Name: _____ NetID: _____

C839 – Blockchains and Decentralized Applications
Midterm Exam

Instructor: Kai Mast Date: 11/11/21

Question:	1	2	3	4	5	6	7	8	9	Total
Points:	5	6	4	5	7	5	4	5	6	47
Score:										

Each part of a question should only require two to three sentences to answer. If you need more space use the back of the page.

1. Blockchains are distributed systems. Before we talk about them we need to cover some basics.
   (a) (2 points) In class we talked about "crash failures" and "Byzantine failures". Briefly describe the difference between the two?

   Solution:

   - In the crash-failure model nodes either operate correctly or stop executing. However, it is not possible to differentiate between a slow, a disconnected, or a crashed node.
   - The Byzantine failure model is a superset of the crash failure model. Here faulty nodes might keep operating, e.g., they could issue faulty or conflicting messages.

   (b) (1 point) If all nodes in my system run within the same organization (meaning, we trust the operators of the nodes), does it ever make sense to consider Byzantine failures? If so, why?

   Solution: It can make sense. For example, in the original PBFT paper, the authors were not so much concerned with malicious actors, but software and hardware bugs that cause unexpected behavior.

   (c) (2 points) We also talked about different synchronicity assumptions. In a synchronous network, messages are guaranteed to arrive within a known time bound and in asynchronous network messages can potentially take forever. What is the "partially synchronous" model and why is it useful?

   Solution:

   - In the partially synchronous model messages eventually (within a finite amount of time) arrive, but the time bound in which they arrive is not known.
   - The model can be seen as a compromise between the synchronous and the asynchronous model. It is easier to reason about than the asynchronous one, but more realistic than the synchronous one.

2. In class we talked about how Bitcoin has slow confirmation times. (Here, we always assume <25% malicious miners)

   (a) (2 points) For a low-value payment (e.g., for a cup of coffee) in Bitcoin it might be sufficient to wait 2 or 3 blocks (≈ half an hour), while for larger sums of money one might wait multiple hours. Why is that?

   Solution:

   - With every additional block a transaction is "buried" in the blockchain, it becomes exponentially less likely that a transaction is reversed by the existence of a longer chain/fork.
   - For low value transactions it might be sufficient to have a somewhat low (e.g., 95%) certainty, while for high-value transactions it makes sense to wait longer to have almost absolute certainty.

   (b) (2 points) Is a transaction in Bitcoin ever 100% confirmed? If so, when? If not, why not?

   Solution: No. Bitcoin is a probabilisitc protocol. The likelihood that a transaction is reverted decreases exponentially with every new block, but never reaches 0.

   (c) (2 points) The previous question talks about three different network models, which one is assumed by Bitcoin? Why do Bitcoin and other permissionless blockchains make this assumption?

   Solution: Bitcoin assumes a synchronous network model, because the participants of the network are unknown. In an partially synchronous (or asynchronous) model there could always be an unknown longer chain due to a network partition.

3. As discussed in the previous question, Bitcoin is probably too slow for many real world applications. Let us use payment channels! (Ignore the existence of payment networks for this set of questions)

   (a) (1 point) In Lightning and similar mechanisms, we (can) have instant confirmation for payments. How does that work?

   Solution: To process a payment on an already established channel, the two parties only need to sign off a new set of commitments. They can then always settle the payment on chain if need, but do not need to do so right away.

   (b) (1 point) Is instant confirmation also possible if two parties have not established a channel yet? Why (not)?

   Solution: No, unless we have access to payment networks.

   (c) (2 points) Alice and Bob want to use a payment channel to transfer money between each other. What is the minimum number of on-chain transactions they will create during the lifetime of the channel? What are they?

   Solution: At least two.

   - One to establish a joint account for the payment channel.
   - One (or two) to settle the channel.

4. Payment channels are great for performance, but Bob also cares about privacy.

(a) (1 point) In cryptocurrencies users are protected by pseudonyms (e.g., their account number). Why is this not sufficient to protect their identity?

Solution: Once, a person uses their cryptocurrency to pay for some real world good, their account (or UTXOs) gets linked to their real-world identity. We can then trace back all their payments by parsing the account's transaction history.

(b) (2 points) In ZCash, there is a "nullifier" associated with each transaction output. What are they used for? Why are they needed in addition to the commitment for the transaction output itself?

Solution:

- In ZCash transaction outputs are represented by commitments stored on the blockchain. We cannot spend those directly, because it would reveal our identity.
- Instead, users reveal the serial number (or "nullifier") associated with the commitment as well as zero-knowledge proof demonstrating the serial number corresponds to a commitment on the blockchian.
- The nullifier is public and prevents the same commitment from being spent more than once.

(c) (2 points) Let us assume we run a committee-based protocol like PBFT over Tor. Why could this cause problems?

Solution:

- Tor has a high and very variable latencies, because all communication is routed through a peer-to-peer network.
- PBFT approximates the current latency using an exponential backup mechanisms. If the latency varies a lot it might prevent the protocol from making (much) progress.

5. Alice is bored of just doing digital payments. Gladly, Ethereum allows executing arbitrary computation on its blockchain through smart contracts.

(a) (2 points) Ethereum replaced fixed transaction fees with a notion of gas. What is gas? Why can we not just use the same transaction fee mechanism as in Bitcoin?

Solution:

- Gas pays for computation. A transactions takes longer to execute, consumes more gas.
- Fixed transaction fees do not work in this setting, because the execution time of a transaction is not known a priori. The gas budget/limit also makes sure a transaction does not execute forever.

(b) (2 points) Instead of supporting a scripting language like Bitcoin Script, Ethereum has a virtual machine (the EVM). Name two advantages of this approach.

Solution:

- The EVM executes bytecode and, as a result, supports many different high-level languages.
- In contrast to a scripting language interpreters, virtual machines only need to support a small set of instructions, which makes them more secure.

(c) (3 points) Smart contracts support implementing fungible tokens (e.g., ERC20 tokens) and non-fungible tokens (NFTs). What is the difference between the two? For each, name one example where they are useful/applicable.

Solution:

- Fungbility means that something can be split or exchanged for items of the same type.
- Fungible tokes are thus similar to currencies, as you can split and exchange those.
- Non-Fungible Tokens (NFTs), on the other hand, do not all represent the same item(s) and usually cannot be split. For example, one NFT might correspond to a particular land deed or the ownership a piece of art.

6. We can even run smart contracts off the chain using mechanisms like Plasma.

(a) (2 points) In Plasma there is something called "Mass Exit". What does it do and when is it useful?

Solution: If a Plasma chains fails. If each user of the chain tried to leave by themselves, the parent chain would probably be overloaded. Mass exits allows many users to leave a Plasma chain in a single transaction to address this.

(b) (1 point) Is Plasma more or less secure than Lightning? Explain why.

Solution: In Plasma progress can be made without all users being involved. In Lightning all participants need to sign off on a state change. It can, thus, be argued that Plasma is less secure.

(c) (2 points) Validators of a plasma chain lock stake in the parent chain. Why do they need to do this? Does everyone who wants to use the Plasma chain have to lock up some amount of stake?

Solution:

- Stake is used to determine voting power of validators, and to punish the validators in case they misbehave.
- Regular users do not need to provide stake up front to interact with the Plasma chain.

7. We also talked about committee-based or "private" blockchains. They are not as exciting but still quite useful.

(a) (1 point) PBFT can only support a small number of nodes. Why is that?

Solution: In PBFT all nodes need to talk to each other. Thus, it has quadratic ($O(n^2)$) communication complexity and cannot scale to many participants.

(b) (1 point) HotStuff scales to a slightly larger number of nodes. How does it improve over PBFT in that aspect?

Solution: HotStuff uses are star communication mechanism, which only has $O(n)$ complexity in the absence of failures. However, here all nodes still need to interact with the leader, so it also will not scale forever.

(c) (2 points) In PBFT and similar protocols there is a "view change"-mechanism. What is it used for? Why do blockchains based on Nakamoto consensus not need an explicit "view change"?

Solution:

- View changes allow electing new leaders, in case the current leader fails, becomes disconnected, or is slow.
- Leader election in Nakamoto consensus happens as part of the mining process. We do not need an explicit mechanism to pick leaders in advance.

8. Bob cares about the environment and does not want to use Proof-of-Work anymore.

   (a) (2 points) How does Proof-of-Work work? Please also explain what the nonce value in a Bitcion block header is used for in PoW.

   Solution:

   - In PoW, miners try to solve a crytpographic puzzle to mine a block (and become leader)
   - Mining is done by trying out different nonce values until the block's hash value is below some threshold.

   (b) (1 point) Ouroboros has a very long safety proof because the attacker has an advantage compared to PoW. Why is it easier for an attacker to trick other nodes in Ouroboros?

   Solution: There is virtually no cost associated with creating blocks in Ouroboros. Attackers can potentially create many blocks in a single slot.

   (c) (2 points) Avalanche can also use Proof-of-Stake. How do Avalanche nodes agree on a transaction? What is stake used for in this mechanism? (You do not need to describe the full protocol just the overall intuition)

   Solution:

   - In Avalanche randomly "ask" other nodes for their current state. Nodes might switch their state to mirror the majority response. Eventually the network converges to a single state.
   - Stake is needed here to prevent Sybil attacks and determine a nodes membership/voting power.

9. Alice recently found out about Selfish Mining and wants to stop using Bitcoin.

   (a) (2 points) What is the Selfish Mining attack? How does it work? (only briefly describe it)

   Solution:

   - In selfish mining, a pool of miners does not publish their blocks. This gives them an advantage as other nodes mine on an outdated version of the chain.
   - The selfish miners only reveal blocks if there is a public chain that is about to overtake their private one.

   (b) (2 points) The authors of the Selfish Mining paper describe a simple fix to the Bitcoin protocol that protects networks with less than 25% malicious nodes against selfish mining. What did they propose? Why does it work?

   Solution:

   - Selfish miners can always win if honest miners pick their revealed blocks to mine on, because they already have an advantage on that chain.
   - The authors propose for honest miners to randomly pick which chain to mine on, if there are two or more equally long chains.

   (c) (2 points) The "Selfish Mining Revisited" paper talks about different difficulty adjustment mechanisms. What is their purpose? Name at least two different kinds of such mechanisms.

   Solution:

   - Difficulty adjustment is needed to adapt to changes in mining power, for example, when miners join or leave the network.
   - In Bitcoin, difficulty is adjusted after a period of 2016 blocks.
   - In Ethereum, difficulty is adjusted incrementally after each block.
   - Some other blockchains, e.g., Monero, use a sliding window mechanism that recomputes the difficulty after each blcok