash value of the above tuple: previous block's hash + transaction data + nonce). Refer to the nonce calculation pseudocode given below.
- Add the successful hash to the blockchain file.

- Continuously loop though these steps to add blocks comprising new data unless the user wants to exit.

Pseudocode of nonce calculation function:

Create a new block = (index + timestamp + data + previous block's hash)
Initialise a nonce variable to zero
Continuous loop:
    Update block by concatenating the nonce
    Calculate the SHA-256 of the updated block
    Check if the message digest has 14 zeros:
        Update the BC file with the successful hash
        Break
    Check if the nonce value exceeds 50000 (this step is performed to prevent an infinite loop):
        Update the BC file with last produced hash
        Break
    Increment the nonce

Requirements
You must include the following with your submission:

- The source code for your implementation of the block chaining a mining system
- Documentation explaining your implementation
- Instructions for running your program to complete the required objectives

Marking key:

- File handling (2)
- Using appropriate hashing libraries from Python (2)
- Block formation (3)
- Nonce calculation (4)
- User interaction (2)
- Correct results (2)

Submission
Your assignment must be submitted via the appropriate link on blackboard.

Notes
Your work must comply with ECU referencing guidelines and plagiarism policy