10挖矿和共识

10.1 介绍

The word "mining" is somewhat misleading. By evoking the extraction of precious metals, it focuses our attention on the reward for mining, the new bitcoin created in each block. Although mining is incentivized by this reward, the primary purpose of mining is not the reward or the generation of new coins. If you view mining only as the process by which coins are created, you are mistaking the means (incentives) as the goal of the process. Mining is the mechanism that underpins the decentralized clearinghouse, by which transactions are validated and cleared. Mining is the invention that makes bitcoin special, a decentralized security mechanism that is the basis for P2P digital cash.
“挖矿”这个词有些误导。
通过想起贵金属的提炼，它把我们的注意力集中于挖矿的奖励，就是每个区块创建的新比特币。
虽然这个奖励激励了挖矿，但挖矿的主要目的不是这个奖励，或创建新比特币。
如果你只是把挖矿视为创建新比特币的过程，则会将挖矿的过程错认为是挖矿的目标。
挖矿是一种机制，它撑起了这去中性化票据交换所的基础，通过它验证和清算交易。
挖矿是比特币与众不同的发明，它是一个去中心化的安全机制，这是P2P数字货币的基础。

Mining *secures the bitcoin system* and enables the emergence of network-wide *consensus without a central authority*. The reward of newly minted coins and transaction fees is an incentive scheme that aligns the actions of miners with the security of the network, while simultaneously implementing the monetary supply.
挖矿保证了比特币系统的安全，在没有中央权威的情况下，实现了网络范围内的共识。
奖励（新比特币和交易费）是一种激励机制，它使矿工的行动都瞄向网络的安全，同时又实现了货币供给。

Tip: The purpose of mining is not the creation of new bitcoin. That’s the incentive system. Mining is the mechanism by which bitcoin’s *security* is *decentralized*.
提示：挖矿的目的不是创建新比特币，创建新比特币只是一种激励机制。
挖矿是一种机制，通过它，以去中心化的方式实现了比特币的安全。

Miners validate new transactions and record them on the global ledger. A new block, containing transactions that occurred since the last block, is "mined" every 10 minutes on average, thereby adding those transactions to the blockchain. Transactions that become part of a block and added to the blockchain are considered "confirmed," which allows the new owners of bitcoin to spend the bitcoin they received in those transactions.
矿工们验证新交易，并记录在区块链中。
每10分钟，会有一个新区块被“挖出”，它包含上个区块以来发生的交易。
这样，这就吧这些交易加入到了区块链中。
包含交易的区块被加入区块链，我们就认为交易“已被确认”，它允许收款人花费这些比特币。

Miners receive two types of rewards in return for the security provided by mining: new coins created with each new block, and transaction fees from all the transactions included in the block. To earn this reward, miners compete to solve a difficult mathematical problem based on a cryptographic hash algorithm. The solution to the problem, called the Proof-of-Work, is included in the new block and acts as proof that the miner expended significant computing effort. The competition to solve the Proof-of-Work algorithm to earn the reward and the right to record transactions on the blockchain is the basis for bitcoin’s security model.

作为对挖矿提供安全的回报，矿工们收到两种类型的奖励：每个新区块创建的新比特币，区块中包含的所有交易的交易费。
为了得到这些奖励，矿工们竞争解决一个数学难题，这个难题是既有一个加密哈希算法。
这个难题的解（称为“工作量证明”）被包含在新区块中，以证明矿工做了大量计算。
这是比特币的安全模型的基础：竞争解决工作量证明算法，以赚钱奖励，并有权将交易记录到区块链上。

The process is called mining because the reward (new coin generation) is designed to simulate diminishing returns, just like mining for precious metals. Bitcoin's money supply is created through mining, similar to how a central bank issues new money by printing bank notes. The maximum amount of newly created bitcoin a miner can add to a block decreases approximately every four years (or precisely every 210,000 blocks). It started at 50 bitcoin per block in January of 2009 and halved to 25 bitcoin per block in November of 2012. It halved again to 12.5 bitcoin in July 2016. Based on this formula, bitcoin mining rewards decrease exponentially until approximately the year 2140, when all bitcoin (20.99999998 million) will have been issued. After 2140, no new bitcoin will be issued.
这个过程称为“挖矿”，因为奖励（新比特币）被设计为模拟收益递减，就像挖掘贵金属一样。
比特币的货币供给是通过挖矿创建的，类似中央银行通过印钞来发新钱。
矿工可以向一个区块中添加的新比特币的最大数量是递减的，大约每四年（210,000个区块）减少一半。

2009-1开始，每个新区块奖励50个比特币；
2012-11，每个新区块奖励25个比特币；
2016-7，每个新区块奖励12.5个比特币。
所以，比特币挖矿奖励以指数方式递减。到2140年，所有比特币（20,999,999,980）全部发行完毕。
即，2140年之后，不会再有新的比特币产生。

Bitcoin miners also earn fees from transactions. Every transaction may include a transaction fee, in the form of a surplus of bitcoin between the transaction's inputs and outputs. The winning bitcoin miner gets to "keep the change" on the transactions included in the winning block. Today, the fees represent 0.5% or less of a bitcoin miner's income, the vast majority coming from the newly minted bitcoin. However, as the reward decreases over time and the number of transactions per block increases, a greater proportion of bitcoin mining earnings will come from fees. Gradually, the mining reward will be dominated by transaction fees, which will form the primary incentive for miners. After 2140, the amount of new bitcoin in each block drops to zero and bitcoin mining will be incentivized only by transaction fees.
矿工们同时也会获取交易费。
每个交易都可能包含一笔交易费，交易费是每个交易记录的输入和输出的差额。
在挖矿过程中，成功“挖出”新区块的矿工可以得到该区块中包含的所有交易费。
目前，交易费占矿工收入的0.5%或更少，大部分收益仍来自挖矿所得的比特币奖励。
然而，随着挖矿奖励的递减，以及每个区块中包含的交易数量增加，交易费在矿工收益中所占的比重会逐渐增加。在2140年之后，矿工的所有收益都只有交易费。

In this chapter, we will first examine mining as a monetary supply mechanism and then look at the most important function of mining: the decentralized consensus mechanism that underpins bitcoin's security.
在本章中，我们先来看看挖矿这种货币供给机制，然后再来了解挖矿的最重要的功能：
去中心化的共识机制，它撑起了比特币的安全的基础。

To understand mining and consensus, we will follow Alice's transaction as it is received and added to a block by Jing's mining equipment. Then we will follow the block as it is mined, added to the blockchain, and accepted by the bitcoin network through the process of emergent consensus.
为了理解“挖矿”和“共识”，我们将跟随Alice的交易，这个交易被Jing的挖矿设备收到，并被加入一个区块中。
然后，我们将跟随这个被挖出的区块，它被加入到区块链中，通过自然发生的共识，被比特币网络接受。

10.1.1 比特币经济学和货币创造

Bitcoin are "minted" during the creation of each block at a fixed and diminishing rate. Each block, generated on average every 10 minutes, contains entirely new bitcoin, created from nothing. Every 210,000 blocks, or approximately every four years, the currency issuance rate is decreased by 50%. For the first four years of operation of the network, each block contained 50 new bitcoin.
创建每个区块的过程中，比特币以固定和递减的速率被挖出。
平均每10分钟产生一个区块，它包含新的比特币。
每隔210,000个区块（大约4年），比特币发行速率被减半。
第一个四年中，每个区块包含50个新比特币。

In November 2012, the new bitcoin issuance rate was decreased to 25 bitcoin per block. In July of 2016 it was decreased again to 12.5 bitcoin per block. It will halve again to 6.25 bitcoin at block 630,000, which will be mined sometime in 2020. The rate of new coins decreases like this exponentially over 32 "halvings" until block 6,720,000 (mined approximately in year 2137), when it reaches the minimum currency unit of 1 satoshi. Finally, after 6.93 million blocks, in approximately 2140, almost 2,099,999,997,690,000 satoshis, or almost 21 million bitcoin, will be issued. Thereafter, blocks will contain no new bitcoin, and miners will be rewarded solely through the transaction fees. Supply of bitcoin currency over time based on a geometrically decreasing issuance rate shows the total bitcoin in circulation over time, as the issuance of currency decreases.
2012-11，比特币的发行速率为每个区块25个比特币。
2016-7，每个区块12.5个比特币。
2020年的某个时候，也就是在区块630,000后，每个区块将降至6.25个比特币。
新比特币的发行速率会进行32次“等分”，直到区块6,720,000（大约在2137年），将达到比特币的最小货币单位1聪。
最终，在经过6,930,000个区块之后，所有比特币将发行完毕，总共 2,099,999,997,690,000聪。
也就是说，到2140年左右，会有大约 2100万个比特币。
在那之后，新的区块不再包含比特币奖励，矿工的收益全部来自交易费。
图1显示了在发行速率不断降低的情况下，比特币总流通量与时间的关系。

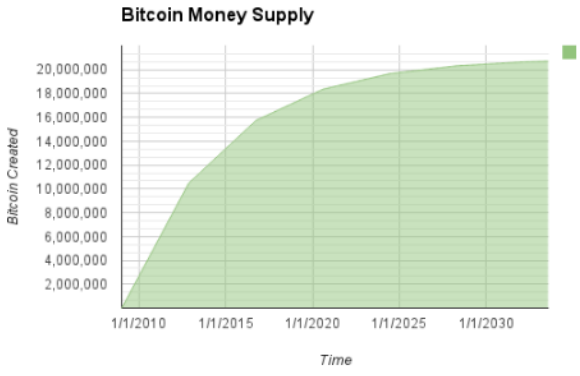

图10-1 比特币发行与时间的关系

Figure 1. Supply of bitcoin currency over time based on a geometrically decreasing issuance rate
图1：比特币随时间的供给

Note：The maximum number of coins mined is the *upper limit* of possible mining rewards for bitcoin. In practice, a miner may intentionally mine a block taking less than the full reward. Such blocks have already been mined and more may be mined in the future, resulting in a lower total issuance of the currency.
注意：挖矿所得的比特币的最大数量是比特币的可能的挖矿奖励的上限。
在实际中，矿工可能故意使挖出的区块奖励少于全额奖励。

这些区块（非全额奖励）已经被挖出，未来可能会有更多区块（非全额奖励）被挖出，就导致了货币发行总量更少。

In the example code in A script for calculating how much total bitcoin will be issued, we calculate the total amount of bitcoin that will be issued.
在例1的代码中，我们计算比特币的发行总量。

Example 1. A script for calculating how much total bitcoin will be issued
例1：计算比特币发行总量的脚本

```
link:code/max_money.py[]
```

Running the max_money.py script shows the output produced by running this script.
例2 显示了上面脚本的输出

Example 2. Running the max_money.py script
例2：运行脚本

```
$ python max_money.py
Total BTC to ever be created: 2099999997690000 Satoshis
```

The finite and diminishing issuance creates a fixed monetary supply that resists inflation. Unlike a fiat currency, which can be printed in infinite numbers by a central bank, bitcoin can never be inflated by printing.
有限和递减地发行导致了抵制通货膨胀的固定货币供应。
中央银行可以印刷无限多的法定货币，而比特币不能通过印刷来导致通货膨胀。

10.1.2通货紧缩

Deflationary Money

The most important and debated consequence of fixed and diminishing monetary issuance is that the currency tends to be inherently *deflationary*. Deflation is the phenomenon of appreciation of value due to a mismatch in supply and demand that drives up the value (and exchange rate) of a currency. The opposite of inflation, price deflation, means that the money has more purchasing power over time.
对于固定和递减的货币发行来说，最重要和最有争议的一个结果是：货币会趋向于通货紧缩。
通货紧缩是由于货币供求失衡导致货币升值（和汇率）的现象。
与通过膨胀相反，价格通货紧缩意味着：随着时间的推移，钱的购买力越来越强。

Many economists argue that a deflationary economy is a disaster that should be avoided at all costs. That is because in a period of rapid deflation, people tend to hoard money instead of spending it, hoping that prices will fall. Such a phenomenon unfolded during Japan's "Lost Decade," when a complete collapse of demand pushed the currency into a deflationary spiral.
许多经济学家认为：通货紧缩对经济是一个灾难，应该尽力避免。
因为在快速通货紧缩时期，人们倾向于囤积货币，而是消费货币，希望会继续下跌。
这种现象在日本“失去的十年”期间展现了出来，当时需求完全塌缩把货币推向通货紧缩螺旋。

Bitcoin experts argue that deflation is not bad per se. Rather, deflation is associated with a collapse in demand because that is the only example of deflation we have to study. In a fiat currency with the possibility of unlimited printing, it is very difficult to enter a deflationary spiral unless there is a complete collapse in demand and an unwillingness to print money. Deflation in bitcoin is not caused by a collapse in demand, but by a predictably constrained supply.
比特币专家认为：通缩本身并不坏。

通货紧缩与需求塌缩起来，是 因为这是我们迄今研究的为一个通货紧缩例子。
对于可能无限印刷的法币中，很难进入通货紧缩螺旋，除非需求完全塌缩，并且不愿印钞。
比特币的通货紧缩不是由需求塌缩引起的，而是由一个可预测的限制供给引起的。

The positive aspect of deflation, of course, is that it is the opposite of inflation. Inflation causes a slow but inevitable debasement of currency, resulting in a form of hidden taxation that punishes savers in order to bail out debtors (including the biggest debtors, governments themselves). Currencies under government control suffer from the moral hazard of easy debt issuance that can later be erased through debasement at the expense of savers.
通货紧缩的积极方面就是通货膨胀的反面。
通货膨胀一起你货币缓慢但不可避免的贬值，造成隐性税收的一种形式，为了拯救债务人而惩罚储蓄者（包括最大的债务人，政府自己）。
政府控制下的货币陷入轻易发行债券的道德危险，这种债券可以通过牺牲储户为代价被抹去。

ddk注：

- 因为货币贬值，对于储蓄者来说，钱的购买力降低了；对于债务人来说，他们还的钱相对借钱时的价值变低了。
- 例如储蓄者卖了一亩地得到1000元，债务人借了1000元买了1一亩地。
  如果货币贬值一半，则储蓄者现在的1000元只能卖半亩地，而债务人只需要卖掉半亩地，就能还清借的1000元。
- 政府控制下的货币与上类似，政府给债权人货币，得到实际的东西，为了偿还债务，就多印货币，导致通货膨胀，这样，债权人的货币贬值，而政府所还的债务比之前减少了。

It remains to be seen whether the deflationary aspect of the currency is a problem when it is not driven by rapid economic retraction, or an advantage because the protection from inflation and debasement far outweighs the risks of deflation.
仍然有待观察：当通货紧缩不是由快速经济收缩引起，货币的通货紧缩方面是否是一个问题；或者，货币的通货紧缩方面是否是一个优点，因为防止通货膨胀和贬值在重要性上远远超过通货紧缩的风险。

ddk说明：（未必准确）
新发行的货币主要通过三条途径流入社会：

- 央行回购国债和央行票据，以及偿付外债（财政紧缺难以周转时才会使用）
  -- 央行完全凭空印了一堆钞票，用于债务，导致市场上的货币增加，但其它都没有变，致使货币贬值，储户受损。
- 成为政府的收入，并经过政府支出进入流通领域（政府财政紧缺时使用）
  -- 这也是凭空印了一堆钞票
- 央行从商业银行手中购进外币，形成央行的外汇储备
  -- 央行用外币作为抵押，向市场注入新货币，这样，法币价值受到汇率和市场的影响。

10.2 去中心化共识

In the previous chapter we looked at the blockchain, the global public ledger (list) of all transactions, which everyone in the bitcoin network accepts as the authoritative record of ownership.
在上一章中，我们了解了区块链，它是所有交易的全局公共账本，
比特币网络中的每个人都把它看作所有权的权威记录。

But how can everyone in the network agree on a single universal "truth" about who owns what, without having to trust anyone? All traditional payment systems depend on a trust model that has a central authority providing a clearinghouse service, basically verifying and clearing all transactions.
但在不必信任任何人的情况下，网络中的每个人如何达成一致：一个全世界的“事实”，即谁拥有什么。
所有的传统支付系统都依赖于一个信任模型，它有一个中央权威，提供票据交换服务，从根本上验证和清算所有交易。

Bitcoin has no central authority, yet somehow every full node has a complete copy of a public ledger that it can trust as the authoritative record. The blockchain is not created by a central authority, but is assembled independently by every node in the network. Somehow, every node in the network, acting on information transmitted across insecure network connections, can arrive at the same conclusion and assemble a copy of the same public ledger as everyone else. This chapter examines the process by which the bitcoin network achieves global consensus without central authority.
比特币没有中央权威，每个全节点都有一份完整的公共账目，节点可以认为它是权威记录。
这个区块链并不是由一个中央权威创建的，而是由比特币网络中的每个节点独立地组装而成。
网络中的每个节点，都对不安全网络连接上传输的信息采取行动，能够达成相同的结论，并组装出相同的公共账本。
本章解释这个过程，比特币网络通过这个过程，在没有中央权威的情况下，实现了全局共识。

Satoshi Nakamoto’s main invention is the decentralized mechanism for *emergent consensus*. Emergent, because consensus is not achieved explicitly—there is no election or fixed moment when consensus occurs. Instead, consensus is an emergent artifact of the asynchronous interaction of thousands of independent nodes, all following simple rules. All the properties of bitcoin, including currency, transactions, payments, and the security model that does not depend on central authority or trust, derive from this invention.
中本聪的主要发明是这种去中心化机制，它实现了自发共识。
“自发”是因为共识不是明确实现的：没有选举，也没有共识发生的固定时刻。
“共识”是数千独立节点异步交互的自发产物，这些节点遵循简单的规则。
比特币的所有属性都是从这个发明导出的，包括：货币、交易、支付、安全模型（它不依赖于中央权威或信任）。

Bitcoin’s decentralized consensus emerges from the interplay of four processes that occur independently on nodes across the network:
- Independent verification of each transaction, by every full node, based on a comprehensive list of criteria
- Independent aggregation of those transactions into new blocks by mining nodes, coupled with demonstrated computation through a Proof-of-Work algorithm
- Independent verification of the new blocks by every node and assembly into a chain
- Independent selection, by every node, of the chain with the most cumulative computation demonstrated through Proof-of-Work

比特币的去中心化共识来自于，节点上独立发生的4个过程的相互作用：
- 根据一个综合标准列表，每个“全节点”独立验证每个交易
- 每个“挖矿节点”独立把交易写入新区块，并通过工作量证明算法表明做了计算
- 每个“节点”独立验证新区块，并组装到一个链上
- 每个“节点”独立选择有最多计算量的那个区块链

In the next few sections we will examine these processes and how they interact to create the emergent property of network-wide consensus that allows any bitcoin node to assemble its own copy of the authoritative, trusted, public, global ledger.
在下面几节中，我们将看看这些过程，了解它们之间如何相互作用来产生自发的网络共识，它允许任何节点可以自己组装这个权威、可信、公开、全局的账本。

10.3 独立验证交易

In [transactions], we saw how wallet software creates transactions by collecting UTXO, providing the appropriate unlocking scripts, and then constructing new outputs assigned to a new owner. The resulting transaction is then sent to the neighboring nodes in the bitcoin network so that it can be propagated across the entire bitcoin network.
钱包软件创建的交易的方法是：收集UTXO、提供合适的解锁脚本、构造新输出给收款人。
然后，把这个交易发给相邻节点，这样就能在整个比特币网络上传播。

However, before forwarding transactions to its neighbors, every bitcoin node that receives a transaction will first verify the transaction. This ensures that only valid transactions are propagated across the network, while invalid transactions are discarded at the first node that encounters them.
但是，在邻居转发这个交易之前，它首先会验证这个交易。
这保证了，只有有效的交易才会在网络上传播，而无效的交易在第一个节点就会被丢弃。

Each node verifies every transaction against a long checklist of criteria:
每个节点在验证每个交易时，是根据一个长长的标准列表：

1. The transaction's syntax and data structure must be correct.
   交易的语法和数据结构必须正确

2. Neither lists of inputs or outputs are empty.
   输入列表或输出列表都不是空

3. The transaction size in bytes is less than MAX_BLOCK_SIZE.
   交易的字节数小于 MAX_BLOCK_SIZE

4. Each output value, as well as the total, must be within the allowed range of values (less than 21m coins, more than the dust threshold).
   每一个输出值，以及总值，必须在允许的范围内（大于粉尘阈值，小于2100万个比特币）。

5. None of the inputs have hash=0, N=–1 (coinbase transactions should not be relayed).
   输入不能有：hash=0，N=-1（不应传播币基交易）
   ddk问题：这是什么意思?

6. nLocktime is equal to INT_MAX, or nLocktime and nSequence values are satisfied according to MedianTimePast.
   nLockTime = INT_MAX，或nLocktime和nSequence的值满足MedianTimePast
   注：MedianTime是这个区块的前面11个区块按照时间戳排序后的中位时间

7. The transaction size in bytes is greater than or equal to 100.
   交易的字节数 >= 100

8. The number of signature operations (SIGOPS) contained in the transaction is less than the signature operation limit.
   交易中的签名操作(SIGOPS)的数量小于签名操作数量上限。

9. The unlocking script (scriptSig) can only push numbers on the stack, and the locking script (scriptPubkey) must match IsStandard forms (this rejects "nonstandard" transactions).
   解锁脚本（scriptSig ）只能够向堆栈压入数字，
   锁定脚本（scriptPubkey）必须要符合isStandard的格式（拒绝非标准交易）。

10. A matching transaction in the pool, or in a block in the main branch, must exist.

"池中"或"主分支的一个区块中"必须存在一个匹配的交易
注：也就是能够花费的交易

11. For each input, if the referenced output exists in any other transaction in the pool, the transaction must be rejected.
对于每个输入，如果引用的输出存在于池中的其它交易中，则必须拒绝这个交易。
注：不能双重花费

12. For each input, look in the main branch and the transaction pool to find the referenced output transaction. If the output transaction is missing for any input, this will be an orphan transaction. Add to the orphan transactions pool, if a matching transaction is not already in the pool.
对于每个输入，在主分支和交易池中寻找引用的输出交易。
如果输出交易缺少任何一个输入，该交易将成为一个孤儿交易。
如果与其匹配的交易还没有出现在池中，则加入到孤儿交易池中。

13. For each input, if the referenced output transaction is a coinbase output, it must have at least COINBASE_MATURITY (100) confirmations.
对于每个输入，如果引用的输出交易是一个币基输出，
则它必须必须至少获得 COINBASE_MATURITY(100)个确认。

14. For each input, the referenced output must exist and cannot already be spent.
对于每个输入，引用的输出必须存在，并且不能已被花费。

15. Using the referenced output transactions to get input values, check that each input value, as well as the sum, are in the allowed range of values (less than 21m coins, more than 0).
使用引用的输出交易获得输入值，检查每个输入值和总值是否在允许的范围内
（大于0，小于2100万个比特币）。

16. Reject if the sum of input values is less than sum of output values.
如果输入值的总和小于输出值的总和，则拒绝这个交易。

17. Reject if transaction fee would be too low (minRelayTxFee) to get into an empty block.
如果交易费太低(minRelayTxFee)以至于无法进入一个空区块，则拒绝该交易。

18. The unlocking scripts for each input must validate against the corresponding output locking scripts.
每个输入的解锁脚本必须依据相应输出的锁定脚本来验证。

These conditions can be seen in detail in the functions AcceptToMemoryPool, CheckTransaction, and CheckInputs in Bitcoin Core. Note that the conditions change over time, to address new types of denial-of-service attacks or sometimes to relax the rules so as to include more types of transactions.
Bitcoin Core的下列函数有这些条件的细节：AcceptToMemoryPool、CheckTransaction、CheckInputs。
注意，这些条件可能会变，为了解决新型DoS攻击，或有时放松规则以包含更多类型的交易。

By independently verifying each transaction as it is received and before propagating it, every node builds a pool of valid (but unconfirmed) transactions known as the *transaction pool*, *memory pool*, or *mempool*.
每个节点在在收到交易和传播之前独立地进行验证，就建立了一个有效交易池（还未被确认），称为"交易池"或"内存池"。

10.4 挖矿节点

Some of the nodes on the bitcoin network are specialized nodes called *miners*.
In [ch01_intro_what_is_bitcoin] we introduced Jing, a computer engineering student in Shanghai, China, who is a bitcoin miner. Jing earns bitcoin by running a "mining rig," which is a specialized computer-hardware system designed to mine bitcoin. Jing's specialized mining hardware is connected to a server running a full bitcoin node. Unlike Jing, some miners mine without a full node, as we will see in Mining Pools. Like every other full node, Jing's node receives and propagates unconfirmed transactions on the bitcoin network. Jing's node, however, also aggregates these transactions into new blocks.
在比特币网络中，有些特殊节点被称为“矿工”。
第1章中，我们介绍了Jing，他就是一位矿工。
Jing通过运行一个矿机来获得比特币，矿机是专门设计用于挖比特币的计算机硬件系统。
Jing的这台专业挖矿设备连接着一个运行比特币全节点的服务器。
与Jing不同，有些矿工是在没有全节点的条件下进行挖矿，我们将在“矿池”中介绍。
与其它它全节点相同，Jing的节点也接收和传播未确认的交易。
但是，Jing节点还能把这些交易聚合到一个新区块中。

Jing's node is listening for new blocks, propagated on the bitcoin network, as do all nodes. However, the arrival of a new block has special significance for a mining node. The competition among miners effectively ends with the propagation of a new block that acts as an announcement of a winner. To miners, receiving a valid new block means someone else won the competition and they lost. However, the end of one round of a competition is also the beginning of the next round. The new block is not just a checkered flag, marking the end of the race; it is also the starting pistol in the race for the next block.
同其它节点一样，Jing节点时刻监听着传播到比特币网络的新区块。
这些新区块对挖矿节点有着特殊的意义。
矿工之间的竞争以新区块的传播而结束，如同宣布谁是最后的赢家。
对于矿工们来说，收到一个有效的新区块意味着有人赢了，而自己输了。
但是，一轮竞争的结束也代表下一轮竞争的开始。新区块不仅象征着本轮竞赛结束，它也是下一个区块竞赛的发令枪。

10.5 把交易聚合到区块中

After validating transactions, a bitcoin node will add them to the *memory pool*, or *transaction pool*, where transactions await until they can be included (mined) into a block. Jing's node collects, validates, and relays new transactions just like any other node. Unlike other nodes, however, Jing's node will then aggregate these transactions into a *candidate block*.
验证交易后，比特币节点将它加入到自己的"内存池"（交易池），这里的交易等待包含在一个区块中。
与其它节点一样，Jing节点收集、验证、传递新的交易。
不同的是，Jing节点会把这些交易聚合到一个候选区块中。

Let's follow the blocks that were created during the time Alice bought a cup of coffee from Bob's Cafe (see [cup_of_coffee]). Alice's transaction was included in block 277,316. For the purpose of demonstrating the concepts in this chapter, let's assume that block was mined by Jing's mining system and follows Alice's transaction as it becomes part of this new block.
我们再来看看Alice的交易，这个交易被放入了区块 277,316中。
为了说明本章中的概念，假设这个区块是由Jing节点挖出的。

Jing's mining node maintains a local copy of the blockchain. By the time Alice buys the cup of coffee, Jing's node has assembled a chain up to block 277,314. Jing's node is listening for transactions, trying to mine a new block and also listening for blocks discovered by other nodes. As Jing's node is mining, it receives block 277,315 through the bitcoin network. The arrival of this block signifies the end of the competition for block 277,315 and the beginning of the competition to create block 277,316.
Jing的挖矿节点在本地维护了一个区块链。
当Alice买咖啡时，Jing节点的区块链已经收集到了区块277,314，并继续监听着网络上的交易，在尝试挖掘新区块的同时，也监听着由其它节点发现的区块。
当Jing节点在挖矿时，它从比特币网络收到了区块277,315。这个标志着区块277,315的竞赛已经结束，与同时开始区块277,316的竞赛。

During the previous 10 minutes, while Jing's node was searching for a solution to block 277,315, it was also collecting transactions in preparation for the next block. By now it has collected a few hundred transactions in the memory pool. Upon receiving block 277,315 and validating it, Jing's node will also compare it against all the transactions in the memory pool and remove any that were included in block 277,315. Whatever transactions remain in the memory pool are unconfirmed and are waiting to be recorded in a new block.
在上个10分钟内，当Jing节点正在寻找区块277,315的解时，它也在收集交易为下一个区块做准备。
目前，它已经收到了几百个交易，将它们放进了内存池。
在收到和验证区块277,315后，Jing节点会检查内存池中的全部交易，移除已经在区块277,315中出现过的交易，确保留在内存池中的交易都是未被确认的，它们等待被记录到新区块中。

Jing's node immediately constructs a new empty block, a candidate for block 277,316. This block is called a *candidate block* because it is not yet a valid block, as it does not contain a valid Proof-of-Work. The block becomes valid only if the miner succeeds in finding a solution to the Proof-of-Work algorithm.
Jing节点立刻构建一个新的空区块，做为区块277,316的候选区块。
称作"候选区块"是因为它还没有包含有效的工作量证明，不是一个有效的区块，而只有在矿工成功找到一个工作量证明的解之后，这个区块才生效。

When Jing's node aggregates all the transactions from the memory pool, the new candidate block has 418 transactions with total transaction fees of 0.09094928 bitcoin. You can see this block in the blockchain using the Bitcoin Core client command-line interface, as shown in Using the command line to retrieve block 277,316.
现在，Jing节点聚合了内存池中所有交易，新的候选区块有418个交易，总的交易费为0.09094925个比特币。
你可以使用Bitcoin Core客户端命令行，来查看这个区块，如例3所示。

Example 3. Using the command line to retrieve block 277,316
例3：使用命令行获取区块277,316

```
$ bitcoin-cli getblockhash 277316
000000000000001b6b9a13b095e96db41c4a928b97ef2d944a9b31b2cc7bdc4

$ bitcoin-cli getblock
000000000000001b6b9a13b095e96db41c4a928b97ef2d944a9b31b2cc7bdc4
{
    "hash"          :
"000000000000001b6b9a13b095e96db41c4a928b97ef2d944a9b31b2cc7bdc4",
    "confirmations"          : 35561,
    "size"          : 218629,
    "height"        : 277316,
    "version"       : 2,
    "merkleroot"    :
"c91c008c26e50763e9f548bb8b2fc323735f73577effbc55502c51eb4cc7cf2e",
    "tx" : [
        "d5ada064c6417ca25c4308bd158c34b77e1c0eca2a73cda16c737e7424afba2f",
        "b268b45c59b39d7596147577l8b9918caf0ba9d97c56f3b91956ff877c503fbe",
        ... 417 more transactions ...
       ],
    "time"          : 1388185914,
    "nonce"         : 924591752,
    "bits"          : "1903a30c",
    "difficulty"    : 1180923195.25802612,
    "chainwork"     :
"0000000000000000000000000000000000000000000000934695e92aaf53afa1a",
    "previousblockhash" :
"0000000000000002a7bbd25a417c0374cc55261021e8a9ca74442b01284f0569"
}
```

10.5.1 币基交易

The first transaction in any block is a special transaction, called a *coinbase transaction*. This transaction is constructed by Jing's node and contains his *reward* for the mining effort.
任何区块中的第一个交易都是一个特殊交易，称为币基交易。
这个交易是Jing节点构造的，包含他的挖矿奖励。

Note：When block 277,316 was mined, the reward was 25 bitcoin per block. Since then, one "halving" period has elapsed. The block reward changed to 12.5 bitcoin in July 2016. It will be halved again in 210,000 blocks, in the year 2020.
注意：当区块277,316被挖出时，每个区块的奖励是25个比特币。此后，已经过了一个“减半”期。2016-7的奖励为12.5个比特币，2020年到达区块210000，区块奖励将再次减半。

Jing's node creates the coinbase transaction as a payment to his own wallet: "Pay Jing's address 25.09094928 bitcoin." The total amount of reward that Jing collects for mining a block is the sum of the coinbase reward (25 new bitcoin) and the transaction fees (0.09094928) from all the transactions included in the block as shown in Coinbase transaction.
Jing节点把笔记交易创建为向他的钱包做一个支付：向Jing的地址支付25.09094928个比特币。
Jing收集到的挖矿奖励的全部金额是：

- 币基奖励（25个新比特币）
- 区块中所有交易的交易费（0.09094928）

Example 4. Coinbase transaction
例4：币基交易

```
$ bitcoin-cli getrawtransaction
d5ada064c6417ca25c4308bd158c34b77e1c0eca2a73cda16c737e7424afba2f 1
```

```
{
    "hex" :
"010000000100000000000000000000000000000000000000000000000000000000000000000000ffffffff0f03443b0
403858402062f503253482fffffffff0110c08d95000000000232102aa970c592640d19de03ff6f329d6fd2eecb02
3263b9ba5d1b81c29b523da8b21ac00000000",
    "txid"                    :
"d5ada064c6417ca25c4308bd158c34b77e1c0eca2a73cda16c737e7424afba2f",
    "version"        : 1,
    "locktime"       : 0,
    "vin" : [
        {
            "coinbase" : "03443b0403858402062f503253482f",
            "sequence" : 4294967295
        }
    ],
    "vout" : [
        {
            "value"  : 25.09094928,
            "n"      : 0,
            "scriptPubKey" : {
                "asm"          :
"02aa970c592640d19de03ff6f329d6fd2eecb023263b9ba5d1b81c29b523da8b21OP_CHECKSIG",
                "hex"          :
"2102aa970c592640d19de03ff6f329d6fd2eecb023263b9ba5d1b81c29b523da8b21ac",
                "reqSigs" : 1,
                "type"   : "pubkey",
                "addresses" : [
                    "1MxTkeEP2PmHSMze5tUZ1hAV3YTKu2Gh1N"
                ]
            }
        }
    ]
}
```


Unlike regular transactions, the coinbase transaction does not consume (spend) UTXO as inputs. Instead, it has only one input, called the *coinbase*, which creates bitcoin from nothing. The coinbase transaction has one output, payable to the miner's own bitcoin address. The output of the coinbase transaction sends the value of 25.09094928 bitcoin to the miner's bitcoin address; in this case it is 1MxTkeEP2PmHSMze5tUZ1hAV3YTKu2Gh1N.
与普通交易不同，币基交易没有输入，不消耗UTXO。
它只包含一个被称作coinbase的输入，它用来创建新比特币。
创币交易有一个输出，可以支付到矿工自己的比特币地址。
创币交易的输出把25.09094928个比特币发送给矿工的比特币地址：
1MxTkeEP2PmHSMze5tUZ1hAV3YTKu2Gh1N。

10.5.2 币基奖励与交易费

To construct the coinbase transaction, Jing's node first calculates the total amount of transaction fees by adding all the inputs and outputs of the 418 transactions that were added to the block. The fees are calculated as:
为了构造这个币基交易，Jing节点首先计算交易费总额：418个交易的输入总额和输出总额。
交易费就是：

```
Total Fees = Sum(Inputs) - Sum(Outputs)
```

In block 277,316, the total transaction fees are 0.09094928 bitcoin.
在区块277,316中，交易费的总额是0.09094925个比特币。

Next, Jing's node calculates the correct reward for the new block. The reward is calculated based on the block height, starting at 50 bitcoin per block and reduced by half every 210,000 blocks. Because this block is at height 277,316, the correct reward is 25 bitcoin.

然后，Jing节点计算出这个新区块的正确奖励。
计算方法是：根据区块高度，以每个区块50个比特币奖励为开始，每产生210,000个区块，则奖励减半一次。这个区块的高度是277,316，所以正确的奖励额是25个比特币。

The calculation can be seen in function GetBlockSubsidy in the Bitcoin Core client, as shown in Calculating the block reward—Function GetBlockSubsidy, Bitcoin Core Client, main.cpp.
Bitcoin Core客户端的GetBlockValue函数中可以看到这个计算，如例5所示。

Example 5. Calculating the block reward—Function GetBlockSubsidy, Bitcoin Core Client, main.cpp
例5：计算区块奖励，Bitcoin Core Client, main.cpp 函数GetBlockValue

```
CAmount GetBlockSubsidy( int nHeight, const Consensus::Params& consensusParams )
{
    int halvings = nHeight / consensusParams.nSubsidyHalvingInterval;

    // Force block reward to zero when right shift is undefined.
    if (halvings >= 64)
        return 0;

    CAmount nSubsidy = 50 * COIN;
    // Subsidy is cut in half every 210,000 blocks which will occur approximately every 4
years.
   nSubsidy >>= halvings;

    return nSubsidy;
}
```

The initial subsidy is calculated in satoshis by multiplying 50 with the COIN constant (100,000,000 satoshis). This sets the initial reward (nSubsidy) at 5 billion satoshis.
初始奖励是50个比特币，用变量nSubsidy记录。

Next, the function calculates the number of halvings that have occurred by dividing the current block height by the halving interval (SubsidyHalvingInterval). In the case of block 277,316, with a halving interval every 210,000 blocks, the result is 1 halving.
然后，这个函数计算当前区块高度的减半次数（halvings）。
每 210,000个区块减半一次，对应本例中的区块277316，所以减半次数为1。

The maximum number of halvings allowed is 64, so the code imposes a zero reward (returns only the fees) if the 64 halvings is exceeded.
变量 halvings 最大值64，如果超出这个值，区块奖励额为0。

Next, the function uses the binary-right-shift operator to divide the reward (nSubsidy) by two for each round of halving. In the case of block 277,316, this would binary-right-shift the reward of 5 billion satoshis once (one halving) and result in 2.5 billion satoshis, or 25 bitcoins. The binary-right-shift operator is used because it is more efficient than multiple repeated divisions. To avoid a potential bug, the shift operation is skipped after 63 halvings, and the subsidy is set to 0.
然后，使用二进制移位操作计算奖励金额。

Finally, the coinbase reward (nSubsidy) is added to the transaction fees (nFees), and the sum is returned.
最后，将币基奖励（nSubsidy ）和交易费（nFee）相加，并返回这个值。

Tip: If Jing's mining node writes the coinbase transaction, what stops Jing from "rewarding" himself 100 or 1000 bitcoin? The answer is that an incorrect reward would result in the block being deemed invalid by everyone else, wasting Jing's electricity used for Proof-of-Work. Jing only gets to spend the reward if the block is accepted by everyone.
提示：如果Jing节点写入了币基交易，那么如何防止Jing奖励自己100或1000个比特币呢?
答案是，不正确的奖励将被其他人视为无效，这浪费了Jing用于工作量证明的投入。
只有这个区块被大家接受，Jing才能花费这个奖励。

10.5.3币基交易的结构

With these calculations, Jing's node then constructs the coinbase transaction to pay himself 25.09094928 bitcoin.
经过计算，Jing节点构造了一个币基交易，支付给自己25.09094928个比特币。

As you can see in Coinbase transaction, the coinbase transaction has a special format. Instead of a transaction input specifying a previous UTXO to spend, it has a "coinbase" input. We examined transaction inputs in [tx_in_structure]. Let's compare a regular transaction input with a coinbase transaction input. The structure of a "normal" transaction input shows the structure of a regular transaction, while The structure of a coinbase transaction input shows the structure of the coinbase transaction's input.
在例4中，币基交易的结构比较特殊，它包含一个“coinbase”输入。
在前面章节中，我们已经介绍了交易的输入。
现在比较一下普通交易输入与币基交易输入。
表1给出了普通交易输入的结构，表2给出了币基交易输入的结构。

Table 1. The structure of a "normal" transaction input

Size	Field	Description
32 bytes	Transaction Hash	Pointer to the transaction containing the UTXO to be spent
4 bytes	Output Index	The index number of the UTXO to be spent, first one is 0
1–9 bytes (VarInt)	Unlocking-Script Size	Unlocking-Script length in bytes, to follow
Variable	Unlocking-Script	A script that fulfills the conditions of the UTXO locking script
4 bytes	Sequence Number	Currently disabled Tx-replacement feature, set to 0xFFFFFFFF

Table 2. The structure of a coinbase transaction input

Size	Field	Description
32 bytes	Transaction Hash	All bits are zero: Not a transaction hash reference
4 bytes	Output Index	All bits are ones: 0xFFFFFFFF
1–9 bytes (VarInt)	Coinbase Data Size	Length of the coinbase data, from 2 to 100 bytes
Variable	Coinbase Data	Arbitrary data used for extra nonce and mining tags. In v2 blocks; must begin with block height
4 bytes	Sequence Number	Set to 0xFFFFFFFF

In a coinbase transaction, the first two fields are set to values that do not represent a UTXO reference. Instead of a "transaction hash," the first field is filled with 32 bytes all set to zero. The "output index" is filled with 4 bytes all set to 0xFF (255 decimal). The "Unlocking Script" (scriptSig) is replaced by coinbase data, a data field used by the miners, as we will see next.
在币基交易中，前两个字段被设置的值表示这不是一个UTXO引用：
“交易哈希”字段填写全0，“输出索引”字段填写全1。
“解锁脚本（scriptSig）”由“币基数据”代替，矿工可以使用这个数据字段。

10.5.4 币基数据

Coinbase transactions do not have an unlocking script (aka, scriptSig) field. Instead, this field is replaced by coinbase data, which must be between 2 and 100 bytes. Except for the first few bytes, the rest of the coinbase data can be used by miners in any way they want; it is arbitrary data.
创币交易不包含“解锁脚本”字段，它被“币基数据”替代，长度为2-100字节。
除了开始的几个字节外，剩余部分可以被矿工写入任意数据。

In the genesis block, for example, Satoshi Nakamoto added the text "The Times 03/Jan/2009 Chancellor on brink of second bailout for banks" in the coinbase data, using it as a proof of the date and to convey a message. Currently, miners use the coinbase data to include extra nonce values and strings identifying the mining pool.
例如，在创世区块中，中本聪在币基数据中写入了这句话：
`The Times 03/Jan/ 2009 Chancellor on brink of second bailout for banks`
(泰晤士报 2009-1-3，财政大臣将再次对银行施以援手)
用它表示日期证明，并且传递一个信息。
目前，矿工使用币基数据来包含额外的nonce值和标识矿池的字符串。

The first few bytes of the coinbase used to be arbitrary, but that is no longer the case. As per BIP-34, version-2 blocks (blocks with the version field set to 2) must contain the block height index as a script "push" operation in the beginning of the coinbase field.
币基数据的前几个字节以前可以任意使用，但现在不行了。
根据BIP-34，版本2区块（区块的vesion字段设置为2）必须在币基字段的开始包含区块高度索引，作为一个交易的压入操作。

In block 277,316 we see that the coinbase (see Coinbase transaction), which is in the unlocking script or scriptSig field of the transaction input, contains the hexadecimal value 03443b0403858402062f503253482f. Let’s decode this value.
在区块277,316中，币基数据是：`03443b0403858402062f503253482f`。
我们来解锁这个值。

The first byte, 03, instructs the script execution engine to push the next three bytes onto the script stack (see [tx_script_ops_table_pushdata]). The next three bytes, 0x443b04, are the block height encoded in little-endian format (backward, least-significant byte first). Reverse the order of the bytes and the result is 0x043b44, which is 277,316 in decimal.
第一个字节是03，它告诉脚本执行引擎把后面3个字节压入这个脚本栈中。
后面3个字节是0x443b04，它们是用小端格式编码的区块高度。
翻转字节序得到0x043b44，表示为十进制是277,316。

```
03 443b04 0385840206 2f503253482f
```

The next few hexadecimal digits (0385840206) are used to encode an extra *nonce* (see The Extra Nonce Solution), or random value, used to find a suitable Proof-of-Work solution.
后面的十六进制数（03858402062）用于编码一个额外的nonce（随机值），用于找到一个工作量证明的解。

The final part of the coinbase data (2f503253482f) is the ASCII-encoded string `/P2SH/`, which indicates that the mining node that mined this block supports the P2SH improvement defined in BIP-16.
币基数据的结尾部分(2f503253482f)是ASCII编码的字符“/P2SH/”，表示挖矿节点支持BIP16定义的P2SH改进方案。

The introduction of the P2SH capability required signaling by miners to endorse either BIP-16 or BIP-17. Those endorsing the BIP-16 implementation were to include /P2SH/ in their coinbase data. Those endorsing the BIP-17 implementation of P2SH were to include

the string p2sh/CHV in their coinbase data. The BIP-16 was elected as the winner, and many miners continued including the string /P2SH/ in their coinbase to indicate support for this feature.
在P2SH功能引入到比特币时，曾经有过一场对P2SH不同实现方式的投票，候选者是BIP16和BIP17。支持BIP16的矿工将“/P2SH/”放入币基数据中，支持BIP17的矿工将“p2sh/CHV”放入币基数据中。最后，BIP16在投票中胜出，直到现在，依然有很多矿工在他们的币基数据中填入“/P2SH/”，以表示支持这个功能。

Extract the coinbase data from the genesis block uses the libbitcoin library introduced in [alt_libraries] to extract the coinbase data from the genesis block, displaying Satoshi's message. Note that the libbitcoin library contains a static copy of the genesis block, so the example code can retrieve the genesis block directly from the library.
例6使用了libbitcoin库，从创世区块中提取币基数据，显示出中本聪留下的信息。libbitcoin库中自带了一份创世区块的静态拷贝，所以示例代码可以直接取自库中的创世区块数据。

Example 6. Extract the coinbase data from the genesis block
例6：从创世区块中提取币基数据

```
link:code/satoshi-words.cpp[]
```

We compile the code with the GNU C++ compiler and run the resulting executable, as shown in Compiling and running the satoshi-words example code.
例7中，我们使用GNU C++编译器编译源代码，并运行可执行程序，

Example 7. Compiling and running the satoshi-words example code
例7：编译和运行示例代码

```
$ # Compile the code
$  g++ -o satoshi-words satoshi-words.cpp $(pkg-config --cflags --libs libbitcoin)

$ # Run the executable
$ ./satoshi-words
^D��<GS>^A^DEThe Times 03/Jan/2009 Chancellor on brink of second bailout for banks
```

10.6 构造区块头

To construct the block header, the mining node needs to fill in six fields, as listed in The structure of the block header.
为了构造区块头，挖矿节点需要填写六个字段，如表3所示。

Table 3. The structure of the block header

Size	Field	Description
4 bytes	Version	A version number to track software/protocol upgrades
32 bytes	Previous Block Hash	A reference to the hash of the previous (parent) block in the chain
32 bytes	Merkle Root	A hash of the root of the merkle tree of this block's transactions
4 bytes	Timestamp	The approximate creation time of this block (seconds from Unix Epoch)
4 bytes	Target	The Proof-of-Work algorithm target for this block
4 bytes	Nonce	A counter used for the Proof-of-Work algorithm

At the time that block 277,316 was mined, the version number describing the block structure is version 2, which is encoded in little-endian format in 4 bytes as 0x02000000.
在区块277,316被挖出的时候，描述区块结构udev版本号是版本2，长度为4字节，小端格式编码为0x20000000。

Next, the mining node needs to add the "Previous Block Hash" (also known as prevhash). That is the hash of the block header of block 277,315, the previous block received from the network, which Jing's node has accepted and selected as the *parent* of the candidate block 277,316. The block header hash for block 277,315 is:
接着，挖矿节点需要填写“父区块哈希”。
在本例中，这个值是Jing节点从网络上接收到的区块277,315 的区块头哈希。
区块277,315的区块头哈希为：

```
000000000000002a7bbd25a417c0374cc55261021e8a9ca74442b01284f0569
```

Tip: By selecting the specific *parent* block, indicated by the Previous Block Hash field in the candidate block header, Jing is committing his mining power to extending the chain that ends in that specific block. In essence, this is how Jing "votes" with his mining power for the longest-difficulty valid chain.
提示：通过选择特定的父区块，Jing将自己的挖矿能力致力于扩展指定区块结尾的链。
从本质上说，这是用他的挖矿能力为最难有效链进行“投票”。

The next step is to summarize all the transactions with a merkle tree, in order to add the merkle root to the block header. The coinbase transaction is listed as the first transaction in the block. Then, 418 more transactions are added after it, for a total of 419 transactions in the block. As we saw in the [merkle_trees], there must be an even number of "leaf" nodes in the tree, so the last transaction is duplicated, creating 420 nodes, each containing the hash of one transaction. The transaction hashes are then combined, in pairs, creating each level of the tree, until all the transactions are summarized into one node at the "root" of the tree. The root of the merkle tree summarizes all the transactions into a single 32-byte value, which you can see listed as "merkle root" in Using the command line to retrieve block 277,316, and here:
下一步是用merkle树归纳所有的交易，目的是将merkle跟填入区块头。
币基交易是这个区块中的第一个交易，后面有 418个其它交易，因此，这个区块共有419个交易。
我们已经讲过Merkle树，叶节点必须是偶数个，所以需要复制最后一个交易作为第420个叶节点，
每个叶节点是是对应交易的哈希。这些交易的哈希逐层、成对地组合，直到最终合并成一个根节点。
merkle数的根节点将全部交易数据归纳为一个32字节的值。

在例3中，能看可以merkel根的值如下：

```
c91c008c26e50763e9f548bb8b2fc323735f73577effbc55502c51eb4cc7cf2e
```

Jing's mining node will then add a 4-byte timestamp, encoded as a Unix "epoch" timestamp, which is based on the number of seconds elapsed from January 1, 1970, midnight UTC/GMT. The time 1388185914 is equal to Friday, 27 Dec 2013, 23:11:54 UTC/GMT.
Jing节点然后添加4字节的时间戳，它是Unix时间。
本例中，1388185914对应的时间是2013-12-27 23:11:54 UTC/GMT。

Jing's node then fills in the target, which defines the required Proof-of-Work to make this a valid block. The target is stored in the block as a "target bits" metric, which is a mantissa-exponent encoding of the target. The encoding has a 1-byte exponent, followed by a 3-byte mantissa (coefficient). In block 277,316, for example, the target bits value is 0x1903a30c. The first part 0x19 is a hexadecimal exponent, while the next part, 0x03a30c, is the coefficient. The concept of a target is explained in Retargeting to Adjust Difficulty and the "target bits" representation is explained in Target Representation.
然后，Jing节点填充Target字段（难度目标值），为了使这个区块有效，Target字段定义了需要满足的工作量证明的难度。
难度在区块中以"尾数-指数"的格式编码和存储，这种格式称作target bits（难度位）。
这种编码的首字节表示指数，后面的3字节表示尾数(系数)。
以区块277316为例，难度位的值为0x1903a30c，0x19是指数，0x03a30c是系数。
这部分的概念在后面有详细的解释。

The final field is the nonce, which is initialized to zero.
最后一个字段是nonce，初始值为0。

With all the other fields filled, the block header is now complete and the process of mining can begin. The goal is now to find a value for the nonce that results in a block header hash that is less than the target. The mining node will need to test billions or trillions of nonce values before a nonce is found that satisfies the requirement.
所有其它字段填写后，这个区块头就完成了，可以开始挖矿过程了。
挖矿的目标是找到一个nonce值，使区块头的哈希小于target。
挖矿节点需要测试非常多的nonce取值，才能找到一个满足条件的nonce值。

10.7 挖掘区块

Now that a candidate block has been constructed by Jing's node, it is time for Jing's hardware mining rig to "mine" the block, to find a solution to the Proof-of-Work algorithm that makes the block valid. Throughout this book we have studied cryptographic hash functions as used in various aspects of the bitcoin system. The hash function SHA256 is the function used in bitcoin's mining process.
既然Jing节点已经构建了一个候选区块，现在Jing的硬件矿机就开始挖掘这个区块，找到工作量证明算法的一个解，使这个区块有效。
从本书中，我们已经学习了加密哈希函数用在比特币系统的各个方面。
比特币挖矿过程使用的是 SHA256哈希函数。

In the simplest terms, mining is the process of hashing the block header repeatedly, changing one parameter, until the resulting hash matches a specific target. The hash function's result cannot be determined in advance, nor can a pattern be created that will produce a specific hash value. This feature of hash functions means that the only way to produce a hash result matching a specific target is to try again and again, randomly modifying the input until the desired hash result appears by chance.
简单地说，挖矿就是不断计算区块头的哈希的过程，修改一个参数，计算哈希，在修改参数，再计算哈希，直到哈希与一个特定target相匹配。

无法预先确定这个哈希函数的结果，也没有模式能产生一个特定的哈希。
哈希函数的这个特性意味着：得到匹配一个特定target的哈希的唯一方法是不断地尝试，每次随机修改输入，直到出现合适的哈希。

10.7.1 工作量证明算法

A hash algorithm takes an arbitrary-length data input and produces a fixed-length deterministic result, a digital fingerprint of the input. For any specific input, the resulting hash will always be the same and can be easily calculated and verified by anyone implementing the same hash algorithm.
哈希算法的输入是一个任意长度的数据，输出一个固定长度、确定性的值，它是输入的一个数字指纹。
对于任意特定的输入，生成的哈希都一样，很容易计算，任何人都可以用相同的哈希函数进行验证。

The key characteristic of a cryptographic hash algorithm is that it is computationally infeasible to find two different inputs that produce the same fingerprint (known as a *collision*). As a corollary, it is also virtually impossible to select an input in such a way as to produce a desired fingerprint, other than trying random inputs.
加密哈希函数的主要特征是：不同的输入几乎不可能出现相同的数字指纹。
因此，这也是几乎不可能的：除了尝试随机的输入，不可能有意选择一个输入，得到一个希望的指纹。

With SHA256, the output is always 256 bits long, regardless of the size of the input. In SHA256 example, we will use the Python interpreter to calculate the SHA256 hash of the phrase, "I am Satoshi Nakamoto."
无论输入的大小，SHA256的输出都是256比特。
在例8中，我们将使用Python解释器来计算这个语句的SHA256哈希：I am Satoshi Nakamoto

Example 8. SHA256 example
例8：SHA256示例

```
$ python
Python 2.7.1
>>> import hashlib
>>> print hashlib.sha256("I am Satoshi Nakamoto").hexdigest()
5d7c7ba21cbbcd75d14800b100252d5b428e5b1213d27c385bc141ca6b47989e
```

SHA256 example shows the result of calculating the hash of "I am Satoshi Nakamoto": 5d7c7ba21cbbcd75d14800b100252d5b428e5b1213d27c385bc141ca6b47989e. This 256-bit number is the *hash* or *digest* of the phrase and depends on every part of the phrase. Adding a single letter, punctuation mark, or any other character will produce a different hash.
在例8中， 输出的是"I am Satoshi Nakamoto"的哈希。
改变原句中的任何一个字母、标点、或增加字母，都会产生不同的哈希。

Now, if we change the phrase, we should expect to see completely different hashes. Let's try that by adding a number to the end of our phrase, using the simple Python scripting in SHA256 script for generating many hashes by iterating on a nonce.
如果我们改变这个句子，得到的应该是完全不同的哈希。
例如，我们在句子末尾加上一个数字，运行例9中的Python脚本。

Example 9. SHA256 script for generating many hashes by iterating on a nonce
例9：通过反复修改 nonce 来生成许多哈希的SHA256脚本

```
link:code/hash_example.py[]
```

Running this will produce the hashes of several phrases, made different by adding a number at the end of the text. By incrementing the number, we can get different hashes, as shown in SHA256 output of a script for generating many hashes by iterating on a nonce.

执行这个脚本就能生成这些只是末尾数字不同的句子的哈希。
例10 中，我们只是改变一个数字，却得到了完全不同的哈希。

Example 10. SHA256 output of a script for generating many hashes by iterating on a nonce
例10：通过修改nonce 来生成不同哈希

```
$ python hash_example.py
I am Satoshi Nakamoto0 => a80a81401765c8eddee25df36728d732...
I am Satoshi Nakamoto1 => f7bc9a6304a4647bb41241a677b5345f...
I am Satoshi Nakamoto2 => ea758a8134b115298a1583ffb80ae629...
I am Satoshi Nakamoto3 => bfa9779618ff072c903d773de30c99bd...
I am Satoshi Nakamoto4 => bce8564de9a83c18c31944a66bde992f...
I am Satoshi Nakamoto5 => eb362c3cf3479be0a97a20163589038e...
I am Satoshi Nakamoto6 => 4a2fd48e3be420d0d28e202360cfbaba...
I am Satoshi Nakamoto7 => 790b5a1349a5f2b909bf74d0d166b17a...
I am Satoshi Nakamoto8 => 702c45e5b15aa54b625d68dd947f1597...
I am Satoshi Nakamoto9 => 7007cf7dd40f5e933cd89fff5b791ff0...
I am Satoshi Nakamoto10 => c2f38c81992f4614206a21537bd634a...
I am Satoshi Nakamoto11 => 7045da6ed8a914690f087690e1e8d66...
I am Satoshi Nakamoto12 => 60f01db30c1a0d4cbce2b4b22e88b9b...
I am Satoshi Nakamoto13 => 0ebc56d59a34f5082aaef3d66b37a66...
I am Satoshi Nakamoto14 => 27ead1ca85da66981fd9da01a8c6816...
I am Satoshi Nakamoto15 => 394809fb809c5f83ce97ab554a2812c...
I am Satoshi Nakamoto16 => 8fa4992219df33f50834465d3047429...
I am Satoshi Nakamoto17 => dca9b8b4f8d8e1521fa4eaa46f4f0cd...
I am Satoshi Nakamoto18 => 9989a401b2a3a318b01e9ca9a22b0f3...
I am Satoshi Nakamoto19 => cda56022ecb5b67b2bc93a2d764e75f...
```

Each phrase produces a completely different hash result. They seem completely random, but you can reproduce the exact results in this example on any computer with Python and see the same exact hashes.
每个句子都生成了一个完全不同的哈希。
它们看起来是完全随机的，但你在任何计算机上用Python执行上面的脚本，都能重现这些相同的哈希。

The number used as a variable in such a scenario is called a *nonce*. The nonce is used to vary the output of a cryptographic function, in this case to vary the SHA256 fingerprint of the phrase.
类似这样在句子末尾的变化的数字称为nonce（随机数）。
Nonce是用来改变加密函数输出的，在这个示例中，改变了这个句子的 SHA256指纹。

To make a challenge out of this algorithm, let's set a target: find a phrase that produces a hexadecimal hash that starts with a zero. Fortunately, this isn't difficult! SHA256 output of a script for generating many hashes by iterating on a nonce shows that the phrase "I am Satoshi Nakamoto13" produces the hash 0ebc56d59a34f5082aaef3d66b37a661696c2b618e62432727216ba9531041a5, which fits our criteria. It took 13 attempts to find it. In terms of probabilities, if the output of the hash function is evenly distributed we would expect to find a result with a 0 as the hexadecimal prefix once every 16 hashes (one out of 16 hexadecimal digits 0 through F). In numerical terms, that means finding a hash value that is less than 0x1000000000000000000000000000000000000000000000000000000000000000. We call this threshold the *target* and the goal is to find a hash that is numerically less than the target. If we decrease the target, the task of finding a hash that is less than the target becomes more and more difficult.
为了使这个算法更有挑战，我们设定一个目标：找到一个句子，使哈希以 0开头。
这很容易！在例10中，"I am Satoshi Nakamoto13" 的哈希满足这个条件。我们得到它用了13次计算。
用概率角度看，如果哈希函数的输出是平均分布，我们可以期望每16次就能得到一个以0开头的哈希。
从数字角度看，我们要找的是小于0x1000000000000000000000000000000000000000000000000000000000000000 的哈希。

我们称它为Target，我们的目的是找到小于这个Target的一个哈希。
如果我们减小这个Target，那找到一个小于它的哈希会越来越难。

To give a simple analogy, imagine a game where players throw a pair of dice repeatedly, trying to throw less than a specified target. In the first round, the target is 12. Unless you throw double-six, you win. In the next round the target is 11. Players must throw 10 or less to win, again an easy task. Let's say a few rounds later the target is down to 5. Now, more than half the dice throws will exceed the target and therefore be invalid. It takes exponentially more dice throws to win, the lower the target gets. Eventually, when the target is 2 (the minimum possible), only one throw out of every 36, or 2% of them, will produce a winning result.
打个比方，人们不断扔一对骰子，目标是得到的点数小于一个特定数字。
第一局目标是12，只要你不扔出两个6，就能赢。
第二局目标是11，你只能扔10或更小的点数才能赢，不过也很简单。
加入目标降为5，现在有50%的概率，扔出来的骰子加起来点数会超过5。
随着目标越来越小，要想赢的话，扔骰子的次数会指数级上升。
当目标为2时（最小可能点数），你平均要扔36次，才能赢。

From the perspective of an observer who knows that the target of the dice game is 2, if someone has succeeded in casting a winning throw it can be assumed that they attempted, on average, 36 throws. In other words, one can estimate the amount of work it takes to succeed from the difficulty imposed by the target. When the algorithm is a based on a deterministic function such as SHA256, the input itself constitutes *proof* that a certain amount of *work* was done to produce a result below the target. Hence, *Proof-of-Work*.
从观察者的角度来看，如果有人要成功中奖，可以认为他平均尝试了36次。
换句话说，从实现目标难度可以估计取得成功所需的工作量。
当算法是基于一个确定性函数（例如SHA256）时，这个输入本身就是证据，必须要有一定的工作量才能得到低于目标的结果。因此，称之为工作量证明。

Tip: Even though each attempt produces a random outcome, the probability of any possible outcome can be calculated in advance. Therefore, an outcome of specified difficulty constitutes proof of a specific amount of work.
提示：尽管每次尝试产生一个随机的结果，但是任何可能的结果的概率可以预先计算。
因此，指定特定难度的结果构成了一个特定工作量的证明。

In SHA256 output of a script for generating many hashes by iterating on a nonce, the winning "nonce" is 13 and this result can be confirmed by anyone independently. Anyone can add the number 13 as a suffix to the phrase "I am Satoshi Nakamoto" and compute the hash, verifying that it is less than the target. The successful result is also Proof-of-Work, because it proves we did the work to find that nonce. While it only takes one hash computation to verify, it took us 13 hash computations to find a nonce that worked. If we had a lower target (higher difficulty) it would take many more hash computations to find a suitable nonce, but only one hash computation for anyone to verify. Furthermore, by knowing the target, anyone can estimate the difficulty using statistics and therefore know how much work was needed to find such a nonce.
在例10中，成功的nonce为13，且这个结果能被所有人独立验证。
任何人可以吧13加到句子 "I am Satoshi Nakamoto" 后面，计算它的哈希，就能确认它比目标值小。
这个正确的结果同时也是工作量证明，因为它证明了：我们确花时间找到了这个nonce。
验证这个哈希只需要计算一次，而找到它却计算了13次。
如果目标更小（难度更大），那需要更多的哈希计算才能找到合适的nonce，但验证时只需要一次计算。
此外，知道目标后，任何人都可以用统计学来估算其难度，因此就能知道找到这个nonce需要多少工作。

Tip: The Proof-of-Work must produce a hash that is *less than* the target. A higher target means it is less difficult to find a hash that is below the target. A lower target means it is more difficult to find a hash below the target. The target and difficulty are inversely related.
提示：工作量证明必须产生小于目标的哈希。

较高的目标意味着寻找不会太难。较低的目标意味着寻找更加困难。目标和难度是成反比。

Bitcoin's Proof-of-Work is very similar to the challenge shown in SHA256 output of a script for generating many hashes by iterating on a nonce. The miner constructs a candidate block filled with transactions. Next, the miner calculates the hash of this block's header and sees if it is smaller than the current *target*. If the hash is not less than the target, the miner will modify the nonce (usually just incrementing it by one) and try again. At the current difficulty in the bitcoin network, miners have to try quadrillions of times before finding a nonce that results in a low enough block header hash.
比特币的工作量证明和例10中的挑战非常类似。
矿工用一些交易构建一个候选区块。接下来，这个矿工计算这个区块头的哈希，看其是否小于当前目标。
如果这个哈希不小于目标，矿工就修改nonce（通常是加1），然后再试一次。
按当前比特币系统的难度，矿工得试10^{15}次才能找到一个合适的nonce，使区块头哈希足够小。

A very simplified Proof-of-Work algorithm is implemented in Python in Simplified Proof-of-Work implementation.
例11是一个简化的工作量证明算法，使用Python实现的。

Example 11. Simplified Proof-of-Work implementation
例11：简化的工作量证明实现

```
link:code/proof-of-work-example.py[]
```

Running this code, you can set the desired difficulty (in bits, how many of the leading bits must be zero) and see how long it takes for your computer to find a solution. In Running the Proof-of-Work example for various difficulties, you can see how it works on an average laptop.
运行这个代码，你可以设置希望的难度（指定开头有多少个比特为0）。
执行这个代码，看看在你的计算机上求解需要多久。
在例12中，你可以看到这个程序在一个普通笔记本电脑上的执行情况。

Example 12. Running the Proof-of-Work example for various difficulties
例12：用各种难度来运行工作量证明

```
$ python proof-of-work-example.py*
Difficulty: 1 (0 bits)
[...]

Difficulty: 8 (3 bits)
Starting search...
Success with nonce 9
Hash is 1c1c105e65b47142f028a8f93ddf3dabb9260491bc64474738133ce5256cb3c1
Elapsed Time: 0.0004 seconds
Hashing Power: 25065 hashes per second

Difficulty: 16 (4 bits)
Starting search...
Success with nonce 25
Hash is 0f7becfd3bcd1a82e06663c97176add89e7cae0268de46f94e7e11bc3863e148
Elapsed Time: 0.0005 seconds
Hashing Power: 52507 hashes per second

Difficulty: 32 (5 bits)
Starting search...
Success with nonce 36
Hash is 029ae6e5004302a120630adcbb808452346ab1cf0b94c5189ba8bac1d47e7903
Elapsed Time: 0.0006 seconds
Hashing Power: 58164 hashes per second

[...]
```

```
Difficulty: 4194304 (22 bits)
Starting search...
Success with nonce 1759164
Hash is 0000008bb8f0e731f0496b8e530da984e85fb3cd2bd81882fe8ba3610b6cefc3
Elapsed Time: 13.3201 seconds
Hashing Power: 132068 hashes per second

Difficulty: 8388608 (23 bits)
Starting search...
Success with nonce 14214729
Hash is 000001408cf12dbd20fcba6372a223e098d58786c6ff93488a9f74f5df4df0a3
Elapsed Time: 110.1507 seconds
Hashing Power: 129048 hashes per second

Difficulty: 16777216 (24 bits)
Starting search...
Success with nonce 24586379
Hash is 0000002c3d6b370fccd699708d1b7cb4a94388595171366b944d68b2acce8b95
Elapsed Time: 195.2991 seconds
Hashing Power: 125890 hashes per second

[...]

Difficulty: 67108864 (26 bits)
Starting search...
Success with nonce 84561291
Hash is 0000001f0ea21e676b6dde5ad429b9d131a9f2b000802ab2f169cbca22b1e21a
Elapsed Time: 665.0949 seconds
Hashing Power: 127141 hashes per second
```

As you can see, increasing the difficulty by 1 bit causes a doubling in the time it takes to find a solution. If you think of the entire 256-bit number space, each time you constrain one more bit to zero, you decrease the search space by half. In Running the Proof-of-Work example for various difficulties, it takes 84 million hash attempts to find a nonce that produces a hash with 26 leading bits as zero. Even at a speed of more than 120,000 hashes per second, it still requires 10 minutes on a laptop to find this solution.
可以看出，随着难度的增加，查找正确结果的时间会呈指数级增长。
如果你考虑整个256比特数字空间，每次要求多一个0，就是把哈希查找空间缩减了一半。
在例12中，难度为哈希开头的26位为0时，共尝试了8千多万次。
即使家用笔记本每秒可以计算270,000次哈希，这个查找依然需要10分钟。

At the time of writing, the network is attempting to find a block whose header hash is less than:
在写本书时，比特币网络要寻找的区块头哈希要小于

```
0000000000000000029AB9000000000000000000000000000000000000000000
```

As you can see, there are a lot of zeros at the beginning of that target, meaning that the acceptable range of hashes is much smaller, hence it's more difficult to find a valid hash. It will take on average more than 1.8 zeta-hashes (thousand billion billion hashes) per second for the network to discover the next block. That seems like an impossible task, but fortunately the network is bringing 3 exa-hashes per second (EH/sec) of processing power to bear, which will be able to find a block in about 10 minutes on average.
可以看出，这个目标哈希开头的0多了很多。
这意味着，可接受的哈希范围大幅缩减，因而找到正确的哈希更加困难。
生成下一个区块需要网络每秒计算1.8 zeta次哈希。
这看起来像是不可能完成的任务，但幸运的是，比特币网络已经拥有每秒3 exa次哈希的处理能力，平均每10分钟就可以找到一个新区块。

10.7.2 Target表示

In Using the command line to retrieve block 277,316, we saw that the block contains the target, in a notation called "target bits" or just "bits," which in block 277,316 has the value of 0x1903a30c. This notation expresses the Proof-of-Work target as a coefficient/exponent format, with the first two hexadecimal digits for the exponent and the next six hex digits as the coefficient. In this block, therefore, the exponent is 0x19 and the coefficient is 0x03a30c.
在例3中，我们看到区块277,316中包含的target：0x1903a30c。
这个值表示为“系数/指数”格式，第1个字节是指数（exponent），后面3字节是系数（coefficient）。在这个区块里，0x19为指数，0x03a30c为系数。

The formula to calculate the difficulty target from this representation is:
计算难度目标的公式是：

- target = coefficient * $2^{(8*(exponent-3))}$

Using that formula, and the difficulty bits value 0x1903a30c, we get:
对于0x1903a30c，可得：

- target = 0x03a30c * $2^{0x08*(0x19-0x03)}$
- => target = 0x03a30c * $2^{(0x08*0x16)}$
- => target = 0x03a30c * 2^{0xB0}

which in decimal is:
十进制表示是：

- => target = 238,348 * 2^{176}
- => target = 22,829,202,948,393,929,850,749,706,076,701,368,331,072,452,018,388,575,715,328

switching back to hexadecimal:
十六进制表示是：

- => target = 0x0000000000000003A30C0000000000000000000000000000000000000000

This means that a valid block for height 277,316 is one that has a block header hash that is less than the target. In binary that number must have more than 60 leading bits set to zero. With this level of difficulty, a single miner processing 1 trillion hashes per second (1 terahash per second or 1 TH/sec) would only find a solution once every 8,496 blocks or once every 59 days, on average.
也就是说，区块277,316的区块头哈希要是小于这个目标。
这个数字的二进制表示中前60位都是0。
在这个难度上，一个每秒可以计算1万亿次哈希的矿工，平均每8496个区块才能找到一个正确结果，
换句话说，平均每59天，才能为某一个区块找到正确的哈希。

10.7.3调整目标难度

Retargeting to Adjust Difficulty

As we saw, the target determines the difficulty and therefore affects how long it takes to find a solution to the Proof-of-Work algorithm. This leads to the obvious questions: Why is the difficulty adjustable, who adjusts it, and how?
如前所述，目标决定了难度，进而影响了求解工作量证明算法所需要的时间。
那么问题来了：为什么这个难度值是可调整的？由谁来调整？如何调整？

Bitcoin's blocks are generated every 10 minutes, on average. This is bitcoin's heartbeat and underpins the frequency of currency issuance and the speed of transaction settlement. It has to remain constant not just over the short term, but over a period of many decades. Over this time, it is expected that computer power will continue to increase at a rapid pace. Furthermore, the number of participants in mining and the computers they use will also constantly change. To keep the block generation time at 10 minutes, the difficulty of mining must be adjusted to account for these changes. In fact, the Proof-of-Work target is a dynamic parameter that is periodically adjusted to meet a 10-minute block interval goal. In simple terms, the target is set so that the current mining power will result in a 10-minute block interval.
比特币的区块平均每10分钟生成一个。
这是比特币的心跳，是货币发行速率和交易结算速度的基础。
不仅是在短期内，而是在几十年内，它都必须要保持恒定。
在此期间，计算机性能会飞速提升。此外，参与挖矿的人和计算机也会不断变化。
为了能保持10分钟生成一个新区块，挖矿的难度必须根据这些变化进行调整。
事实上，难度是一个动态的参数，会定期调整，以达到每10分钟一个新区块的目标。
简单地说，难度被设定为：无论挖矿能力如何，新区块产生速率都保持在10分钟一个。

How, then, is such an adjustment made in a completely decentralized network? Retargeting occurs automatically and on every node independently. Every 2,016 blocks, all nodes retarget the Proof-of-Work. The equation for retargeting measures the time it took to find the last 2,016 blocks and compares that to the expected time of 20,160 minutes (2,016 blocks times the desired 10-minute block interval). The ratio between the actual timespan and desired timespan is calculated and a proportionate adjustment (up or down) is made to the target. In simple terms: If the network is finding blocks faster than every 10 minutes, the difficulty increases (target decreases). If block discovery is slower than expected, the difficulty decreases (target increases).
那么，在一个完全去中心化的网络中，这样的调整是如何做到的呢?
难度的调整是在每个节点独立自发进行的。
每2016个区块，所有节点都会调整难度。
难度的调整公式是：由最近2016个区块的花费时长，与20160分钟比较得出的。
难度是根据实际时长与期望时长的比值进行相应调整的（变难或变易）。
简单来说，如果发现区块产生速率比10分钟短，就增加难度。如果发现比10分钟长，就降低难度。

The equation can be summarized as:
这个公式是:

```
New Target = Old Target * (Actual Time of Last 2016 Blocks / 20160 minutes)
```

新目标 = 旧目标 * （前2016个区块的实际时间 / 20160分钟）
新目标 * 20160分钟 = 旧目标 * 前2016个区块的实际时间

Retargeting the Proof-of-Work—CalculateNextWorkRequired() in pow.cpp shows the code used in the Bitcoin Core client.
例13展示了Bitcoin Core客户端中使用的代码。

Example 13. Retargeting the Proof-of-Work—CalculateNextWorkRequired() in pow.cpp
例13：调整难度，pow.cpp，CalculateNextWorkRequired()

```
    // Limit adjustment step
    int64_t nActualTimespan = pindexLast->GetBlockTime() - nFirstBlockTime;
    LogPrintf("  nActualTimespan = %d  before bounds\n", nActualTimespan);
    if (nActualTimespan < params.nPowTargetTimespan/4)
        nActualTimespan = params.nPowTargetTimespan/4;
    if (nActualTimespan > params.nPowTargetTimespan*4)
        nActualTimespan = params.nPowTargetTimespan*4;

    // Retarget
    const arith_uint256 bnPowLimit = UintToArith256(params.powLimit);
    arith_uint256 bnNew;
    arith_uint256 bnOld;
```

```
    bnNew.SetCompact(pindexLast->nBits);
    bnOld = bnNew;
    bnNew *= nActualTimespan;
    bnNew /= params.nPowTargetTimespan;

    if (bnNew > bnPowLimit)
        bnNew = bnPowLimit;
```

Note：While the target calibration happens every 2,016 blocks, because of an off-by-one error in the original Bitcoin Core client it is based on the total time of the previous 2,015 blocks (not 2,016 as it should be), resulting in a retargeting bias toward higher difficulty by 0.05%.
注意：虽然目标校准使用2016个区块的时间，但是由于Bitcoin Core客户端的一个错误，它是基于之前的2015个区块的时间，导致难度提高了0.05%。

The parameters Interval (2,016 blocks) and TargetTimespan (two weeks as 1,209,600 seconds) are defined in *chainparams.cpp*.
参数Interval(2016个区块)和TargetTimespan(1,209,600秒，即两周)的定义在文件chainparams.cpp中。

To avoid extreme volatility in the difficulty, the retargeting adjustment must be less than a factor of four (4) per cycle. If the required target adjustment is greater than a factor of four, it will be adjusted by a factor of 4 and not more. Any further adjustment will be accomplished in the next retargeting period because the imbalance will persist through the next 2,016 blocks. Therefore, large discrepancies between hashing power and difficulty might take several 2,016 block cycles to balance out.
为了防止难度的变化过快，每个周期的调整幅度必须小于一个因子（值为4）。
如果要调整的幅度大于4倍，则按4倍调整。
由于在下一轮2016个区块的周期不平衡的情况会继续存在，所以进一步的难度调整会在下一周期进行。
因此，平衡哈希计算能力和难度的巨大差异有可能需要花费几个2016区块周期才会完成。

Tip：The difficulty of mining a bitcoin block is approximately '10 minutes of processing' for the entire network, based on the time it took to mine the previous 2,016 blocks, adjusted every 2,016 blocks. This is achieved by lowering or raising the target.
提示：寻找一个比特币区块需要整个网络花费10分钟来处理，每发现2016个区块时，会根据前2016个区块完成的时间对难度进行调整。

Note that the target is independent of the number of transactions or the value of transactions. This means that the amount of hashing power and therefore electricity expended to secure bitcoin is also entirely independent of the number of transactions. Bitcoin can scale up, achieve broader adoption, and remain secure without any increase in hashing power from today's level. The increase in hashing power represents market forces as new miners enter the market to compete for the reward. As long as enough hashing power is under the control of miners acting honestly in pursuit of the reward, it is enough to prevent "takeover" attacks and, therefore, it is enough to secure bitcoin.
注意，目标难度与交易的数量和金额无关。即，哈希算力的强弱和所耗电力，与交易数量完全无关。
换句话说，当比特币的规模变得更大，使用它的人数更多时，即使哈希算力保持当前的水平，比特币的安全性也不会受到影响。
哈希算力的增加表明，更多的人为得到比特币回报而加入了挖矿队伍。
只要为了回报，公平正当地从事挖矿的矿工群体保持足够的哈希算力，就能防止“强行控制”攻击，就能保证比特币足够安全。

The difficulty of mining is closely related to the cost of electricity and the exchange rate of bitcoin vis-a-vis the currency used to pay for electricity. High-performance mining systems are about as efficient as possible with the current generation of silicon fabrication, converting electricity into hashing computation at the highest rate possible. The primary influence on the mining market is the price of one kilowatt-hour of electricity in bitcoin, because that determines the profitability of mining and therefore the incentives to enter or exit the mining market.
挖矿难度与电费和比特币汇率密切相关，因为人们要用货币支付电费。

高性能挖矿系统就是要用当前硅芯片以最高效的方式将电力转化为哈希算力。
对挖矿市场的主要影响是每度电的比特币价格。
因为这决定着挖矿的营利，因此也刺激着人们选择进入或退出挖矿市场。

10.8 成功地挖出区块

As we saw earlier, Jing's node has constructed a candidate block and prepared it for mining. Jing has several hardware mining rigs with application-specific integrated circuits, where hundreds of thousands of integrated circuits run the SHA256 algorithm in parallel at incredible speeds. Many of these specialized machines are connected to his mining node over USB or a local area network. Next, the mining node running on Jing's desktop transmits the block header to his mining hardware, which starts testing trillions of nonces per second. Because the nonce is only 32 bits, after exhausting all the nonce possibilities (about 4 billion), the mining hardware changes the block header (adjusting the coinbase extra nonce space or timestamp) and resets the nonce counter, testing new combinations.
前面已经看到，Jing节点创建了一个候选区块，准备拿它来挖矿。
Jing有几个ASIC矿机，可以超高速并行运行SHA256算法。
这些定制的硬件通过USB连接到他的挖矿节点上。
接下来，运行在Jing的桌面电脑上的挖矿节点将区块头传送给这些硬件，让它们以每秒亿万次的速度进行nonce测试。

Almost 11 minutes after starting to mine block 277,316, one of the hardware mining machines finds a solution and sends it back to the mining node.
在对区块277,316的挖矿工作开始大概11分钟后，其中一个ASIC矿机找到了一个解，发回给挖矿节点。

When inserted into the block header, the nonce 924,591,752 produces a block hash of:
当把nonce 4,215,469,401放入区块头时，就会产生一个哈希：

```
0000000000000001b6b9a13b095e96db41c4a928b97ef2d944a9b31b2cc7bdc4
```

which is less than the target:
它小于难度目标：

```
00000000000000003A30C00000000000000000000000000000000000000000000
```

Immediately, Jing's mining node transmits the block to all its peers. They receive, validate, and then propagate the new block. As the block ripples out across the network, each node adds it to its own copy of the blockchain, extending it to a new height of 277,316 blocks. As mining nodes receive and validate the block, they abandon their efforts to find a block at the same height and immediately start computing the next block in the chain, using Jing's block as the "parent." By building on top of Jing's newly discovered block, the other miners are essentially "voting" with their mining power and endorsing Jing's block and the chain it extends.
Jing的挖矿节点立刻将这个区块发给它的所有相邻节点。
这些节点在接收并验证这个新区块后，会继续传播这个区块。
当这个新区块在网络中扩散时，每个节点都会将它作为区块277,316加到自己的区块链中。
当挖矿节点收到并验证了这个新区块后，它们会放弃之前构建区块277,316的计算，立即开始下一个区块的计算工作。

In the next section, we'll look at the process each node uses to validate a block and select the longest chain, creating the consensus that forms the decentralized blockchain.
在下一节中，介绍每个节点用于验证区块和选择最长链的过程，从而达成共识，这个共识形成了去中心化的区块链。

10.9验证新区块

The third step in bitcoin's consensus mechanism is independent validation of each new block by every node on the network. As the newly solved block moves across the network, each node performs a series of tests to validate it before propagating it to its peers. This ensures that only valid blocks are propagated on the network. The independent validation also ensures that miners who act honestly get their blocks incorporated in the blockchain, thus earning the reward. Those miners who act dishonestly have their blocks rejected and not only lose the reward, but also waste the effort expended to find a Proof-of-Work solution, thus incurring the cost of electricity without compensation.
比特币共识机制的第三步是，网络中的每个节点独立验证每个新区块。
当新区块在网络中传播时，每个节点在转发它之前，会进行一系列的测试去验证它。这确保了只有会传播有效的区块。
独立验证还确保了：

- 诚实的矿工生成的区块可以被纳入到区块链中，从而获得奖励。
- 不诚实的矿工所生成的区块将被拒绝，这不但使他们失去奖励，而且也浪费了努力，因而导致没有补偿的电力成本。

When a node receives a new block, it will validate the block by checking it against a long list of criteria that must all be met; otherwise, the block is rejected. These criteria can be seen in the Bitcoin Core client in the functions CheckBlock and CheckBlockHeader and include:
当节点收到一个新区块时，它将对照一个长长的标准清单对该区块进行验证，若没有通过验证，这个区块将被拒绝。
这些标准可以在`Bitcoin Core`客户端的`CheckBlock`和`CheckBlockHead`函数中获得，包括：

- The block data structure is syntactically valid
  区块数据结构的语法是有效的
- The block header hash is less than the target (enforces the Proof-of-Work)
  区块头哈希小于目标难度（确认包含足够的工作量证明）
- The block timestamp is less than two hours in the future (allowing for time errors)
  区块时间戳在未来两个小时以内（允许时间误差）
- The block size is within acceptable limits
  区块大小在可接受范围内
- The first transaction (and only the first) is a coinbase transaction
  第一个交易（且只有第一个）是币基交易
- All transactions within the block are valid using the transaction checklist discussed in Independent Verification of Transactions
  使用前面介绍的交易验证列表，验证区块中的所有交易。

The independent validation of each new block by every node on the network ensures that the miners cannot cheat. In previous sections we saw how miners get to write a transaction that awards them the new bitcoin created within the block and claim the transaction fees. Why don't miners write themselves a transaction for a thousand bitcoin instead of the correct reward? Because every node validates blocks according to the same rules. An invalid coinbase transaction would make the entire block invalid, which would result in the block being rejected and, therefore, that transaction would never become part of the ledger.
网络上每个节点对每个新区块的独立验证，保证了矿工无法欺骗。
在前面章节中，我们看到了矿工们如何获取新比特币和交易费。
为什么矿工不为他们自己记录一笔交易，以获得数以千计的比特币呢?
这是因为，每个节点根据相同的规则对区块进行验证。
一个无效的币基交易将使整个区块无效，这样，币基交易就不会成为区块链的一部分。

The miners have to construct a perfect block, based on the shared rules that all nodes follow, and mine it with a correct solution to the Proof-of-Work. To do so, they expend a lot

of electricity in mining, and if they cheat, all the electricity and effort is wasted. This is why independent validation is a key component of decentralized consensus.
矿工们必须构建一个完美的区块，基于所有节点使用的共同的规则，并且根据正确工作量证明算法进行挖矿，他们要花费大量的电力才能做到这一点。
如果他们作弊，所有的电力和努力都会被浪费掉。
所以，“独立验证”是“去中心化共识”的重要组成部分。

10.10 区块链的组装与选择

The final step in bitcoin's decentralized consensus mechanism is the assembly of blocks into chains and the selection of the chain with the most Proof-of-Work. Once a node has validated a new block, it will then attempt to assemble a chain by connecting the block to the existing blockchain.
比特币的"去中心化的共识机制"的最后一步是，把区块组装到链中，并选择有最大工作量证明的链。
一旦节点验证了一个新区块，它就尝试把这个区块连接的现有区块链上。

Nodes maintain three sets of blocks: those connected to the main blockchain, those that form branches off the main blockchain (secondary chains), and finally, blocks that do not have a known parent in the known chains (orphans). Invalid blocks are rejected as soon as any one of the validation criteria fails and are therefore not included in any chain.
节点维护三个区块集合：

- 已连接到主区块链上
- 形成了主区块链的分支
- 在已知区块链中没有找到父区块

在验证过程中，如果发现有不符合标准的地方，就返回失败，这个区块被拒绝，不会加入到任何链中。

The "main chain" at any time is whichever *valid* chain of blocks has the most cumulative Proof-of-Work associated with it. Under most circumstances this is also the chain with the most blocks in it, unless there are two equal-length chains and one has more Proof-of-Work. The main chain will also have branches with blocks that are "siblings" to the blocks on the main chain. These blocks are valid but not part of the main chain. They are kept for future reference, in case one of those chains is extended to exceed the main chain in work. In the next section (Blockchain Forks), we will see how secondary chains occur as a result of an almost simultaneous mining of blocks at the same height.
在任何时候，主链都是累计了最多工作量证明的有效区块链。
在多数情况下，主链也是包含最多区块的，除非有两个等长的链，并且其中一个有更多的工作量证明。
主链还会有分支，分支中的区块与主链上的区块是"兄弟"关系。
这些区块是有效的，但不是主链的一部分。
保留这些区块的为了在这种情况下使用：如果在未来它们中的一个延长到超过主链的工作量。
在下一节中，我们会看到次链，在同样的区块高度，几乎同时挖出区块时发生。

When a new block is received, a node will try to slot it into the existing blockchain. The node will look at the block's "previous block hash" field, which is the reference to the block's parent. Then, the node will attempt to find that parent in the existing blockchain. Most of the time, the parent will be the "tip" of the main chain, meaning this new block extends the main chain. For example, the new block 277,316 has a reference to the hash of its parent block 277,315. Most nodes that receive 277,316 will already have block 277,315 as the tip of their main chain and will therefore link the new block and extend that chain.
当节点接收到一个新区块时，它会尝试将这个区块插入到现有区块链中。
节点会看这个区块的"父区块链哈希"字段，并尝试在已存在的区块链中找出这个父区块。
大多数情况下，父区块在主区块链的"顶端"，这意味着，这个新区块延长了主链。
举个例子，一个新区块277,316引用了父区块277,315。收到区块277316的大部分节点都已经将区块277315放在主链的顶端，因此，这新区块就延长了主区块链。

Sometimes, as we will see in Blockchain Forks, the new block extends a chain that is not the main chain. In that case, the node will attach the new block to the secondary chain it extends and then compare the work of the secondary chain to the main chain. If the secondary chain has more cumulative work than the main chain, the node will *reconverge* on the secondary chain, meaning it will select the secondary chain as its new main chain, making the old main chain a secondary chain. If the node is a miner, it will now construct a block extending this new, longer, chain.
有时，新区块所延长的区块链并不是主链。
在这种情况下，节点将新区块添加到次链上，并比较次链和主链的难度。

如果次链比主链积累了更多的工作量，节点将收敛于次链，这意味着，节点将选择次链作为新的主链，而之前那个老的主链则成为了次链。
如果节点是矿工，它将开始构造新的区块，来延长这个新的主链。

If a valid block is received and no parent is found in the existing chains, that block is considered an "orphan." Orphan blocks are saved in the orphan block pool where they will stay until their parent is received. Once the parent is received and linked into the existing chains, the orphan can be pulled out of the orphan pool and linked to the parent, making it part of a chain. Orphan blocks usually occur when two blocks that were mined within a short time of each other are received in reverse order (child before parent).
如果节点收到了一个有效的区块，而在现有的区块链中未找到它的父区块，那么这个区块被认为是“孤儿区块”。孤儿区块被保存在孤儿区块池中，直到节点收到它们的父区块。
一旦收到了父区块，并且将其连接到现有区块链上，节点就会将孤儿区块从孤儿区块池中取出，并且连接到它的父区块，让它作为区块链的一部分。
当两个区块在很短的时间间隔内被挖出，节点有可能会以相反的顺序收到它们，这时就会出现这种现象。

By selecting the greatest-cumulative-work valid chain, all nodes eventually achieve network-wide consensus. Temporary discrepancies between chains are resolved eventually as more work is added, extending one of the possible chains. Mining nodes "vote" with their mining power by choosing which chain to extend by mining the next block. When they mine a new block and extend the chain, the new block itself represents their vote.
通过选择最大累积工作量的有效区块链，所有节点最终实现了全网范围内的共识。
随着更多的工作量被添加到链中，链的暂时性差异最终会得到解决。
挖矿节点通过“投票”来选择它们想要延长的区块链，当它们挖出一个新区块，并且延长了一个链，新区块本身就代表它们的投票。

In the next section we will look at how discrepancies between competing chains (forks) are resolved by the independent selection of the greatest-cumulative-work chain.
在下一节中，我们将研究，如何通过最大的累积工作链的独立选择，来解决竞争链（分叉）之间的差异。

10.10.1 区块链分叉

Because the blockchain is a decentralized data structure, different copies of it are not always consistent. Blocks might arrive at different nodes at different times, causing the nodes to have different perspectives of the blockchain. To resolve this, each node always selects and attempts to extend the chain of blocks that represents the most Proof-of-Work, also known as the longest chain or greatest cumulative work chain. By summing the work recorded in each block in a chain, a node can calculate the total amount of work that has been expended to create that chain. As long as all nodes select the greatest-cumulative-work chain, the global bitcoin network eventually converges to a consistent state. Forks occur as temporary inconsistencies between versions of the blockchain, which are resolved by eventual reconvergence as more blocks are added to one of the forks.
因为区块链是一个去中心化的数据结构，所以不同副本之间不是总能保持一致。
区块可能在不同时间到达不同节点，导致节点有不同的区块链全貌。
解决的办法是：每个节点总是选择并尝试延长代表累计了最大工作量证明的区块链，也就是最长或最大累计工作量的链。
节点通过累加链上每个区块的工作量，得到建立这个链所要付出的工作量证明的总量。
只要所有节点都选择最长累计工作量的区块链，整个比特币网络最终会收敛到一致的状态。
分叉是在不同区块链间发生的临时差异，当更多的区块添加到了某个分叉时，这个问题便最终解决。

`Tip`: The blockchain forks described in this section occur naturally as a result of transmission delays in the global network. We will also look at deliberately induced forks later in this chapter.
提示：由于全局网络中的传输延迟，本节中描述的区块链分叉会自然发生。
我们也将在本章稍后再看看故意引起的分叉。

In the next few diagrams, we follow the progress of a "fork" event across the network. The diagram is a simplified representation of the bitcoin network. For illustration purposes, different blocks are shown as different shapes (star, triangle, upside-down triangle, rhombus), spreading across the network. Each node in the network is represented as a circle.
在接下来的几个图中，我们将通过网络跟踪“分叉”事件的进展。
这些图是比特币网络的简化表示。
为了便于描述，不同的区块被显示为不同的形状（星形，三角形，倒三角形，菱形），遍布这个网络。
网络中的每个圆表示一个节点。

Each node has its own perspective of the global blockchain. As each node receives blocks from its neighbors, it updates its own copy of the blockchain, selecting the greatest-cumulative-work chain. For illustration purposes, each node contains a shape that represents the block that it believes is currently the tip of the main chain. So, if you see a star shape in the node, that means that the star block is the tip of the main chain, as far as that node is concerned.
每个节点都有自己的全局区块链视图。
当每个节点从其邻居接收区块时，它会更新其自己的区块链，选择最大累积工作量的链。
为便于描述，每个节点包含一个图形，表示它的主链顶端的区块。
因此，如果在节点里是星形，就意味着，该节点认为星形区块位于主链顶端。

In the first diagram (Before the fork—all nodes have the same perspective), the network has a unified perspective of the blockchain, with the star block as the tip of the main chain.
在图2中，网络有一个统一的区块链视角，都以星形区块为主链的“顶点”。

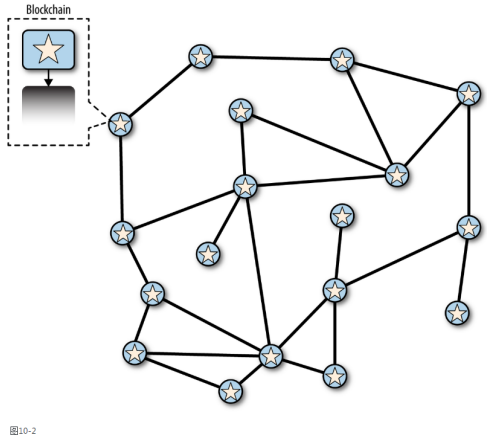

图10-2

Figure 2. Before the fork—all nodes have the same perspective
图2 在分叉前，所有节点有相同的视图

A "fork" occurs whenever there are two candidate blocks competing to form the longest blockchain. This occurs under normal conditions whenever two miners solve the Proof-of-Work algorithm within a short period of time from each other. As both miners discover a solution for their respective candidate blocks, they immediately broadcast their own "winning" block to their immediate neighbors who begin propagating the block across the network.
当有两个候选区块竞争来形成最长区块链时，就发生了分叉。
正常情况下，分叉发生在两个矿工在较短的时间内，各自都算出了工作量证明的解。
两个矿工在各自发现解之后，便立刻把自己的“获胜”区块传播到网络中，先是传播给邻近的节点，而后传播到整个网络。

Each node that receives a valid block will incorporate it into its blockchain, extending the blockchain by one block. If that node later sees another candidate block extending the same parent, it connects the second candidate on a secondary chain. As a result, some nodes will "see" one candidate block first, while other nodes will see the other candidate block and two competing versions of the blockchain will emerge.
每个收到有效区块的节点都会将其并入区块链，并延长这个区块链。

如果节点随后又收到了一个候选区块，而它有相同的父区块，那么节点会把这个区块连接到次链上。
其结果是，一些节点先收到了一个候选区块，另一些节点先收到了另一个候选区块，这时就出现了两个竞争版本的区块链。

In Visualization of a blockchain fork event: two blocks found simultaneously, we see two miners (Node X and Node Y) who mine two different blocks almost simultaneously. Both of these blocks are children of the star block, and extend the chain by building on top of the star block. To help us track it, one is visualized as a triangle block originating from Node X, and the other is shown as an upside-down triangle block originating from Node Y.
在图3中，矿工X和Y几乎同时挖到了两个不同的区块。
这两个区块是顶点区块（星形区块）的子区块，可以延长这个区块链。
为了方便说明，我们把节点X产生的区块标记为三角形，把节点Y生产的区块标记为倒三角形。

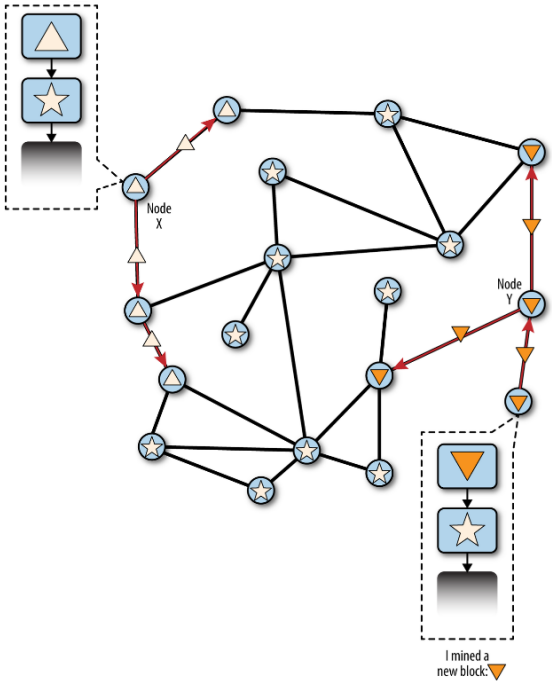

图10-3

Figure 3. Visualization of a blockchain fork event: two blocks found simultaneously
图3 区块链分叉：同时发现两个区块

Let’s assume, for example, that a miner Node X finds a Proof-of-Work solution for a block "triangle" that extends the blockchain, building on top of the parent block "star." Almost simultaneously, the miner Node Y who was also extending the chain from block "star" finds a solution for block "upside-down triangle," his candidate block. Now, there are two possible blocks; one we call "triangle," originating in Node X; and one we call "upside-down triangle," originating in Node Y. Both blocks are valid, both blocks contain a valid solution to the Proof-of-Work, and both blocks extend the same parent (block "star"). Both blocks likely contain most of the same transactions, with only perhaps a few differences in the order of transactions.
例如，节点X挖出了一个三角形区块，放在在星形父区块的上面。
与此同时，节点Y挖出了一个倒三角形区块，也放在星型区块的上面。
现在有两个可能的块，这两个区块都是有效的，均包含有效的工作量证明的解，并延长同一个父区块。
这个两个区块可能包含了几乎相同的交易，只是在交易的排序上有些许不同。

As the two blocks propagate, some nodes receive block "triangle" first and some receive block "upside-down triangle" first. As shown in Visualization of a blockchain fork event: two blocks propagate, splitting the network, the network splits into two different perspectives of the blockchain; one side topped with a triangle block, the other with the upside-down-triangle block.
当两个区块在网络上传播时，一些节点先接收到了三角形区块，另一些节点先接收了倒三角形区块。

如图4所示，比特币网络上的节点对于区块链的顶点产生了分歧，一派以三角形区块为顶点，另一派以倒三角形区块为顶点。

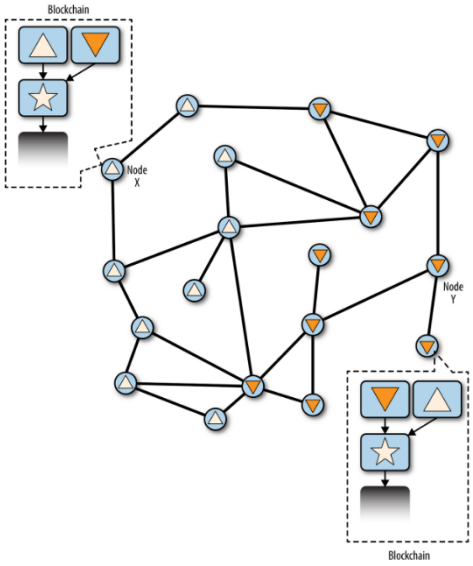

图10-4

Figure 4. Visualization of a blockchain fork event: two blocks propagate, splitting the network

图4 区块链分叉：两个区块传播，划分了网络

In the diagram, a randomly chosen "Node X" received the triangle block first and extended the star chain with it. Node X selected the chain with "triangle" block as the main chain. Later, Node X also received the "upside-down triangle" block. Since it was received second, it is assumed to have "lost" the race. Yet, the "upside-down triangle" block is not discarded. It is linked to the "star" block parent and forms a secondary chain. While Node X assumes it has correctly selected the winning chain, it keeps the "losing" chain so that it has the information needed to reconverge if the "losing" chain ends up "winning."

在图中，节点x先收到三角形区块，并用它扩展星形链。节点x选择三角形区块为主链。

之后，节点x又收到了倒三角区块。由于是后收到的，因此它判定这个倒三角形区块是竞争的失败者。

然而，它不会丢弃这个倒三角形区块。它被链接到星形链的父区块，并形成次链。

虽然节点x认为自己已经正确选择了获胜链，但是它还会保存“失败”链，这样，如果“失败”链最终“获胜”，它还具有重新收敛所需的信息。

On the other side of the network, Node Y constructs a blockchain based on its own perspective of the sequence of events. It received "upside-down triangle" first and elected that chain as the "winner." When it later received "triangle" block, it connected it to the "star" block parent as a secondary chain.

在网络的另一端，节点y根据自己的视角构建一个区块链。

首先收到倒三角形区块，并选择这条链为“获胜”。

当它稍后收到三角形区块时，它将三角形区块连接到星形链的父区块，作为次链。

Neither side is "correct," or "incorrect." Both are valid perspectives of the blockchain. Only in hindsight will one prevail, based on how these two competing chains are extended by additional work.

没有哪一方是“正确的”，或“不正确的”。

这两个都是区块链的有效视图。

只有在事后看来，一个链才能获胜，这是基于这两个相互竞争的链条是如何通过额外的工作来延伸的。

Mining nodes whose perspective resembles Node X will immediately begin mining a candidate block that extends the chain with "triangle" as its tip. By linking "triangle" as the parent of their candidate block, they are voting with their hashing power. Their vote supports the chain that they have elected as the main chain.

节点x阵营的其它节点将立即开始挖掘候选区块，以“三角形区块”作为扩展区块链的顶端。

通过将三角形作为候选区块的父区块，它们用自己的哈希算力进行投票。
它们的投票表明支持自己选择的链为主链。

Any mining node whose perspective resembles Node Y will start building a candidate node with "upside-down triangle" as its parent, extending the chain that they believe is the main chain. And so, the race begins again.
同样，节点Y阵营的其它节点，开始构建一个以倒三角形区块为父节点的候选区块，扩展它们认为是主链的链。比赛再次开始。

Forks are almost always resolved within one block. While part of the network's hashing power is dedicated to building on top of "triangle" as the parent, another part of the hashing power is focused on building on top of "upside-down triangle." Even if the hashing power is almost evenly split, it is likely that one set of miners will find a solution and propagate it before the other set of miners have found any solutions. Let's say, for example, that the miners building on top of "triangle" find a new block "rhombus" that extends the chain (e.g., star-triangle-rhombus). They immediately propagate this new block and the entire network sees it as a valid solution as shown in Visualization of a blockchain fork event: a new block extends one fork, reconverging the network.
分叉问题几乎总是在一个区块内就被解决了。
网络中的一部分算力专注于“三角形区块”为父区块，在其之上建立新的区块，另一部分算力则专注在“倒三角形区块”上。
即便算力在这两个阵营中平均分配，也总有一个阵营抢在另一个阵营前发现工作量证明的解，并将其传播出去。

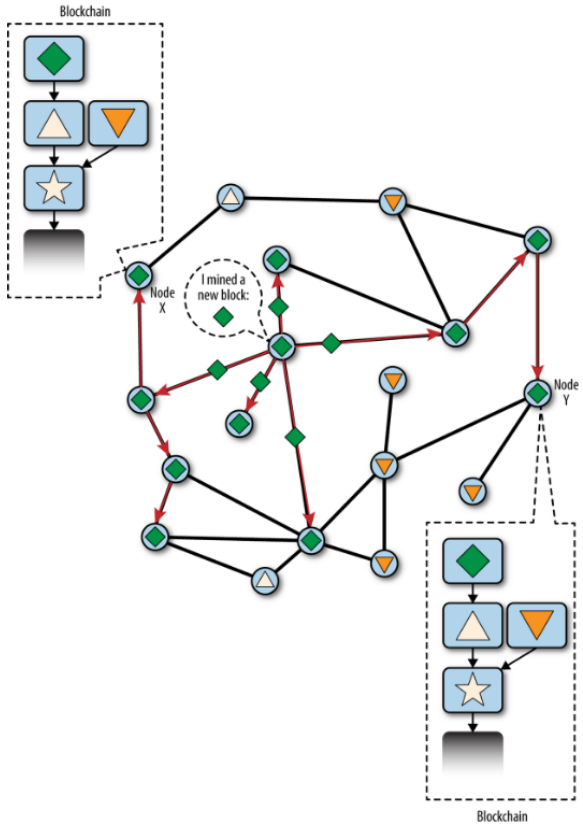

图10-5

Figure 5. Visualization of a blockchain fork event: a new block extends one fork, reconverging the network
图5 区块链分叉：一个新区块扩展了一个分叉，收敛了这个网络

All nodes that had chosen "triangle" as the winner in the previous round will simply extend the chain one more block. The nodes that chose "upside-down triangle" as the winner, however, will now see two chains: star-triangle-rhombus and star-upside-down-triangle. The chain star-triangle-rhombus is now longer (more cumulative work) than the other chain. As a result, those nodes will set the chain star-triangle-rhombus as the main chain and change the star-upside-down-triangle chain to a secondary chain, as shown in Visualization of a blockchain fork event: the network reconverges on a new longest chain.
在上一轮中，选了“三角形区块”作为获胜区块的节点，只需要在主链上延伸一个区块。
选了“倒三角形区块”作为获胜区块的节点，将看到两个链：星形–三角形–菱形、星形–到三角形。
“星形–三角形–菱形”现在是最长链，它有更多的累积工作量。
因此，那些节点会把它设置为主链，把另一个设置为次链，见图6。

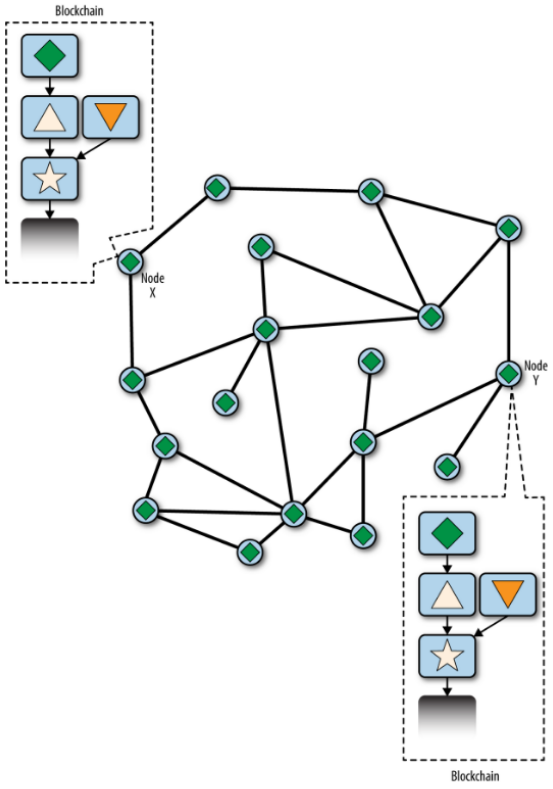

图10-6

Figure 6. Visualization of a blockchain fork event: the network reconverges on a new longest chain
图6 区块链分叉：网络在新的最长链上重新收敛

This is a chain reconvergence, because those nodes are forced to revise their view of the blockchain to incorporate the new evidence of a longer chain. Any miners working on extending the chain star-upside-down-triangle will now stop that work because their candidate block is an "orphan," as its parent "upside-down-triangle" is no longer on the longest chain. The transactions within "upside-down-triangle" that are not within "triangle" are re-inserted in the mempool for inclusion in the next block to become a part of the main chain. The entire network reconverges on a single blockchain star-triangle-rhombus, with "rhombus" as the last block in the chain. All miners immediately start working on candidate blocks that reference "rhombus" as their parent to extend the star-triangle-rhombus chain.
这个一个链重新收敛，因为那些节点被迫修改它们的区块链视图，结合一条较长链的新证据。
在“倒三角形区块”中，但不在“三角形区块”中的交易，被重新插入内存池中（交易未被确认），以包含在一个区块中，使之成为主链的一部分。
任何一个致力于延长“星形–倒三角形”区块链的矿工，现在都将停止这个工作，因为它们的候选区块是一个孤儿，因为它的父区块不在最长链上。
整个网络重新聚合在一个区块链“星形–三角形–菱形”，其中“菱形”是这个链的最后一个区块。

所有矿工立即开始以“菱形”为父区块开始挖掘新的候选区块。

It is theoretically possible for a fork to extend to two blocks, if two blocks are found almost simultaneously by miners on opposite "sides" of a previous fork. However, the chance of that happening is very low. Whereas a one-block fork might occur every day, a two-block fork occurs at most once every few weeks.
从理论上说，一个分叉可能延伸到两个区块，如果两个区块被双方同时找到。
但是，这种情况发生的概率非常低。
单区块分叉每天都可能发生，而双区块分叉则几周才发生一次。

Bitcoin’s block interval of 10 minutes is a design compromise between fast confirmation times (settlement of transactions) and the probability of a fork. A faster block time would make transactions clear faster but lead to more frequent blockchain forks, whereas a slower block time would decrease the number of forks but make settlement slower.
比特币的区块间隔是10分钟，这个设计是“快速确认次数”和“分叉概率”之间的平衡。
更快的区块产生会使清算更快，但导致更频繁的区块链分叉；
更慢的区块产生会减少分叉数量，但使清算更慢。

10.11挖矿和哈希竞赛

Bitcoin mining is an extremely competitive industry. The hashing power has increased exponentially every year of bitcoin's existence. Some years the growth has reflected a complete change of technology, such as in 2010 and 2011 when many miners switched from using CPU mining to GPU mining and field programmable gate array (FPGA) mining. In 2013 the introduction of ASIC mining lead to another giant leap in mining power, by placing the SHA256 function directly on silicon chips specialized for the purpose of mining. The first such chips could deliver more mining power in a single box than the entire bitcoin network in 2010.
比特币挖矿是一个极富竞争性的行业。
自从比特币存在开始，哈希算力每年都以指数级增长。
一些年份的增长还反应出技术的彻底变革，例如2010和2011年，很多矿工从使用CPU挖矿，转换到使用GPU和FGPA挖矿。2013年，引入了ASIC挖矿，它把SHA256功能直接放在挖矿专用的硅芯片上，引起了算力的另一次巨大飞跃。这种芯片首次使用时，一台矿机提供的挖矿算力，比2010年整个比特币网络的算力还要大。

The following list shows the total hashing power of the bitcoin network, over the first eight years of operation:
下表显示了比特币网络前八年中的总哈希算力：

```
2009         0.5 MH/sec —    8 MH/sec     (      16× growth )
2010           8 MH/sec — 116 GH/sec       ( 14,500× growth )
2011         116 GH/sec —    9 TH/sec     (      78× growth )
2012           9 TH/sec —  23 TH/sec       (     2.5× growth )
2013          23 TH/sec —  10 PH/sec       (     450× growth )
2014          10 PH/sec — 300 PH/sec       (      30× growth )
2015         300 PH/sec — 800 PH/sec       (    2.66× growth )
2016         800 PH/sec — 2.5 EH/sec       (    3.12× growth )
```

In the chart in Total hashing power, terahashes per second (TH/sec), we can see that bitcoin network's hashing power increased over the past two years. As you can see, the competition between miners and the growth of bitcoin has resulted in an exponential increase in the hashing power (total hashes per second across the network).
在图7中，可以看到过去两年中比特币网络的哈希算力增长。
矿工和比特币的增长已经导致了哈希算力的指数级增长（网络中每秒的总哈希）。

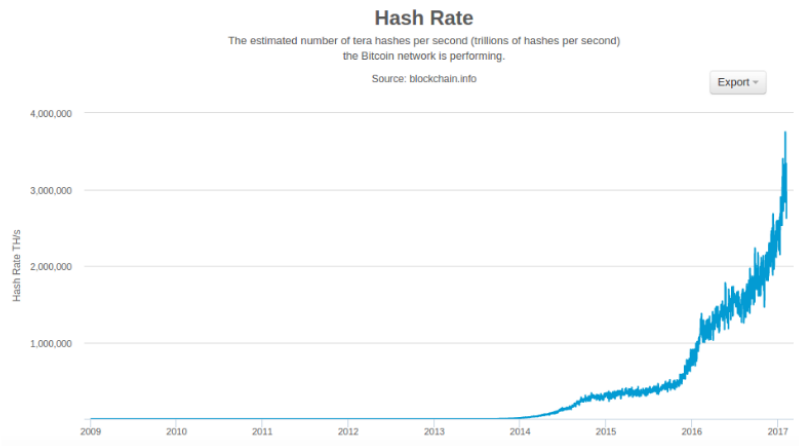

Figure 7. Total hashing power, terahashes per second (TH/sec)
图7：总哈希算力，TH/sec

As the amount of hashing power applied to mining bitcoin has exploded, the difficulty has risen to match it. The difficulty metric in the chart shown in Bitcoin's mining difficulty

metric is measured as a ratio of current difficulty over minimum difficulty (the difficulty of the first block).
随着比特币挖矿算力的爆炸性增长，难度也相应地在增长。
在图8中，显示了当前难度与最小难度（第一个区块的难度）的比例。

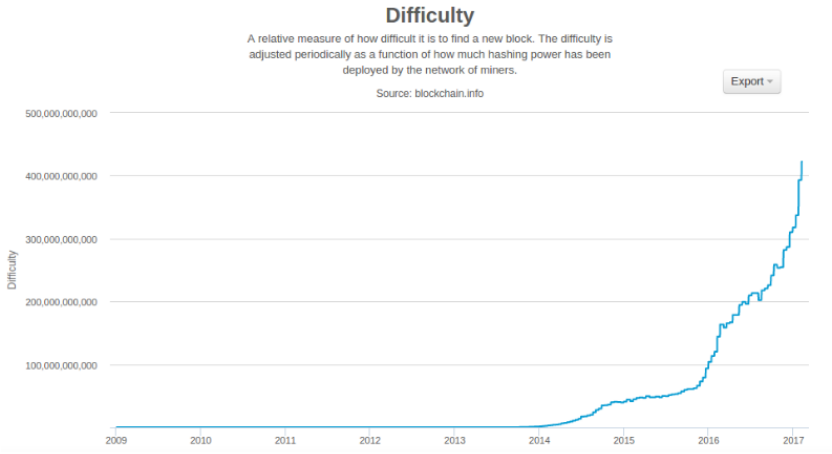

Figure 8. Bitcoin's mining difficulty metric
图8：比特币的挖矿难度

In the last two years, the ASIC mining chips have become increasingly denser, approaching the cutting edge of silicon fabrication with a feature size (resolution) of 16 nanometers (nm). Currently, ASIC manufacturers are aiming to overtake general-purpose CPU chip manufacturers, designing chips with a feature size of 14 nm, because the profitability of mining is driving this industry even faster than general computing. There are no more giant leaps left in bitcoin mining, because the industry has reached the forefront of Moore's Law, which stipulates that computing density will double approximately every 18 months. Still, the mining power of the network continues to advance at an exponential pace as the race for higher density chips is matched with a race for higher density data centers where thousands of these chips can be deployed. It's no longer about how much mining can be done with one chip, but how many chips can be squeezed into a building, while still dissipating the heat and providing adequate power.
在过去两年，ASIC芯片变得日益密集，特征尺寸接近芯片制造业前沿的16纳米。
目前，ASIC制造商的目标是超越通用CPU制造商，设计特征尺寸为14纳米的芯片，因为挖矿利润在驱使这个行业要比通用计算更快。
比特币矿业不再有巨大的飞跃，因为这个行业已经到达了摩尔定律的前沿，摩尔定律指出：计算能力每18个月增加一倍。
尽管如此，网络的挖矿算力仍以指数级在前进，因为更高密度的芯片与更高密度的数据中心在竞争，数据中心可以部署成千上万个这样的芯片。
现在的竞争已经不再是比较一个芯片的挖矿能力，而是在一个地方能够集合多少芯片，同时处理好散热和供电问题。

10.11.1额外的随机数

Since 2012, bitcoin mining has evolved to resolve a fundamental limitation in the structure of the block header. In the early days of bitcoin, a miner could find a block by iterating through the nonce until the resulting hash was below the target. As difficulty increased, miners often cycled through all 4 billion values of the nonce without finding a block. However, this was easily resolved by updating the block timestamp to account for the elapsed time. Because the timestamp is part of the header, the change would allow miners to iterate through the values of the nonce again with different results.
自2012以来，比特币挖掘已经发展到了要解决区块头结构的一个基本限制。
在比特币的早期，矿工可以通过遍历nonce（随机数），来获得符合要求的哈希。

随着难度的增长，矿工经常在尝试了所有的40亿个值后，仍然没有合适的哈希。
但是，这很容易通过更新区块的时间戳来解决。
因为时间戳是区块头的一部分，它的变化可以让矿工用不同的随机值再次遍历。

Once mining hardware exceeded 4 GH/sec, however, this approach became increasingly difficult because the nonce values were exhausted in less than a second. As ASIC mining equipment started pushing and then exceeding the TH/sec hash rate, the mining software needed more space for nonce values in order to find valid blocks. The timestamp could be stretched a bit, but moving it too far into the future would cause the block to become invalid.
当挖矿硬件超过4GH/秒，这种方法也日益困难，因为随机数的值在一秒内就被用尽了。
当ASIC矿机开始推进，超过TH/秒的哈希速率后，挖矿软件需要更多的随机值空间，以找到有效区块。
时间戳可以延伸一点，但延伸太远会使区块变为无效。

A new source of "change" was needed in the block header. The solution was to use the coinbase transaction as a source of extra nonce values. Because the coinbase script can store between 2 and 100 bytes of data, miners started using that space as extra nonce space, allowing them to explore a much larger range of block header values to find valid blocks. The coinbase transaction is included in the merkle tree, which means that any change in the coinbase script causes the merkle root to change.
区块头需要一个新的“修改”源。
解决方案是使用币基交易作为额外的随机值来源。
因为币基脚本可以储存2–100字节的数据，矿工们开始使用这个空间作为额外随机值空间，允许他们探索更大范围的区块头值，以找到有效的区块。
币基交易包含在merkle树中，这意味着，修改币基脚本会导致Merkle根发生变化。

Eight bytes of extra nonce, plus the 4 bytes of "standard" nonce allow miners to explore a total 2^{96} (8 followed by 28 zeros) possibilities *per second* without having to modify the timestamp. If, in the future, miners could run through all these possibilities, they could then modify the timestamp. There is also more space in the coinbase script for future expansion of the extra nonce space.
8字节的额外随机数，加上4字节的标准nonce，允许矿工每秒尝试2^{96}种可能，而无需修改时间戳。
如果未来矿工超越所有这些可能，他们可能要修改时间戳。
在币基脚本中，还有更多的空间用于未来的额外随机数空间的扩展。

10.11.2 矿池

In this highly competitive environment, individual miners working alone (also known as solo miners) don’t stand a chance. The likelihood of them finding a block to offset their electricity and hardware costs is so low that it represents a gamble, like playing the lottery. Even the fastest consumer ASIC mining system cannot keep up with commercial systems that stack tens of thousands of these chips in giant warehouses near hydroelectric powerstations.
在这个激烈竞争的环境中，个体矿工单独工作（独立矿工）没有一点机会。
他们找到一个区块以抵消电力和硬件成本的可能性非常小，以至于可以称得上是赌博，就像是买彩票。
就算是最快的消费型ASIC挖矿系统，也无法跟上商业系统的步伐，这些商业系统有上万的芯片，放在巨大的机房中，临近水电站。

Miners now collaborate to form mining pools, pooling their hashing power and sharing the reward among thousands of participants. By participating in a pool, miners get a smaller share of the overall reward, but typically get rewarded every day, reducing uncertainty.
矿工现在合作组成矿池，汇集他们的哈希算力，来共享奖励。
通过参加矿池，矿工们得到整体回报的一小部分，但通常每天都能得到，因而减少了不确定性。

Let’s look at a specific example. Assume a miner has purchased mining hardware with a combined hashing rate of 14,000 gigahashes per second (GH/s), or 14 TH/s. In 2017 this

equipment costs approximately $2,500 USD. The hardware consumes 1375 watts (1.3 kW) of electricity when running, 33 kW-hours a day, at a cost of $1 to $2 per day at very low electricity rates. At current bitcoin difficulty, the miner will be able to solo mine a block approximately once every 4 years. How do we work out that probability? It is based on a network-wide hashing rate of 3 EH/sec (in 2017), and the miner's rate of 14 TH/sec:
我们来看一个例子。
假设一名矿工购买了算力为14TH/S的设备，在2017年，它的价值大约是2500美元。
该设备运行功率为1.3 KW，每日耗电32度，每日平均成本1或2美元。
以目前的比特币难度，该矿工单独挖出一个区块平均需要4年。
我们是怎么得到这个概率呢?
这是基于2017年网络哈希算力是3 EH/S，而这个矿工的哈希算力是14 TH/S

- $P = (14 * 10^{12} / 3 * 10^{18}) * 210240 = 0.98$

…where 21240 is the number of blocks in four years. The miner has a 98% probability of finding a block over four years, based on the global hash rate at the beginning of the period.
21240是四年中的区块总数。
根据这个时期开始时的总哈希算力，这个矿工在四年中有98%的概率知道一个区块。

If the miner does find a single block in that timeframe, the payout of 12.5 bitcoin, at approximately $1,000 per bitcoin, will result in a single payout of $12,500, which will produce a net profit of about $7,000. However, the chance of finding a block in a 4-year period depends on the miner's luck. He might find two blocks in 4 years and make a very large profit. Or he might not find a block for 5 years and suffer a bigger financial loss. Even worse, the difficulty of the bitcoin Proof-of-Work algorithm is likely to go up significantly over that period, at the current rate of growth of hashing power, meaning the miner has, at most, one year to break even before the hardware is effectively obsolete and must be replaced by more powerful mining hardware.
如果这个矿工在这个时限内挖出一个区块，奖励是12.5个比特币，如果比特币价格为1000美元，可以得到12500美元，得到的净利润大约是7000美元。
但是，在4年内找到一个区块要看这个矿工的运气。他可能在4年中找到两个区块，也可能5年都没有找到一个区块。
更糟的是，比特币的工作量证明算法的难度可能在这段时间内显著上升，按照目前算力增长的速度，这意味着，矿工在设备被下一代更高效的矿机取代之前，最多有1年的时间取得成果。

If this miner participates in a mining pool, instead of waiting for a once-in-four-years $12,500 windfall, he will be able to earn approximately $50 to $60 per week. The regular payouts from a mining pool will help him amortize the cost of hardware and electricity over time without taking an enormous risk. The hardware will still be obsolete in one or two years and the risk is still high, but the revenue is at least regular and reliable over that period. Financially this only makes sense at very low electricity cost (less than 1 cent per kW-hour) and only at very large scale.
如果这个矿工加入矿池，而不是等待4年可能出现一次暴利，他每周能赚取大约50–60美元。
矿池的定期收入能帮他摊销硬件和电力成本，并且不用承担巨大的风险。
1–2年后，硬件仍然会过时，风险仍然很高，但在此期间，收入至少是定期的和可靠的。
从财务上说，只有在这种情况下才是合理的：非常低的电力成本（每千瓦不到1美分），非常大的规模。

Mining pools coordinate many hundreds or thousands of miners, over specialized pool-mining protocols. The individual miners configure their mining equipment to connect to a pool server, after creating an account with the pool. Their mining hardware remains connected to the pool server while mining, synchronizing their efforts with the other miners. Thus, the pool miners share the effort to mine a block and then share in the rewards.
矿池通过专门的矿池协议协调成百上千的矿工。
个体矿工在在矿池中创建一个账号，设置他们的挖矿设备以连接到矿池服务器。
在挖矿、与其它矿工同步工作时，挖矿硬件保持与矿池服务器的连接。
这样，矿池中的矿工分担挖取一个区块的努力，然后分享奖励。

Successful blocks pay the reward to a pool bitcoin address, rather than individual miners. The pool server will periodically make payments to the miners' bitcoin addresses, once their share of the rewards has reached a certain threshold. Typically, the pool server charges a percentage fee of the rewards for providing the pool-mining service.
成功的区块把奖励支付给一个矿池比特币地址，而不是支付给个体矿工。
一旦奖励达到一定数额，矿池服务器会定期支付奖励给矿工的比特币地址。
通常情况下，矿池服务器会要收取一定比例的费用，作为提供矿池服务的回报。

Miners participating in a pool split the work of searching for a solution to a candidate block, earning "shares" for their mining contribution. The mining pool sets a higher target (lower difficulty) for earning a share, typically more than 1,000 times easier than the bitcoin network's target. When someone in the pool successfully mines a block, the reward is earned by the pool and then shared with all miners in proportion to the number of shares they contributed to the effort.
参加矿池的矿工把搜寻候选区块的工作进行分割，并根据他们挖矿的贡献赚取“份额”。
矿池为赚取“份额”设置了一个低难度的目标，通常比比特币网络难度低1000倍以上。
当有人成功挖出一个区块，矿池获得奖励，并和所有矿工按照他们做出贡献的“份额”的比例进行分配。

Pools are open to any miner, big or small, professional or amateur. A pool will therefore have some participants with a single small mining machine, and others with a garage full of high-end mining hardware. Some will be mining with a few tens of a kilowatt of electricity, others will be running a data center consuming a megawatt of power. How does a mining pool measure the individual contributions, so as to fairly distribute the rewards, without the possibility of cheating? The answer is to use bitcoin's Proof-of-Work algorithm to measure each pool miner's contribution, but set at a lower difficulty so that even the smallest pool miners win a share frequently enough to make it worthwhile to contribute to the pool.
矿池对所有矿工开放，无论大小、专业或业余。
一个矿池的参与者中，有人只有一台小矿机，而有些人有一车库高端挖矿硬件。
有人只用几十度电挖矿，也有人会用一个数据中心消耗兆瓦级的电量。
矿池如何衡量每个人的贡献，既能公平分配奖励，又避免作弊的可能?
答案是在设置一个较低难度，使用比特币的工作量证明算法来衡量每个矿工的贡献。
这样，即使是池中最小的矿工也经常能分得奖励，这足以激励他们为矿池做出贡献。

By setting a lower difficulty for earning shares, the pool measures the amount of work done by each miner. Each time a pool miner finds a block header hash that is less than the pool target, she proves she has done the hashing work to find that result. More importantly, the work to find shares contributes, in a statistically measurable way, to the overall effort to find a hash lower than the bitcoin network's target. Thousands of miners trying to find low-value hashes will eventually find one low enough to satisfy the bitcoin network target.
通过设置一个较低难度，矿池可以计量出每个矿工完成的工作量。
每当矿工发现一个小于矿池难度的区块头哈希，就证明了它已经完成了寻找结果所需的哈希计算。
更重要的是，这些是为取得份额贡献而做的工作（以一个统计学上可衡量的方法），而整体上是在寻找一个比特币网络的目标值。
成千上万的矿工尝试找到低值哈希，最终可以找到符合比特币网络要求的结果。

Let's return to the analogy of a dice game. If the dice players are throwing dice with a goal of throwing less than four (the overall network difficulty), a pool would set an easier target, counting how many times the pool players managed to throw less than eight. When pool players throw less than eight (the pool share target), they earn shares, but they don't win the game because they don't achieve the game target (less than four). The pool players will achieve the easier pool target much more often, earning them shares very regularly, even when they don't achieve the harder target of winning the game. Every now and then, one of the pool players will throw a combined dice throw of less than four and the pool wins. Then, the earnings can be distributed to the pool players based on the shares they earned. Even though the target of eight-or-less wasn't winning, it was a fair way to measure dice throws for the players, and it occasionally produces a less-than-four throw.
我们用骰子游戏来说明。

如果网络难度是结果小于4，矿池可以设置一个更容易的目标，统计池中玩家扔出的结果小于8的次数。
当池中玩家扔出的结果小于8，他们得到一个份额，但他们没有赢得这个游戏，因为游戏目标是小于4。
但池中玩家会更经常达到矿池份额目标，不断地赚取他们的份额，尽管他们没有完成赢得比赛的目标。
时不时地，池中的一个成员有可能会扔出一个小于4的结果，这时矿池就赢了。
然后，根据池中玩家获得的份额比例来分配奖励。
尽管目标设置为8没有赢得游戏，但是这是一个衡量玩家们扔出的点数的公平方法，同时也偶尔会产生一个小于4的结果。

Similarly, a mining pool will set a (higher and easier) pool target that will ensure that an individual pool miner can find block header hashes that are less than the pool target often, earning shares. Every now and then, one of these attempts will produce a block header hash that is less than the bitcoin network target, making it a valid block and the whole pool wins.
类似的，矿池会将矿池难度设置为保证个体矿工能够频繁地找到一个符合矿池难度的区块头哈希，从而赢取份额。时不时的，某次尝试会产生一个符合比特币网络目标的区块头哈希，这就产生一个有效区块，然后整个矿池就获胜了。

10.11.2.1管理矿池

Most mining pools are "managed," meaning that there is a company or individual running a pool server. The owner of the pool server is called the *pool operator*, and he charges pool miners a percentage fee of the earnings.
多数矿池是“托管的”，意思是有一个公司或个人经营一个矿池服务器。
矿池服务器的所有者称为矿池管理员，他从矿工的收入中收取一定比例的费用。

The pool server runs specialized software and a pool-mining protocol that coordinate the activities of the pool miners. The pool server is also connected to one or more full bitcoin nodes and has direct access to a full copy of the blockchain database. This allows the pool server to validate blocks and transactions on behalf of the pool miners, relieving them of the burden of running a full node.
矿池服务器运行专门的软件和矿池挖矿协议，它们协调池中矿工们的活动。
矿池服务器同时也连接到一个或更多比特币全节点，并一个可以直接访问的完整区块链数据库。
这使矿池服务器可以代替矿池中的矿工验证区块和交易，消除他们运行一个全节点的负担。

For pool miners, this is an important consideration, because a full node requires a dedicated computer with at least 100 to 150 GB of persistent storage (disk) and at least 2 to 4 GB of memory (RAM). Furthermore, the bitcoin software running on the full node needs to be monitored, maintained, and upgraded frequently. Any downtime caused by a lack of maintenance or lack of resources will hurt the miner’s profitability. For many miners, the ability to mine without running a full node is another big benefit of joining a managed pool.
对于池中的矿工，这是一个重要的考虑，因为一个全节点要求拥有至少100–150GB的磁盘空间和最少2–4GB的内存。
此外，对运行一个全节点的比特币软件来说，需要对它进行监控、维护和频繁升级。
由于缺乏维护或资源，导致的任何宕机都会损害到矿工的利润。
对于很多矿工来说，不需要跑一个全节点就能挖矿，也是加入管理矿池的一大好处。

Pool miners connect to the pool server using a mining protocol such as Stratum (STM) or GetBlockTemplate (GBT). An older standard called GetWork (GWK) has been mostly obsolete since late 2012, because it does not easily support mining at hash rates above 4 GH/s. Both the STM and GBT protocols create block *templates* that contain a template of a candidate block header. The pool server constructs a candidate block by aggregating transactions, adding a coinbase transaction (with extra nonce space), calculating the merkle root, and linking to the previous block hash. The header of the candidate block is then sent to each of the pool miners as a template. Each pool miner then mines using the block template, at a higher (easier) target than the bitcoin network target, and sends any successful results back to the pool server to earn shares.

矿工连接到矿池服务器使用一个挖矿协议，例如：Stratum(STM)或 GetBlockTemplate(GBT)。
有一个旧标准GetWork(GWK)，但从2012年底已基本过时，因为它不支持在哈希速度超过4GH/S时挖矿。
STM和GBT协议都创建包含候选区块头模板的区块模板。
矿池服务器通过打包交易，添加币基交易（和额外的随机数空间），计算merkle根，并添加父区块哈希，从而建立一个候选区块。
这个候选区块的头部作为模板分发给每个矿工，矿工用这个区块模板在低于比特币网络的难度下挖矿，并发送成功的结果给矿池服务器，以赚取份额。

10.11.2.2 P2P矿池

Managed pools create the possibility of cheating by the pool operator, who might direct the pool effort to double-spend transactions or invalidate blocks (see Consensus Attacks). Furthermore, centralized pool servers represent a single-point-of-failure. If the pool server is down or is slowed by a denial-of-service attack, the pool miners cannot mine. In 2011, to resolve these issues of centralization, a new pool mining method was proposed and implemented: P2Pool, a peer-to-peer mining pool without a central operator.
管理矿池存在管理员作弊的可能，管理员可以利用矿池进行双重支付或使区块无效。
此外，中心化的矿池服务器代表着单点故障。如果因为DoS攻击导致服务器出问题，矿工就不能采矿。
在2011年，为了解决由中心化造成的这些问题，提出和实施了一个新的矿池挖矿方法：P2Pool，它是一个P2P的矿池，没有中心管理员。

P2Pool works by decentralizing the functions of the pool server, implementing a parallel blockchain-like system called a *share chain*. A share chain is a blockchain running at a lower difficulty than the bitcoin blockchain. The share chain allows pool miners to collaborate in a decentralized pool by mining shares on the share chain at a rate of one share block every 30 seconds. Each of the blocks on the share chain records a proportionate share reward for the pool miners who contribute work, carrying the shares forward from the previous share block. When one of the share blocks also achieves the bitcoin network target, it is propagated and included on the bitcoin blockchain, rewarding all the pool miners who contributed to all the shares that preceded the winning share block. Essentially, instead of a pool server keeping track of pool miner shares and rewards, the share chain allows all pool miners to keep track of all shares using a decentralized consensus mechanism like bitcoin's blockchain consensus mechanism.
P2Pool通过去中性化矿池服务器的功能来工作，实现一个并行的类似区块链的系统，名叫份额链。
一个份额链是难度低于比特币区块链的一个区块链系统。
份额链允许矿工在一个去中心化的池中合作，以每30秒一个份额区块的速度在份额链上挖矿。
份额链上的区块记录了矿工贡献的工作份额，并且继承了之前份额区块上的份额记录。
当一个份额区块上还实现了比特币网络的难度目标时，它将被广播并包含到比特币的区块链上，并奖励所有已经在份额链区块中取得份额的矿工。
本质上说，比起用一个矿池服务器记录矿工的份额和奖励，份额链允许所有矿工通过类似比特币区块链系统以去中心化的共识机制跟踪所有份额。

P2Pool mining is more complex than pool mining because it requires that the pool miners run a dedicated computer with enough disk space, memory, and internet bandwidth to support a full bitcoin node and the P2Pool node software. P2Pool miners connect their mining hardware to their local P2Pool node, which simulates the functions of a pool server by sending block templates to the mining hardware. On P2Pool, individual pool miners construct their own candidate blocks, aggregating transactions much like solo miners, but then mine collaboratively on the share chain. P2Pool is a hybrid approach that has the advantage of much more granular payouts than solo mining, but without giving too much control to a pool operator like managed pools.
P2Pool挖矿比矿池挖矿更复杂，因为它要求矿工运行一个专用计算机（空间、内存、带宽充足）来支持一个比特币全节点和P2Pool节点软件。
P2Pool矿工连接他们的挖矿硬件到本地的P2Pool节点，本地P2Pool节点模拟矿池服务器的功能：发送区块模板给挖矿硬件。

在P2Pool中，各个矿工构建自己的候选区块，像独立矿工一样打包交易，但是他们在份额链上合作挖矿。

P2Pool是一种混合方法，它比独立挖矿有更细粒度的收入优势，但不像托管理矿池那样给管理员太多权力。

Even though P2Pool reduces the concentration of power by mining pool operators, it is conceivably vulnerable to 51% attacks against the share chain itself. A much broader adoption of P2Pool does not solve the 51% attack problem for bitcoin itself. Rather, P2Pool makes bitcoin more robust overall, as part of a diversified mining ecosystem.
即使P2Pool减少了矿池管理员的权力集中，但它也可能容易受到份额链本身的51%的攻击。
P2Pool的广泛采用并不能解决比特币本身的51%攻击问题。
作为多样化挖矿生态系统的一部分，P2Pool使得比特币整体上更加健壮。

10.12 共识攻击

Bitcoin’s consensus mechanism is, at least theoretically, vulnerable to attack by miners (or pools) that attempt to use their hashing power to dishonest or destructive ends. As we saw, the consensus mechanism depends on having a majority of the miners acting honestly out of self-interest. However, if a miner or group of miners can achieve a significant share of the mining power, they can attack the consensus mechanism so as to disrupt the security and availability of the bitcoin network.
至少在理论上，比特币的共识机制是易受矿工（或矿池）攻击，他们试图使用哈希算力进行欺骗或破坏。
就像前面讲的，共识机制依赖于：大多数的矿工为了自身利益而表现诚实。
但是，如果矿工能够占据挖矿算力的多数份额，他们就能攻击这个共识机制，从而破坏比特币网络的安全性和可用性。

It is important to note that consensus attacks can only affect future consensus, or at best, the most recent past (tens of blocks). Bitcoin’s ledger becomes more and more immutable as time passes. While in theory, a fork can be achieved at any depth, in practice, the computing power needed to force a very deep fork is immense, making old blocks practically immutable. Consensus attacks also do not affect the security of the private keys and signing algorithm (ECDSA). A consensus attack cannot steal bitcoin, spend bitcoin without signatures, redirect bitcoin, or otherwise change past transactions or ownership records. Consensus attacks can only affect the most recent blocks and cause denial-of-service disruptions on the creation of future blocks.
值得注意的是，共识攻击只影响未来的共识，或最近的过去（几十个区块）。
随着时间的推移，比特币的账本越来越不可修改。
虽然理论上说，修改一个非常深的分叉所需的算力是非常巨大的，这使老的区块实际上是不可修改。
共识攻击也不会影响“用户的私钥”和“签名算法（ECDSA）”。
因为没有签名，所以，共识攻击不能：偷盗比特币、在没有签名的情况下花费比特币、重定向比特币、改变过去的交易或所有权记录。
共识攻击只能影响最近的区块，会对“未来区块的创建”进行DoS破坏。（ddk：就是说，使得最近和未来的区块无效，也就是使它们包含的交易无效。）

One attack scenario against the consensus mechanism is called the "51% attack." In this scenario a group of miners, controlling a majority (51%) of the total network’s hashing power, collude to attack bitcoin. With the ability to mine the majority of the blocks, the attacking miners can cause deliberate "forks" in the blockchain and double-spend transactions or execute denial-of-service attacks against specific transactions or addresses. A fork/double-spend attack is where the attacker causes previously confirmed blocks to be invalidated by forking below them and re-converging on an alternate chain. With sufficient power, an attacker can invalidate six or more blocks in a row, causing transactions that were considered immutable (six confirmations) to be invalidated. Note that a double-spend can only be done on the attacker’s own transactions, for which the attacker can produce a valid signature. Double-spending one’s own transactions is profitable if by invalidating a transaction the attacker can get an irreversible exchange payment or product without paying for it.
针对共识机制的一个攻击场景吃“51%攻击”。
在这个场景中，一群矿工控制了整个网络哈希算力的51%，他们联合起来攻击比特币。
由于能够挖掘多数区块，攻击者可以故意分叉和双重花费，或针对特定交易或地址进行DoS攻击。
一个分叉/双重花费攻击是，攻击者使以前确认的区块无效，方法是在这些区块之前分叉，重新收敛于另一个链。
有了足够的算力，攻击者就能使6个或更多区块无效，导致那些被任务无能修改的交易（6个确认）变为无效。
注意，双重花费只能在攻击者自己的交易上实现，因为攻击者可以对自己的交易产生有效的签名。
如果是下面这种情况，双重花费自己的交易是有利可图的：通过使一个交易无效，攻击者可以获得一个交易支付或产品，而不用为它付钱。

Let’s examine a practical example of a 51% attack. In the first chapter, we looked at a transaction between Alice and Bob for a cup of coffee. Bob, the cafe owner, is willing to accept payment for cups of coffee without waiting for confirmation (mining in a block), because the risk of a double-spend on a cup of coffee is low in comparison to the convenience of rapid customer service. This is similar to the practice of coffee shops that accept credit card payments without a signature for amounts below $25, because the risk of a credit-card chargeback is low while the cost of delaying the transaction to obtain a signature is comparatively larger.
让我们看一个“51%攻击”的实际例子。
Alice 使用比特币在 Bob的咖啡店买了一杯咖啡。
Bob 愿意在 Alice 给自己的转账交易确认数为零时就向她提供咖啡，这是因为这种小额交易遭遇“51%攻击”的风险（风险很小），与顾客购物的即时性比起来（客户需要），微不足道。
这就和大部分咖啡店的做法一样：对低于25美元的信用卡消费，不会费力地向顾客索要签名，因为与顾客撤销这笔信用卡支付的风险比起来（风险很小），向用户索要信用卡签名的成本更高（客户需要）。

In contrast, selling a more expensive item for bitcoin runs the risk of a double-spend attack, where the buyer broadcasts a competing transaction that spends the same inputs (UTXO) and cancels the payment to the merchant. A double-spend attack can happen in two ways: either before a transaction is confirmed, or if the attacker takes advantage of a blockchain fork to undo several blocks. A 51% attack allows attackers to double-spend their own transactions in the new chain, thus undoing the corresponding transaction in the old chain.
相比之下，比特币支付的大额交易就会有更高的双重支付攻击的风险，这时买家广播一个竞争交易，它花费相同的UTXO，取消了给商家的支付。
双重支付可以有两种方式：在交易被确认之前，或者攻击者利用区块链分叉来取消几个区块。
一个51%攻击允许攻击者在一个新链上双重花费他们自己的交易，从而取消了老链上的对应交易。

In our example, malicious attacker Mallory goes to Carol’s gallery and purchases a beautiful triptych painting depicting Satoshi Nakamoto as Prometheus. Carol sells "The Great Fire" paintings for $250,000 in bitcoin to Mallory. Instead of waiting for six or more confirmations on the transaction, Carol wraps and hands the paintings to Mallory after only one confirmation. Mallory works with an accomplice, Paul, who operates a large mining pool, and the accomplice launches a 51% attack as soon as Mallory’s transaction is included in a block.
例如，Mallory在Carol的画廊买了一幅画，他给Carol支付了价值25万美金的比特币。
在等到了一个交易确认后，Carol把这幅画包好，交给了Mallory。
这时，Mallory 的同伙Paul，Paul运营了一个有很大算力的矿池，在这笔交易写入区块链时，Paul开始了51%攻击。

Paul directs the mining pool to remine the same block height as the block containing Mallory’s transaction, replacing Mallory’s payment to Carol with a transaction that double-spends the same input as Mallory’s payment. The double-spend transaction consumes the same UTXO and pays it back to Mallory’s wallet, instead of paying it to Carol, essentially allowing Mallory to keep the bitcoin.
首先，Paul利用自己矿池算力重新计算包含这笔交易的区块，并且在新区块里将原来的交易替换成了另外一笔交易（例如把钱转给Mallory 的另一个钱包地址），从而实现了“双重支付”。
这笔“双重支付”交易使用了跟原有交易一样的UTXO，但收款人被替换成了Mallory的钱包地址。

Paul then directs the mining pool to mine an additional block, so as to make the chain containing the double-spend transaction longer than the original chain (causing a fork below the block containing Mallory’s transaction). When the blockchain fork resolves in favor of the new (longer) chain, the double-spent transaction replaces the original payment to Carol. Carol is now missing the three paintings and also has no bitcoin payment. Throughout all this activity, Paul’s mining pool participants might remain blissfully unaware of the double-spend attempt, because they mine with automated miners and cannot monitor every transaction or block.
然后，Paul利用矿池又挖出一个区块，这样，包含这笔“双重支付”交易的区块链就比原有的区块链多出了一个区块。

到此，更长的分叉区块链取代了原有的区块链，“双重支付”交易取代了原来给Carol的交易，Carol既没有收到价值25万美金的比特币，那副画也被Mallory白白拿走了。
在整个过程中，Paul矿池里的其他矿工可能自始至终都没有觉察到这笔“双重支付”交易有什么异样，因为挖矿程序都是自动在运行，不可能监控每个交易和每个区块。

To protect against this kind of attack, a merchant selling large-value items must wait at least six confirmations before giving the product to the buyer. Alternatively, the merchant should use an escrow multisignature account, again waiting for several confirmations after the escrow account is funded. The more confirmations elapse, the harder it becomes to invalidate a transaction with a 51% attack. For high-value items, payment by bitcoin will still be convenient and efficient even if the buyer has to wait 24 hours for delivery, which would correspond to approximately 144 confirmations.
为了避免这类攻击，商家应该在交易得到全网6次确认之后再交付商品。
或者，商家应该使用一个托管多签名账户，在托管账户可以使用之前，需要等待几次确认。
确认次数越多，就越难使用51%攻击使一个交易无效。
对于高价值商品，即使买家要等待24小时（144次确认）才能发货，使用比特币仍然是方便和高效的。

In addition to a double-spend attack, the other scenario for a consensus attack is to deny service to specific bitcoin participants (specific bitcoin addresses). An attacker with a majority of the mining power can simply ignore specific transactions. If they are included in a block mined by another miner, the attacker can deliberately fork and remine that block, again excluding the specific transactions. This type of attack can result in a sustained denial-of-service against a specific address or set of addresses for as long as the attacker controls the majority of the mining power.
除了“双重支付”攻击，还有一种攻击场景是针对特定比特币地址的DoS攻击。
一个拥有了系统中绝大多数算力的攻击者，可以轻易地忽略某一笔特定的交易。
如果这笔交易存在于另一个矿工所产生的区块中，攻击者可以故意分叉，然后重新产生这个区块，并且把想忽略的交易从这个区块中移除。
这种攻击造成的结果是，只要这个攻击者拥有系统中的绝大多数算力，那么他就可以持续地干预某个或某批特定钱包地址产生的所有交易，从而达到拒绝为这些地址服务的目的。

Despite its name, the 51% attack scenario doesn’t actually require 51% of the hashing power. In fact, such an attack can be attempted with a smaller percentage of the hashing power. The 51% threshold is simply the level at which such an attack is almost guaranteed to succeed.
需要注意的是，51%攻击并非需要攻击者至少有51%的算力才能发起攻击，实际上，即使拥有不到51%的系统算力，仍然可以尝试发起这种攻击。
之所以命名为51%攻击，只是因为在攻击者的算力达到51%这个值时，发起的攻击尝试几乎肯定会成功。

A consensus attack is essentially a tug-of-war for the next block and the "stronger" group is more likely to win. With less hashing power, the probability of success is reduced, because other miners control the generation of some blocks with their "honest" mining power.
本质上看，共识攻击，就是系统中所有矿工的算力被分成了两组，一组是诚实算力，一组是攻击者算力，两组人都在争先恐后地计算区块链上的新区块，只是攻击者算力挖出的是精心构造、包含或剔除了某些交易的区块。因此，攻击者拥有的算力越少，在这场竞赛中获胜的可能性就越小。

One way to look at it is that the more hashing power an attacker has, the longer the fork he can deliberately create, the more blocks in the recent past he can invalidate, or the more blocks in the future he can control. Security research groups have used statistical modeling to claim that various types of consensus attacks are possible with as little as 30% of the hashing power.
从另一个角度讲，攻击者的哈希算力越多，它就可以故意创造更长的分叉，可以使更多最近的区块无效，或者他可以控制更多未来的区块。
安全研究组织已经使用统计模型得出结论，只要有30%的哈希算力，各种类型的共识攻击都是有可能的。

The massive increase of total hashing power has arguably made bitcoin impervious to attacks by a single miner. There is no possible way for a solo miner to control more than a small percentage of the total mining power. However, the centralization of control caused

by mining pools has introduced the risk of for-profit attacks by a mining pool operator. The pool operator in a managed pool controls the construction of candidate blocks and also controls which transactions are included. This gives the pool operator the power to exclude transactions or introduce double-spend transactions. If such abuse of power is done in a limited and subtle way, a pool operator could conceivably profit from a consensus attack without being noticed.
全网哈希算力的急剧增长已经使得比特币系统不再可能被一个矿工攻击。
因为一个矿工不可能控制很多挖矿算力。
但是，中心化控制的矿池则引入了风险：矿池管理员为牟利而发动攻击。
矿池管理员控制候选区块的构造，还控制包含哪些交易。这使他有权剔除特定交易，或包含双花交易。
如果这种权力以微妙而有节制的方式使用的话，那么矿池管理员就可以在不被发现的情况下发动共识攻击并获益。

Not all attackers will be motivated by profit, however. One potential attack scenario is where an attacker intends to disrupt the bitcoin network without the possibility of profiting from such disruption. A malicious attack aimed at crippling bitcoin would require enormous investment and covert planning, but could conceivably be launched by a well-funded, most likely state-sponsored, attacker. Alternatively, a well-funded attacker could attack bitcoin's consensus by simultaneously amassing mining hardware, compromising pool operators, and attacking other pools with denial-of-service. All of these scenarios are theoretically possible, but increasingly impractical as the bitcoin network's overall hashing power continues to grow exponentially.
但是，并不是所有的攻击者都是为了利益。
一个潜在的攻击场景是，攻击者的目的是破坏整个比特币系统，而不是为了牟利。
这种攻击需要巨大的投入和精心的计划，因此可以想象，这种攻击很有可能来自政府资助的组织。
或者，一个资金雄厚的攻击者可以同时使用一下方法攻击比特币的共识：积累挖矿硬件、贿赂矿池管理员、使用DoS攻击攻击其它矿池。
所有这些场景在理论上都是可能的，但随着比特币网络的整体哈希算力继续呈指数增长，这些场景变得越来越不切实际。

Undoubtedly, a serious consensus attack would erode confidence in bitcoin in the short term, possibly causing a significant price decline. However, the bitcoin network and software are constantly evolving, so consensus attacks would be met with immediate countermeasures by the bitcoin community, making bitcoin more robust.
毫无疑问，一次严重的共识攻击势必会打击人们对比特币系统的信心，进而可能导致比特币价格的跳水。
但是，比特币网络和软件在不断演进，所以，共识攻击可能会遇到比特币社区的快速应对措施，使比特币更加健壮。

10.13改变共识规则

The rules of consensus determine the validity of transactions and blocks. These rules are the basis for collaboration between all bitcoin nodes and are responsible for the convergence of all local perspectives into a single consistent blockchain across the entire network.
共识规则决定了交易和区块的有效性。
这些规则是所有比特币节点之间协作的基础，并且负责把所有本地视图收敛到一个一致的区块链。

While the consensus rules are invariable in the short term and must be consistent across all nodes, they are not invariable in the long term. In order to evolve and develop the bitcoin system, the rules have to change from time to time to accommodate new features, improvements, or bug fixes. Unlike traditional software development, however, upgrades to a consensus system are much more difficult and require coordination between all the participants.
虽然共识规则在短期内是不变的，并且在所有节点必须一致，但长期来看，它们并非不可改变。
为了演进和开发比特币系统，规则必须随时间而改变，以容纳新功能、改进或修复错误。
但是，与传统软件开发不同，对共识系统的升级要困难得多，需要所有参与者之间的协调。

10.13.1硬分叉

In Blockchain Forks we looked at how the bitcoin network may briefly diverge, with two parts of the network following two different branches of the blockchain for a short time. We saw how this process occurs naturally, as part of the normal operation of the network and how the network reconverges on a common blockchain after one or more blocks are mined.
在前面章节中，我们看到了比特币网络如何短暂地分叉，网络中的两个部分在短时间内追随区块链的两个不同分支。
我们看到这个过程是如何自然发生的，作为网络通常操作的一部分；以及如何在一个或多个区块被挖出之后，网络再次收敛于一个公共的区块链上。

There is another scenario in which the network may diverge into following two chains: a change in the consensus rules. This type of fork is called a *hard fork*, because after the fork the network does not reconverge onto a single chain. Instead, the two chains evolve independently. Hard forks occur when part of the network is operating under a different set of consensus rules than the rest of the network. This may occur because of a bug or because of a deliberate change in the implementation of the consensus rules.
另一个使网络分成两个链的场景是：修改了共识规则。
这种分叉称为“硬分叉”，因为这种分叉后，网络不会重新收敛到一个链上。这两条链会独立发展。
当比特币网络的一部分节点按照与网络的其余部分节点不同的规则运行时，就会发生硬分叉。
这可能是由于错误或故意改变了规则而发生的。

Hard forks can be used to change the rules of consensus, but they require coordination between all participants in the system. Any nodes that do not upgrade to the new consensus rules are unable to participate in the consensus mechanism and are forced onto a separate chain at the moment of the hard fork. Thus, a change introduced by a hard fork can be thought of as not "forward compatible," in that nonupgraded systems can no longer process the new consensus rules.
硬分叉可用于改变共识规则，但需要在系统中所有参与者之间进行协调。
没有升级到新的共识规则的任何节点都不能参与这个共识机制，并且在硬分叉时被强制到另一个链上。
因此，硬分叉引入的变化可以被认为是“向前不兼容”，因为未升级的系统不能再处理新的共识规则。

Let’s examine the mechanics of a hard fork with a specific example.
我们用一个例子来看一下硬分叉的机制。

A blockchain with forks shows a blockchain with two forks. At block height 4, a one-block fork occurs. This is the type of spontaneous fork we saw in Blockchain Forks. With the mining of block 5, the network reconverges on one chain and the fork is resolved.
图9显示的区块链有两个分叉。
在区块高度4处，发生了一个区块分叉，这是比特币分叉中的自发分叉类型。
经过区块5的挖掘，网络在一条链上重新收敛，分叉被解决。

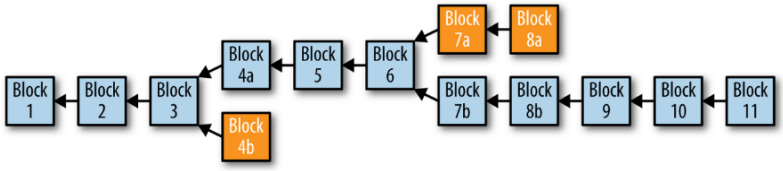

Figure 9. A blockchain with forks
图9：有分叉的区块链

Later, however, at block height 6, a hard fork occurs. Let's assume that a new implementation of the client is released with a change in the consensus rules. Starting on block height 7, miners running this new implementation will accept a new type of digital signature, let's call it a "Smores" signature, that is not ECDSA based. Immediately after, a node running the new implementation creates a transaction that contains a Smores signature and a miner with the updated software mines block 7b containing this transaction.
但是，后来在区块高度6处发生了硬分叉。
我们假设原因是由于新共识规则的修改而发布了一个新的客户端版本。
从块高度7开始，矿工运行新的版本，需要接受新类型的数字签名，我们称之为“Smores”签名，它不是基于ECDSA的签名。紧接着，运行新版本的节点创建了一各包含Smores签名的交易，它挖出了包含此交易的区块7b。

Any node or miner that has not upgraded the software to validate Smores signatures is now unable to process block 7b. From their perspective, both the transaction that contained a Smores signature and block 7b that contained that transaction are invalid, because they are evaluating them based upon the old consensus rules. These nodes will reject the transaction and the block and will not propagate them. Any miners that are using the old rules will not accept block 7b and will continue to mine a candidate block whose parent is block 6. In fact, miners using the old rules may not even receive block 7b if all the nodes they are connected to are also obeying the old rules and therefore not propagating the block. Eventually, they will be able to mine block 7a, which is valid under the old rules and does not contain any transactions with Smores signatures.
任何未升级软件的节点在验证Smores签名时，都无法处理区块7b。
从它们的角度看，包含Smores签名的交易和包含该交易的区块7b都是无效的，因为它们是根据旧的共识规则进行计算的。这些节点将拒绝这个交易和区块，并且不会传播它们。
正在使用旧规则的矿工都不接受区块7b，并且继续挖掘其父区块为6的候选块。
实际上，如果它们连接的所有节点也都遵守旧的规则，遵守旧规则的矿工甚至可能接收不到块7b，因此不会传播这个区块。最终，他们将能够挖出区块7a，这个旧的规则是有效的，其中不包含有Smores签名的任何交易。

The two chains continue to diverge from this point. Miners on the "b" chain will continue to accept and mine transactions containing Smores signatures, while miners on the "a" chain will continue to ignore these transactions. Even if block 8b does not contain any Smores-signed transactions, the miners on the "a" chain cannot process it. To them it appears to be an orphan block, as its parent "7b" is not recognized as a valid block.
这两个链从这一点开始分叉。
b链的矿工将继续接受并开采含有Smores签名的交易，而a链上的矿工将继续忽视这些交易。
即使区块8b不包含任何Smores签名的交易，a链上的矿工也无法处理。对他们来说，它是一个孤儿区块，因为它的父区块7b被认为是一个无效的区块。

10.13.2硬分叉：软件、网络、挖矿和链

For software developers, the term "fork" has another meaning, adding confusion to the term "hard fork." In open source software, a fork occurs when a group of developers choose to follow a different software roadmap and start a competing implementation of an open source project. We've already discussed two circumstances that will lead to a hard fork: a bug in the consensus rules and a deliberate modification of the consensus rules. In the case of a deliberate change to the consensus rules, a software fork precedes the hard fork. However, for this type of hard fork to occur, a new software implementation of the consensus rules must be developed, adopted, and launched.
对于软件开发者来说，“分叉”有另一个含义，对“硬分叉”一词增加了混淆。
在开源软件中，当一组开发人员选择遵循不同的软件路线图，并启动开源项目的竞争实施时，会发生叉。
我们已经讨论了两种情况，这将导致硬分叉：共识规则中的错误、对共识规则的故意修改。
在故意改变共识规则的情况下，软件分叉在硬分叉之前。
但是，对于这种类型的硬分叉，新的共识规则必须通过开发、采用和启动新的软件实现。

Examples of software forks that have attempted to change consensus rules include Bitcoin XT, Bitcoin Classic, and most recently Bitcoin Unlimited. However, none of these software forks have resulted in a hard fork. While a software fork is a necessary precondition, it is not in itself sufficient for a hard fork to occur. For a hard fork to occur, the competing implementation must be adopted and the new rules activated, by miners, wallets, and intermediary nodes. Conversely, there are numerous alternative implementations of Bitcoin Core, and even software forks, that do not change the consensus rules and barring a bug, can coexist on the network and interoperate without causing a hard fork.
试图改变共识规则的软件分叉的例子包括Bitcoin XT、Bitcoin Classic、Bitcoin Unlimited。

但是，这些软件分叉都没有产生硬分叉。
虽然软件分叉是一个必要的前提条件，但它本身不足以发生硬分叉。
对于硬分叉发生，必须是由于采取相互竞争的实施方案，并且规则需要由矿工、钱包和中间节点激活。
相反，有许多Bitcoin Core的替代实现方案，甚至软件分叉，这些没有改变共识规则，它们可以在网络上共存并互操作，最终并未导致硬分叉。

Consensus rules may differ in obvious and explicit ways, in the validation of transactions or blocks. The rules may also differ in more subtle ways, in the implementation of the consensus rules as they apply to bitcoin scripts or cryptographic primitives such as digital signatures. Finally, the consensus rules may differ in unanticipated ways because of implicit consensus constraints imposed by system limitations or implementation details. An example of the latter was seen in the unanticipated hard fork during the upgrade of Bitcoin Core 0.7 to 0.8, which was caused by a limitation in the Berkley DB implementation used to store blocks.
共识规则可能在一些方面明显不同，在验证交易或区块上不同。
共识规则也可能有细微差别，比如共识规则应用于比特币脚本或加密原语（如数字签名）。
最后，共识规则的差别还可能由于意料之外，比如由于系统限制或实现细节所产生的隐含共识约束。
在将Bitcoin Core 0.7升级到0.8时，就发生了一个意外的硬分叉，那是由于用于存储区块的Berkley DB实现中的一个限制引起的。

Conceptually, we can think of a hard fork as developing in four stages: a software fork, a network fork, a mining fork, and a chain fork.
从概念上讲，我们可以把硬分叉看做四个发展阶段：软件分叉、网络分叉、挖矿分叉、区块链分叉。

The process begins when an alternative implementation of the client, with modified consensus rules, is created by developers.
这个过程开始于开发人员创建了客户端的另一个实现，它修改了共识规则。

When this forked implementation is deployed in the network, a certain percentage of miners, wallet users, and intermediate nodes may adopt and run this implementation. A resulting fork will depend upon whether the new consensus rules apply to blocks,

transactions, or some other aspect of the system. If the new consensus rules pertain to transactions, then a wallet creating a transaction under the new rules may precipitate a network fork, followed by a hard fork when the transaction is mined into a block. If the new rules pertain to blocks, then the hard fork process will begin when a block is mined under the new rules.
当这个分叉实现部署在网络中时，一定比例的矿工、钱包用户和中间节点可能采用和运行了这个实现。
产生的分叉依赖于新共识规则是否应用于区块、交易或系统其它方面。
如果新共识规则与交易有关，那么当钱包用新共识规则创建一个交易时，可能会产生出一个网络分叉。
随后，当这个交易被放到一个区块时，就产生了硬分叉。
如果新规则与区块相关，则这个硬分叉过程开始于：使用新规则挖掘一个区块时。

First, the network will fork. Nodes based on the original implementation of the consensus rules will reject any transactions and blocks that are created under the new rules. Furthermore, the nodes following the original consensus rules will temporarily ban and disconnect from any nodes that are sending them these invalid transactions and blocks. As a result, the network will partition into two: old nodes will only remain connected to old nodes and new nodes will only be connected to new nodes. A single transaction or block based on the new rules will ripple through the network and result in the partition into two networks.
首先，是网络分叉。基于旧的共识规则的节点将拒绝根据新规则创建的任何交易和块。
此外，遵循旧的共识规则的节点将暂时禁止和断开发送这些无效交易和块的任何节点。
因此，网络将分为两部分：旧节点将只保留连接到旧节点，新节点只能连接到新节点。
基于新规则的一个交易或区块将逐渐通过网络，导致分成了两个网络。

Once a miner using the new rules mines a block, the mining power and chain will also fork. New miners will mine on top of the new block, while old miners will mine a separate chain based on the old rules. The partitioned network will make it so that the miners operating on separate consensus rules won't likely receive each other's blocks, as they are connected to two separate networks.
一旦使用新规则的矿工开采了一个区块，挖矿算力和区块链也将分叉。
新的矿工将在新区块之上挖掘，而老矿工将根据旧的规则挖掘另一个的链。
被划分的网络将使得按照各自共识规则运行的矿工将不会接收彼此的区块，因为它们连接到两个分开的网络。

10.13.3分离矿工和难度

As miners diverge into mining two different chains, the hashing power is split between the chains. The mining power can be split in any proportion between the two chains. The new rules may only be followed by a minority, or by the vast majority of the mining power.
随着矿工们被分裂开始开采两条不同的链，链上的哈希算力也被分裂。
两个链之间的挖矿能力可以分成任意比例。
新的规则可能只有少数人跟随，或者也可能是绝大多数矿工。

Let's assume, for example, an 80%–20% split, with the majority of the mining power using the new consensus rules. Let's also assume that the fork occurs immediately after a retargeting period.
The two chains would each inherit the difficulty from the retargeting period. The new consensus rules would have 80% of the previously available mining power committed to them. From the perspective of this chain, the mining power has suddenly declined by 20% vis-a-vis the previous period. Blocks will be found on average every 12.5 minutes, representing the 20% decline in mining power available to extend this chain. This rate of block issuance will continue (barring any changes in hashing power) until 2016 blocks are mined, which will take approximately 25,200 minutes (at 12.5 minutes per block), or 17.5 days. After 17.5 days, a retarget will occur and the difficulty will adjust (reduced by 20%) to produce 10-minute blocks again, based on the reduced amount of hashing power in this chain.
我们假设是80%–20%的划分，大多数挖矿算力使用了新的共识规则。

我们还假设分叉在改变目标（retarget）后立即出现。
这两条链将从改变目标之后各自接受自己的难度。
新的共识规则拥有80%的挖矿算力。从这个链的角度来看，与上一周期相比，挖矿能力下降了20%。
区块将会平均每12分钟发现一次。这个块发行速度将持续下去（除非有任何改变哈希算力的因素出现）。
直到挖到第2016个区块，这将需要大约24,192分钟（每个块需要12分钟）或16.8天。
16.8天后，基于此链中哈希算力的减少，改变目标将再次发生，并将难度调整（减少20%）每10分钟产生一个区块。

The minority chain, mining under the old rules with only 20% of the hashing power, will face a much more difficult task. On this chain, blocks will now be mined every 50 minutes on average. The difficulty will not be adjusted for 2016 blocks, which will take 100,800 minutes, or approximately 10 weeks to mine. Assuming a fixed capacity per block, this will also result in a reduction of transaction capacity by a factor of 5, as there are fewer blocks per hour available to record transactions.
少数矿工认可的那条链，根据旧规则继续挖矿，现在只有20%的哈希算力，将面临更加艰巨的任务。
在这条链上，平均每隔50分钟挖出一个区块。
这个难度将不会在2016个块之前进行调整，这将需要100,800分钟，或大约10周的时间。
假设每个区块有固定容量，这也将导致交易容量减少5倍，因为每小时可用于记录交易的块大幅减少了。

10.13.4有争议的硬分叉

This is the dawn of consensus software development. Just as open source development changed both the methods and products of software and created new methodologies, new tools, and new communities in its wake, consensus software development also represents a new frontier in computer science. Out of the debates, experiments, and tribulations of the bitcoin development roadmap, we will see new development tools, practices, methodologies, and communities emerge.
这是共识软件开发的黎明。
正如开源开发改变了软件的方法和产品，创造了新的方法论，新工具和新社区，共识软件开发也代表了计算机科学的新前沿。
对比特币发展路线图的辩论、实验和纠结之中，我们将看到新的开发工具、实践、方法和社区的出现。

Hard forks are seen as risky because they force a minority to either upgrade or remain on a minority chain. The risk of splitting the entire system into two competing systems is seen by many as an unacceptable risk. As a result, many developers are reluctant to use the hard fork mechanism to implement upgrades to the consensus rules, unless there is near-unanimous support from the entire network. Any hard fork proposals that do not have near-unanimous support are considered too "contentious" to attempt without risking a partition of the system.
硬分叉被视为有风险，因为它们迫使少数人被迫选择升级或是必须保留在少数派链条上。
将整个系统分为两个竞争系统的风险被许多人认为是不可接受的风险。
结果，许多开发人员不愿使用硬分叉机制来实现对共识规则的升级，除非整个网络都能达成一致。
任何没有被所有人支持的硬分叉建议也被认为太“有争议”，以至于不能冒此风险。

The issue of hard forks is highly controversial in the bitcoin development community, especially as it relates to any proposed changes to the consensus rules controlling the maximum block size limit. Some developers are opposed to any form of hard fork, seeing it as too risky. Others see the mechanism of hard fork as an essential tool for upgrading the consensus rules in a way that avoids "technical debt" and provides a clean break with the past. Finally, some developers see hard forks as a mechanism that should be used rarely, with a lot of advance planning and only under near-unanimous consensus.
硬分叉的问题在比特币开发社区也是非常有争议，尤其是与控制区块大小限制相关的共识规则修改提议。
一些开发者反对任何形式的硬分叉，认为太冒险了。
另一些人认为硬分叉机制是提升共识规则的重要工具，避免了“技术债务”，与过去做一个干净的了断。
最后，有些开发者看到硬分叉作为一种应该很少使用的机制，应该经过认真的计划，并且只有在近乎一致的共识下才建议使用。

Already we have seen the emergence of new methodologies to address the risks of hard forks. In the next section, we will look at soft forks, and the BIP-34 and BIP-9 methods for signaling and activation of consensus modifications.
我们已经看到出现了新的方法来解决硬分叉的危险。
在下一节中，我们将看看软分叉，以及BIP-34和BIP-9方法，它们用于告知和激活共识修改。

10.13.5软分叉

Not all consensus rule changes cause a hard fork. Only consensus changes that are forward-incompatible cause a fork. If the change is implemented in such a way that an unmodified client still sees the transaction or block as valid under the previous rules, the change can happen without a fork.
并非所有的共识规则修改都会导致硬分叉，只有前向不兼容的共识规则的变化才会导致分叉。
如果共识规则修改也能够让未升级的客户端仍然按照先前的规则对待交易或区块，那么就可以在不进行分叉的情况下实现共识修改。这就是软分叉。

The term *soft fork* was introduced to distinguish this upgrade method from a "hard fork." In practice, a soft fork is not a fork at all. A soft fork is a forward-compatible change to the consensus rules that allows unupgraded clients to continue to operate in consensus with the new rules.
实际上，软分叉不是分叉。
软分叉是与共识规则的前向兼容并作些变化，允许未升级的客户端程序继续与新规则同时工作。

One aspect of soft forks that is not immediately obvious is that soft fork upgrades can only be used to constrain the consensus rules, not to expand them. In order to be forward compatible, transactions and blocks created under the new rules must be valid under the old rules too, but not vice versa. The new rules can only limit what is valid; otherwise, they will trigger a hard fork when rejected under the old rules.
软分叉的一个不是很明显的方面就是，软分叉升级只能用于约束共识规则，而不能扩展共识规则。
为了向前兼容，根据新规则创建的交易和区块也必须在旧规则下有效，但是反过来不行。
新规则只能增加限制，否则，当就规则拒绝它们时，还会产生硬分叉。

Soft forks can be implemented in a number of ways—the term does not specify a particular method, rather a set of methods that all have one thing in common: they don’t require all nodes to upgrade or force nonupgraded nodes out of consensus.
软分叉可以通过多种方式实现，这个术语并不指一个特定的方法，而是指一套方法，它们都有一个共同点：
它们不要求所有节点都需要升级，或强制未升级节点脱离共识。

10.13.5.1软分叉：重新定义NOP操作码

A number of soft forks have been implemented in bitcoin, based on the re-interpretation of NOP opcodes. Bitcoin Script had ten opcodes reserved for future use, NOP1 through NOP10. Under the consensus rules, the presence of these opcodes in a script is interpreted as a null-potent operator, meaning they have no effect. Execution continues after the NOP opcode as if it wasn’t there.
基于对NOP操作码的重新解释，比特币中已经实现了一些软分叉。
比特币脚本有10个操作码保留供将来使用：NOP1到NOP10。
根据共识规则，这些操作码在脚本中被解释为无效的操作码，这意味着它们没有任何效果。
在NOP操作码之后继续执行，就好像它不存在一样。
因此，软分叉可以修改NOP操作码的语义，给它新的含义。

A soft fork therefore can modify the semantics of a NOP code to give it new meaning. For example, BIP-65 (CHECKLOCKTIMEVERIFY) reinterpreted the NOP2 opcode. Clients implementing BIP-65 interpret NOP2 as OP_CHECKLOCKTIMEVERIFY and impose an

absolute locktime consensus rule on UTXO that contain this opcode in their locking scripts. This change is a soft fork because a transaction that is valid under BIP-65 is also valid on any client that is not implementing (ignorant of) BIP-65. To the old clients, the script contains an NOP code, which is ignored.
例如，BIP-65（CHECKLOCKTIMEVERIFY）重新解释了NOP2操作码。
实施BIP-65的客户将NOP2解释为OP_CHECKLOCKTIMEVERIFY，并在其锁定脚本中包含该操作码的UTXO上，强制了绝对锁定的共识规则。
这种变化是一个软分叉，因为在BIP-65下有效的交易在任何没有实现BIP-65的客户端上也是有效的。
对于旧的客户端，该脚本包含一个NOP代码，它被忽略。

10.13.5.2软分叉升级的其它方式

The reinterpretation of NOP opcodes was both planned for and an obvious mechanism for consensus upgrades. Recently, however, another soft fork mechanism was introduced that does not rely on NOP opcodes for a very specific type of consensus change. This is examined in more detail in [segwit]. Segwit is an architectural change to the structure of a transaction, which moves the unlocking script (witness) from inside the transaction to an external data structure (segregating it). Segwit was initially envisioned as a hard fork upgrade, as it modified a fundamental structure (transaction). In November 2015, a developer working on Bitcoin Core proposed a mechanism by which segwit could be introduced as a soft fork. The mechanism used for this is a modification of the locking script of UTXO created under segwit rules, such that unmodified clients see the locking script as redeemable with any unlocking script whatsoever. As a result, segwit can be introduced without requiring every node to upgrade or split from the chain: a soft fork.
NOP操作码的重新解释既是对共识升级的一个规划机制，也是一个明显机制。
然而，最近，引入了另一种软分叉机制，它不依赖于NOP操作码，用于非常特定类型的共识改变。
这在隔离见证（segwit）中有更详细的解释。
segwit是交易结构的一个体系结构变化，它将解锁脚本（见证）从交易内部移到外部数据结构。
segwit最初被设想为硬分叉升级，因为它修改了一个基本的结构（交易）。
2015年11月，一个Bitcoin Core开发人员提出了一种机制，通过这种机制可以用软分叉引入segwit。
这个机制是修改了segwit规则下创建的UTXO的锁定脚本，使未修改的客户端把锁定脚本看做可以使用任何解锁脚本都可以赎回的。
这样，segwit不要求每个节点都升级，或者与链分离，所以这是一个软分叉。

It is likely that there are other, yet to be discovered, mechanisms by which upgrades can be made in a forward-compatible way as a soft fork.
有可能还有其它尚未发现的机制，可以以向前兼容的方式（软分叉）进行升级。

10.13.6对软分叉的批评

Soft forks based on the NOP opcodes are relatively uncontroversial. The NOP opcodes were placed in Bitcoin Script with the explicit goal of allowing non-disruptive upgrades.
基于NOP操作码的软分叉相对没有争议。
NOP操作码被放置在比特币脚本中，明确的目标是允许无中断升级。

However, many developers are concerned that other methods of soft fork upgrades make unacceptable tradeoffs. Common criticisms of soft fork changes include:
但是，许多开发人员担心软分叉升级的其它方法会产生不可接受的权衡。
对软分叉修改的常见批评包括如下。

Technical debt 技术债务
Because soft forks are more technically complex than a hard fork upgrade, they introduce *technical debt*, a term that refers to increasing the future cost of code

maintenance because of design tradeoffs made in the past. Code complexity in turn increases the likelihood of bugs and security vulnerabilities.
由于软叉在技术上比硬分叉升级更复杂，所以引入了技术债务，这是指由于过去的设计权衡而增加代码维护的未来成本。代码复杂性又增加了错误和安全漏洞的可能性。

Validation relaxation 验证放松
Unmodified clients see transactions as valid, without evaluating the modified consensus rules. In effect, the unmodified clients are not validating using the full range of consensus rules, as they are blind to the new rules. This applies to NOP-based upgrades, as well as other soft fork upgrades.
未修改的客户端将交易视为有效，而不评估修改的共识规则。
实际上，未修改的客户端不会使用新规则来验证，因为它们不了解新规则。
这适用于基于NOP的升级，以及其它软分叉升级。

Irreversible upgrades 不可逆转的升级
Because soft forks create transactions with additional consensus constraints, they become irreversible upgrades in practice. If a soft fork upgrade were to be reversed after beings activated, any transactions created under the new rules could result in a loss of funds under the old rules. For example, if a CLTV transaction is evaluated under the old rules, there is no timelock constraint and it can be spent at any time. Therefore, critics contend that a failed soft fork that had to be reversed because of a bug would almost certainly lead to loss of funds.
因为软分叉创建了有额外共识约束的交易，所以它们在实际中成为不可逆转的升级。
如果软分叉升级在激活后被回退，根据新规则创建的任何交易都可能导致旧规则下的资金损失。
例如，如果根据旧规则对CLTV交易进行评估，则不存在任何时间锁定约束，并且可以花费在任何时间。
因此，批评者认为，由于错误而不得不被回退的失败的软分叉，几乎肯定会导致资金的损失。

10.14使用“区块版本”作软分叉信号

Since soft forks allow unmodified clients to continue to operate within consensus, the mechanism for "activating" a soft fork is through miners signaling readiness: a majority of miners must agree that they are ready and willing to enforce the new consensus rules. To coordinate their actions, there is a signaling mechanism that allows them to show their support for a consensus rule change. This mechanism was introduced with the activation of BIP-34 in March 2013 and replaced by the activation of BIP-9 in July 2016.
由于软分叉允许未修改的客户在共识内继续运行，“激活”软分叉的机制是通过向矿工发出信号准备：大多数矿工必须同意，他们准备并愿意执行新的共识规则。
为了协调他们的行动，有一个信号机制，使他们能够表达他们对共识规则改变的支持。
该机制是在2013-3激活BIP-34时引入，并在2016-7月激活BIP-9时所取代。

10.14.1 BIP-34信号和激活

The first implementation, in BIP-34, used the block version field to allow miners to signal readiness for a specific consensus rule change. Prior to BIP-34, the block version was set to "1" by *convention* not enforced by *consensus*.
BIP-34中的第一个实现使用“区块版本”字段来允许矿工表达对特定共识规则改变的意见。
在BIP-34之前，按照惯例将块版本设定为“1”，而不是以共识方式执行。

BIP-34 defined a consensus rule change that required the coinbase field (input) of the coinbase transaction to contain the block height. Prior to BIP-34, the coinbase could contain any arbitrary data the miners chose to include. After activation of BIP-34, valid blocks had to contain a specific block-height at the beginning of the coinbase and be identified with a version number greater than or equal to "2."
BIP-34定义了一个共识规则变更，要求将币基交易的coinbade字段（输入）包含区块高度。
在BIP-34之前，Coinbase可以让矿工选择包含的任意数据。
在BIP-34激活之后，有效区块必须在Coinbase的开始处包含特定的区块高度，并且使用大于或等于“2”的版本号进行标识。

To signal the change and activation of BIP-34, miners set the block version to "2," instead of "1." This did not immediately make version "1" blocks invalid. Once activated, version "1" blocks would become invalid and all version "2" blocks would be required to contain the block height in the coinbase to be valid.
为了标记BIP-34的更改和激活，矿工们将区块版本设置为“2”而不是“1”。
这没有立即使版本“1”区块无效。
一旦激活，版本“1”区块将变得无效，并且将需要所有版本“2”区块以包含Coinbase中的区块高度才能有效。

BIP-34 defined a two-step activation mechanism, based on a rolling window of 1000 blocks. A miner would signal his or her individual readiness for BIP-34 by constructing blocks with "2" as the version number. Strictly speaking, these blocks did not yet have to comply with the new consensus rule of including the block-height in the coinbase transaction because the consensus rule had not yet been activated. The consensus rules activated in two steps:
BIP-34基于1000个块的滚动窗口定义了两步启动机制。
矿工将以“2”作为版本号来构建区块，从表示为BIP-34做好了准备。
严格来说，由于共识规则尚未被激活，这些区块还没有遵守新的共识规则，也就是将区块高度包括在币基交易中。
激活共识规则分为两个步骤：

- If 75% (750 of the most recent 1000 blocks) are marked with version "2," then version "2" blocks must contain block height in the coinbase transaction or they are

rejected as invalid. Version "1" blocks are still accepted by the network and do not need to contain block-height. The old and new consensus rules coexist during this period.
当75%（最近1000个块中的750个）标有版本“2”，则版本“2”区块必须在币基交易中包含区块高度，否则被认为无效。
版本“1”区块仍然被网络接受，不需要包含区块高度。在这个时期，新旧共识规则共存。

- When 95% (950 of the most recent 1000 blocks) are version "2," version "1" blocks are no longer considered valid. Version "2" blocks are valid only if they contain the block-height in the coinbase (as per the previous threshold). Thereafter, all blocks must comply with the new consensus rules, and all valid blocks must contain block-height in the coinbase transaction.
当95%（最近1000块中的950）是版本“2”时，版本“1”区块不再被视为有效。
版本“2”区块只有当它们在币基中包含区块高度时才有效。
此后，所有块必须符合新的共识规则，所有有效块必须在币基交易中包含区块高度。

After successful signaling and activation under the BIP-34 rules, this mechanism was used twice more to activate soft forks:
- BIP-66 Strict DER Encoding of Signatures was activated by BIP-34 style signaling with a block version "3" and invalidating version "2" blocks.
- BIP-65 CHECKLOCKTIMEVERIFY was activated by BIP-34 style signaling with a block version "4" and invalidating version "3" blocks.

根据BIP-34规则，成功发出信号和激活后，该机制又被使用了两次以激活下列软分叉：
- BIP-66通过BIP-34信号激活，区块版本为“3”，使版本“2”区块失效。
- BIP-65通过BIP-34信号激活，区块版本为“4”，使版本“3”区块失效。

After the activation of BIP-65, the signaling and activation mechanism of BIP-34 was retired and replaced with the BIP-9 signaling mechanism described next.
The standard is defined in BIP-34 (Block v2, Height in Coinbase).
BIP-65激活后，BIP-34的信号和激活机制退出，并用BIP-9信号机制代替。
BIP-34: Block v2, Height in Coinbase

10.14.2 BIP-9信号和激活

The mechanism used by BIP-34, BIP-66, and BIP-65 was successful in activating three soft forks. However, it was replaced because it had several limitations:
- By using the integer value of the block version, only one soft fork could be activated at a time, so it required coordination between soft fork proposals and agreement on their prioritization and sequencing.
- Furthermore, because the block version was incremented, the mechanism didn’t offer a straightforward way to reject a change and then propose a different one. If old clients were still running, they could mistake signaling for a new change as signaling for the previously rejected change.
- Each new change irrevocably reduced the available block versions for future changes.

BIP-34，BIP-66和BIP-65使用的机制成功地激活了三个软分叉。
然而，它又被替换了，因为它有几个限制：
- 通过使用区块版本的整数值，一次只能激活一个软分叉，因此需要软分叉建议之间的协调，以及对其优先级和排序进行协商。
- 此外，由于区块版本增加，所以这个机制并没有提供一种直接的方式来拒绝变更，而是提出一个不同的方法。如果老客户仍然在运行，他们可能会错误地将信号转换为新的更改，作为以前拒绝的更改的信号。
- 每个新的更改不可逆转地减少可用的供将来更改的区块版本。

BIP-9 was proposed to overcome these challenges and improve the rate and ease of implementing future changes.
BIP-9克服了这些挑战，提高了实施未来变更的速度和便利性。

BIP-9 interprets the block version as a bit field instead of an integer. Because the block version was originally used as an integer, versions 1 through 4, only 29 bits remain available to be used as a bit field. This leaves 29 bits that can be used to independently and simultaneously signal readiness on 29 different proposals. BIP-9 also sets a maximum time for signaling and activation. This way miners don't need to signal forever. If a proposal is not activated within the TIMEOUT period (defined in the proposal), the proposal is considered rejected. The proposal may be resubmitted for signaling with a different bit, renewing the activation period.
BIP-9将区块版本解释为比特字段，而不是整数字段。
因为区块版本最初作为整数，版本1到4，只有29位仍可用作比特字段。
这剩下的29位，可以独立使用，同时对29个不同的提案表示是否支持。
BIP-9还设置了信号和激活的最大时间。矿工们不需要永远发出信号。
如果提案在TIMEOUT期间（在提案中定义）未激活，则该提案被视为被拒绝。
该提议可以使用不同位的信号重新提交，更新激活周期。

Furthermore, after the TIMEOUT has passed and a feature has been activated or rejected, the signaling bit can be reused for another feature without confusion. Therefore, up to 29 changes can be signaled in parallel and after TIMEOUT the bits can be "recycled" to propose new changes.
此外，在TIMEOUT过去之后，一个特征被激活或拒绝，信号位可用于另一个特征，而不会混淆。
因此，最多可以有29个更改并行发出信号，TIMEOUT后可将这些位再用于新的更改。

Note：While signaling bits can be reused or recycled, as long as the voting period does not overlap, the authors of BIP-9 recommend that bits are reused only when necessary; unexpected behavior could occur due to bugs in older software. In short, we should not expect to see reuse until all 29 bits have been used once.
注意：虽然信号位可以重复使用或回收利用，但在投票期间不能重叠，BIP-9的作者就建议仅在必要时重复使用位；主要是由于旧软件中的bug，可能会发生意外的行为。
总之，我们不应该期望在所有29位都被使用一次之前看到重用。

Proposed changes are identified by a data structure that contains the following fields:
建议的更改由一个数据结构标识，它有包含下列字段：

name

A short description used to distinguish between proposals. Most often the BIP describing the proposal, as "bipN," where N is the BIP number.

名称：用于区分提案的简短描述。通常，BIP将该提案描述为“bipN”，其中N是BIP编号。

bit

0 through 28, the bit in the block version that miners use to signal approval for this proposal.

位：0到28，矿工使用的区块版本中的位用于表示此提案的支持。

starttime

The time (based on Median Time Past, or MTP) that signaling starts after which the bit's value is interpreted as signaling readiness for the proposal.

开始时间：信号开始的时间（基于中值时间），之后该位的值被解释为提示的信号准备。

endtime

The time (based on MTP) after which the change is considered rejected if it has not reached the activation threshold.

结束时间：该时间（基于中值时间），如果尚未达到激活阈值，则认为该更改被拒绝。

Unlike BIP-34, BIP-9 counts activation signaling in whole intervals based on the difficulty retarget period of 2016 blocks. For every retarget period, if the sum of blocks signaling for a proposal exceeds 95% (1916 of 2016), the proposal will be activated one retarget period later.
BIP-9 offers a proposal state diagram to illustrate the various stages and transitions for a proposal, as shown in BIP-9 state transition diagram.
与BIP-34不同，BIP-9根据2016个区块的难度改变目标（retarget）周期，在整个间隔周期中计算激活信号的数量。在每个难度调整周期内，如果提案的激活信号数量总和超过95%（2016中的1916），则该提案将在下一个难度调整周期内激活。
BIP-9提供了一个提案状态图，以说明一个提案的各个阶段和转换，如图10所示。

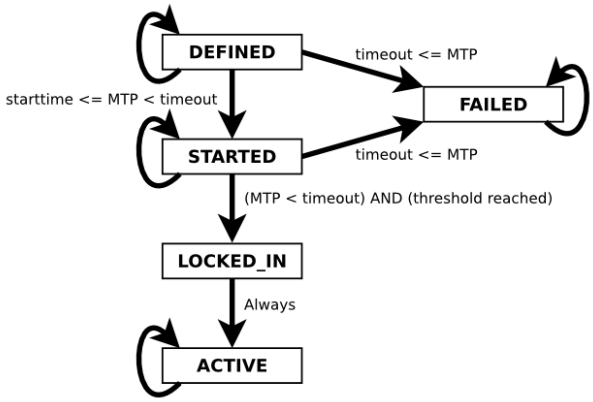

Figure 10. BIP-9 state transition diagram
图10：BIP-9状态转换图

Proposals start in the DEFINED state, once their parameters are known (defined) in the bitcoin software. For blocks with MTP after the start time, the proposal state transitions to STARTED. If the voting threshold is exceeded within a retarget period and the timeout has not been exceeded, the proposal state transitions to LOCKED_IN. One retarget period later, the proposal becomes ACTIVE. Proposals remain in the ACTIVE state perpetually once they reach that state. If the timeout elapses before the voting threshold has been reached, the proposal state changes to FAILED, indicating a rejected proposal. FAILED proposals remain in that state perpetually.
一旦它们的参数在比特币软件中被知道（定义），提案将在DEFINED状态下开始。
对于具有MTP的区块在开始时间之后，提议状态转换到STARTED。
如果在改变目标期间超过了投票阈值，并且未超过超时，则提案状态转换为LOCKED_IN。
一个改变目标期后，该提案变为活动。
一旦达到这个状态，提案仍然处于活动状态。
如果在达到投票阈值之前超时时间，提案状态将更改为“已故”，表示已拒绝的提案。 REJECTED的建议永远在这个状态。

BIP-9 was first implemented for the activation of CHECKSEQUENCEVERIFY and associated BIPs (68, 112, 113). The proposal named "csv" was activated successfully in July of 2016. The standard is defined in BIP-9 (Version bits with timeout and delay).
BIP-9首次实施用于激活CHECKSEQUENCEVERIFY和相关BIP（68,112,113）。
名为“csv”的建议在2016年7月成功激活。
BIP-9 ： Version bits with timeout and delay

10.15共识软件开发

Consensus software continues to evolve and there is much discussion on the various mechanisms for changing the consensus rules. By its very nature, bitcoin sets a very high bar on coordination and consensus for changes. As a decentralized system, it has no "authority" that can impose its will on the participants of the network. Power is diffused between multiple constituencies such as miners, core developers, wallet developers, exchanges, merchants, and end users. Decisions cannot be made unilaterally by any of these constituencies. For example, while miners can theoretically change the rules by simple majority (51%), they are constrained by the consent of the other constituencies. If they act unilaterally, the rest of the participants may simply refuse to follow them, keeping the economic activity on a minority chain. Without economic activity (transactions, merchants, wallets, exchanges), the miners will be mining a worthless coin with empty blocks. This diffusion of power means that all the participants must coordinate, or no changes can be made. Status quo is the stable state of this system with only a few changes possible if there is strong consensus by a very large majority. The 95% threshold for soft forks is reflective of this reality.
共识软件开发不断发展，对于改变共识规则的各种机制进行了很多讨论。
按照其本质，比特币在协调和改变共识方面树立了非常高的标杆。
作为一个去中心化的机制，不存在将权力强加于网络参与者的“权威”。
权力分散在多个支持者，如矿工、核心开发者、钱包开发者、交易所、商家和用户之间。
这些支持者不能单方面做出决定。
例如，矿工在理论上可以通过简单多数（51%）来改变规则，但受到其他支持者的同意的限制。
如果他们单方面行事，其他参与者可能会拒绝遵守，将经济活动保持在少数链条上。
没有经济活动（交易，商人，钱包，交易所），矿工们将用空的区块开采一个无价值的货币。
这种权力的扩散意味着所有参与者必须协调，或者不能做出任何改变。
现状是这个机制的稳定状态，只有在很大程度上达成一致的情况下，才能有几个变化。
软分叉的95%门槛反映了这一现实。

It is important to recognize that there is no perfect solution for consensus development. Both hard forks and soft forks involve tradeoffs. For some types of changes, soft forks may be a better choice; for others, hard forks may be a better choice. There is no perfect choice; both carry risks. The one constant characteristic of consensus software development is that change is difficult and consensus forces compromise.
重要的是要认识到，对于共识发展没有完美的解决办法。
硬分叉和软分叉都涉及权衡。
对于某些类型的更改，软分叉可能是一个更好的选择；对于别人来说，硬分叉可能是一个更好的选择。
没有完美的选择，都带有风险。
共识软件开发的一个不变特征是变革是困难的，是共识力量的妥协。

Some see this as a weakness of consensus systems. In time, you may come to see it as I do, as the system’s greatest strength.
有些人认为这是共识制度的弱点。
随着时间的推移，你可能会像我一样，把它看做系统最强大的力量。