

# Bitcoin and Cryptocurrency Technologies

## Lecture 4: Bitcoin Data Model

Yuri Zhykin

Apr 27, 2022

# Bitcoin Data Model

- **Bitcoin Block Chain** (or **Bitcoin Time Chain** is a distributed, highly-redundant database of transaction records that strongly guarantees the *existence, validity and order* of transactions.
- **Bitcoin Protocol** is a distributed protocol of maintaining the Bitcoin transaction database that imposes strict rules about transaction validity and enforces the database guarantees through the **Proof-of-Work** system.
- If a transaction record is added to the database, one can be sure that it
  - definitely happened,
  - is provably valid,
  - it happened strictly before or after the other transactions.

# Transaction

- **Tx**
  - **version**
  - **inputs**
  - **outputs**
  - **witnesses**
  - **locktime**
- **Inputs** - a list of transaction inputs - references to the outputs of other transactions, that are consumed by this transaction.
- **Outputs** - a list of newly created outputs that specify, where all the bitcoin from the outputs referenced by inputs is being transferred.
- **Locktime** restricts the moment in time, when the transaction can be included in the transaction records database.

# Transaction ID

- **Transaction ID** is not included in the transaction data structure, since it can be computed from the binary representation of the transaction:

$$TXID = SHA256(SHA256(TX_{binary}))$$

- *TXID* is a 32-byte sequence and is usually represented as a 64-character hex-encoding of the 32-byte sequence:

169e1e83e930853391bc6f35f605c6754cfead57cf8387639d3b4096c54f18f4

# Transaction Output

- **TxOutput**
  - **amount**
  - **lock-script**
- **Amount** - an amount of bitcoin currency units, called “satoshis” represented as an integer ( $1 \text{ BTC} = 10^8 \text{ satoshis}$ ).
- **Lock-script** - a computational (usually “supply a valid signature”) problem that must be solved in order to spend this output, expressed as Bitcoin Script program.

# Transaction Input

- **TxInput**
  - **previous-tx-id**
  - **previous-tx-index**
  - **unlock-script**
- Transaction input is a reference to the transaction output being spent along with the solution to the lock-script of that output.
- **Previous transaction ID** - a TXID of the transaction that created the output.
- **Previous transaction index (VOUT)** - an index of the output in the output list of the transaction referenced by previous transaction ID.
- **Unlock-script** - a solution to the *lock-script* of the output being spent represented as Bitcoin Script program.

# Transaction Witness

- **Witness** is an additional structure in the transaction that was introduced as a protocol upgrade (called **SegWit** - **Segregated Witness**) in 2017 as a first step in a long-term plan to improve Bitcoin security, scalability and flexibility.
- **Witness** allows to store complex *unlock-scripts* (solutions) to the *lock-scripts*, which accounts for a large part of the transaction size.

# UTXO (Unspent Transaction Output) Set

- All bitcoin in existence is represented via a set of **unspent transaction outputs (UTXOs)** - a set of records of the form (*amount*, *owner*) that can be provably verified as not used in any known transaction.
- Every regular Bitcoin transaction consumes some existing UTXOs and generates new UTXOs.
- An entity “owns” bitcoin if the UTXO set contains outputs that have that entity as the *owner* part.



# Transaction Fee

- **Transaction fee** is a difference between the amounts in the spent outputs and the amounts in the newly generated outputs:

$$TxFee = \sum_{i=1}^n InputAmount(txin_i) - \sum_{j=1}^m Amount(txout_j),$$

- **Block**
  - **header**
  - **transactions**
- **Header** is a structure that contains metadata about the block and all components necessary for the Proof-of-Work system.
- **Transactions** is an ordered list of transactions that were included in the given block.

# Block Header 1/4

- **BlockHeader**
  - **version**
  - **previous-block-hash**
  - **transaction-merkle-tree-root**
  - **timestamp**
  - **proof-of-work-bits**
  - **proof-of-work-nonce**
- **Block hash** (or **block ID** is calculated as

$$BlockID = SHA256(SHA256(BlockHeader_{binary}))$$

- **Previous block hash** is the “chain” part of the “block chain” term: every next block references the previous one, and modifying any information in any block modifies the hashes of all the blocks that follow (avalanche effect of the cryptographic hash functions).

## Block Header 2/4

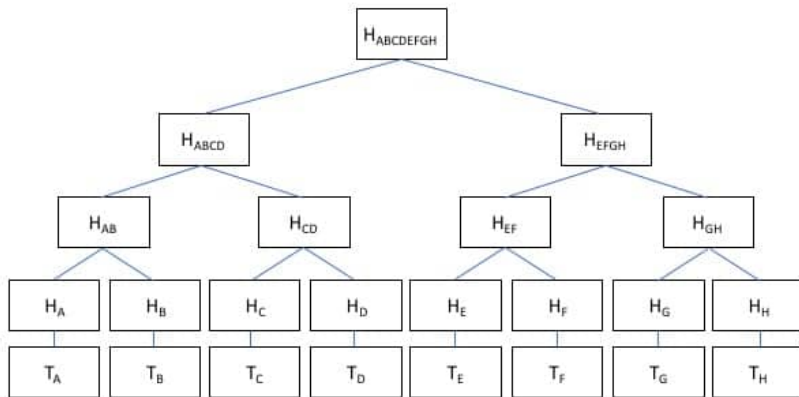
- **Proof-of-Work bits** is an encoded target value for the Proof-of-Work algorithm: the resulting PoW solution must be less than the number represented by this field:

$$BlockID = SHA256(SHA256(BlockHeader)) < Target$$

- **Bits** value is recalculated every 2016 blocks (roughly 2 weeks) to ensure that the average time to solve the PoW problem is kept around 600 seconds (10 minutes).
- This recalculation is called **difficulty adjustment**: if the average time for the last 2016 blocks was less than 10 minutes, increase PoW difficulty by selecting a smaller target value, otherwise decrease it.
- **Proof-of-Work nonce** (**nonce** = Number used **ONCE**) - a value that is incremented when running brute force search for the PoW solution.

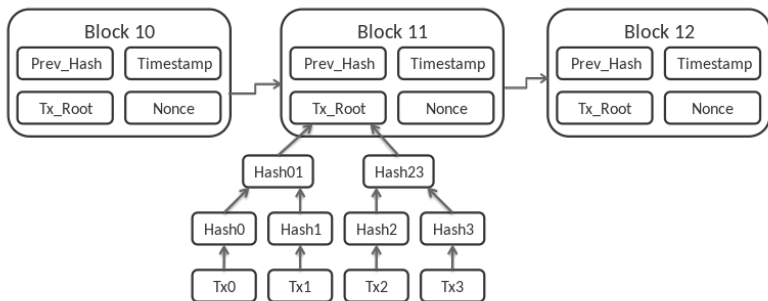
## Block Header 3/4

- **Transaction Merkle Tree Root** - a root of a Merkle Tree - a 32-byte sequence that cryptographically includes all transactions in the block and ensures their sequence:



## Block Header 4/4

- Previous block hashes link blocks and transactions they contain into a linear sequence chain via a cryptographic commitments.
- Changing a single bit in a transaction completely changes the merkle root, which changes block hash, which, in turn changes hash of the next block and so on.



# Mining

- **Mining** is a process of computing Proof-of-Work solutions for new blocks.
- Miner
  - selects a number of pending transactions,
  - builds a Merkle tree and uses its root to construct the new block header with the last known block's hash as the previous block hash.
  - performs a brute force search of the Proof-of-Work solution

$$SHA256(SHA256(BlockHeader)) < Target$$

- if the PoW solution is found **before** an alternative solution is received over the peer-to-peer network, it can be published for the network to accept it as the new highest known block.

# Coinbase Transaction and Halving Events

- In order to incentivize miners to do their job, Bitcoin Protocol allows to add a special transaction at the beginning of the transaction list of each new block.
- This transaction is called **coinbase transaction** and has no inputs, only outputs, thus “generating” new bitcoins.
- Additionally, miners can add all transaction input-output amount differences (*transaction fees*) to the coinbase transaction amount.
- **Halving events** - in order to keep bitcoin supply limited, the amount of new bitcoins started with 50 BTC and is reduced by the factor of 2 every 210000 blocks, eventually reaching 0, at which point the bitcoin supply will stop growing.



- Learn me a Bitcoin by Greg Walker - a great resource that explains multiple technical details about Bitcoin
  - <https://learnmeabitcoin.com/>

# The End

Thank you!