Blockchain and Future Society
Homework Set 3

2021/4/12
Instructor: Heung-No Lee
Due date: 2021/4/19

Problem 1 Suppose that there are only three groups of bitcoin miners in the worldwide bitcoin network. The first group A uses AnMiner S1 miners, the second group B uses AntMiner S3, the third group C uses Ebit E10. The percentage of each group is 55% for A, 40% for B, and 5% for C. Provide an estimation on the total number of miners working in the bitcoin network on March 14th, 2018. Give the number of miners in each group.

1.1 Use the reference for ASIC miners at https://en.bitcoin.it/wiki/Mining_hardware_comparison
1.2 Use the estimated hashrate published at https://www.blockchain.com/charts/hash-rate

Problem 2 Use the setting from Problem 1.

2.1 Find the probability of mining success per block or the percentage of the mining success per block of group C.
2.2 What is the estimate amount of money the group C pays for electricity bill per day?
2.3 What is the amount of money the group expects to earn for a day?
2.4 Find the average rate at the on-line information post at Kepco and provide your source.
2.5 Use this and give an estimate of the energy spent per day the bitcoin network consumes.

Problem 3 (RSA Encryption/Decryption) Use the RSA Calculator on line. https://www.devglan.com/online-tools/rsa-encryption-decryption. Suppose we use RSA for chain of signatures. Alice owns a Pizza shop. Bob wants to buy a Pizza (1000 Satoshi) from Alice using his bitcoin.

3.1 Generate a private and public key pair for Bob. Show your answer.
3.2 Do the same for Alice.
3.3 Write down the sequences of events that must occur to complete the pizza ordering transaction. Start it with what Alice needs to do, and end it with the event that Alice sends the Pizza to Bob in delivery.

Problem 4 Repeat the experiment in Problem 3 but using the elliptic curve cryptography defined in the downloaded BIT-ECC package or Python package. The private key is first generated by randomly picking a number; then, the public key is generated; then, the bitcoin address is generated; finally, Base58Check encoding is done. Read https://www.oreilly.com/library/view/mastering-bitcoin/9781491902639/ch04.html. See Example 4-2.

4.1 Obtain a private key of yours, say *k*.

4.2 Obtain the public key

$$K = k \times G \tag{1.1}$$

corresponding to the private key *k* generated in 4.1.

1.1 Generate the public key hash (double hashed) address A, i.e.,

$$A = RIPEMD160(SHA256(K)). \tag{1.2}$$

4.3 Obtain the Base58Check encoded address (0x00 version prefix).

Problem 5 Use SHA-256. Bob has found that the input file x_0 has the hash value y_0.

5.1 (a) He selects a file x_1 at random from his desktop computer and runs it thought SHA-256. What is the probability that this output is the same as the first output y_0?

5.2 (b) He does the same experiment like (a) 10^6 times. What's the probability that he has found the same hash?

5.3 (c) How many repetition of experiments is required so that the probability he finds the same hash y_0 is greater than 1e-6.

5.4 (d) Repeat the question (c) but this time with a little change. Let us use a growing pool of all the produced hashes in previous experiments for comparison. That is, at the *n*-th experiment, the new hash y_n is to be compared with all the previously produced hashes that are distinct with each other, i.e., $\mathcal{Y}_{past} := \{y_0, y_1, \cdots, y_{n-1}\}$. What is the probability that the new hash is the same as one the hashes he has found in the set $\mathcal{Y}_{past}$?

Problem 6 (SHA256) Write a C++ program of your own to run the SHA256 algorithm.

6.1 Obtain the SHA256 of the spelling of your name using your program.

6.2 Obtain the SHA256 of the exact same spelling of your name in 6.1. Are they the same? What does, do you think, this verify?

6.3 Add a character "." somewhere within the spelling of your name and obtain the hash. Compare it with the hash 6.2.

6.4 Repeat variations like 6.3 million times and draw the cumulative distribution function of the hash values. Does it look like a uniform distribution?

6.5 Include the program you have written.

Problem 7 (Genesis Block) Create a genesis block of your own. Use the current time of your doing homework, post 2021/4/13. Set the difficulty to be 1.000000.

1.2 Include a message showing your name into the block body.

1.3 Create the merkleroot of your block body. Just do SHA256 once.

1.4 Set the version to be a meaningless number, say 100.

1.5 Do the mining and find a good block hash.

1.6 Show the bytecode of your block.

1.7 Include a part-by-part interpretation of the bytecode of your genesis block.

Problem 8 (Currency Generation Schedule) Design a CGS program with start_block_reward = 100 and reward_interval is 105000. Reward is reduced by 30% at the end of each reward_interval.

1.8 Use Reference: https://www.oreilly.com/library/view/mastering-bitcoin/9781491902639/ch08.html
1.9 Draw the CGS for 10 years from the inception.
1.10 Discuss the good and bad aspects of this schedule, compared to the bitcoin's CGS. Is it deflationary currency?

Problem 9 What is nonce in bitcoin? How is it used? Why do you need it? What happens when you ran out of nonce? In the bitcoin system, what do we mean by data immutability? Is it really immutable? If yes, how is it done? Justify your answer.

Problem 10 The hash rate is 20 Exa hashes/sec in bitcoin network. Use your hand calculation to provide an approximate answer to the following question. Just follow the order of magnitude. Let the first 20 hexadecimals be zeros for a *good* block summary (the target hash).

1.11 (a) What is the size of *good* hash output set $\mathcal{Y}$ in decimal number (e.g., 2^256 = 1.16e77)? Note that a good hash is to mean that

$$y < \text{the target hash} \tag{1.3}$$

1.12 (b) What is the probability that you find a hash once and it happens to be a good hash?
1.13 (c) What is the probability distribution $p(k)$ such that you have to run a new hash repeatedly, say $k = 1, 2, 3, \cdots$ times, until you hit the first good hash? What is the average number of hashes you need to draw to see a single good hash?
1.14 (d) What is the size of the set of all the produced hashes in a month?
1.15 (e) Now draw a new input *x* at random and calculate the hash y of *x* and compare it with all the produced hashes in the month. What is the probability that hash collision would have occurred in the month?