Bitcoin:
A P2P E-Cash System

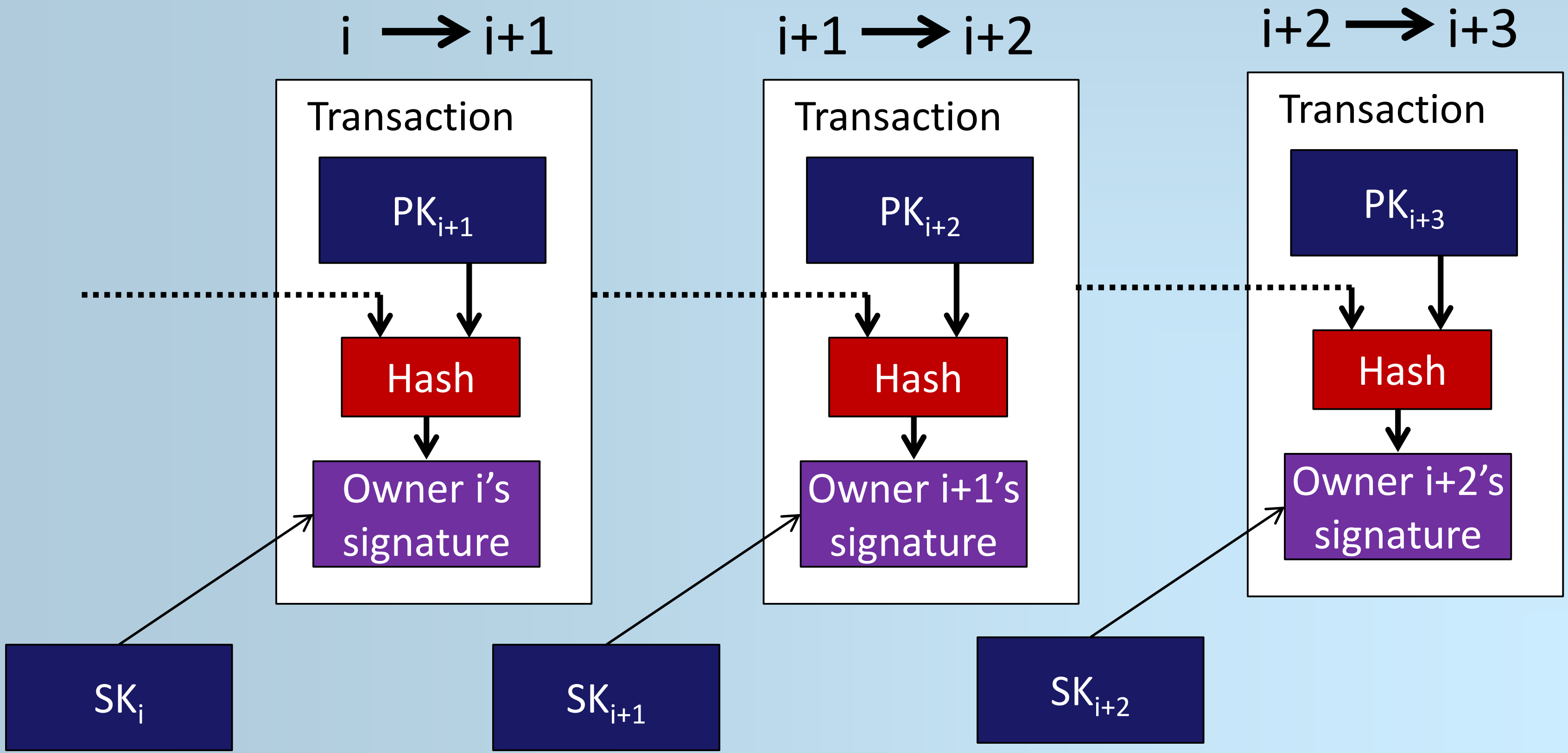
Erkay Savas
Sabanci University

Basics of Bitcoin

- Transaction
  - Transfer of money
- Double spending problem
- Proof-of-Work
- Incentive
- Compacting Transactions

Transactions

- E-coin is a chain of digital certificates
  - A bitcoin owner transfers the coin to next by digitally signing
    - Hash of the previous transaction and
    - Public key of the next owner

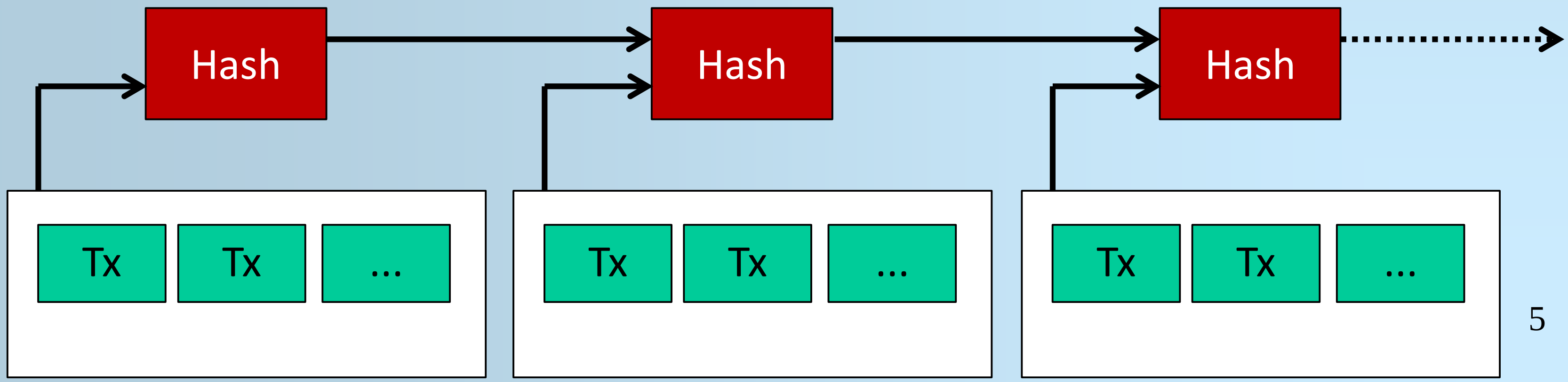


Double-Spending

- Payee cannot verify that one of the previous owners did not double-spend the coin.
- Trusted Central Authority
  - All transactions are sent to TCA
  - All transactions are kept in a ledger
  - We can catch if it is double spent by checking the ledger
- However, it is a P2P network
- No trusted central authority
  - Transactions are publicly announced

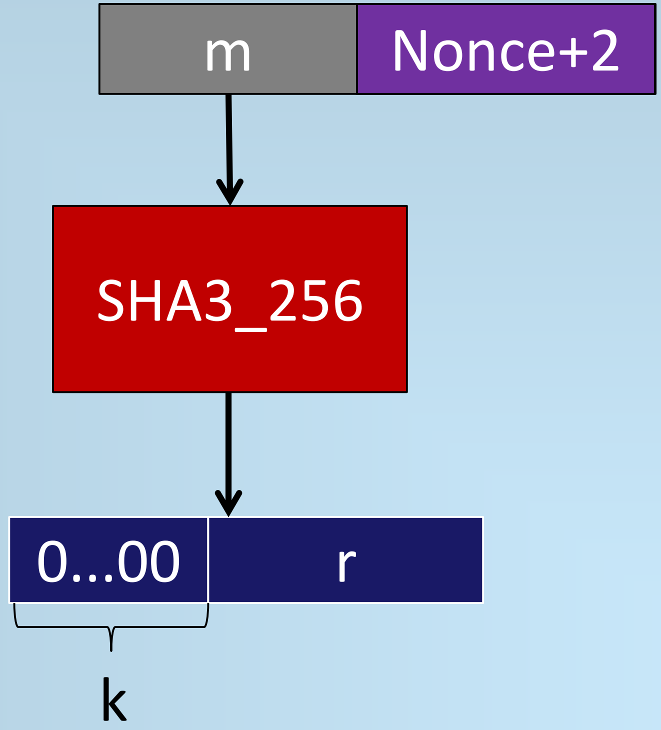
Timestamping Transactions

- A timestamp server would order transactions
- We can compute the hash of a block of transactions and publishes the hash
  - Each timestamp includes the previous timestamp in its hash, forming a chain of hashes
- Hash chains establishes an order between the transactions



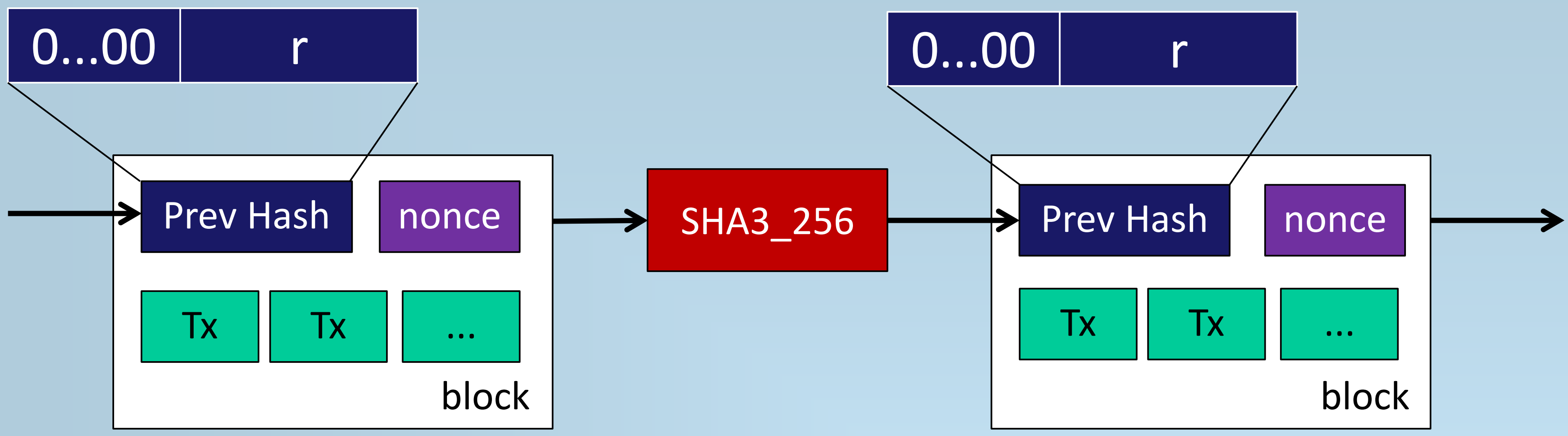
Proof-of-Work

- Computing a certain form of hash value given a message is most probably impossible



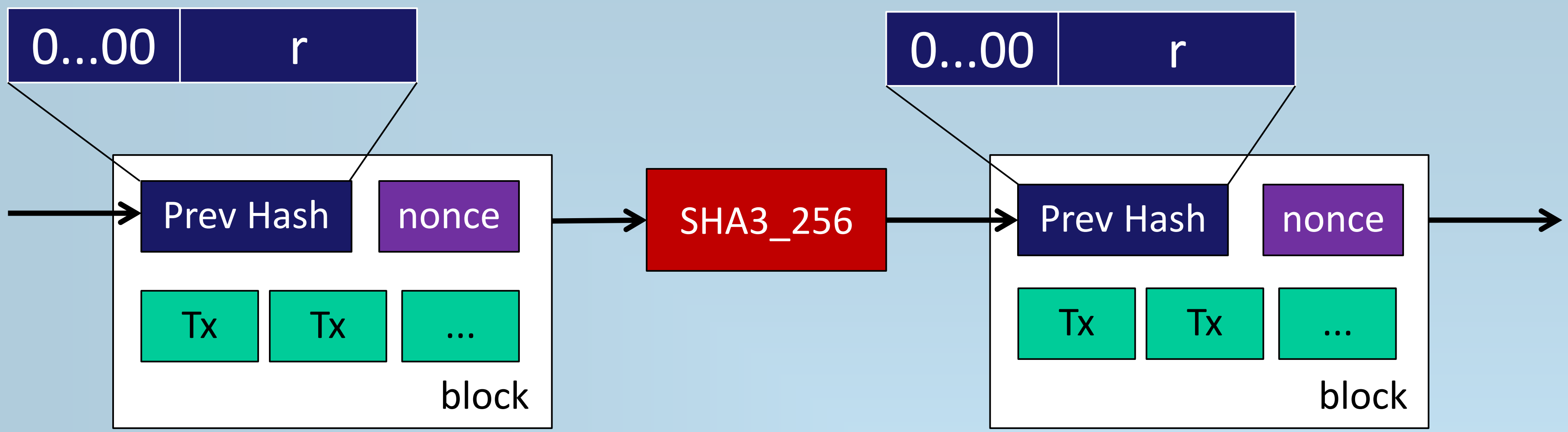
- This can be feasible depending on the magnitude of k,
  - but it takes some time and require some (computational) effort

Proof-of-Work



- To find the hash in the required form (i.e., proof-of-work) , a peer increments the nonce until it is found
  - It takes some time (e.g., ten minutes on a CPU core) → proof-of-work
- There may be more than one chain,
  - The longest one must be chosen

Longest Chain



- A malicious peer wants to change a transaction in a block
  - It needs to recompute the hash of the block
- Honest users keep computing hashes of new blocks
  - A malicious peer needs to work harder to create a longer chain
- As long as honest peers are in majority
  - Malicious peer will never catch up

P2P Network

- Steps to run the network:

1. New transactions are broadcast to all peers
2. Each peer collects new transactions into a block
3. Each peer works on finding a proof-of-work for its block
4. When a peer finds a proof-of-work, it broadcasts the block to all peers
5. Peers accept the block only if all transactions in it are valid and not already spent
6. Peers express their acceptance of the block by working on creating the next block in the chain, using the proof-of-work as the previous hash.

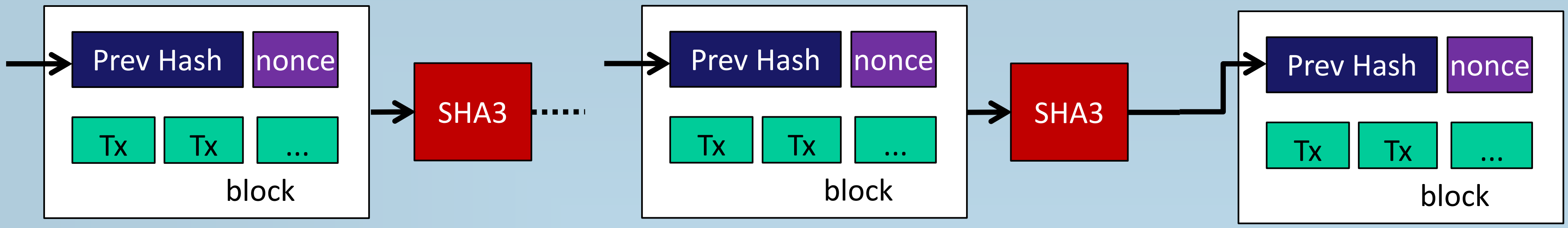
Incentive for Peers

- Why peers should work on proof-of-work
- Incentive:
  - The first transaction of each block is a new coin that will be rewarded to the peer who found the proof-of-work for the block
  - It is 25 BTC (bitcoin) for every block
  - It was 50 BTC in 2009-01-03 for the first block
  - The bitcoin block mining reward halves every 210,000 blocks
  - In 2024, it is expected to drop to 6.25 BTC
- When the reward reaches to 0 (well almost)
  - No new coin is created
  - Incentive will be transaction fee (as in the bank)

Keeping the Incentive

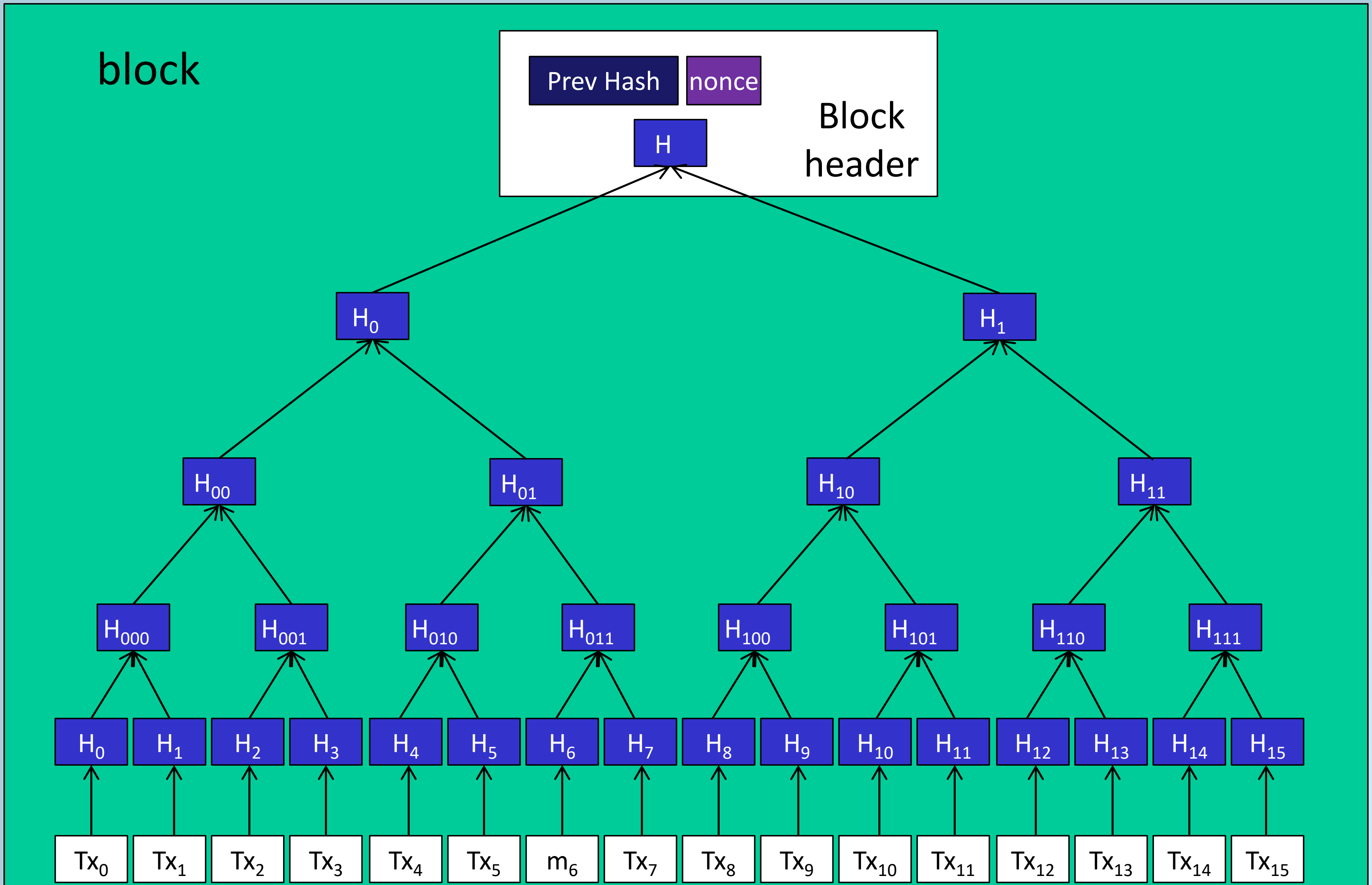
- The incentive should encourage peers to stay honest
  - If a peer earns more by staying honest, he will stay honest
- If the incentive is not satisfactory,
  - a greedy peer (or group of peers) is able to assemble more CPU power than all the honest peers
  - S/he can steal back his payments (delete them from the transactions)

Valid Transactions

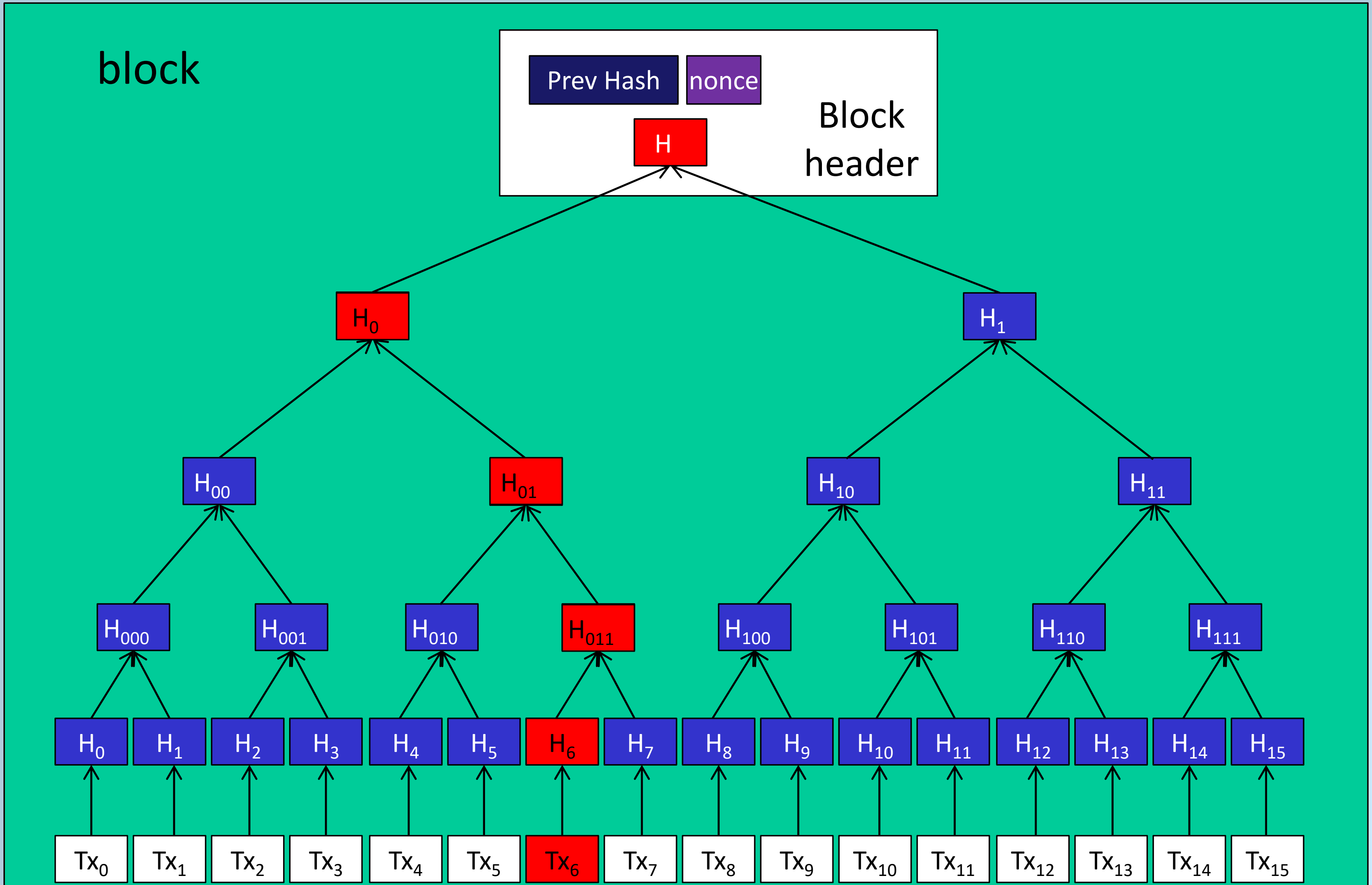


- To verify a transaction is valid
  - We need to determine if the current owner has the bitcoin
  - We need to check a former transaction in which his amount of bitcoin is transferred to the current owner.
  - We may have to verify all transactions in all blocks
  - A block can contain many transactions
  - If we know the block, we can use Merkle Tree to check if the transaction is a part of a valid block

Merkle Tree


block
Prev Hash
nonce
H
Block header
H_0
H_1
H_{00}
H_{01}
H_{10}
H_{11}
H_{000}
H_{001}
H_{010}
H_{011}
H_{100}
H_{101}
H_{110}
H_{111}
H_0 H_1 H_2 H_3 H_4 H_5 H_6 H_7 H_8 H_9 H_{10} H_{11} H_{12} H_{13} H_{14} H_{15}
Tx_0 Tx_1 Tx_2 Tx_3 Tx_4 Tx_5 m_6 Tx_7 Tx_8 Tx_9 Tx_{10} Tx_{11} Tx_{12} Tx_{13} Tx_{14} Tx_{15}

Merkle Tree


block
Prev Hash
nonce
H
Block header
H0
H1
H00
H01
H10
H11
H000
H001
H010
H011
H100
H101
H110
H111
H0
H1
H2
H3
H4
H5
H6
H7
H8
H9
H10
H11
H12
H13
H14
H15
Tx0
Tx1
Tx2
Tx3
Tx4
Tx5
Tx6
Tx7
Tx8
Tx9
Tx10
Tx11
Tx12
Tx13
Tx14
Tx15

Simplified Payment Verification

- A user needs to keep a copy of the block headers of the longest proof-of-work chain
  - He can get the longest chain from the peers
  - Obtain the branch of the Merkle Tree linking the transaction to the block it is timestamped in.
  - If the root hash in the Merkle Tree is good, this means a peer has accepted it
  - Every block after that will further confirm the network has accepted it.
- A block header is about 80 B
  - If a block is generated every ten minutes, 80 B $\times$ 6 $\times$ 24 $\times$365 = 4.2 MB per year.

Some Calculations

- A scenario:
  - an attacker can generate an alternate chain faster than the honest chain.
  - Even so, the attacker cannot include a nonexistent money in a transaction
    - Other peers will not accept it
  - But, the attacker can try to change one of his own transactions to take back money he recently spent
- The probabilities
  - p: probability an honest node finds the next block
  - q: probability the attacker finds the next block
  - q_z: probability the attacker will ever catch up from z blocks behind

Binomial Random Walk

- Gambler's Ruin problem
  - a gambler with unlimited credit starts at a deficit and plays potentially an infinite number of trials to try to reach breakeven.
- Formula:
  - p: probability an honest node finds the next block
  - q: probability the attacker finds the next block
  - q_z: probability the attacker will ever catch up from z blocks behind
  - $q_z = 1$ if $p \leq q$
  - $q_z = (q/p)^z$ if $p > q$