BITCOIN, PoW

Blockchain technologies, **lecture 4**

Course overview

- Consensus protocols
- PoW algorithm and PoW protocol
- Block validation rules
- Hard forks – soft forks

CONSENSUS PROTOCOLS

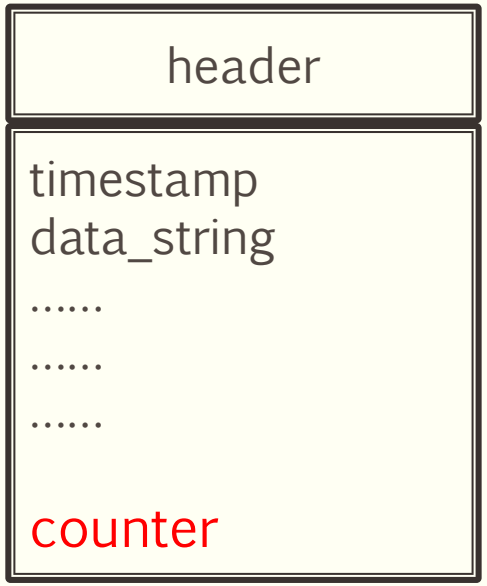
Consensus protocols

- **Distributed database:** order of transactions
- **Blockchain**: All nodes maintain the same distributed ledger, maintain the same history of transactions, i.e., the same blockchain.
- **PoW:** (power of computation)
- **PoS:** (security depositis)
- **PBFT:** Tolerate malicious nodes.

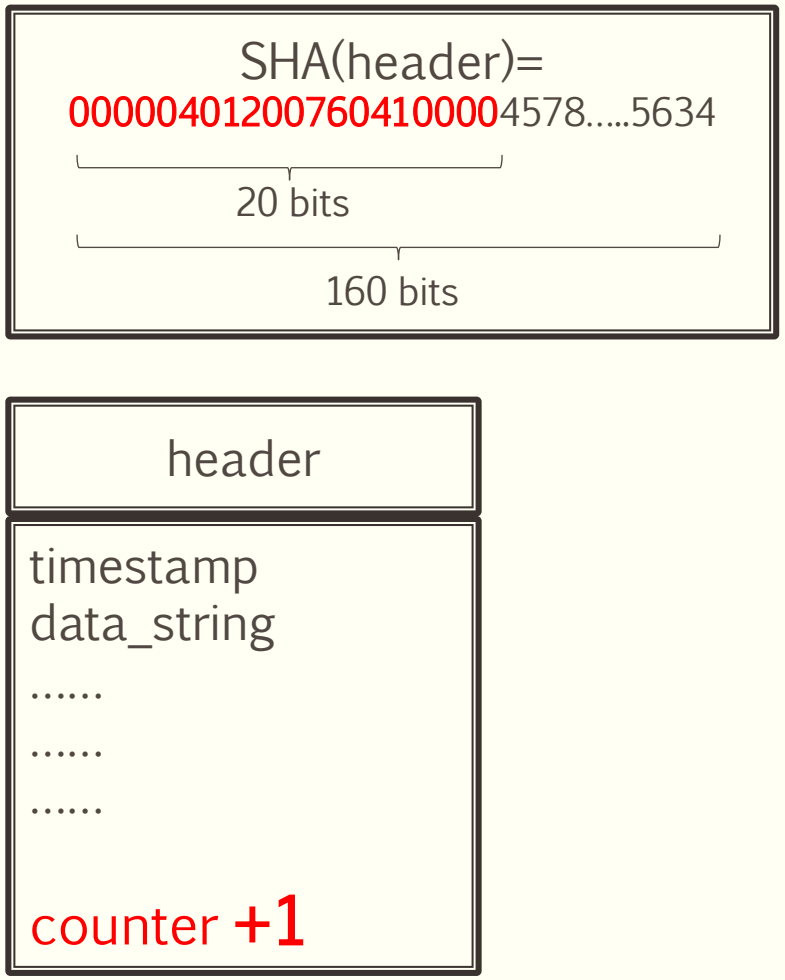
HASH CASH

Bitcoin PoW history

Hashcash (1997)
PoW and reusable PoW (2005)

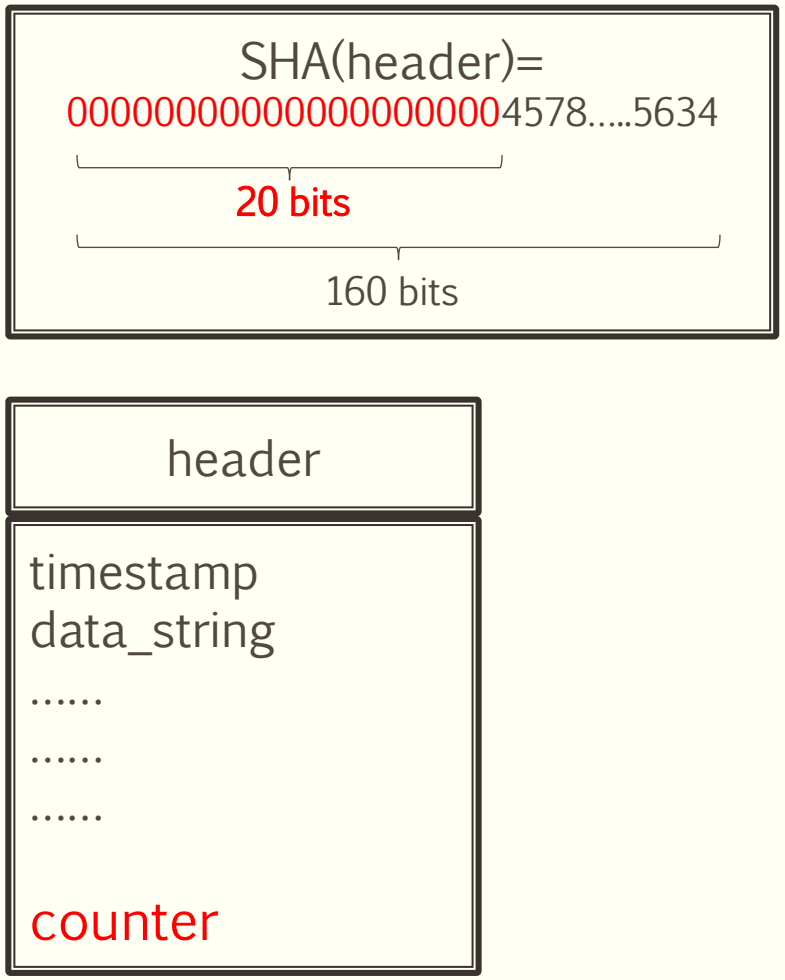


- Detect spam emails
- Sender must fill a counter value, initialized with a random number.
- A correct value for counter is the proof the sender spent some CPU to generate the email.



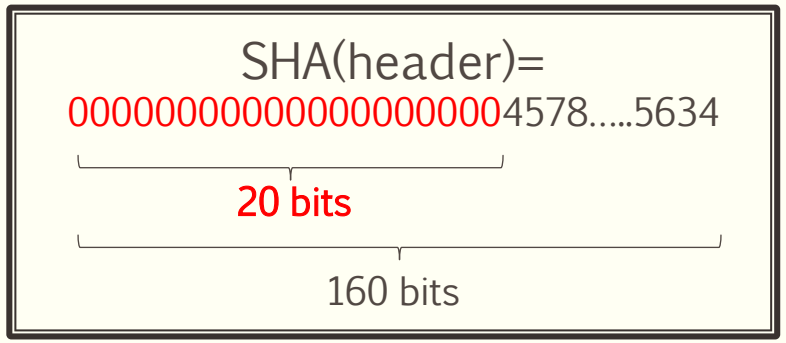
Hashcash (1997)
PoW and reusable PoW (2005)

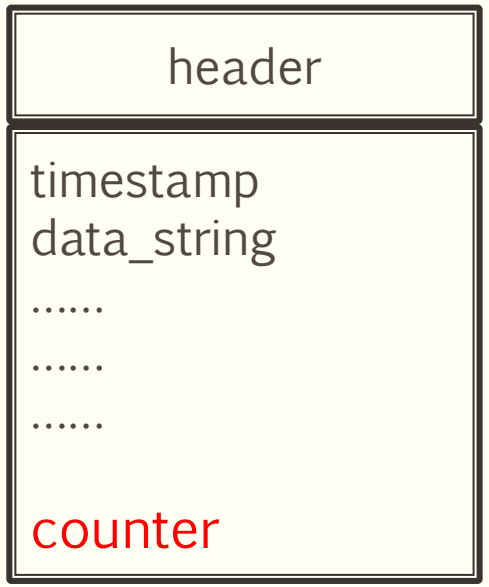
- If SHA-1(header) has the first 20 bits set to 0, the header is valid
- If header is not valid (first 20 bits are not set to 0) increment counter.
- pre-image resistance



Hashcash (1997)
PoW and reusable PoW (2005)

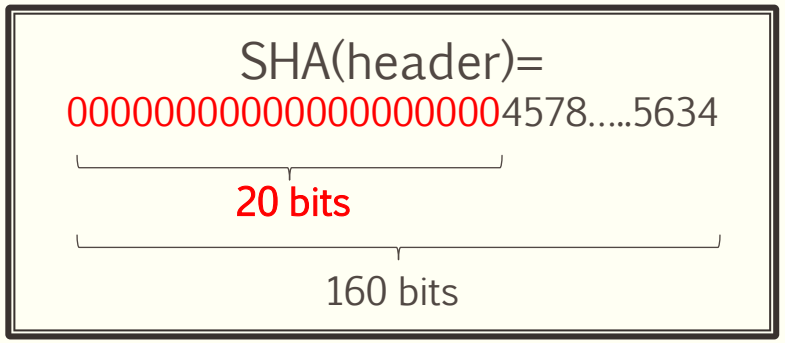
- If SHA-1(header) has the first 20 bits set to 0, the header is valid.
- hash(header) < target
- If header is not valid (first 20 bits are not set to 0) increment counter.
- pre-image resistance

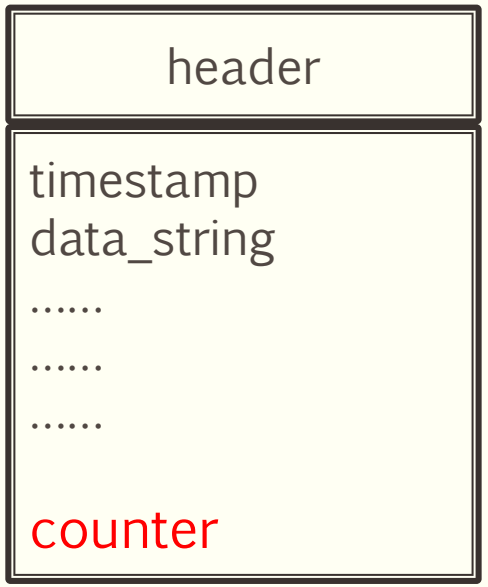




Hashcash (1997)
PoW and reusable PoW (2005)

- pre-image resistance
- it is improbably to guess a value with the hash value of 0
- anyone can verify the hash of the header in constant time.



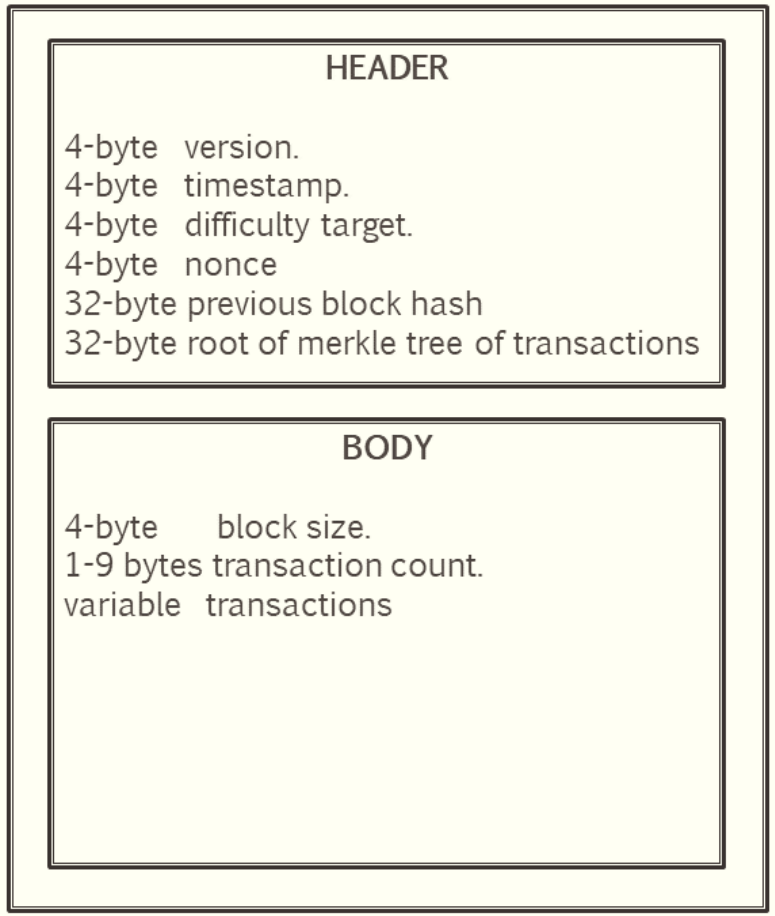


Hashcash (1997)
PoW and reusable PoW (2005)

- Is the counter value reusable?

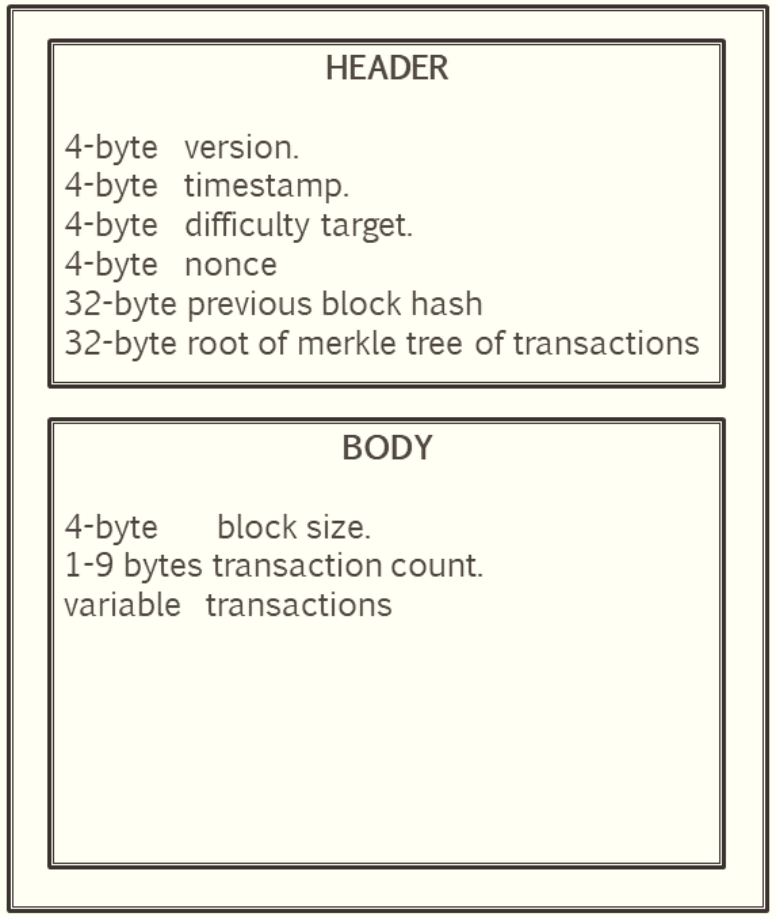
BLOCK STRUCTURE

Bitcoin Block structure -- Header



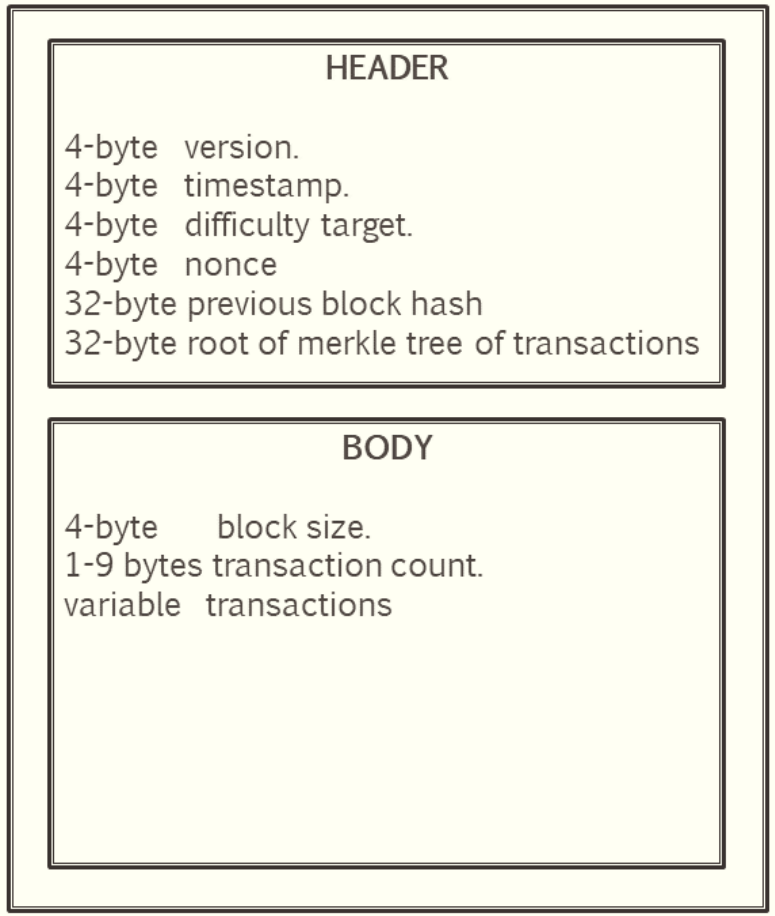
- 4-byte version.
- 4-byte timestamp.
- 4-byte difficulty target.
- 4-byte nonce
- 32-byte previous block hash
- 32-byte Merkle root

Bitcoin Block structure -- Header



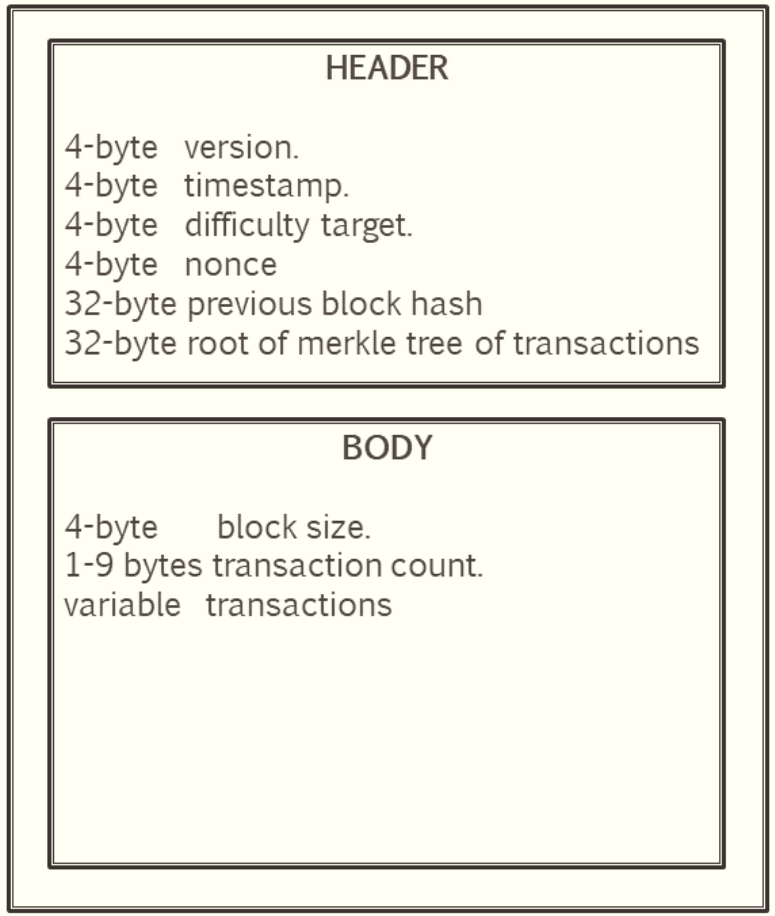
- 4-byte version.
- 4-byte timestamp.
- **4-byte difficulty target.**
- **4-byte nonce**
- 32-byte previous block hash
- 32-byte Merkle root

Bitcoin Block structure -- Header



- 4-byte version.
- 4-byte timestamp.
- 4-byte difficulty target.
- 4-byte nonce
- 32-byte previous block hash
- 32-byte Merkle root transaction hashes

Bitcoin Block structure -- Body



- 4-byte block size.
- 1-9 bytes transaction count.
- variable transactions.

BLOCK CREATION - POW

TRANSACTION RULES

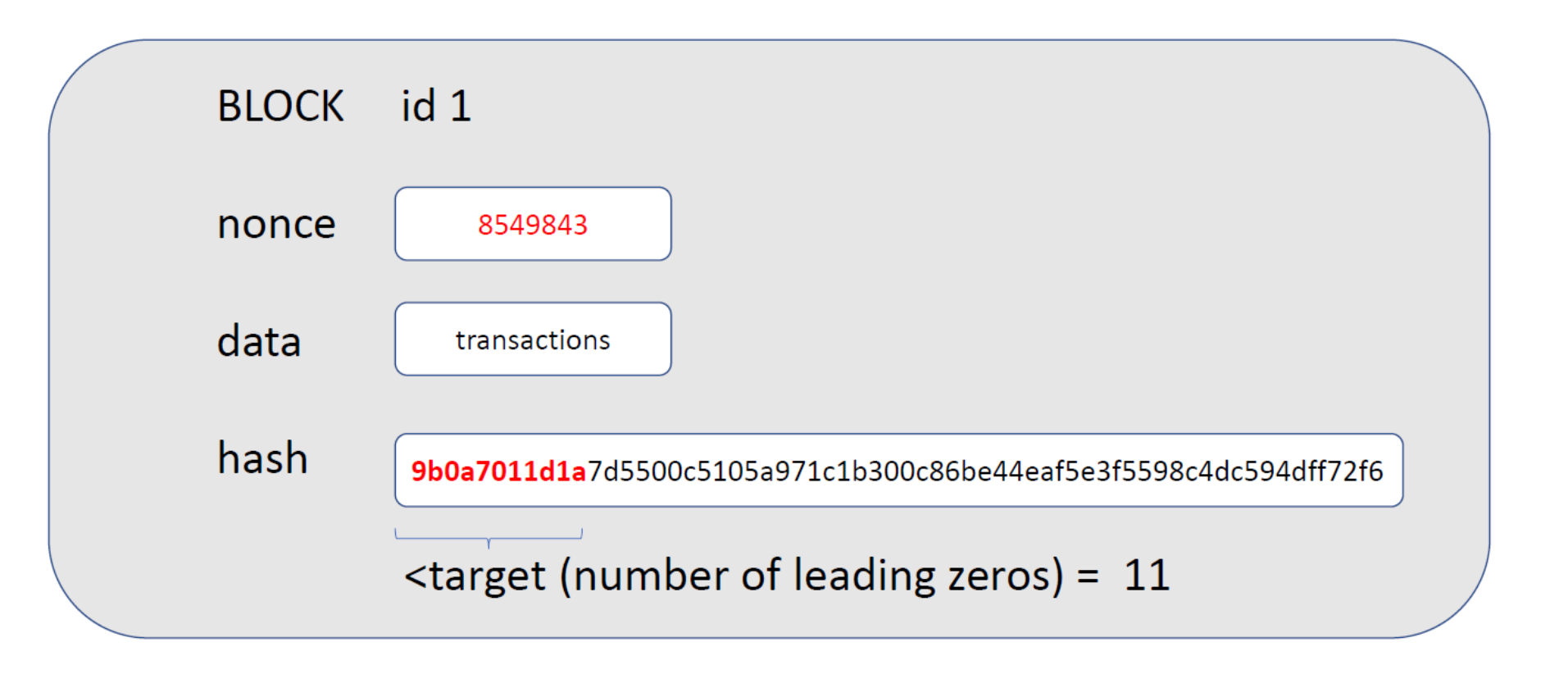
- Transactions are checked:
  - ScriptSig successfully unlocks the previous ScriptPubKey
  - Valid signatures
  - inputs equals outputs
  - inputs are not previously spent
- All nodes synchronize their transaction pools (transactions not included in the blockchain) via P2P messages.

BLOCK GENARATION -- PoW

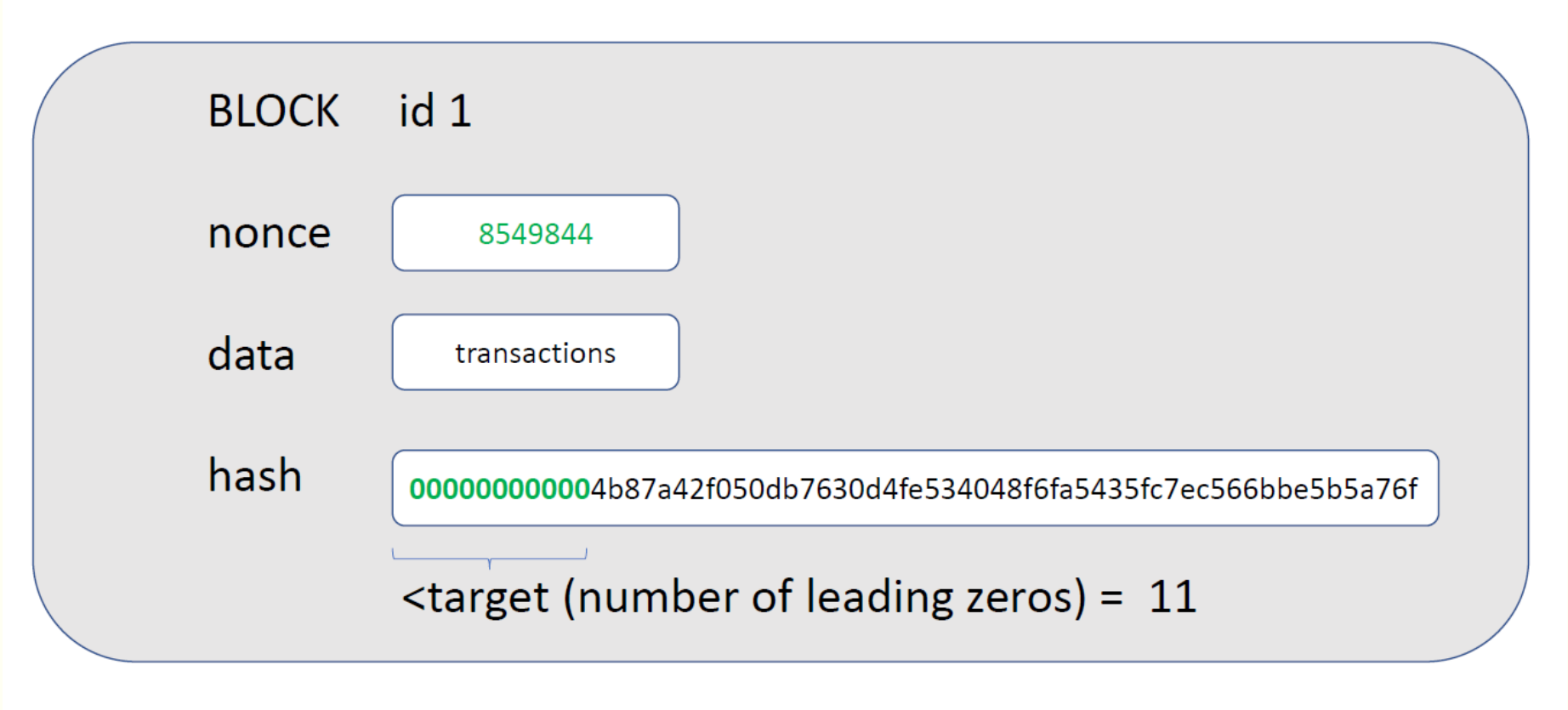
Algorithm 1 PoW finding block header nonce

```
nonce ← 10
while nonce < 2^32 do
    digest ← SHA_256(SHA_256(bl_header))
    if digest < target then
        return nonce
    else
        nonce ← nonce + 1
    end if
end while
```

BLOCK GENARATION PoW



BLOCK GENARATION -- PoW



DIFFICULTY

- **block time**: expected/average time spent to mine a block **10 minutes**.
- **difficulty**: measures how difficult it is to mine a block. (i.e., find a nonce such that the block hash is valid)
- If [average block time] > [expected block time] ➔ reduce difficulty
- If [average block time] < [expected block time] ➔ increase difficulty

DIFFICULTY

- Target is recalculated each 2016 block to adjust difficulty

$$difficulty\_new = difficulty\_old * 20160 / difficulty\_epoch\_time$$

BLOCK VALIDATION

BLOCK CONFIRMATION

- Miner advertise the new block. Block is accepted if:
  - Hash of the block does not exceed target -- PoW
  - The block data structure is syntactically valid.
  - All transactions are valid.
  - Hash of the previous block is valid and on the main chain.
  - Block timestamp must not be more than two hours in the future.
  - The block size is within acceptable limits.
  - The first transaction is a coinbase generation transaction
- The majority decision is represented by the chain which has the greatest proof-of-work effort invested in it.

BLOCK CONFIRMATION

- Miner advertise the new block. Block is accepted if:
  - The block data structure is syntactically valid.
  - All transactions are valid.
  - Hash of the previous block is valid and on the main chain.
  - Hash of the block does not exceed target -- PoW
  - Block timestamp must not be more than two hours in the future.
  - The block size is within acceptable limits.
  - The first transaction is a coinbase generation transaction
- Nodes express their acceptance of the block by adding it to their accepted version of the chain, i.e using the hash of the accepted block as the previous hash in the block they continue to work on.

BLOCK CONFIRMATION – LONGEST CHAIN RULE

- Hash of the previous block is valid and on the main chain.

- "orphan" blocks. When a node adopts the longer chain, all valid transactions of the blocks inside the shorter chain are re-added to the transaction pool and will be included in another block. The reward for the orphan blocks will be lost.

- The miner of a block is allowed to spend the reward only after 100 blocks have passed after the block is processed. The 100-block maturation rule gives time for chain reorganization.

BLOCK CONFIRMATION

- Miner advertise the new block. Block is accepted if:
  - The block data structure is syntactically valid.
  - All transactions are valid.
  - Hash of the previous block is valid and on the main chain.
  - Hash of the block does not exceed target -- PoW
  - Block timestamp must not be more than two hours in the future.
  - The first transaction is a coinbase generation transaction
  - The block size is within acceptable limits.

- 2020 halving day $8821.42 => $10,943.00.

BLOCK CONFIRMATION

- Miner advertise the new block. Block is accepted if:

  - The block data structure is syntactically valid.
  - All transactions are valid.
  - Hash of the previous block is valid and on the main chain.
  - Hash of the block does not exceed target -- PoW
  - Block timestamp must not be more than two hours in the future.
  - The first transaction is a coinbase generation transaction
  - The block size is within acceptable limits.

HARD FORKS AND SOFT FORKS

HARD FORKS

- **Change** the rules of the protocol.
- Mitigates security risks.
- Correct the effect of an attack.
- Add new functionalities.
- It is **not** backward compatible (can see previous transactions as invalid).
- All nodes must adhere to the new rules.

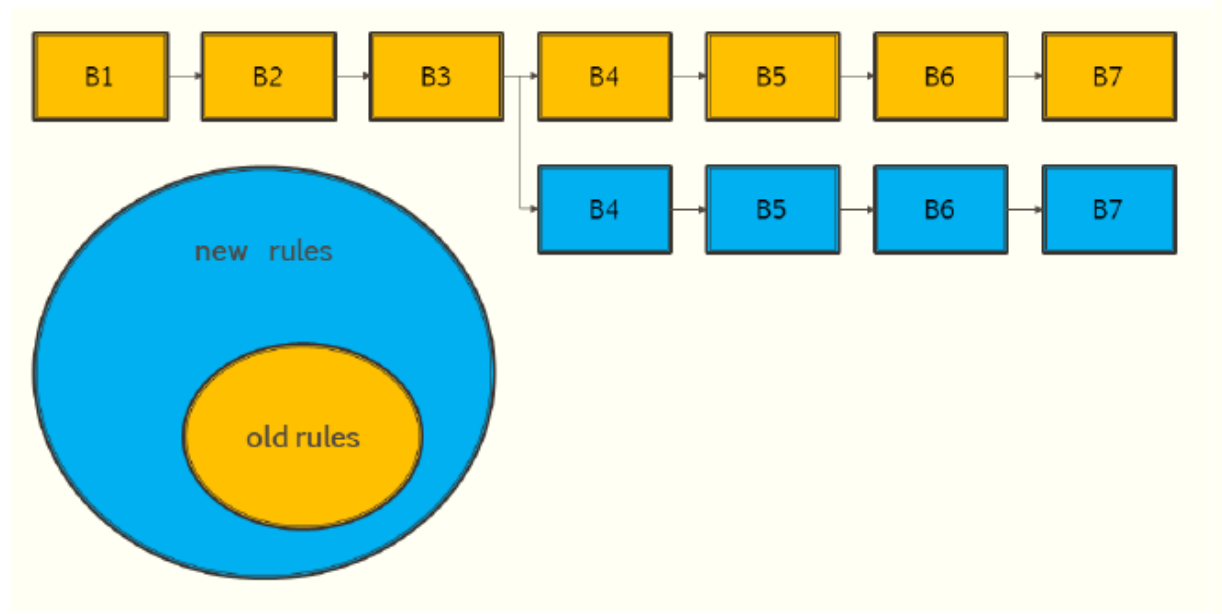
HARD FORKS

- After a hard fork the block chain network may permanently split into:
  - nodes adhering to the new protocol,
  - nodes running the old protocol.

- Example: **Bitcoin-cash (2017)**

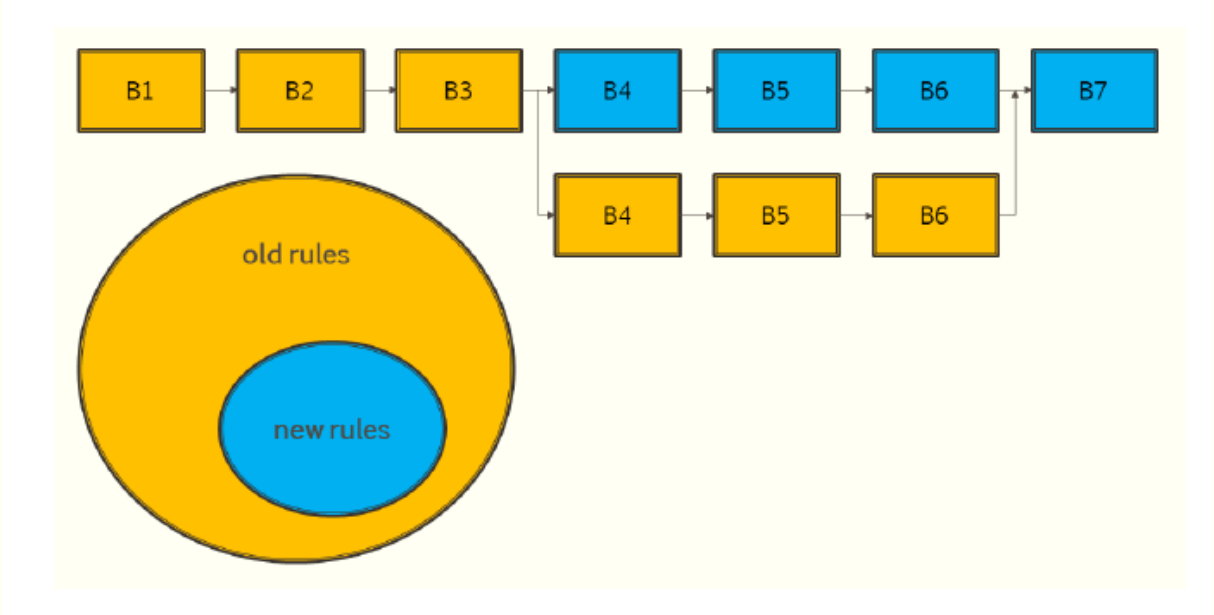
HARD FORKS

- After a hard fork the block chain network may permanently split into:
  - nodes adhering to the new protocol.
  - nodes running the old protocol.
- Hard fork: less restrictive rules



SOFT FORKS

- It is backward compatible (see previous transactions as valid).
- Allows both previous and new blocks to be accepted, keeping two versions of the block chain until all nodes reach consensus



SOFT FORKS -SegWit

- **Witness C**ryptographic condition placed on an unspent transaction output (UTXO) – signature in the unlocking script.

- SegWit separates the signatures from the transaction – reduce transaction size.

SOFT FORKS

- SegWit 2017:
  - **increases transaction throughput.**
    - Block size limit 4MB. non-witness data * 4, witness data * 1/4
    - Old blocks 1MB → 4MB.
  - **eliminates third-party transaction malleability.**

SOFT FORKS - SegWit

- SegWit 2017:
  - **eliminates third-party transaction malleability.**
    - Alice sends Bob 1BTC with transaction id TX_ID1
    - Bob forges a new transaction TX_ID2, copying the first transaction but changing only the signature.
    - If Bob inform Alice that he never get the money, Alice will seek for transaction TX_ID1 which will never get confirmed and will eventually decide to send Bob another 1 BTC.