Table of contents

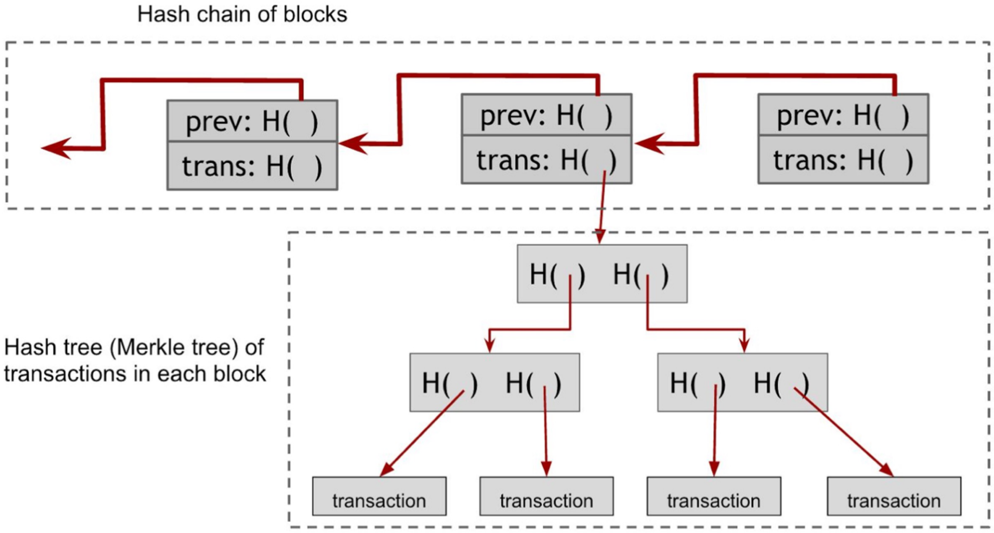
- Bitcoin miners
- Mining hardware
- Mining pool
- Changing protocol

Bitcoin miners

- The task of Bitcoin miners
  - Maintain blockchain
    - Validate Tx
      - Listen for new Tx
      - Assemble a candidate block
      - Find a valid nonce for the candidate block
        - Broadcast
      - Hope block is accepted
        - Profit
    - Validate blocks from the network
      - Listen for new blocks
      - Validate these blocks

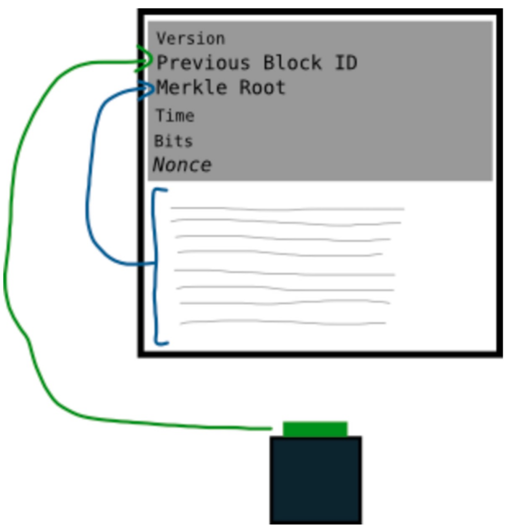
Bitcoin miners

- Bitcoin block
  - Bitcoin blockchain contains two different hash structures
    - Hash chain of blocks
    - Merkle tree of Tx within blocks



Bitcoin miners

- Bitcoin block
  - Block header is a summary (metadata) of the data in the block



Bitcoin miners

- Bitcoin block

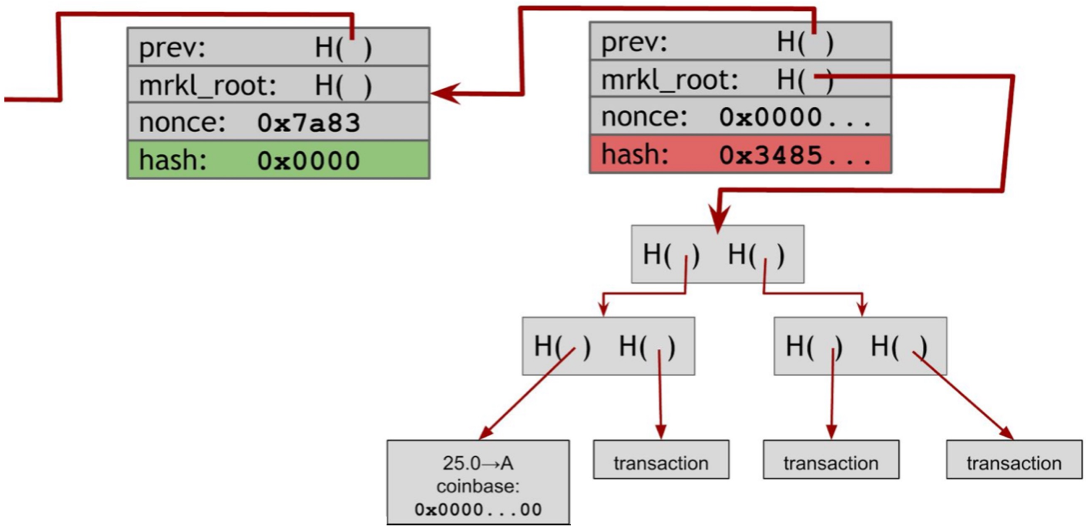
Field	Description
Version	Bitcoin version/rules-set
Previous Block ID	Hash of previous block, where chain is extending
Merkle Root	All Tx in this block hashed (single-line summary of all Tx) together
Time	Unix time, when a miner is trying to mine (hashing) this block the
Bits	A shortened version of target
Nonce	Miners change this 4 bytes (32 bits; 2^{32} possibilities starting from 0) value to get a hash of block header (block hash) below target

Bitcoin miners

- Finding a valid block
  - Miners
    - Identify latest (previous) block
    - Compute the merkle root
      - Pick a set of pending Tx
        - Take care of max size of block
        - Validate these Tx
      - Compile a coinbase Tx
    - Compute target
    - Pick a nonce value
    - Hash the block
      - Check if hash satisfies target
        - If yes, broadcast
        - Otherwise, try again with a new nonce value

Bitcoin miners

- Finding a valid block



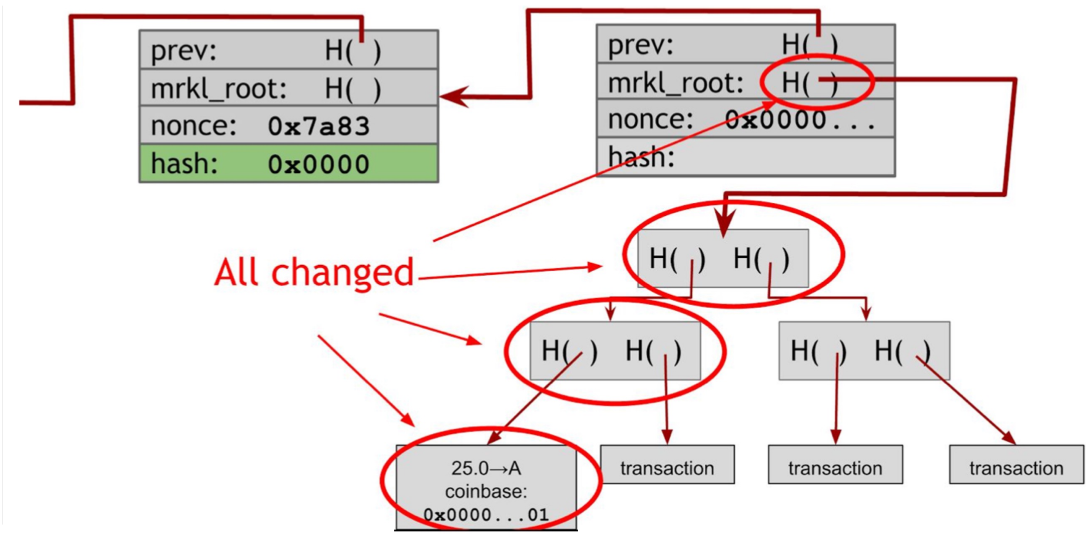
Bitcoin miners

- Finding a valid block
  - What if we exhaust nonce values?

  - Changeable elements in block header
    - Nonce
      - We just exhausted nonce values
    - Time (rollnTime in getwork)
      - Limited scalability
    - Merkle root

Bitcoin miners

- Change merkle root
  - Change Tx, costly
  - Change coinbase Tx

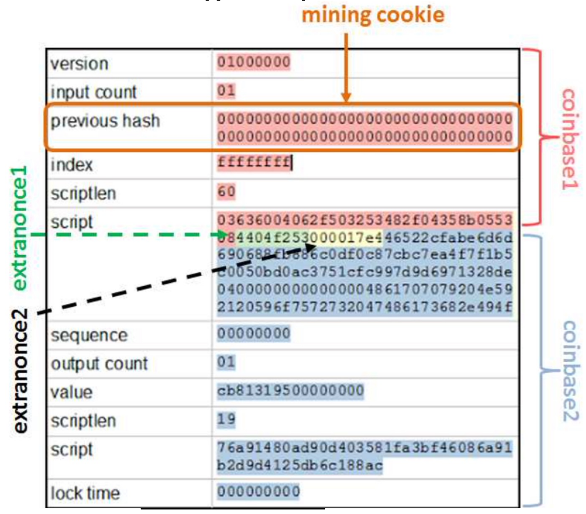


Bitcoin miners

- Change coinbase Tx
  - In regular Tx
    - Take scriptPubKey from previous Tx and scriptSig from current Tx
  - Coinbase Tx has single blank input
    - No reference to previous output with scriptPubKey
    - scriptSig in current coinbase Tx can be arbitrary bytes
      - Of course, avoiding caveats (*OP_CHECKSIG*)
  - extraNonce solution
  - extraNonce isn't part of the protocol
    - There is no extraNonce field in blocks or Tx

Bitcoin miners

- Change coinbase Tx
  - coinBase1/2 represent input/output (overlapping)
  - Change extraNonce (extraNonce1 || extraNonce2)
    - extraNonce1 is unique (pseudo-random) in Stratum (we'll see it later)
    - Only option is to change 4 bytes extraNonce2



Bitcoin miners

- Change coinbase Tx
  - Overall increment/change extraNonce
  - Coinbase Tx = coinBase1 || extraNonce1 || extraNonce2 || coinBase2
  - Compute H(coinbase Tx)
  - Rebuild merkle root
  - Reset nonce, recompute hash

Bitcoin miners

- Finding a valid block
  - Is everyone solving the same puzzle?
    - No!
    - Miners choose Tx and build candidate block independently
    - What if different miners choose identical Tx?
      - Blocks would still differ
        - Miners specify their own address in coinbase Tx
    - What if they share the address?
      - Pooled mining
        - Try different ranges of nonce, avoid duplicate work

Bitcoin miners

- When you create and broadcast a Tx
  - It goes to a node
  - Node stores/queues it in its mempool
    - Broadcast further
  - Tx reside in mempool until miner picks it
    - Prioritize tx based on fees/age
      - More fee, more chances to be picked
- What if miner never picks it?
  - May be you paid a very low/no fee
    - Miner is least interested to pick it
      - Can I increase fee?
        - Replace By Fee (RBF)

Bitcoin miners

- RBF
  - A node policy
  - Allows an unconfirmed Tx in a mempool to be replaced with a different Tx
    - Spends at least one of same inputs
    - Pays a higher Tx fee

Bitcoin miners

- RBF
  - Full RBF
    - Unconditionally allows a Tx to replace older ones as long as it pays a sufficient fee
  - Opt-in RBF
    - Only allows replacement when Tx being replaced have explicitly signaled to allow replacement
      - Via nSequence/sequence field
      - Both full and opt-in RBF may lead to double spend

Bitcoin miners

- RBF
  - First-seen-safe RBF
    - Only allows replacement if replacement/new Tx keeps same outputs as Tx being replaced
  - Delayed RBF
    - Allows Tx to be replaced unconditionally, but only after a given number of blocks have been mined since Tx being replaced were first seen by node

Bitcoin miners

- RBF
  - nSequence
    - 4 bytes field in an Tx input
    - Usually only shown in rawTx
    - Send a Tx with a lock_time & sequence number 0
      - Tx is then not considered "final" by network
        - Can't include in a block until lock_time
    - Before lock_time expires
      - Replace Tx with as many new versions as you want
      - Newer versions have higher sequence numbers
    - To lock Tx permanently
      - Set sequence number to 0xFFFFFFFF
      - Tx is considered final, even if lock_time has not reached

Bitcoin miners

- Orphan Tx
  - When Tx are transmitted across network
    - May arrive to nodes in different order
      - A child might arrive before parent
  - Upon seeing a Tx (child)
    - That references a Tx (parent) that is not yet known
    - Nodes don't reject child
      - Put it in temporary orphan Tx pool
      - Where orphan Tx wait for parent's arrival
  - When parent arrives
    - Child Tx is retrieved, revalidated recursively, inserted in mempool

Bitcoin miners

- Orphan block
  - When a fork occurs
    - A block may not be included in main chain
      - Block becomes orphaned
  - All Tx in it simply go back in node's queue/pool
  - Revalidated, if their input has been spent
    - If so, evicted
  - Wait to be added to next block

Mining hardware

- Computing SHA-256 hash function is the core of miners' tasks
- CPU mining
  - First generation mining was done on general purpose CPUs
  - Simply compute SHA-256 in software, check the result
  - Speed?
    - ~10-20 MH/s
    - Mining on a regular CPU at current difficulty level is impossible

Mining hardware

- GPU mining
  - Designed for processing videos/graphics
    - Large number of floating point units
      - Not used at all in SHA-256
  - Poor cooling characteristics (especially, in farms)
  - High electricity computation
  - Speed?
    - ~200-300 MH/s
    - Ok, but not sufficient

Mining hardware

- FPGA mining
  - Better performance over GPUs
    - "Bit fiddling" operations is trivial to specify
    - Use all transistors on the card
  - Cooling is easier
  - Speed?
    - ~1 GH/s
    - Better, but still not sufficient
  - Cost-per-performance was marginally improved over GPUs
    - Short-lived phenomenon

Mining hardware

- ASIC mining
  - Application-Specific Integrated Circuits
  - Optimized for the sole purpose of mining
  - Lifetime of ASICs was quite short initially
    - Due to rapidly increasing network hash rate
  - Now ASIC equipments have longer life time
    - After growth rate of hash power has stabilized

Mining hardware

- Economics of mining haven't been favorable to small miner
  - Mining cost = H/w cost + operation cost (electricity, cooling, etc.)
- Today, professional mining
  - Mining farms
  - Mining pools
- Future, find alternatives

Mining pool

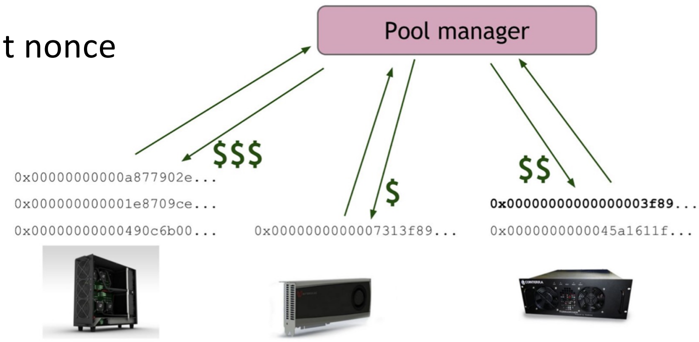
- A group of miners form a pool
- Attempt to mine a block
- Use a designated coinbase recipient
  - Recipient is called pool manager
- No matter who actually finds nonce
  - Pool manager will receive reward

Mining pool

- Pool manager takes that revenue
  - Distribute it among participants
    - Based on amount of work done by participants
  - Takes a cut
- How does pool manager know if miners are actually working?
  - If yes, how to estimate amount of work?
    - Miners can cheat
      - Show proof?

Mining pool

- Mining shares
  - Miners probabilistically prove amount of work done
  - Consider a target begins with 30 zeros
    - Block's valid hash must have >=30 zeros
  - Miners find many hashes, some are close to target
    - Miners show these nearly valid hashes as proof
      - Threshold can be set, e.g., >=20 zeros
  - More work, more profit
    - No bonus for finding right nonce



Mining pool

- Mining shares
  - Pay-per-share model
    - Pool manager pays a flat fee for shares above threshold
    - Best for miners
    - Pool manager absorbs the risk
      - Must pay miners even if right nonce is not found
  - Proportional model
    - Payment depends on whether or not pool actually finds right nonce
    - If found, reward is distributed based on contributions
    - Otherwise, no/low reward
  - Miners do pool hopping to optimize revenue

Mining pool

- Are mining pools good?
  - Pros
    - Easier for smaller miners to get involved
    - Easier to upgrade network
      - Central pool manager can force miners to upgrade
  - Cons
    - Issue of centralization
      - 51% mining pools
    - All are miners, less full nodes

Mining pool

- Mining pool protocols
  - Three major protocols
    - getwork
    - getblocktemplate (gbt)
    - Stratum

Mining pool

- Mining pool protocols
  - getwork
    - Original mining method
    - HTTP-based requests and responses
    - JSON Remote Procedure Calls (RPC) to request a block
    - Server gives only block headers to client
      - No Tx or way to modify block (except nonce)
    - Client is limited to try all nonce values
      - If exhausted, get more work from server
    - High bandwidth usage for modern mining h/w
    - Worst approach in today's scenario
      - rollnTime extension
        - Allows a client to generate own work by modifying timestamp on block header
          - Limited scalability

Mining pool

- Mining pool protocols
    - getblocktemplate (gbt)
        - HTTP-based
        - JSON RPC for communication between client and poolserver
        - Server gives templates to client
            - Client uses it to generate own work
        - Full block data is given
            - Allows the client to modify the block
                - Miner can choose which Tx to include in block
        - Very high bandwidth usage
        - Well documented, detailed specifications

Mining pool

- Mining pool protocols
    - Stratum
        - Not HTTP-based
            - Direct TCP connections
        - JSON RPC for communication between client and poolserver
        - Uses parts of gbt's message specification
        - Server gives templates to client
            - Client uses it to generate own work
        - Block headers and only Tx hashes are given
            - Change extraNonce
        - Least bandwidth usage
        - Very fast, efficient switching to new work
        - No real specification

Mining pool

- Mining pool protocols
  - Stratum


```
{
    "result": [
        [
            "mining.notify",
            "ae6812eb4cd7735a302a8a9dd95cf71f"
        ],
        "60021014", extraNonce1
        4 extraNonce2 size (bytes)
    ],
    "id": 1,
    "error": null
}

{
    "params": [
        512 Difficulty
    ],
    "id": null,
    "method": "mining.set_difficulty"
}

{
    "params": [
        "5c04", JobID
        "da0dadb0eda4381df442bde08d23d54d7d371d5ce7af3ee716bd2a7e017eacb8", Previous block hash
coinBase1 "010000000100000000000000000000000000000000000000000000000000000000000000ffffffff2a03700a08062f503253482f0495
coinBase2 "102f776166666c65706f6f6c2e636f6d2f0000000001d07e582a010000001976a9145d8f33b0a7c94c878d572c40cbff22a49268467d88
        [
            "50a4a386ab344d40d29a833b6e40ea27dab6e5a79a2f8648d3bc0d1aa65ecd3f",
            "7952ecc836fb104f41b2cb06608eeeaa6d1ca2fe4391708fb13bb10ccf8da179",
            "9400ec6453aac577fb6807f11219b4243a3e50ca6d1c727e6d05663211960c94",
            "c11a630fa9332ab51d886a47509b5cbace844316f4fc52b493359b305fd489ae",
            "85891e7c5773f234d647f1d5fca7fbcabb59b261322d16c0ae486ccf5143383d",
            "faa26bbc17f99659f64136bea29b3fc8d772b339c52707d5f2ccfe1195317f43"
        ], Merkle branches
        "00000002", Version
        "1b10b60e", Bits
Time    "531a3f95",
        false
    ],
    "id": null,
    "method": "mining.notify"
}
```


Changing protocol

- How to introduce new features in Bitcoin protocol?
  - Just release an update and ask all nodes to upgrade?
    - Not really!
      - Not every node would upgrade
      - Some nodes would fail to get or to get it in time
  - Two types of changes
    - One causes a hard fork
    - Other causes a soft fork

Changing protocol

- Hard fork
  - Introduces new features that were previously considered invalid
    - New nodes would accept blocks as valid that old nodes would reject
  - After majority of nodes have upgrade
    - Longest branch will contain blocks that are invalid for old nodes
    - So, old nodes will consider their (shorter) branch to be longest valid branch
  - Such changes cause a hard fork
    - Blockchain splits
    - Branches will never join together again

Changing protocol

- Hard fork
  - E.g., fixing the bug in *OP_CHECKMULTISIG* would need a hard fork
  - What can we add with a hard fork
    - Adding new opcodes to protocol
    - Changing limits on block/Tx size

Changing protocol

- Soft fork
  - Adding features that make validation rules stricter
    - Old node would accept all blocks while new nodes would reject some
  - After majority of nodes have upgraded
    - New nodes will enforce new, tighter rules
    - Old miners might mine invalid blocks
      - Include some Tx invalid under new rules
      - But, they will figure out soon after their blocks are rejected
        - Without understanding reason
        - Ask their operator to check/upgrade
  - Such changes cause a soft fork
    - There will be many small, temporary forks

Changing protocol

- Soft fork
  - E.g., introduction of P2SH scripts
    - Old nodes would still verify a valid P2SH correctly
      - Hash the data value, check if hash matches value in scriptPubKey
      - OP_EQUAL pushes 1 on top of stack
    - Old nodes don't know (now required) step of running that value itself
  - What can we add with a soft fork
    - New cryptographic schemes
    - Some extra metadata in coinbase parameter
    - Some specific formatting to a field
  - 2016 “The DAO” hack
    - Soft/hard fork proposals