CS251: Cryptocurrencies and Blockchain Technologies **Fall 2020**

Assignment #2

Due: 11:59pm on Mon., **Oct. 12, 2020**
Submit via Gradescope (each answer on a separate page) code: **92ZNG8**

Problem 1. Why is the difficulty of the proof of work in Bitcoin set to ten minutes? What would go wrong if it were changed to ten seconds?

Problem 2. Suppose two groups independently implement the Bitcoin protocol. Some miners run implementation A and other miners run implementation B. At some point an attacker finds a vulnerability in implementation A that causes miners running that implementation to accept transactions that double spend a UTXO. Implementation B treats such transactions as invalid.

a. Suppose 80% of the mining power runs the buggy implementation and 20% runs the non-buggy one. What will happen to the blockchain once a block containing a double-spending transaction is submitted to the network?

b. What will happen to the blockchain in the reverse situation where 20% of the mining power runs the buggy implementation and 80% runs the non-buggy one?

Problem 3. In this exercise we look at two estimates for the amount of energy consumed by the Bitcoin network. Assume in your answer that the current exchange rate is 1BTC = US$10000 and that there are no transaction fees (only the block reward of 6.25BTC per block). Recall that energy is measured in killoWatt-hours (kWH). You may assume that one bitcoin block is generated every 10 minutes exactly.

a. Estimate the network's hourly energy consumption assuming the entire block reward is spent on electricity for mining. Use US$0.05/kWH as the price of energy and express your answer in kWH.

b. Next, estimate the network's hourly energy consumption assuming all mining is done using Antminer S9 Hydro devices. Each device has a hash rate of 18 terra-hash/sec and consumes 1.7 kW of power (running the device for an hour consumes 1.7 kWH of energy). Assume the current difficulty of generating a bitcoin block is $D = 2^{75}$.

c. Can the difficulty ever become so great that your answer for part (b) becomes larger than your answer for part (a)? For your answer, you may assume that Antminer S9 Hydro is the best mining device available.

Problem 4. Streamlet. In class we discussed the StreamLet protocol where nodes *notarize* a block if it has 2/3rd of the vote, and *finalize* a chain up to the second to last block if the chain ends with three notarized blocks from consecutive epochs, e.g. 7-8-9. Suppose we relax the Streamlet finalization rule so that nodes consider a chain finalized up to the second to last block if the chain ends with three notarized blocks from *four* consecutive epochs, e.g. 7-8-10. Come up with an adversary that can break consistency. Here you may assume that the network is not synchronized so honest nodes can receive blocks out of order.

Problem 5. Mining pool sabotage. Recall that in section we discussed how mining pools enable individual miners to lower the variance of their earnings, while keeping the same expected returns. Participants repeatedly submit shares (PoW solutions that are valid at a lower difficulty) to prove how much work they are doing. Whenever the pool finds a block, the coinbase from that block is split among the participants in proportion to the number of shares each submitted. One risk of this is sabotage, in which a participant submits shares, but withholds full solutions if they are found (no coinbase is awarded for these withheld solutions).

a. Consider a participant with mining power $\beta \in [0, 1]$ (as a fraction of global mining power) in a pool with total mining power $\alpha \in [0, 1]$ (as a fraction of global mining power), where $\beta < \alpha$. What is the expected fraction of the overall mining rewards (the rewards collectively earned by the entire network) that this individual will earn if he or she mine honestly? Assume rewards are distributed proportionally to the number of shares submitted by each participant. **Hint:** there is no need for complicated expectation calculations throughout this entire question.

b. What is the expected fraction of the overall mining rewards (the rewards collectively earned by the entire network) that this individual will earn if it devotes all of its power to sabotage? **Hint:** Because β power is no longer used to find blocks, the total network mining power is now only $(1 - \beta)$ times its full power. Therefore, P's useful mining power, as a fraction of the entire network, is now $(\alpha - \beta)/(1 - \beta)$.

c. Now consider two pools, P_1 and P_2 with mining power α_1 and α_2, respectively. What will P_2's expected share of the total earnings be it it dedicates $\beta < \alpha_2$ power towards sabotaging P_1? Note that when P_2 finds a block, it gets the entire coinbase. When P_1 finds a block, P_2 receives a fraction of the coinbase proportional to the number of shares P_2 generated while mining for P_1.
Hint: P_1's total mining power is now $\alpha_1 + \beta$, but only α_1 is used for finding a new block.

d. Provide concrete values for $\alpha_1, \alpha_2, \beta \in [0, 1]$ in which this attack is profitable for P_2 over honest behavior.

Problem 6. Synchronizing the mempool across miners. Miners A and B each have a set of transactions in their mempool. Suppose that miner A's set is a superset of miner B's. Miner A wants to send to B the transactions that B is missing. The problem is that A does not know which transactions B is missing.

a. Suppose B is only missing one transaction. Show that A can send a single 32-byte message to B that quickly lets B identify the missing transaction hash. B will send the missing transaction hash to A, and A will send back the transaction data.
Hint: think of computing the xor of all the transaction hashes in A's mempool.

b. Suppose B is missing k transactions, for some small k. There is a simple algebraic solution for sending the k missing transactions to B, but here we will look at an interactive solution instead. Show that A and B can engage in a protocol that takes at most $\log_2 n$ rounds to identify the k missing transaction hashes, where n is the total number of transactions in A's mempool. In each round A sends up to $2k$ hash values to B, and B responds with up to k hash values.
Hint: In the first round try applying the method from part (**a.**) twice – once to A's transactions in the lower half of the hash range and once to A's transactions in the upper half.

Problem 7. Bob posts the following wallet contract to Ethereum to manage his personal finances:

```
contract BobWallet {
    function pay(address dest, uint amount) {
       if (tx.origin == HardcodedBobAddress) dest.send(amount);
} }
```

The function `pay` lets Bob send funds to anyone he wants. Suppose Mallory can trick Bob into calling a method on a contract she controls. Explain how Mallory can transfer all the funds out of Bob's wallet into her own account.