Software Engineering for Business Information Systems
Informatik
Technical University of Munich

Eexam
Place student sticker here

Note:

- During the attendance check a sticker containing a unique code will be put on this exam.
- This code contains a unique number that associates this exam with your registration number.
- This number is printed both next to the code and to the signature field in the attendance check list.

Blockchain-based Systems Engineering

Exam:	IN2359 / Endterm	**Date:**	Thursday 4th August, 2022
Examiner:	Prof. Dr. Florian Matthes	**Time:**	08:15 – 09:45

	P 1	P 2	P 3	P 4	P 5	P 6	P 7	P 8	P 9
I									

Working instructions

- This exam consists of **16 pages** with a total of **9 problems**.
  Please make sure now that you received a complete copy of the exam.
- The total amount of achievable credits in this exam is 90 credits.
- Detaching pages from the exam is prohibited.
- Allowed resources: none
- Subproblems marked by * can be solved without results of previous subproblems.
- Do not write with red or green colors nor use pencils.
- Physically turn off all electronic devices, put them into your bag and close the bag.

Left room from _____ to _____ / Early submission at _____

Problem 1 Cryptographic Basics (7 credits)

For her new online payment service, Alice wants to create a one-time-password system. A login system like that is based on a chain of hashes that each user generates over an initial secret s using a hash function h:

$$h^i = h(h^{i-1}(s)) \text{ with } h^1 = h(s)$$

Bob wants to use Alice's service and goes through the following steps:

1. Bob generates a secret s and keeps it to himself.
2. Bob calculates the hash chain on his secret up to a certain point n, such as $n = 1000$.
3. Bob shares h^n with Alice's service on signup. It is used as Bob's next password hash h^k.

Everytime Bob logs into the service, the following will happen:

1. Bob uses h^{k-1} as his password when sending a login request to the service.
2. The service verifies his password by checking that $h(h^{k-1}) = h^k$, where h^k is the last value Bob used to login (initially $h^k = h^n$).
3. The service stores $h^k = h^{k-1}$ to be ready for the next login.

0 1 2

a)* Which property of cryptographic hash functions discussed in the lecture is required for h? And why is it critical?

0 1

b) * Which type of attack (that you also know from blockchain systems) does this password scheme prevent?

0 1 2

c)* Is it possible, that this chain h^i is cyclic? Give a short explanation for your answer.

0 1 2

d)* In search for a good hash algorithm h for her system, Alice found several different hash algorithms: MD4, MD5, SHA-1, SHA-2, and SHA-3. Wich ones of these are sufficiently secure for this use case? What factor determines if a given cryptographic hash algorithm is deemed secure?

Problem 2 **Blockchain Basics (7 credits)**

On an abstract level, an Ethereum account is a 4-tuple looking like Figure 2.1.

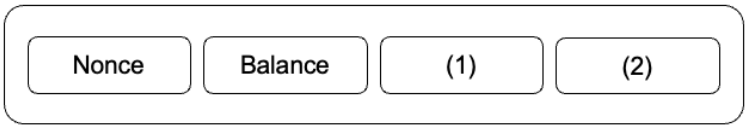


Figure 2.1: A 4-tuple representing an Ethereum account

a)* Name the two missing fields.

0 1

b) * What does it mean if *(1)* and *(2)* fields are **empty**?

0 1

c)* **Nonce** is a counter that indicates the number of transactions sent from an account. Briefly argue whether nonce will increase after a mined transaction that fails.

0 1 2

d)* Although a single Bitcoin block includes more transactions than an Ethereum block, the average number of transactions confirmed per day is 1.1 million in Ethereum, while it is 275,000 in Bitcoin. Briefly explain why this is the case.

0 1

Turing completeness of a system means that it can solve any reasonable computational problem given enough time and memory. While Bitcoin Script is **not** Turing complete (e.g., does not enable loops), Ethereum is argued to have Turing completeness in Solidity.

e)* One of the reasons why Bitcoin prefers to be not Turing complete is to avoid Denial-of-Service (DoS) attacks where nodes execute scripts that run into an infinite loop. Briefly explain how Ethereum handles this problem.

0 1 2

Problem 3 **Blockchain Transactions (15 credits)**

Here you can see one set of (slightly simplified) transactions within a block from Bitcoin and one from Ethereum. The order of transactions is given by the *position index*, which can also be used to refer to a transaction.

position index	sender	recipient	nonce	value	startgas	gas used	gas price	data
1	0xaaa...	0xddd...	3	0	**?1**	140300	7	0xb76ea962ff...
2	0xbbb...	0xaaa...	46	3x10^18	40000	**?2**	7	0x
3	0xccc...	0xddd...	42	60x10^18	800000	348 120	7	0x6080604052...
4	0xbbb...	0xccc...	**?3**	2x10^18	40000	**?2**	8	0x
...								

Table 3.1: Block A

position index	tx_{in}	tx_{out} (amount $\rightarrow$ recipient address)
1	-	6,4*10^8 $\rightarrow$ bc1aaa...
2	#1[0]	**?4** $\rightarrow$ 1bbb... 2x10^8 $\rightarrow$ bc1aaa... 2x10^8 $\rightarrow$ bc1ddd...
3	#2[0]	1x10^8 $\rightarrow$ bc1ccc... 1.2x10^8 $\rightarrow$ bc1ddd...
4	#2[2]	**?5** $\rightarrow$ bc1aaa... **?6** $\rightarrow$ bc1ddd...
...		

Table 3.2: Block B

0 1

a)* Which block is from Bitcoin and which from Ethereum? Why?

b)* Name values for the variables in both tables, such that all transactions are valid/accepted.

(?1):

(?2):

(?3):

(?4):

(?5):

(?6):

0 1 2 3 4 5

c)* What role does *bc1aaa...* most likely occupy in the network? Explain your answer.

0 1 2

d)* What kind of account is behind *0xddd...*? Explain your answer.

0 1 2

e)* The block in 3.2 shows a special transaction at index 1. Why does 3.1 not have a similar transaction? Explain how the respective mechanism works for the network in 3.1.

0 1 2

f)* There are two different types of addresses in 3.2. The ones starting with “bc1” are the newest version introduced with an upgrade known as SegWit. What goal did it have, and how did it reach it?

0 1 2

g)* Remembering or backing up a raw account address and private key is not very human-friendly and prone to errors. In what format are accounts usually exported, imported, and stored?

0 1

Problem 4 Finding Consensus in a Network (15 credits)

In a July 2022 Forbes article comparing Bitcoin and Ethereum[1], Proof of Work (PoW) is listed as Bitcoin's consensus mechanism.

0 1 2

a)* In popular media, proof of work is often discussed as a consensus mechanism. Why is it not technically correct, when Forbes calls PoW a consensus mechanism? Name and explain what type of mechanism PoW actually is.

0 1 2

b)* Explain the purpose of a consensus mechanism and name one example.

0 1 2

c)* PoW has been shown to incentivize miners to band together in pools, to stabilize their reward frequency. When a pool miner finally finds a valid block, there might be some temptation to betray the pool and keep the entire block reward to himself. Why is that not possible?

0 1 2

d)* The consensus achieved by most public blockchains like Bitcoin and Ethereum is probabilistic. Explain what that means.

[1] https://www.forbes.com/advisor/investing/cryptocurrency/bitcoin-vs-ethereum/

Alice wants to sell her used car to Bob for 14 Ether. Alice knows she needs to be absolutely sure that Bob sent the Ether before she hands over the car.

e)* In an effort to better understand a transaction on Ethereum, Bob has created the following list of stages his transaction will go through:

0 1 2 3

1. Transaction Creation
2. Transaction Broadcast
3. Block Building
4. Miner Validates Transaction
5. Miner Finds Block Hash
6. Block Broadcast
7. Contract Execution
8. Full Node State DB Update

Name the two mistakes he made and briefly explain them.

f)* If Bob's transaction ends up in an orphan block there are two broad ways this deal can possibly end. One where Bob is honest and one where he is not. Explain them.

0 1 2

g)* Up until what point does Alice roughly wait until handing over the car? Please name a concrete number of confirmations. What is the main factor determining that number for this sale transaction?

0 1 2

Problem 5 Blockchain Evolution and Challenges (15 credits)

In June 2016, a famous smart contract on Ethereum called *The DAO* was hacked and $60 million worth of Ether was stolen. After much debate, a fork that rolled back Ethereum's history to before the attack and allowed the investors to withdraw their funds was proposed. While the majority of the network agreed upon the fork, some refused to accept to roll back the history of Ethereum and stayed with the pre-forked version of the chain which is known as *Ethereum Classic (ETC)* today. The chain that implemented the fork is currently known as *Ethereum (ETH)*.

0 1 2

a) * What was the type of the fork that led to Ethereum Classic and Ethereum? Briefly explain why.

0 1 2

b)* Which key property of the blockchains did Ethereum violate by adopting this fork? Briefly explain why.

0 1 2

c)* It was later discovered that the attack exploited a re-entrancy vulnerability on the DAO smart contract. Name **two** measures that can be taken to prevent such an attack.

0 1 2 3

d)* As happened in the DAO fork case, forks can sometimes lead to so-called *chain splits* where the community splits and follows different versions of the underlying blockchain.

Assume that there are 100 miners and 100 full nodes in the network. $F_{fullnodes}$ and F_{miners} represent the number of full nodes and miners that adopted the fork. $L_{fullnodes}$ and L_{miners} represent the number of legacy full nodes and miners (i.e., nodes following the old rules). For each of the given fork adoption scenarios, determine whether a chain split will occur or not and briefly explain why.

1. **Soft Fork**: $F_{fullnodes} = 1$, $F_{miners} = 1$, $L_{fullnodes} = 99$, $L_{miners} = 99$
2. **Hard Fork**: $F_{fullnodes} = 99$, $F_{miners} = 1$, $L_{fullnodes} = 1$, $L_{miners} = 99$

Alice is a crypto-enthusiast who regularly orders pizza from Bob's and pays with ETH. After the DAO fork, Alice keeps ordering from Bob's and continues to pay with ETH. After some time, she decides to use her ETC to pay for the pizza but she realizes that her ETC balance is much less than what she expected. Upon examining the transaction history on Ethereum Classic, she finds out that Bob was the attacker.

e)* State the name of the attack that Bob executed on Alice and briefly explain how he accomplished it.

0 1 2

f)* Briefly explain why Alice would **not** be vulnerable to the same attack if she had joined Ethereum and Ethereum Classic after the DAO fork.

0 1 2

To get revenge on Bob, Alice starts working on a classic double spending attack. She decides to use **Bitcoin** to realize the attack and plans the following steps:

1. Orders pizza from Bob's and issues a Bitcoin transaction (Tx#1) to pay for it.
2. Right after getting a delivery confirmation email from Bob (assume that Bob accepts payment on Bitcoin as soon as he sees a valid transaction in the mempool), she issues a transaction (Tx#2) to herself where she spends the same UTXO she used in Tx#1.

g) * Briefly explain why Alice has to execute the second step.

0 1

h) * How can Alice incentivize the miners to include Tx#2 over Tx#1 in a block?

0 1

Problem 6 Ethereum Smart Contracts - Solidity (14 credits)

BBSea.io is a decentralized application (like the famous OpenSea.io) where users can buy and sell non-fungible tokens (NFT) of the famous *SEBISPunks* collection. The application uses Ethereum smart contracts to implement its business logic on-chain. You are hired as a Solidity developer to complete the missing functionalities on the contract. The requirements of the contract are:

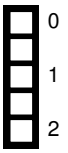
- A user can list a SEBISPunks NFT using the **listNFT** function. The function must ensure that:
  - Caller of the function is the owner of the NFT
  - The NFT is not already listed
  - Asked price is at least 1 ETH
- A user can buy a SEBISPunks NFT using the **buyNFT** function. The function must ensure that:
  - The seller has not sold more than 5 items to the buyer previously as this may indicate suspicious activity
  - 5% of the paid price is taken as a fee and the rest is transferred to the seller
  - If the seller has sold more than 10 items in total previously, only 2% fee is taken
  - The NFT is transferred to the buyer

You are allowed to use all functionality that Solidity provides. As you can see on the next page, the smart contract assumes a current version of Solidity, but the older syntax will be accepted. This Solidity version includes SafeMath and thus you do not need to account for over/underflow of variables. Some of the following code snippets may be useful:

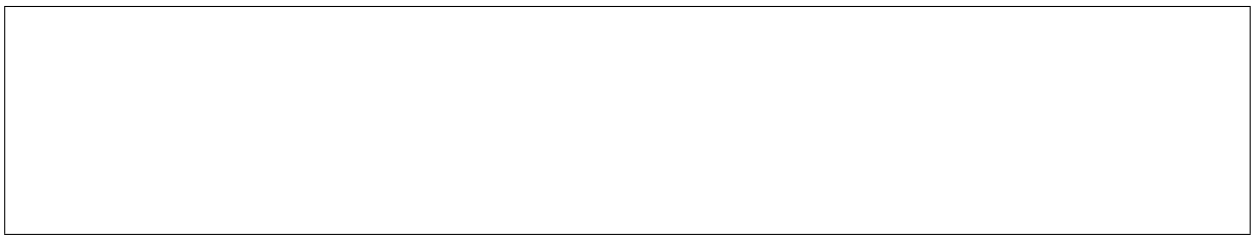
- Sender of the transaction: msg.sender
- Amount sent with the transaction: msg.value
- Enforcing conditions: require(...)
- Transfer assets: recipient.transfer(...)
- Unit literals: ether, finney, wei
- Casting arbitrary data to uint: uint(...)
- Casting an address a to a payable address: payable(a)
- Empty address: address(0)
- Returns the Ether balance of the contract: address(this).balance
- Returns the owner of an ERC721 token: tokenContract.ownerOf(tokenId)
- Transfers an ERC721 token: tokenContract.transferFrom(from, to, tokenId)

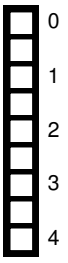
Convert the instructions on the previous page into code.

```
pragma solidity >=0.8.4;

import "./SEBISPunks.sol";

contract BBSea{
    SEBISPunks private sebisPunksContract;

    uint public constant MAX_SAFE_TRADES = 5;
    uint public constant MIN_PRICE = 10**18;
    uint public constant DISCOUNT_THRESHOLD = 10;

    mapping (address => mapping(address => uint)) itemsSold;
    mapping (address => uint) totalItemsSold;

    mapping (uint => bool) itemListed;
    mapping (uint => uint) itemPrices;

    // ensure item is listed
a)  modifier listed(uint tokenId) {
```

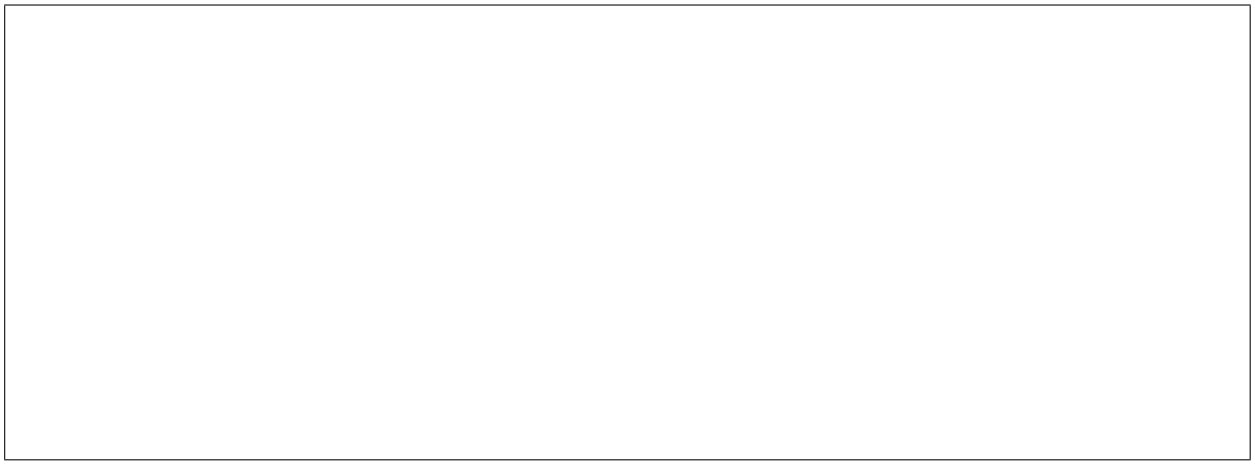


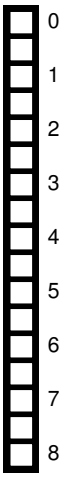

```
    }

    constructor (address _sebisPunksContract) public {
        sebisPunksContract = _sebisPunksContract;
    }

b)  function listNFT(uint tokenId, uint price) public {
```






```
        itemListed[tokenId] = true;
    }

    function buyNFT(uint tokenId) payable public listed(tokenId) {
        require(itemListed[tokenId] == true);
        require(itemPrices[tokenId] == msg.value);
        itemListed[tokenId] = false;
        itemPrices[tokenId] = 0;

c)      uint fee = 0; // initial dummy value
```






```
    }
} // end of the contract
```

Problem 7 **Tezos (5 credits)**

0 1

a) * What is the name of the entity in Tezos which has the most similar role with an Ethereum **miner**?

0 1 2

b) * Describe the mechanism which enables someone who does **not** run a block producing node to take part in Tezos' block production and explain the motivation to do so.

0 1 2

c)* What was the motivation for Tezos to hold an initial coin offering (ICO) and issue pre-minted tokens to the contributors instead of an airdrop (i.e., free giveaway of tokens)? Answer the problem with respect to the Sybil control mechanism adopted by Tezos.

Problem 8 **Hyperledger Fabric (4 credits)**

a)* What is the name of the component governing network access and what technology does it use?

0
1

b)* Name an advantage regarding lifecycle-management that Fabric chaincode has over Ethereum smart contracts.

0
1

c)* Fabric's newest ordering service uses the Raft algorithm. What kind of guarantee does it provide? How does it compare to what Ethereum guarantees in terms of attack resiliency?

0
1

d)* Assume a network of 20 nodes maintaining a Fabric ledger for supply chain tracing. How many malicious nodes are required to disrupt the ledger operation?

0
1

Problem 9 **Micro-Lectures (8 credits)**

0 ☐
1 ☐
2 ☐
3 ☐
4 ☐
5 ☐
6 ☐
7 ☐
8 ☐

* Check whether the following statements are **true** or **false**. If they are false, write correction to the solution box. Use the index number to refer to a statement.

Index	Statement	True	False
1	NFTs themselves are immutable, their metadata is not.		
2	MEV can only be extracted by miners.		
3	MEV is significantly less common in Bitcoin compared to Ethereum.		
4	Total existing MEV in Ethereum can be quantified.		
5	A verifier has to communicate with the issuer to verify a Verifiable Credential.		
6	A Verifiable Presentation can contain multiple Verifiable Credentials.		
7	The DID Specification does not standardize any specific DID Methods.		
8	An SSI ecosystem requires a blockchain.		

Additional space for solutions–clearly mark the (sub)problem your answers are related to and strike out invalid solutions.