Security Without a Node

Darren Tapp

Dr. Tapp's Mathematical Endeavors LLC

Free State Blockchain Digital Assets Conference

Hash Function

What is a hash function?

- A function that takes any digital information and outputs a large number.
- It's *very* difficult to predict the output from the input.
- Given the output, it's hard to determine the input.
- Can be easy to compute or hard to compute.
- Used in cryptography and Bitcoin security.

Hash Function

What is a hash function?

- A function that takes any digital information and outputs a large number.
- It's *very* difficult to predict the output from the input.
- Given the output, it's hard to determine the input.
- Can be easy to compute or hard to compute.
- Used in cryptography and Bitcoin security.

Hash Function

What is a hash function?

- A function that takes any digital information and outputs a large number.
- It's *very* difficult to predict the output from the input.
- Given the output, it's hard to determine the input.
- Can be easy to compute or hard to compute.
- Used in cryptography and Bitcoin security.

Hash Function

What is a hash function?

- A function that takes any digital information and outputs a large number.
- It's *very* difficult to predict the output from the input.
- Given the output, it's hard to determine the input.
- Can be easy to compute or hard to compute.
- Used in cryptography and Bitcoin security.

Hash Function

What is a hash function?

- A function that takes any digital information and outputs a large number.
- It's *very* difficult to predict the output from the input.
- Given the output, it's hard to determine the input.
- Can be easy to compute or hard to compute.
- Used in cryptography and Bitcoin security.

Hash Function

What is a hash function?

- A function that takes any digital information and outputs a large number.
- It's *very* difficult to predict the output from the input.
- Given the output, it's hard to determine the input.
- Can be easy to compute or hard to compute.
- Used in cryptography and Bitcoin security.

Satoshi

Citation

Security without having a full node goes back to the original paper from Satoshi.

Notice the following screen shot from Satoshi's paper.

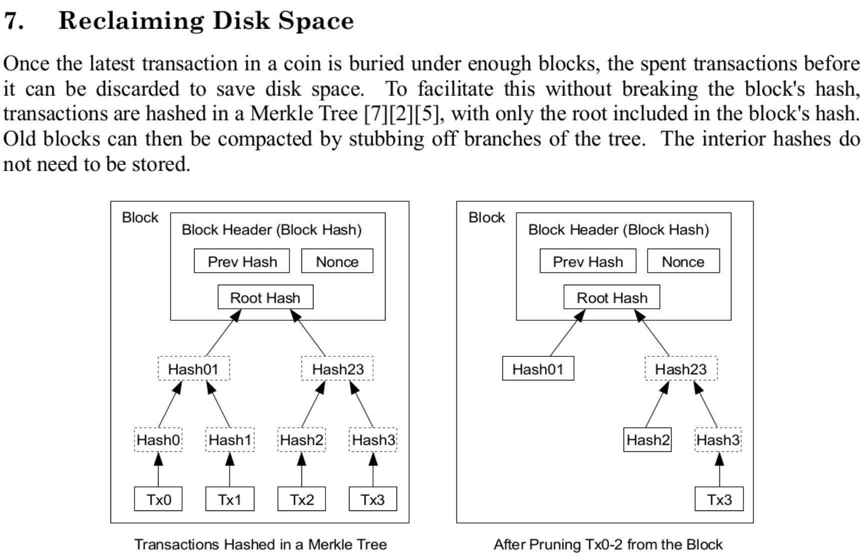
Satoshi

Citation

Security without having a full node goes back to the original paper from Satoshi.

Notice the following screen shot from Satoshi's paper.

Merkle Trees



7. Reclaiming Disk Space

Once the latest transaction in a coin is buried under enough blocks, the spent transactions before it can be discarded to save disk space. To facilitate this without breaking the block's hash, transactions are hashed in a Merkle Tree [7][2][5], with only the root included in the block's hash. Old blocks can then be compacted by stubbing off branches of the tree. The interior hashes do not need to be stored.

Transactions Hashed in a Merkle Tree

After Pruning Tx0-2 from the Block

Bitcoin Blocks

Bitcoin blocks have:

- A block header
- List of transactions

Bitcoin Blocks

Bitcoin blocks have:

- A block header
- List of transactions

Bitcoin Blocks

Bitcoin blocks have:

- A block header
- List of transactions

Block Headers

Block headers include:

- Version
- Hash of previous block header
- Merkle root for transactions in the block
- Time
- Difficulty
- Nonce

Block Headers

Block headers include:

- Version
- Hash of previous block header
- Merkle root for transactions in the block
- Time
- Difficulty
- Nonce

Block Headers

Block headers include:

- Version
- Hash of previous block header
- Merkle root for transactions in the block
- Time
- Difficulty
- Nonce

Block Headers

Block headers include:

- Version
- Hash of previous block header
- Merkle root for transactions in the block
- Time
- Difficulty
- Nonce

Block Headers

Block headers include:

- Version
- Hash of previous block header
- Merkle root for transactions in the block
- Time
- Difficulty
- Nonce

Block Headers

Block headers include:

- Version
- Hash of previous block header
- Merkle root for transactions in the block
- Time
- Difficulty
- Nonce

Block Headers

Block headers include:

- Version
- Hash of previous block header
- Merkle root for transactions in the block
- Time
- Difficulty
- Nonce

Block Headers

Block headers include:

- Version
- Hash of previous block header
- **Merkle root for transactions in the block**
- Time
- Difficulty
- Nonce

Proof of Work

Six valid proof of work blocks require about the energy in 4,000 gallons of gasoline.

The Bitcoin (SegWit) network requires this amount of energy every hour.

This makes the proof of work hard to spoof.

Proof of Work

Six valid proof of work blocks require about
the energy in 4,000 gallons of gasoline.
The Bitcoin (SegWit) network requires this amount of energy every hour.
This makes the proof of work hard to spoof.

Proof of Work

Six valid proof of work blocks require about
the energy in 4,000 gallons of gasoline.
The Bitcoin (SegWit) network requires this amount of energy every hour.
This makes the proof of work hard to spoof.

Simplified Payment Verification

Simplified Payment Verification works by:

- Checking with several nodes for the longest proof of work chain of block headers.
- Asking for transactions and merkle *proofs* associated with addresses.
- This is hard to spoof.
- Vulnerability of lying by omission addressed by querying several nodes.

Simplified Payment Verification

Simplified Payment Verification works by:

- Checking with several nodes for the longest proof of work chain of block headers.
- Asking for transactions and merkle *proofs* associated with addresses.
- This is hard to spoof.
- Vulnerability of lying by omission addressed by querying several nodes.

Simplified Payment Verification

Simplified Payment Verification works by:

- Checking with several nodes for the longest proof of work chain of block headers.
- Asking for transactions and merkle *proofs* associated with addresses.
- This is hard to spoof.
- Vulnerability of lying by omission addressed by querying several nodes.

Simplified Payment Verification

Simplified Payment Verification works by:

- Checking with several nodes for the longest proof of work chain of block headers.
- Asking for transactions and merkle *proofs* associated with addresses.
- This is hard to spoof.
- Vulnerability of lying by omission addressed by querying several nodes.

Simplified Payment Verification

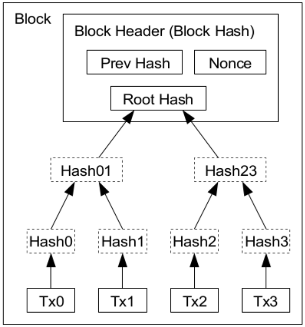
Simplified Payment Verification works by:

- Checking with several nodes for the longest proof of work chain of block headers.
- Asking for transactions and merkle *proofs* associated with addresses.
- This is hard to spoof.
- Vulnerability of lying by omission addressed by querying several nodes.

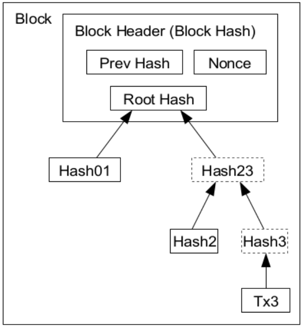
Satoshi Revisited

7. Reclaiming Disk Space

Once the latest transaction in a coin is buried under enough blocks, the spent transactions before it can be discarded to save disk space. To facilitate this without breaking the block's hash, transactions are hashed in a Merkle Tree [7][2][5], with only the root included in the block's hash. Old blocks can then be compacted by stubbing off branches of the tree. The interior hashes do not need to be stored.



Transactions Hashed in a Merkle Tree



After Pruning Tx0-2 from the Block

Thank You

Thank You!
Any Questions?