# 技术实现类 1-5

## 面试问答题（中英文）

## 技术实现类 Technology Realization Category（1-5）

1、代理需要哪种特殊的 CALL 才能工作?

What kind of special CALL is needed for proxies to work?

答：

Solidity中有三种调用函数可以实现合约间的函数调用，它们分别是call、delegatecall和callcode。其中，delegatecall是一种特殊的调用方式，它允许我们在主合约的上下文中加载和调用另一个合约的代码。被调用合约的代码被执行，但被调用合约所做的任何状态改变实际上是在主合约的存储中进行的，而不是在被调用合约的存储中。

There are three call functions in Solidity that enable function calls between contracts, they are call, delegatecall, and callcode. Delegatecall is a special call that allows us to load and call another contract's code in the context of the master contract. The called contract's code is executed, but any state changes made by the called contract are actually made in the master contract's storage, not in the called contract's storage.

2、在 EIP-1559 之前，如何计算以太坊交易的成本?

Before EIP-1559, how was the cost of an Ether transaction calculated?

答：

在EIP-1559之前，以太坊交易的成本由矿工通过拍卖机制来决定。矿工会选择最高出价的交易，并将其包含在下一个区块中。交易的成本由两个因素决定：Gas Price和Gas Limit。Gas Price是以太坊网络中的一种计量单位，用于衡量交易的复杂性。Gas Limit是指交易可以使用的最大Gas数量。交易的成本等于Gas Price乘以Gas Limit。因此，交易的成本取决于Gas Price和Gas Limit的值，这些值由交易的发送者设置。

Prior to EIP-1559, the cost of an Ether transaction was determined by miners through an auction mechanism. Miners would select the highest bidding transaction and include it in the next block. The cost of a transaction is determined by two factors: the Gas Price and the Gas

Limit. Gas Priceis a unit of measurement in the Ether network that measures the complexity of a transaction.The Gas Limit is the maximum amount of Gas that can be used for a transaction. The cost of a transaction is equal to the Gas Price multiplied by the Gas Limit.Therefore, the cost of a transaction depends on the values of Gas Price and Gas Limit, which are set by the sender of the transaction.

3、ETH转账的底层原理是什么?

What is the underlying principle of ETH transfers?

答:

底层原理就是修改链上数据，可以回答详细一点就是交易过程发生了什么。

以下是ETH转账的详细过程:

1.连接到以太坊网络并加载您的私钥。您可以使用go-ethereum客户端来完成此操作。

2.获取发送帐户的随机数（nonce）。每个新事务都需要一个nonce，这是一个仅使用一次的数字。如果是发送交易的新帐户，则该随机数将为“0”。

3.将ETH转换为wei，因为这是以太坊区块链所使用的货币单位。以太网支持最多18个小数位，因此1个ETH为1加18个零。

4.设置您将要转移的ETH数量，以及燃气限额和燃气价格。燃气限额应设上限为“21000”单位，而燃气价格必须以wei为单位设定。

5.需要确定您要将ETH发送给哪个地址。这是接收地址。

6.接下来，您需要生成未签名的以太坊事务。这个函数需要接收nonce，地址，值，燃气上限值，燃气价格和可选发的数据。发送ETH的数据字段为“nil”（nil是go语言的null）。

7.使用发件人的私钥对事务进行签名。为此，您可以使用go-ethereum客户端提供的SignTx方法，该方法接受一个未签名的事务和您之前构造的私钥。

8.在客户端上调用“SendTransaction”将已签名的事务广播到整个网络。这将向网络中的所有节点发送交易，并等待矿工将其打包到区块中。

The underlying principle is the modification of the data on the chain, which can be answered in a little more detail is what happens during the transaction.

Here is the detailed process of ETH transfer:

1. Connect to the ethereum network and load your private key. You can use the go-ethereum client to do this.

2. Get a random number (nonce) of sending accounts. Each new transaction requires a nonce, which is a number that is only used once. If it is a new account sending the transaction, the random number will be "0".

3. Convert ETH to wei, as this is the unit of currency used by the Ethernet blockchain. Ethernet supports up to 18 decimal places, so 1 ETH is 1 plus 18 zeros.

4. Set the amount of ETH you want to transfer, as well as the gas limit and gas price. The gas limit should be capped at "21000" units and the gas price must be set in wei.

5. You need to determine to which address you want to send the ETH. This is the receiving address.

6. Next, you need to generate an unsigned Ether transaction. This function needs to receive the nonce, address, value, gas cap, gas price and optionally send data. The data field for sending ETH is "nil" (nil is the go language for null).

7. Sign the transaction using the sender's private key. To do this, you can use the SignTx method provided by the go-ethereum client, which accepts an unsigned transaction and the private key you constructed earlier.

8. Call "Send Transaction" on the client to broadcast the signed transaction to the entire network. This sends the transaction to all nodes in the network and waits for the miner to package it into a block.

4、在区块链上创建随机数的挑战是什么?

What is the challenge of creating random numbers on a blockchain?

答:

在区块链上创建随机数的挑战是确保生成的随机数是真正随机的。由于区块链是一个公开的分布式账本，因此任何人都可以查看和验证交易。这意味着，如果随机数生成算法不够随机，那么攻击者可能会通过分析交易来预测随机数的值，从而破坏系统的安全性。为了解决这个问题，一些技术已经被提出，例如使用区块链上的随机数生成器，或者使用多个随机数生成器来生成随机数，以确保生成的随机数是真正随机的。

The challenge of creating random numbers on a blockchain is to ensure that the random numbers generated are truly random. Since the blockchain is a public distributed ledger, anyone can view and verify transactions. This means that if the random number generation algorithm is not random enough, then an attacker may be able to predict the value of the random number by analyzing the transactions, thus compromising the security of the system. To address this problem, several techniques have been proposed, such as using a random number generator on the blockchain, or using multiple random number generators to generate random numbers to ensure that the random numbers generated are truly random.

5、如何用solidity实现随机数生成器?

How to implement a random number generator with solidity?

答：

在这个示例中，我们使用了Solidity中的keccak256哈希函数来生成随机数。我们将当前时间戳、发送方地址和nonce值作为输入，然后对它们进行哈希运算，得到一个随机数。nonce值是一个递增的计数器，用于确保每次调用random函数时都会生成一个新的随机数。请注意，这种方法并不是完全随机的，因为它仍然依赖于输入参数的值。但是，它可以提供足够的随机性，以满足大多数应用程序的需求。

In this example, we have used the keccak256 hash function in Solidity to generate random numbers. We take as input the current timestamp, the sender address, and the nonce value, and then hash them to get a random number. The nonce value is an incremental counter that is used to make sure that a new random number is generated each time the random function is called. Note that this method is not completely random, since it still depends on the value of the input argument. However, it can provide enough randomness to meet the needs of most applications.