Exam

„Blockchain-based Systems Engineering“

Software Engineering for Business Information Systems (sebis)

Date: Thursday, 26th July 2018

Summer Term 2018

Examiner: Prof. Dr. Florian Matthes

Name, first name: ______________________

Degree program: ______________________

Matriculation nr.: ______________________

Signature: ______________________

Important Information

- A processing time of 60 minutes is available.
- A maximum of 60 points can be achieved in this exam.
- Points are awarded for correct intermediate results even if the final result is incorrect or missing. Explain your solutions as precisely as possible!
- In the case of attempts to deceive, the exam is rated 5.0 and appropriate steps will be taken.
- Please do NOT write in red or green, nor in pencil. Such solutions will NOT be considered during the correction!
- Use the concepts presented in the lecture or exercise to solve the tasks.

Auxiliaries

- None.

1	2	3	4	5	∑

1. Exercise – Cryptographic Basics (6 P.)

1.1. Name three key properties of **cryptographic** hash-functions and explain them briefly. (3 P.)

1.2. You are provided with three transactions (tx_1, tx_2, tx_3) and a hash function h: ByteArray → ByteArray and a concatenation function +: (ByteArray x ByteArray) → ByteArray. Draw a Merkle tree which contains these three transactions as leaves. For each node of the tree write the formula to compute the node value. (3 P.)

2. Exercise – Bitcoin and Consensus (21 P.)

2.1. Explain the purpose of a **coinbase transaction**. Which entity of the network creates the coinbase transaction? (2 P.)

2.2. For solving the mining puzzle, the miners have to change some contents of the block. Name two fields in the **block header** that could be changed **directly** for that purpose. What additional changes can be introduced in the coinbase transaction? (2 P.)

2.3. To store data in the Bitcoin Blockchain, users send bitcoins to addresses containing data, for example “15ARZiNUjKEFAspvJaZnLQn1VcybPQNCBu”. However, in the lecture, we introduced other ways to store arbitrary data into the Blockchain of Bitcoin. Describe two ways to store data. Be precise about the way/location the data is stored. (2 P.)

2.4. Explain if it is possible to generate new coins (in addition to the regular rewards) with a 51% attack. (2 P.)

2.5. Below you see a diagram with four transactions through which Alice obtains from sebis two unspent transaction outputs of 12,5 bitcoins each. Transaction fees are ignored.

Tx #0	
∅ *(TXin)*	12,5 → **Sebis** *(TXOut)*

Tx #1	
∅	12,5 → **Sebis**

Tx #2	
#0[0]	12,5 → **Alice**

Tx #3	
#1[0]	12,5 → **Alice**

Alice has to do two payments. First, she wants to pay Bob 15 bitcoins and second Carol 8 bitcoins. Draw the transactions analogue to the above diagram. (3 P.)

2.6. Explain the term change address and explain their reason for existence in Bitcoin. (2 P.)

2.7. Bitcoin Script: Consider following TxOut-script and write a TxIn-script that, concatenated with the given script, correctly executes. Further details are found below (6 P.)

OP_DROP
OP_DUP
OP_HASH160
PubKeyHash1
OP_EQUALVERIFY
OP_CHECKSIG

The TxOut-script.

Your Tx-In script.

For your convenience, we provide you with a table of sufficient OP_CODES for this exercise. Please do not use other OP_CODES, the given ones are sufficient.

Opcode	Input	Output	Description
OP_DUP	<data>	<data><data>	Duplicates the top item on the stack.
OP_HASH160	<data>	H(<data>)	Removes the top item, hashes it, places it on top of the stack.
<data>	-	<data>	Pushes <data> on top of the stack. *
OP_EQUALVERIFY	<d1><d2>	- or false	Checks if <d1> and <d2> are equal. If equal, remove both of them, otherwise fails.
OP_CHECKSIG	<pubkey> <sig>	false or true	Takes two items of the stack and validates whether the signature was created with the private key corresponding to the pubkey.
OP_DROP	<data>	-	Drops the top item.

If you want to use the <data> op-code, you can access following information.

User	Public key	Hash of Public key	Signature
Alice	PubKey1	PubKeyHash1	Sig1
Bob	PubKey2	PubKeyHash2	Sig2
Carol	PubKey3	PubKeyHash3	Sig3

2.8. Referring to 2.7, can you tell who the recipient of the TxOut-Script is? (1 P.)

2.9. Referring to 2.7, can you tell who created the TxOut-Script? (1 P.)

3. Exercise – Ethereum (21 P)

3.1. Briefly explain two properties of the world-computer concept of Ethereum. (2 P.)

3.2. Briefly explain the concept of Gas. (2 P.)

3.3. What is the difference between STARTGAS and GASPRICE? (1 P.)

3.4. What is the difference between a transaction and a message? (1 P.)

3.5. What could be a reason why the developers of Solidity did not include a function to generate arbitrary random numbers? (1 P.)

3.6. What account type is represented by this tuple (nonce:0x10, balance:0x9383, contract_code: 0x0, storage: 0x0)? What leads you to this assumption? (1 P.)

3.7. Briefly explain how it is possible to make use of external data, e.g. from an arbitrary REST API, in a smart contract? (2 P.)

3.8. The following contract has only one function which adds two 8-bit unsigned integer numbers.

```
contract Math {
   function add(uint8 a, uint8 b) pure returns(uint8 c) {
      return a+b;
   }
}
```

- Assume that another contract uses this *Math* contract. Why would this be a potential security risk? (1 P.)

- What is the meaning of the pure keyword? (1 P.)

- How could you change the **body of the function** to make it secure? The function should still add two 8-bit unsigned integers and the return value should be the correct addition of them. (2 P.)

3.9. You decide to become one of those infamous and highly paid ICO advisors. Therefore, you want to use an Ethereum smart contract in Solidity to collect your salary. Write a smart contract that fulfills the following requirements: (7 P.)

- One payable function to which accounts can send Ether to,
- stores the amount of Ether and its corresponding sender address,
- only accepts payments that are larger than 2 Ether, and
- implements a function to check whether an account made a payment or not.

The following might be helpful:
You can define mappings via: mapping (address => uint) map;
msg.value : The value of the sent message in wei
msg.sender : The account address of the sender
The unit literals ether, wei, finney

4. Exercise – Alternative Distributed Ledger Technologies (12 P.)

4.1. Briefly describe the concept of channels in Hyperledger Fabric. (2 P.)

4.2. Name two different roles that a node/peer in Hyperledger Fabric can have. (2 P.)

4.3. Check whether the following statements are true or false. If they are false, please correct them. (8 P.)

Statement	**Correct?**		**Reason (only if no)**
	yes	no	
IOTA uses a Blockchain as main data structure.			
Smart Contracts in Hyperledger are called Chaincode.			
Hyperledger Fabric: All nodes in one channel must have installed all chain codes for this channel.			
Hyperledger Fabric: Each channel maintains its own ledger.			
Hyperledger Fabric: Each committing peer is always an endorsing peer.			
IOTA uses a mixture between an account and an UTXO scheme.			
A fully confirmed transaction (referenced by all tips) in IOTA can lose its status when a single valid transaction is attached to the tangle.			
The number of transactions in a transaction bundle is independent of the security level for the derived private key.			