Bitcoin: A Peer-to-Peer Electronic Cash System

https:// bitcoin.org / bitcoin.pdf

Overview

- Blockchain
- Proof-of-work
- Peer-to-Peer Network
- Incentives
- Scalability
- Privacy
- Security

Spooky Language

- Public & Private Keys
- Digital Signatures
- Hashing Algorithms
- Merkle Trees
- Byzantine Faults

What is a Blockchain?

What problems does a Blockchain solve?

- Authenticity of data at the time of publishing.
- Consensus on the ordering of transactions.
- The double-spending of transactions.

What is Double Spending?

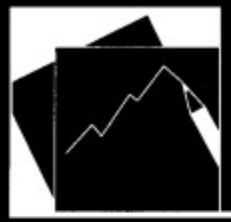
- Alice wants to sell Bob a widget for $500. Bob shows proof of $500 deposit.
- Bob signs a check for $500 and gives it to Alice.
- Alice leaves to deposit the check. Carol walks in.
- Carol wants to sell Bob a widget for $500. Bob shows proof of $500 deposit.
- Bob signs a check for $500 and gives it to Carol.
- Carol leaves to deposit the check. Bob's check to Alice clears.
- Carol’s check bounces at the bank. Carol is screwed!
- Bob receives two widgets for the price of one!

What is Double Spending?

Banker's Edition

- Alice deposits $500 with Bob for safe-keeping. Bob now has a reserve.
- Carol wants to borrow $500 from Bob. Bob credits Carol $500.
- Eve wants to borrow $500 from Bob. Bob credits Eve $500.
- Carol and Eve spend $1,000 at Dave's shop, who banks with Bob.
- Alice and Dave go to withdraw their money from Bob. What happens?

* The bank goes bust! *



IMF Working Paper

The (sizable) Role of Rehypothecation in the Shadow Banking System

Manmohan Singh and James Aitken

Bitcoin: A Peer-to-Peer Electronic Cash System

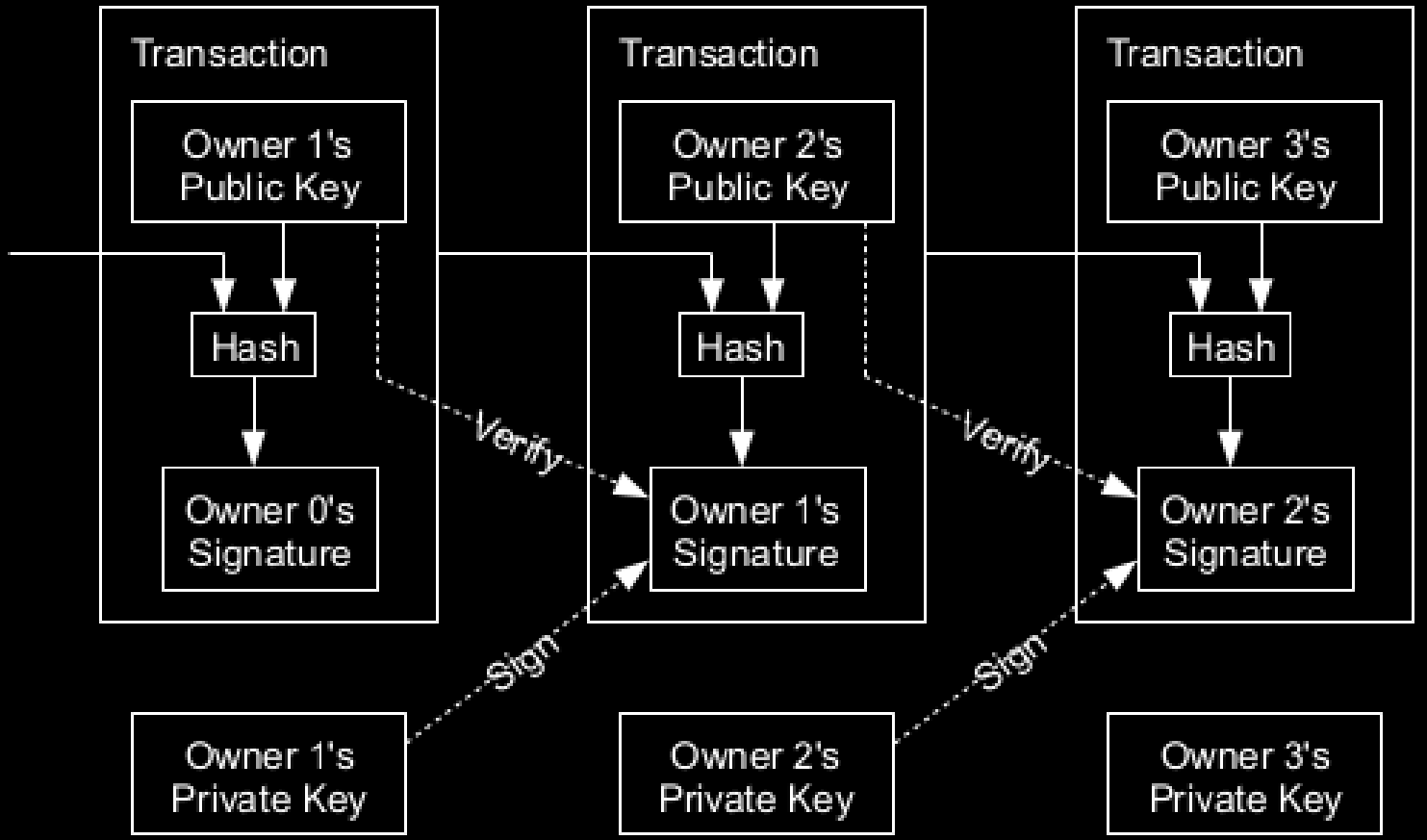
Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

Abstract

- Satoshi proposes a peer-to-peer version of electronic cash, that allows online payments to be sent directly to others, without having to settle through a financial institution.
- He mentions digital signatures as part of the solution, but they do not prevent double-spending attacks and still require trust in a central authority.

Transactions

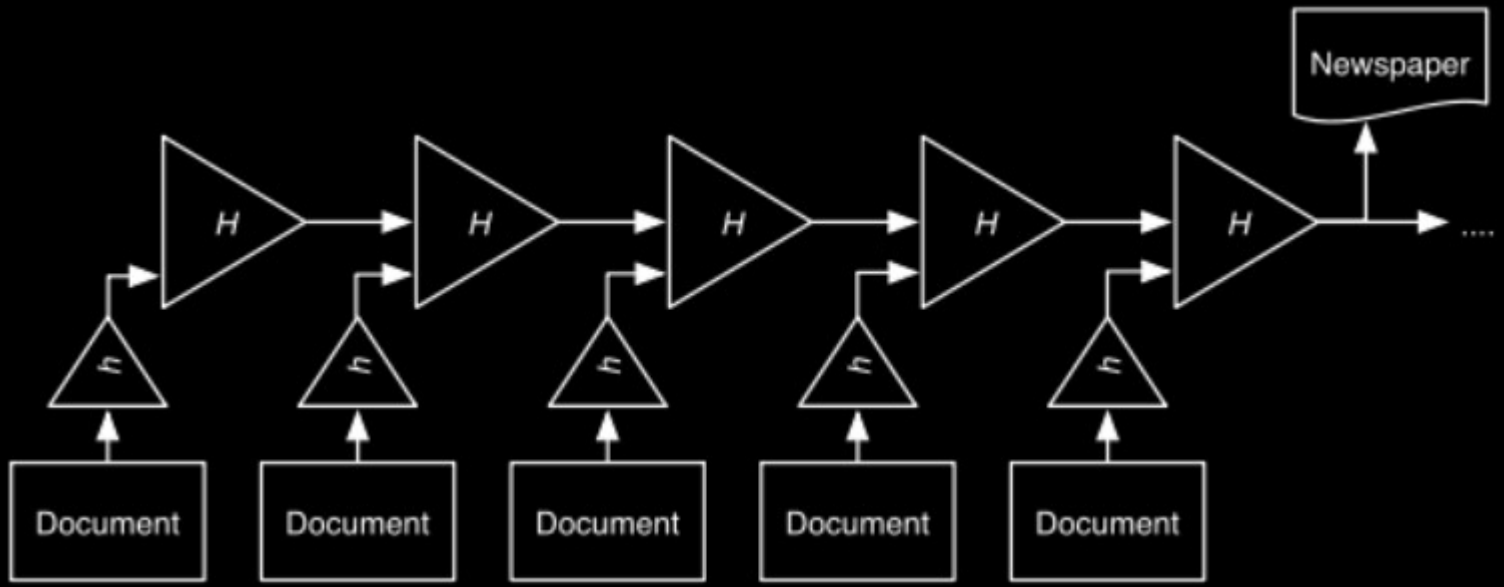


```
{
  "txid": "2b59c31bc232c0399acee4c2a381b564b6fec295c21044fbcbb899ffa56c3da5",
  "hash": "2b59c31bc232c0399acee4c2a381b564b6fec295c21044fbcbb899ffa56c3da5",
  "version": 2,
  "size": 85,
  "vsize": 85,
  "weight": 340,
  "locktime": 0,
  "vin": [
    {
      "txid": "ca4898d8f950df03d6bfaa00578bd0305d041d24788b630d0c4a32debcac9f36",
      "vout": 0,
      "scriptSig": {
        "asm": "",
        "hex": ""
      },
      "sequence": 4294967295
    }
  ],
  "vout": [
    {
      "value": 0.00001000,
      "n": 0,
      "scriptPubKey": {
        "asm": "OP_DUP OP_HASH160 e7c1345fc8f87c68170b3aa798a956c2fe6a9eff OP_EQUALVERIFY OP_CHECKSIG",
        "hex": "76a914e7c1345fc8f87c68170b3aa798a956c2fe6a9eff88ac",
        "reqSigs": 1,
        "type": "pubkeyhash",
        "addresses": [
          "n2eMqTT929pb1RDNuqEnxdaLau1rxy3efi"
        ]
      }
    }
  ]
}
```

Requirements

- We need a way for the recipient to know that a sender did not sign the transaction over to someone else.
- The only way to confirm this is to publicly announce all transactions to the network.
- We need a system for participants to agree on a single transaction history.
- The recipient needs proof that when he accepts a transaction, the network will agree that his claim was first.

Linked Time-stamping

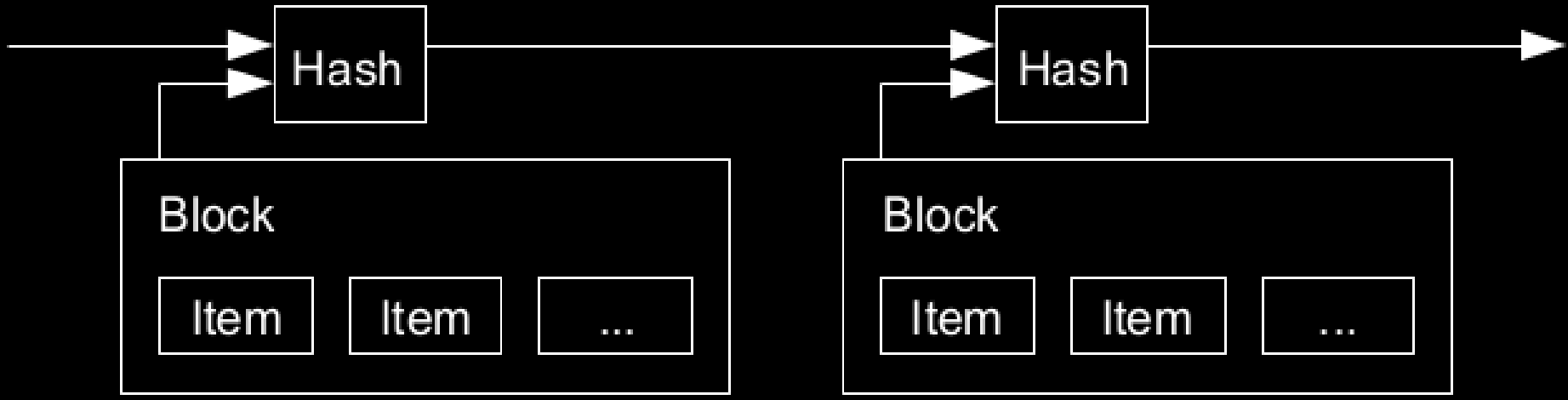


What is “Hashing”?

- It is a one-way computation.
- Reduces any size message into a fixed-length of pseudo-random characters.
- Infeasible to predict or reverse the computation.
- The same message always results in the same hash.

```
> sha256sum bitcoin.pdf
> b1674191a88ec5cdd733e4240a81803105dc412d6c6708d53ab94fc248f4f553
```

Timestamp Server



Chain of Block Hashes

- Hash a block of data, then widely publish the hash.
- This proves that the data must have existed at the time of publishing, in order to calculate the hash.
- Each block includes the previous hash, forming a chain, where each newly hashed block reinforces the ones before it.
- Problem: Who gets to publish the hash? How do we all agree?

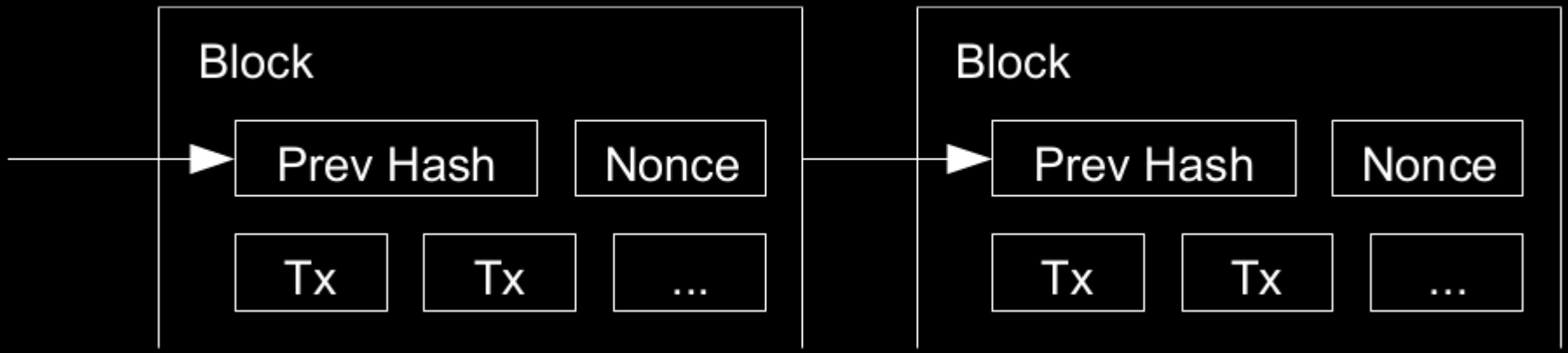
Hash Cash

- Proposed as a mechanism to stop abuse of resources on the internet (such as email).
- The server issues a random cryptographic puzzle to the user, and tasks them to solve it.
- The user must run computations until they find a solution to the puzzle.
- The user offers their solution, server quickly verifies it, and returns the resource.

Proof of Work

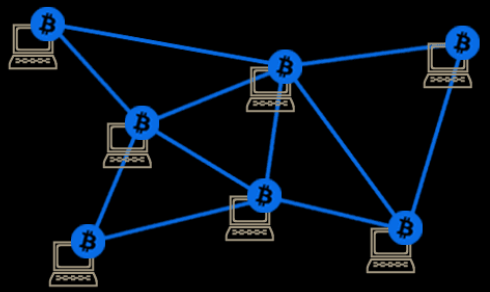
- Hash(Block + variable) = 256 random bits.
- Change 1 bit of input, and the output will change completely.
- What are the odds of finding 256 random bits that has a numerical value below 2^250?
- How would you prove your work?

Proof-Of-Work Chain



```
  "id":
"00000000000000000008692864d3247c5d6c5a389353ec98343bbe84f0142afb",
  "height": 750867,
  "version": 536870912,
  "timestamp": 1661323786,
  "tx_count": 2654,
  "size": 1533321,
  "weight": 3993447,
  "merkle_root":
"490f4da18b11ea3d45cbc54ddaf4f237ae8726be5527e0df35e1a9d58eecc80a",
  "previousblockhash":
"00000000000000000006b77f478f544ebc513ec268f5a62d4a8cc73f3f04ee7a",
  "mediantime": 1661320505,
  "nonce": 657930235,
  "bits": 386526600,
  "difficulty": 28351606743493
```

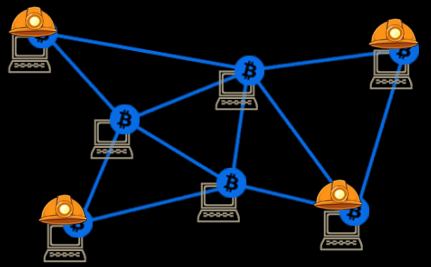
Peer-to-Peer Network



- Each new transaction is broadcast to all peers in the network.
- Each peer collects transactions, validates them, and passes them along.
- Nodes also cache new transactions in memory (known as the mempool), and use them to quickly validate new blocks.

Image source: gbhackers.com

Producing Blocks



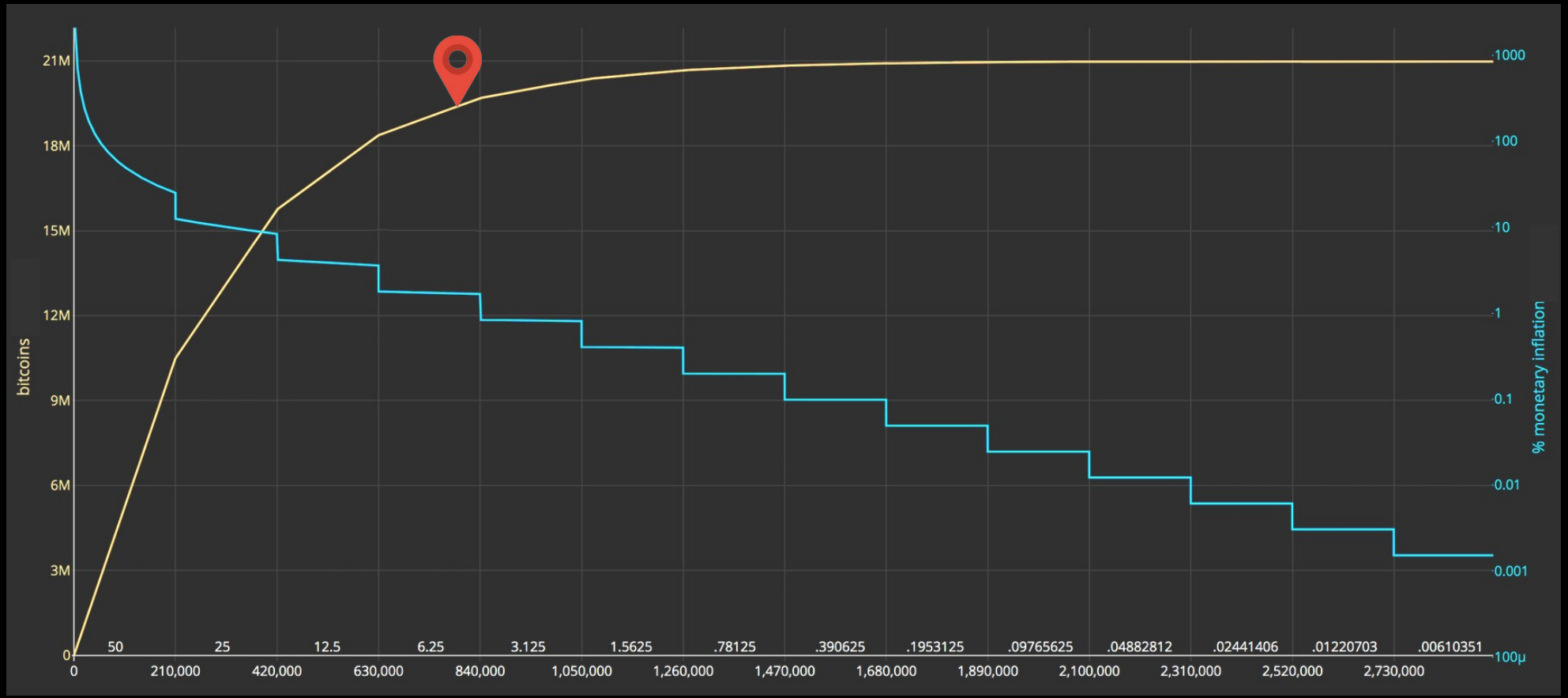
- Collect new transactions into a block.
- Try to calculate a hash for that block that is below a certain value.
- When a hash is found, broadcast your block to the network.
- Other nodes will check the block to see if your block is valid, then add it to the tip of their chain.

Image source: gbhackers.com

Incentive / Reward

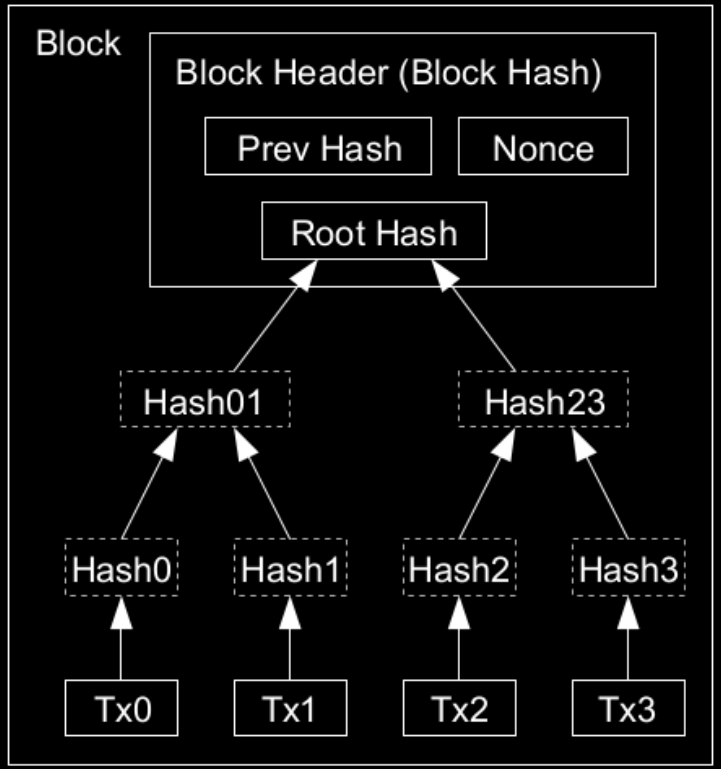
$$\frac{\sum_{i=0}^{32} 210000 \times \left[\frac{50 \times 10^8}{2^i}\right]}{10^8} \approx 21{,}000{,}000$$

Distribution / Inflation

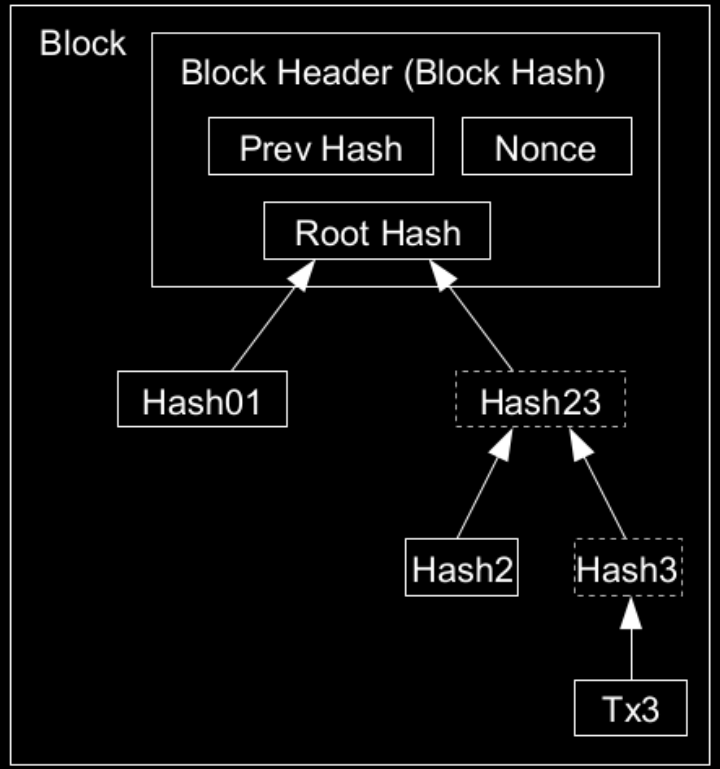


Source: https://cvj.ch/wp-content/uploads/2020/05/Bitcoin-Halving-Chart.png

Reclaiming Disk Space

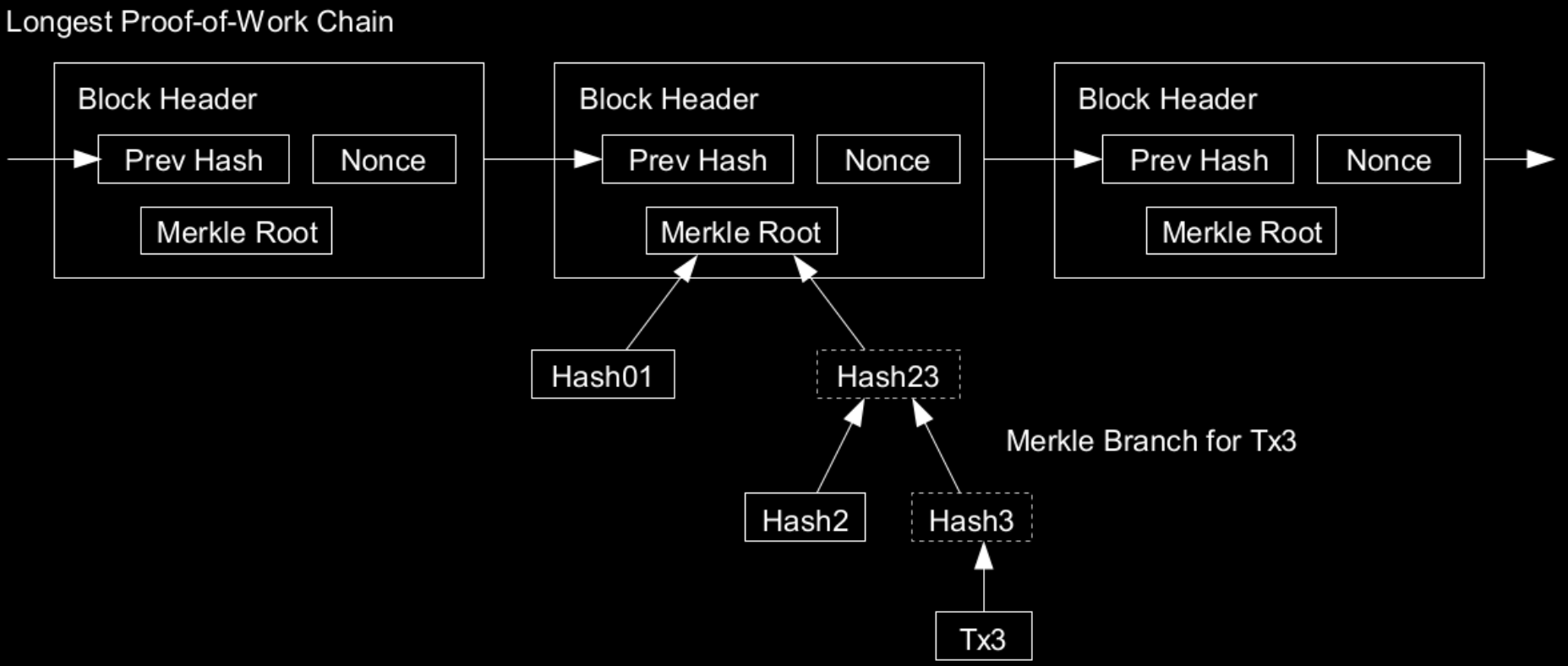


Transactions Hashed in a Merkle Tree

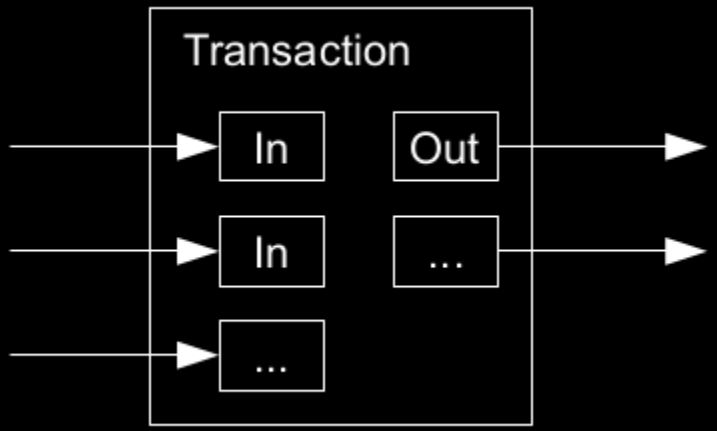


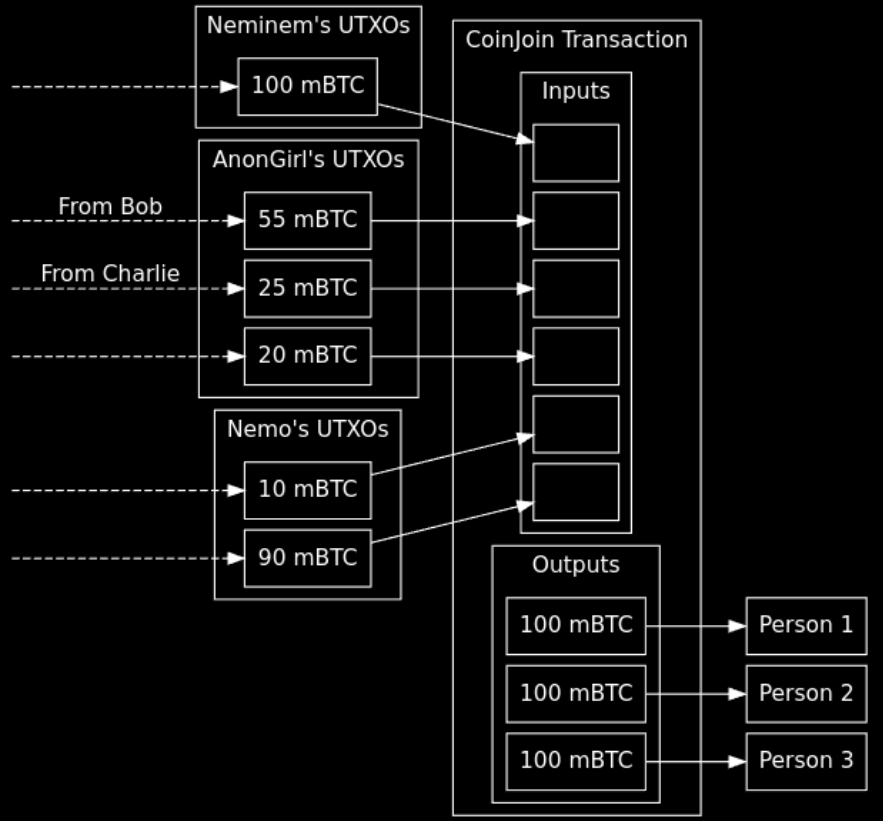
After Pruning Tx0-2 from the Block

Simple Payment Verification

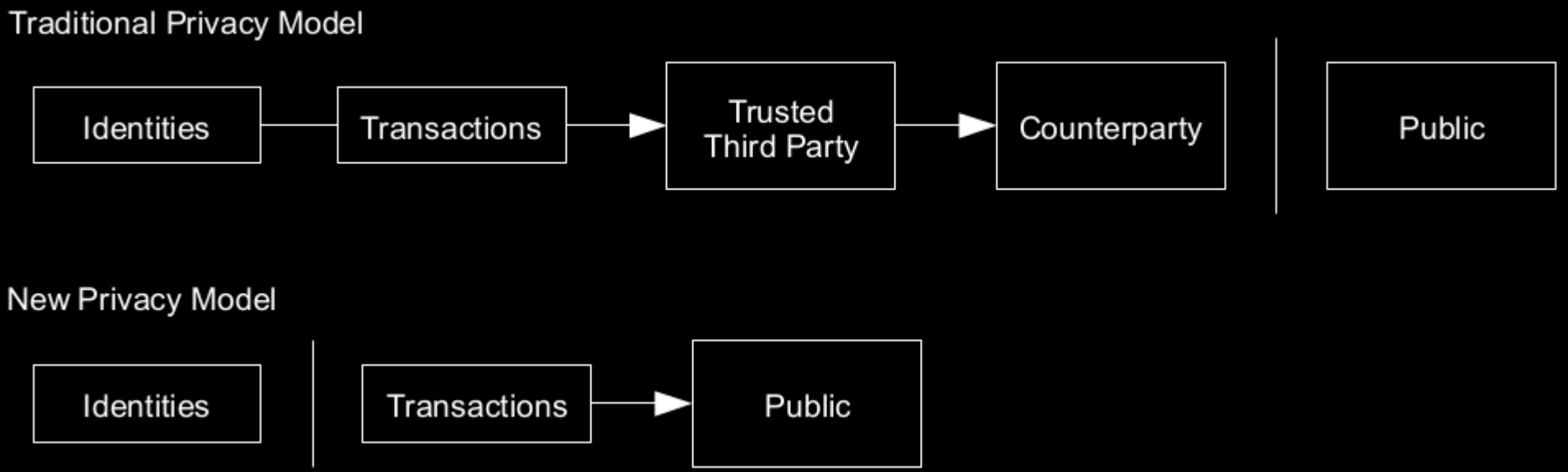


Combine and Split Values





Privacy Model



The Byzantine Generals Problem

LESLIE LAMPORT, ROBERT SHOSTAK, and MARSHALL PEASE
SRI International

Reliable computer systems must handle malfunctioning components that give conflicting information to different parts of the system. This situation can be expressed abstractly in terms of a group of generals of the Byzantine army camped with their troops around an enemy city. Communicating only by messenger, the generals must agree upon a common battle plan. However, one or more of them may be traitors who will try to confuse the others. The problem is to find an algorithm to ensure that the loyal generals will reach agreement. It is shown that, using only oral messages, this problem is solvable if and only if more than two-thirds of the generals are loyal; so a single traitor can confound two loyal generals. With unforgeable written messages, the problem is solvable for any number of generals and possible traitors. Applications of the solutions to reliable computer systems are then discussed.

Categories and Subject Descriptors: C.2.4. [**Computer-Communication Networks**]: Distributed Systems—*network operating systems*; D.4.4 [**Operating Systems**]: Communications Management—*network communication*; D.4.5 [**Operating Systems**]: Reliability—*fault tolerance*

General Terms: Algorithms, Reliability

Additional Key Words and Phrases: Interactive consistency

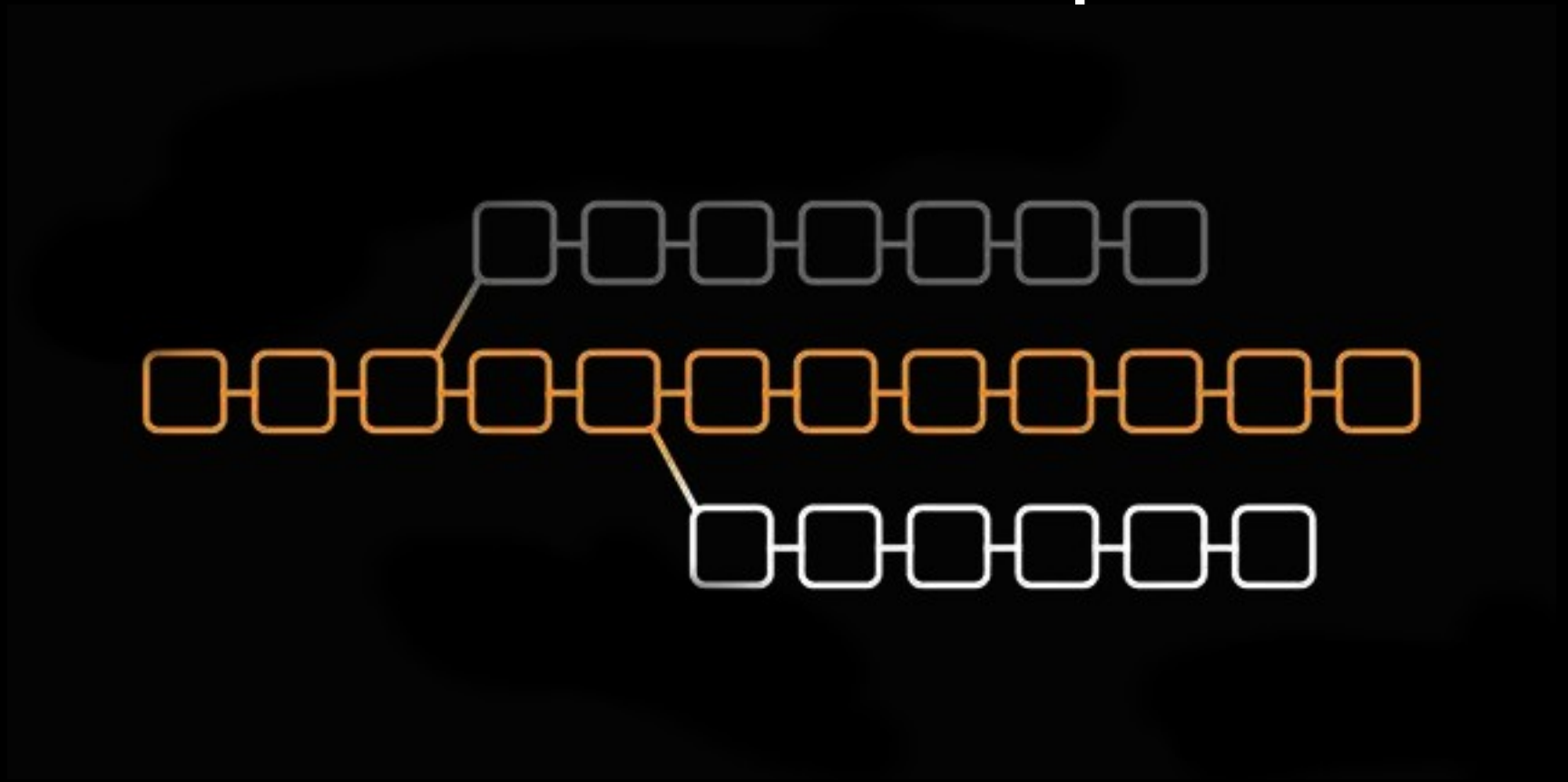
Byzantine Fault

- A faulty device can report conflicting states to different peers (true and false).
- If 33% or more peers are causing conflict, the system will fail to reach consensus.
- Distributed systems require 3n + 1 valid peers in order to achieve consensus.

Calculations

- Attacks on the blockchain cannot steal or create coins, only try to double-spend.
- Even with luck, attackers with < 50% of hash generation will fall behind the honest chain.
- This solves the previous 3n+1 Generals problem, increasing fault tolerance to 49%!

Forks / Chain Splits



Source: Coin Central

Special Thanks

- Satoshi Nakamoto
- Super Testnet
- Tristan Bietch
- Everyone at Pleb Lab!

Download a copy of all slides and resources here:
https://github.com/cmdruid/bitcoin-programming