University of Stavanger

Faculty of Science and Technology

| | |
|---|---|
| Exam in subject: | **DAT650 Blockchain technology** |
| Date: | **Eksamen, 29th November 2022** |
| Duration: | **4 hours** |
| Allowed remedies: | **None** |
| The exam consists of: | **10 exercises on 4 pages** |
| Contact during exam: | **Leander Jehl, tlf. 94120764** |
| Remarks: | None |
| Attachments: | |

---

## Question 1: Trees (13%)

(a) (4%) Sketch a Merkle tree with 7 data elements. What data needs to be included in an inclusion proof for the third data element?

(b) (3%) If Alice did send some bitcoin to Bob, how can she convince Bob that the payment happened? How can she do this efficiently?

(c) (6%) Alice wants to *read* the value of a *owner* variable in a smart-contract deployed on Ethereum. Alice does not run her own Ethereum node. How can she get an answer she trusts. How can this be done efficiently?

## Question 2: Unspent transaction output (UTXO) (11%)

(a) (6%) What are the different elements of a transaction in Bitcoin? How is a transaction validated?

(b) (2%) How does Bitcoin prevent the following attack:

- Alice wants to send bitcoin to Bob. She issues a transaction for this transfer. Charly intercepts this transaction before it reaches the Bitcoin network. He replaces Bobs public key with his own, submits the new transaction, and thus receives the payment from Alice.

(c) (3%) Name 3 reasons, why technology like Bitcoin is not suited to be used in everyday payments.

## Question 3: Proof of work (19%)

(a) (6%) We had the following definition for a proof of work function:

Def: *For an integer d, the proof-of-work (PoW) function with difficulty d takes a data item and returns a nonce (random bits) and a hash value:*

$$(h_{PoW}, nonce) = f_{PoW}(Data)$$

*The proof of work is* valid, *if a)* $h_{PoW}$ *is the hash of the data, concatenated with the nonce*

$$h_{PoW} \stackrel{?}{=} H(Data||nonce))$$

*and b) the first* $d$ *bits of* $h_{PoW}$ *are 0.*

What are the short-commings of this definition and how can it be adjusted?

(b) (4%) Assume Alice finds a nonce that allows her to solve the PoW puzzle and create a new block for the Bitcoin blockchain. Alice sends this block to the bitcoin network.

Assume further that Bob is the first node in the network to receive Alices block. Can Bob take the Nonce from Alices block and cash in the block reward himself?

(c) (4%) Assume Alice runs a bitcoin miner. After searching for a solution for 10 minutes, Bob finds a block. If Alice stops her initial mining, and instead mines on top of the block that Bob has found, does she lose an advantage? Explain.

(d) (1%) a proof of work function needs three properties, *Fast Verification*, *Adjustable difficulty*, and *Progress freedom*. Which of the three properties of a PoW function is relevant for the example in Part c.

(e) (4%) We say that PoW is fair if in the longest chain, the amount of blocks from every miner is proportional to his mining power/hash rate.

- Under which conditions, is a PoW scheme like in Bitcoin not fair? Give an example.
- Explain one technique to alleviate this problem.

## Question 4: Attacks (3%)

(a) (3%) How can a miner benefit from a selfish-mining attack? Will he mine more blocks?

## Question 5: Proof of Stake (6%)

In PPCoin (Peercoin) a miner identified by *addr*, that has deposited `coin`(*addr*) can supply the current block, if

$$H(\texttt{prevBlockHash}||addr||\texttt{timeinseconds}) < d_0 \cdot \texttt{coin}(addr)$$

- Here $d_0$ is a base difficulty. The probability that a miner with a specific address *addr* can mine the next block is proportional to `coin`(*addr*).
- `timeinseconds` shows time in seconds. Thus, a miner gets a change to submit a solution every second.

(a) (3%) Give an example, where a PoS-miner can benefit from an attack similar to proof of work.

(b) (3%) Explain the nothing-at-stake problem. How can that problem be solved?

## Question 6: Off-Chain technology (9%)

Off-chain technology includes Payment-Channels, Payment-Channel networks and Commit Chains.

(a) (5%) What are the advantages and disadvantages or limitations of payment channels, compared to on chain transactions?

(b) (4%) What are the advantages and disadvantages of payment channels, compared to commit chains?

## Question 7: Ethereum (6%)

How are transaction fees set in Ethereum, and how can a user control, predict, or limit the fees? How can the user set fees to ensure inclusion in the blockchain?

## Question 8: Solidity (12%)

The contract given in Algorithm 1 is vulnerable to re-entrancy.

(a) (4%) Describe how the vulnerability can be exploited by an attacker.

(b) (4%) Rewrite the `deposit` function to remove the vulnerability. You should still use `call` to send money.

(c) (4%) What is the advantage of using `call`, rather than `transfer` when sending money. Why is the first vulnerable and the second is not?

**Algorithm 1** Reentrancy

```
1: pragma solidity =0.5.11
2: contract SimpleBank {
3:
4:    mapping(address => uint) balances;
5:
6:    // deposit money.
7:    function deposit() public payable {
8:       require(msg.value > 0);
9:       uint balances[msg.sender] = balances[msg.sender] + msg.value;
10:   }
11:
12:   function withdraw(uint value) public {
13:      require(balances[msg.sender] > value);
14:      (bool success) = msg.sender.call.value(value);
15:      if (success) {
16:         balances[msg.sender] = balances[msg.sender]-value;
17:      }
18:   }
19: }
```

## Question 9: Ethereum consensus (8%)

How are blocks created in Ethereum 2.0?

## Question 10: State and transactions (8%)

(a) (2%) What is recorded on the bitcoin blockchain? Where are balances stored?

(b) (2%) What is stored in the Ethereum blocks? How is this different from Bitcoin?

(c) (4%) Why is the state root important in Ethereum? What can it be used for?