How does Mining work?

In Bitcoin, mining is the term commonly used for creating blocks and minting new coins. Mining is a main part of the innovation of Bitcoin because it creates a balanced financial incentive for users to support, rather than attack, the network. It also provides a mechanism whereby participants in the network can all agree on the current state of the coin supply.

Creating a new block

Miners work on new blocks and submit their solutions to the network for validation. In order to create a new block, and claim the reward, the miner must submit a block that meets the following criteria (among others):

- The block hash must be less than or equal to the current difficulty target
- Timestamp must be valid
 - must be greater than the median time of the last 11 blocks
 - o cannot be greater than 2 hours in the future
- · Transactions must be valid
 - o no double spends
 - o spends are all authorized, i.e. scriptSigs (plus any witness data) satisfy the scriptPubKeys
 - the first tx is a valid coinbase transaction

For a list of other rules see Protocol Rules

Finding a valid block hash

This step is the process commonly referred to as mining. This is where the most work is done, and provides the basis for a secure network. Let's start by looking at the genesis block again:

```
$ bitcoin-cli -regtest getblockhash 0
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206
```

Block Header:

Block Hash: 0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206

The block hash is created by taking the block header, and providing it as an input to the SHA256 hashing algorithm. The nonce is continually changed until a valid solution is found. Miners are, quite literally, optimized guessing machines hoping the next random nonce will result in a valid hash.

Hash Functions

Hash functions are one-way functions, meaning they can only be computed in one direction. You can easily hash an input, but you cannot calculate an input that will result in a specific output. In order to do so, you have to keep providing a new random input until the solution is found. In Bitcoin, this is also known as work, the number of guesses, on average, that a miner must make in order to calculate a valid hash.

To validate the hash, use the provided script hash256:

\$ hash256

result: 06226e46111a0b59caaf126043eb5bbf28c34f3a5e332a1fc7b2b73cf188910f

\$ reverse_endian 06226e46111a0b59caaf126043eb5bbf28c34f3a5e332a1fc7b2b73cf188910f
reversed 0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206

Mining Example Now say we want to hit a lower target (higher difficulty). The current target can be calculated by converting the nBits field from the block header:

\$ reverse_endian 2000ffff
ffff0020

Now the header becomes:

nonce: 02000018 nonce: 02000019 nonce: 0200001a nonce: 0200001b nonce: 0200001c

Now, lets use cpp_miner to mine the block, we'll turn on verbose mode to see what's happening under the hood:

\$ cpp miner mine Verbosity enabled. Level is 1 nonce: 02000001 nonce: 02000002 nonce: 02000003 nonce: 02000004 nonce: 02000005 nonce: 02000006 nonce: 02000007 nonce: 02000008 nonce: 02000009 nonce: 0200000a nonce: 0200000b nonce: 0200000c nonce: 0200000d nonce: 0200000e nonce: 0200000f nonce: 02000010 nonce: 02000011 nonce: 02000012 nonce: 02000013 nonce: 02000014 nonce: 02000015 nonce: 02000016 nonce: 02000017

nonce: 0200001d nonce: 0200001e nonce: 0200001f nonce: 02000020 nonce: 02000021 nonce: 02000022 nonce: 02000023 nonce: 02000024 nonce: 02000025 nonce: 02000026 nonce: 02000027 nonce: 02000028 nonce: 02000029 nonce: 0200002a nonce: 0200002b nonce: 0200002c nonce: 0200002d nonce: 0200002e nonce: 0200002f nonce: 02000030 nonce: 02000031 nonce: 02000032 nonce: 02000033 nonce: 02000034 nonce: 02000035 nonce: 02000036 nonce: 02000037 nonce: 02000038 nonce: 02000039 nonce: 0200003a nonce: 0200003b nonce: 0200003c nonce: 0200003d nonce: 0200003e nonce: 0200003f nonce: 02000040 nonce: 02000041 nonce: 02000042 nonce: 02000043 nonce: 02000044 nonce: 02000045 nonce: 02000046 nonce: 02000047 nonce: 02000048 nonce: 02000049 nonce: 0200004a nonce: 0200004b nonce: 0200004c nonce: 0200004d nonce: 0200004e nonce: 0200004f nonce: 02000050 nonce: 02000051 nonce: 02000052 nonce: 02000053 nonce: 02000054 nonce: 02000055 nonce: 02000056 nonce: 02000057 nonce: 02000058 nonce: 02000059 nonce: 0200005a nonce: 0200005b nonce: 0200005c nonce: 0200005d nonce: 0200005e nonce: 0200005f nonce: 02000060 nonce: 02000061 nonce: 02000062

nonce: 02000063 nonce: 02000064 nonce: 02000065 nonce: 02000066 nonce: 02000067 nonce: 02000068 nonce: 02000069 nonce: 0200006a nonce: 0200006b nonce: 0200006c nonce: 0200006d nonce: 0200006e nonce: 0200006f nonce: 02000070 nonce: 02000071 nonce: 02000072 nonce: 02000073 nonce: 02000074 nonce: 02000075 nonce: 02000076 nonce: 02000077 nonce: 02000078 nonce: 02000079 nonce: 0200007a nonce: 0200007b nonce: 0200007c nonce: 0200007d nonce: 0200007e nonce: 0200007f nonce: 02000080 nonce: 02000081 nonce: 02000082 nonce: 02000083 nonce: 02000084 nonce: 02000085 nonce: 02000086 nonce: 02000087 nonce: 02000088 nonce: 02000089 nonce: 0200008a nonce: 0200008b nonce: 0200008c nonce: 0200008d nonce: 0200008e nonce: 0200008f nonce: 02000090 nonce: 02000091 nonce: 02000092 nonce: 02000093 nonce: 02000094 nonce: 02000095 nonce: 02000096 nonce: 02000097 nonce: 02000098 nonce: 02000099 nonce: 0200009a nonce: 0200009b nonce: 0200009c nonce: 0200009d nonce: 0200009e nonce: 0200009f nonce: 020000a0 nonce: 020000a1 nonce: 020000a2 nonce: 020000a3 nonce: 020000a4 nonce: 020000a5 nonce: 020000a6 nonce: 020000a7 nonce: 020000a8 nonce: 020000a9 nonce: 020000aa nonce: 020000ab nonce: 020000ac nonce: 020000ad nonce: 020000ae nonce: 020000af nonce: 020000b0 nonce: 020000b1 nonce: 020000b2 nonce: 020000b3 nonce: 020000b4 nonce: 020000b5 nonce: 020000b6 nonce: 020000b7 nonce: 020000b8 nonce: 020000b9 nonce: 020000ba nonce: 020000bb nonce: 020000bc nonce: 020000bd nonce: 020000be nonce: 020000bf nonce: 020000c0 nonce: 020000c1 nonce: 020000c2 nonce: 020000c3 nonce: 020000c4 nonce: 020000c5 nonce: 020000c6 nonce: 020000c7 nonce: 020000c8 nonce: 020000c9 nonce: 020000ca nonce: 020000cb nonce: 020000cc nonce: 020000cd nonce: 020000ce nonce: 020000cf nonce: 020000d0 nonce: 020000d1 nonce: 020000d2 nonce: 020000d3 nonce: 020000d4 nonce: 020000d5 nonce: 020000d6 nonce: 020000d7 nonce: 020000d8 nonce: 020000d9 nonce: 020000da nonce: 020000db nonce: 020000dc nonce: 020000dd nonce: 020000de nonce: 020000df nonce: 020000e0 nonce: 020000e1 nonce: 020000e2 nonce: 020000e3 nonce: 020000e4 nonce: 020000e5 nonce: 020000e6 nonce: 020000e7 nonce: 020000e8 nonce: 020000e9 nonce: 020000ea nonce: 020000eb nonce: 020000ec nonce: 020000ed nonce: 020000ee nonce: 020000ef
block header:

nonce: 0x020000ef

 $\verb|block| hash: 00a46955ac128fb7b0e0f4758dadb97dfd94e465639333b5b4de770a0928b446| \\$

٠,

And we finally found a valid block hash!