1 **Cryptographic primitives**

a What is a cryptographic hash function, and how does this differ from hash functions such as those used in hash table implementations?

b Provide a formal definition of the following cryptographic hash function properties:

i) Pre-image resistant.

ii) Second pre-image resistant.

iii) Collision resistant.

c What does it mean to say that a cryptographic hash function is *puzzle-friendly*, and how does this relate, if at all, to the properties listed in (b)?

d Construct a hash function h that is collision resistant but not pre-image resistant for some range of h. (Hint: what simple starting assumption could be made?)

e Is collision resistance more or less difficult to achieve than pre-image resistance? Why?

f i) For hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^{256}$ it is expected to take Carol's computer 8 minutes to find a hash with at least 12 leading zeros. What is her hashrate?

ii) How long should Carol expect it to take to calculate a hash with 15 leading zeros?

The six parts carry, respectively, 10%, 30%, 10%, 15%, 20%, and 15% of the marks.

2 **Blockchain primitives**

a Consider the statement: Proof-of-work is necessary but not sufficient for a consensus mechanism.

i) In what sense is proof-of-work necessary for a consensus mechanism? What is/are the purpose(s) of using proof-of-work?

ii) In relation to Bitcoin, what else is needed such that in combination with proof-of-work there is a *complete* consensus mechanism?

b If a broadcast is *consistent*, what is ensured? How does this differ from *reliable* broadcast?

c Consider the following version of the Byzantine Generals Problem. The governments of n countries want to execute a Denial-of-Service attack on the networking infrastructure of another nation state. For the DoS attack to work, all n countries must attack at the same time. Any country can stipulate the time that the DoS attack will occur. However, communication between the n countries is not instantaneous, and as a result if more than one country announces the proposed attack time at a similar time, some countries may receive the attack time proposal from one country first while other countries may receive another attack time proposal first.

i) What do consistency and reliability mean in the context of this problem?

ii) How could proof-of-work be used to ensure the countries attack at the same time?

d Define consensus in the context of distributed systems, state the impossibility result of Fischer, Lynch and Patterson (1985), and explain how this relates to achieving consensus in blockchains.

e Related to Bitcoin specifically, how do the block reward and transaction fees incentivise node behaviour?

f i) What is an *orphan block* (i.e. stale block) and why is this name misleading?

ii) What parameters influence the stale block rate? Derive a mathematical expression for the rate involving the parameters you have identified.

The six parts carry, respectively, 15%, 10%, 25%, 15%, 15%, and 20% of the marks.

3 **Smart contracts**

The Ethereum Virtual Machine or EVM is the runtime environment for smart contracts in Ethereum. Solidity is one of the most popular high-level languages for implementing smart contracts.

a i) Please list the three areas where the EVM can store data and briefly describe them.

ii) A Solidity contract can have exactly one unnamed function called a fallback function. What is the usage of this function?

iii) Solidity offers some special variables and functions globally that are mainly used to provide information about the blockchain or are general-use utility functions. For example, block.timestamp presents the current block timestamp as seconds since the UNIX Epoch. Is it reliable to use block.timestamp as a source of randomness and why?

b The following smart contract is a game: the first raider passing through the three gates wins the game and can snatch all the treasures.

```
 1. pragma solidity ^0.5.13;
 2.
 3. contract Treasury {
 4.
 5.   address public raider;
 6.
 7.   constructor() public payable {
 8.       require(msg.value > 0);
 9.   }
10.
11.   modifier gateOne() {
12.     require(msg.sender != tx.origin);
13.     _;
14.   }
15.
16.   modifier gateTwo() {
17.     uint x;
18.     // extcodesize returns the size of
19.     // the code at specified address
20.     assembly { x := extcodesize(caller) }
21.     require(x == 0);
22.     _;
23.   }
```

```
24.
25.   modifier gateThree(bytes8 _gateKey) {
26.     require(
27.       uint64(bytes8(keccak256(
28.           abi.encodePacked(msg.sender)
29.       ))) ^ uint64(_gateKey) == uint64(0) - 1
30.     );
31.     _;
32.   }
33.
34.   function enter(
35.       bytes8 _gateKey
36.   ) public gateOne gateTwo gateThree(_gateKey) {
37.     require(raider == address(0));
38.     raider = tx.origin;
39.   }
40.
41.   function snatch() public {
42.       require(msg.sender == raider);
43.       selfdestruct(msg.sender);
44.   }
45. }
```

i) Explain the differences between msg.sender and tx.origin.

ii) How can you pass through the three gates? Give your solution with explanations. (Hint: you can write a contract to do this in less than 10 lines.)

iii) If you successfully snatch the funds and remove the contract through the function selfdestruct, could anyone else know who (i.e. which address) received the funds? Why/why not?

The two parts carry equal marks.

4 Scaling and Layer-Two Protocols

a In the case of Bitcoin, what are the two primary factors which limit transaction throughput? Explain intuitively how these factors constrain the throughput.

b Transaction fees in Bitcoin reflect an equilibrium between supply and demand in a marketplace where block-space is the commodity.

i) Explain the dynamics of this market.

ii) What trade-off is a user making when they select the transaction fee they are offering?

c Why might we expect interplay between block-size and decentralisation?

d Payment channels offer an alternative approach to scaling blockchains. Such channels depend on a variety of state replacement techniques. Describe at least 3 such state replacement techniques.

e Alice wants to set up a shared account (i.e. a multisig) with Bob. However, Alice wants to ensure that if Bob disappears without providing his signature to spend the multisig output she will be refunded. Therefore she opts for two transactions: a setup transaction and a refund transaction (cf. Figure 1). Draw a protocol sequence diagram describing the steps taken in setting up these two transactions. (Hint: assume Bob first tells Alice how much he would like to contribute to the shared account).

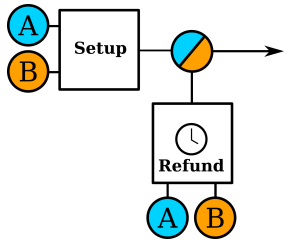

Fig. 1: A multisignature setup.

f Consider Figure 2, which shows an *Invalidation Tree*, an innovation which facilitates duplex (i.e. bi-directional) micropayment channels.

i) Why is this called an invalidation tree? Which state replacement techniques does this construction use, and how?

ii) Why are new branches created in the tree? Explain how the tree structure develops.

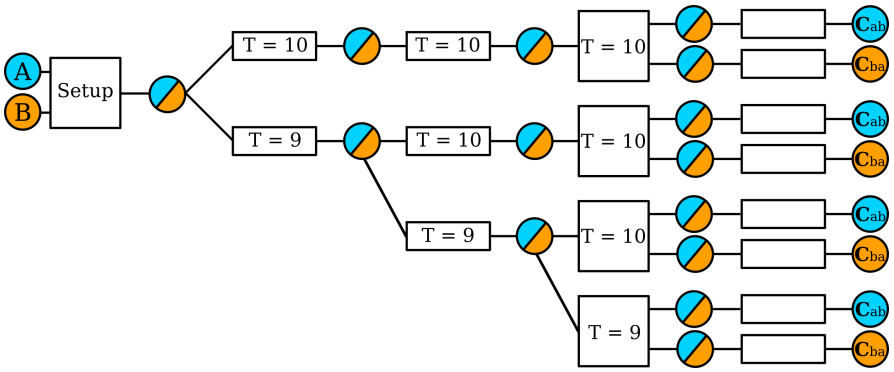

Fig. 2: An invalidation tree. T represents the timelock of a transaction.

The six parts carry, respectively, 10%, 20%, 15%, 20%, 15%, and 20% of the marks.