Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto, 2008

Joerg Osterrieder

Digital Finance

2025

The Abstract and the Problem

Abstract (Nakamoto, 2008)

"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."

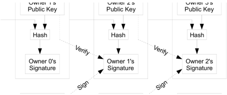
Key Insight from the Introduction

"What is needed is an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party."

Core Building Blocks: Transactions, Timestamps, and Proof-of-Work

Transactions

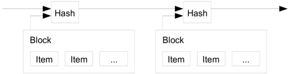
"We define an electronic coin as a chain of digital signatures."



Each owner signs a hash of the previous transaction and the next owner's public key.

Timestamp Server

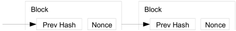
"Each timestamp includes the previous timestamp in its hash, forming a chain."



A timestamp server hashes blocks of items and publishes the hash widely.

Proof-of-Work

"The proof-of-work involves scanning for a value that when hashed, such as with SHA-256, the hash begins with a number of zero bits."



One-CPU-one-vote ensures majority decision making.

Running the Network: Steps, Incentive, and Disk Space

Network Protocol (Section 5)

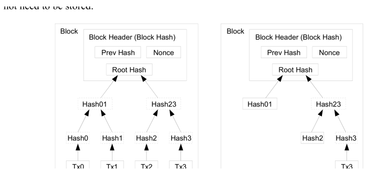
1. New transactions are broadcast to all nodes.
2. Each node collects new transactions into a block.
3. Each node works on finding a difficult proof-of-work for its block.
4. When a node finds a proof-of-work, it broadcasts the block to all nodes.
5. Nodes accept the block only if all transactions in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

Incentive (Section 6)

"The steady addition of a constant of amount of new coins is analogous to gold miners expending resources to add gold to circulation. In our case, it is CPU time and electricity that is expended."

Reclaiming Disk Space (Section 7)

"Transactions are hashed in a Merkle Tree [7][2][5], with only the root included in the block's hash."

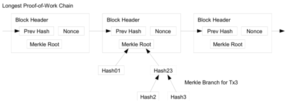


Old blocks can be compacted by stubbing off spent branches – interior hashes do not need to be stored.

Lightweight Clients, Value Transfer, and Privacy

Simplified Payment Verification

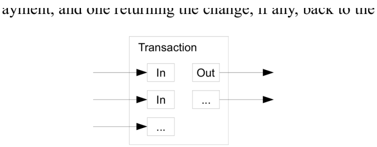
“A user only needs to keep a copy of the block headers of the longest proof-of-work chain.”



Verification without running a full node – query the Merkle branch linking a transaction to its block.

Combining and Splitting Value

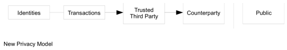
“To allow value to be split and combined, transactions contain multiple inputs and outputs.”


ayment, and one returning the change, if any, back to the

Typically: one or two inputs from prior transactions, two outputs (payment + change).

Privacy Model

“Privacy can still be maintained by breaking the flow of information in another place: by keeping public keys anonymous.”



New key pair per transaction recommended. Public sees amounts, not identities.

Security Analysis and Conclusion

Attacker Catching Up – Gambler's Ruin (Section 11)

"The probability of an attacker catching up from a given deficit is analogous to a Gambler's Ruin problem."

$$q_z = \begin{cases} 1 & \text{if } p \leq q \\ (q/p)^z & \text{if } p > q \end{cases}$$

Probability attacker catches up ($q = 0.1$):

z	P
0	1.0000000
1	0.2045873
2	0.0509779
3	0.0131722
4	0.0034552
5	0.0009137
6	0.0002428
7	0.0000647
8	0.0000173
9	0.0000046
10	0.0000012

Conclusion (Section 12)

"We have proposed a system for electronic transactions without relying on trust. We started with the usual framework of coins made from digital signatures, which provides strong control of ownership, but is incomplete without a way to prevent double-spending. To solve this, we proposed a peer-to-peer network using proof-of-work to record a public history of transactions that quickly becomes computationally impractical for an attacker to change if honest nodes control a majority of CPU power."

Key Design Principles

- **No trusted third party** – cryptographic proof replaces institutional trust
- **One-CPU-one-vote** – majority hash power determines truth
- **Incentive-compatible** – honest behavior is more profitable than attacking
- **Minimal structure** – nodes join/leave freely, longest chain wins

S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.

`bitcoin.org/bitcoin.pdf`