Ethereum Basics

1. What is gas in the context of Ethereum?

   Every instruction of the EVM has a specific cost assigned, denominated in gas. In general, more complex instructions cost more gas because they consume more computing power of the nodes in the network. A transaction sender sets a price in Ether that they are willing to pay per gas unit and the maximum amount of gas they want to consume. The transaction fee is the product of the used gas units and gas price.

2. What happens with spare gas if too much gas is provided for a transaction?

   If a transaction uses less gas than the sender provided, the excess amount gets refunded to the sender.

3. In contrast to the transaction-based ledger of Bitcoin, Ethereum uses an account-based ledger to maintain the world state. In Bitcoin, double spending attacks are avoided by UTXOs. Once a UTXO is spent (i.e., used as an input), it cannot be spent again, creating a coin ownership chain. How does Ethereum prevent double spending?

   Ethereum, being an account-based ledger, cannot use UTXOs. Instead, each Ethereum account has a property called "nonce" (not to be confused with the nonce, the random value in a block for mining) representing the number of transactions an account has issued. Every transaction includes the nonce of its sender.

   A typical double-spending attempt initiates a second transaction with a higher gas price after the original payment. The double spender hopes the block proposer will prefer including the second transaction as it has a higher gas price. In Ethereum, this is not possible as the second transaction must have a higher nonce and will be rejected by the block proposer. If both transactions have the same nonce, then only one (most likely the second) will be included in the block, and hence no double spending will occur.

   Say the double spender has 10 ETH, a nonce of 5, and makes a 10 ETH payment with nonce 6. If he tries to send 10 ETH to another address he owns using a nonce 7, block proposers will reject this transaction as the account nonce is 5 and transaction nonce is 7 (account nonce and transaction nonce must be consecutive). Suppose the attacker publishes another transaction with nonce 6 sending 10 ETH to his account with a higher gas price. In that case, the latter transaction will probably be included in the block instead of the initial payment. Hence only one of the 10 ETH spending transactions becomes confirmed.

4. In Ethereum, while each transaction is explicitly written to the blockchain, messages are only stored "virtually". Explain why messages are not published to the blockchain and how the blockchain could be in the same state among all nodes without writing the messages.

   The Ethereum Yellow Paper describes transactions as "Transaction: A piece of data, signed by an External Actor." Think of the Ethereum VM as a closed system and wallets interacting with the system and therefore changing the state. Ethereum is based on strict determinism. The state of the VM must be the same across all nodes at a certain point. To reach the current state, all nodes run the Ethereum VM and process mined blocks, which include transactions.
   A node that enters the network must first sync with the current state of the blockchain, i.e., start from the genesis block and go through all blocks until the most recent one. **During these steps, the node will execute all the transactions and the resulting messages from them**. In other words, the node will simulate every interaction of any wallet and all the messages these interactions trigger. Thus, the syncing node will eventually produce and execute every message. Therefore there is no need to write the messages to the blockchain as the information is inherently contained in the transactions.

5. Name four reasons why a transaction sent to the Ethereum network might not get mined(included in a block).

   (a) The transaction has a wrong signature.
   (b) Sending account does not have enough funds.
   (c) Sender set a too-low gas price to be added by a block proposer.
   (d) Transaction consumes more gas than the block gas limit.

6. Name two different ways how a mined transaction can fail.

   (a) The transaction runs out of gas.
   (b) The transaction gets reverted (e.g., by executing an illegal instruction, the REVERT instruction, or a JUMP to an illegal destination).

7. Why does a mined transaction that fails still costs gas?

   To determine that a transaction fails, it has to be executed. At the point of failure, the transaction has already consumed computing power. Therefore, the sender should pay for it. This measure prevents spam. If a failed transaction would not cost any gas, a spammer could send many transactions with complex instructions that fail in the end. The spammer would not be charged, but the transactions would consume a significant amount of the network members' computing power.

8. Find the following transaction *0xdc33d512db0730614ad930907546330ffa73dae318077874e169a814b4f13b40* on `https://etherscan.io/` and answer the questions based on it.
Hint: Use the provided recap slides to inform yourself about EIP-1559 transactions.

   (a) The transaction has paid a 0.001942 ETH transaction fee. Show how this calculation is conducted using the provided data fields.

   In EIP-1559 transactions, the transaction fee equals to

   $$\underbrace{\min(base\_fee + priority\_fee, max\_fee)}_{\text{gas price}} * gas\_consumed$$

   Hence, for this transaction, the paid fee is

   $$\min(30.63 + 0.1, 44.46) * 63,197 \approx 1.942MGwei = 0.001942ETH$$

   (b) The transaction has burnt 0.001935 ETH and saved 0.00086 ETH. Show how these calculations are conducted using the provided data fields.

   In EIP-1559 transactions, the base fee of every transaction gets burnt. Hence, the burnt amount for this transaction is

   $$30.63 * 63,197 \approx 1.935MGwei = 0.001935ETH$$

   The savings of a transaction refers to the user's surplus earned by paying a gas price less than the max fee (i.e., the user was willing to pay up to the max fee but ended up paying less).

   $$Savings = (max\_fee - (base\_fee + priority\_fee)) * gas\_consumed$$

   Hence, this transaction saved

   $$(44.46 - (30.63 + 0.1)) * 63,197 \approx 867kGwei = 0.00086ETH$$

   (c) The transaction is included in block 16947410, which has a base fee of 30.63 Gwei. The following block has a base fee of 30.31 Gwei. Briefly explain what caused the decrease in the base fee.

   The base fee of a block is determined by how full/congested the previous block was relative to the target gas limit of 15M. In this case, block 16947410 was 8% below the target gas limit as it only consumed 13.71M gas. Hence, the network adjusted the base fee by decreasing it to a certain amount($\leq$ 12.5%).

9. An Ethereum block header has a more complex structure than a Bitcoin block header due to the advanced world state Ethereum maintains. Name the three Merkle Patricia Trie (MPT) roots contained in an Ethereum block header (skip storage root) and name one use case for each.

- **State Root**: Root of the MPT, which stores the world state after applying/executing the transactions included in the current block.
  Use case: Check an account balance.
- **Transaction Root**: Root of the MPT, which stores the transactions included in the current block.
  Use case: Check if a specific transaction is included in the block.
- **Receipt Root**: Root of the MPT, which stores the receipts (i.e., outcomes) of transactions included in the current block.
  Use case: Find all instances of the Transfer event emitted by a certain contract in the last ten days.