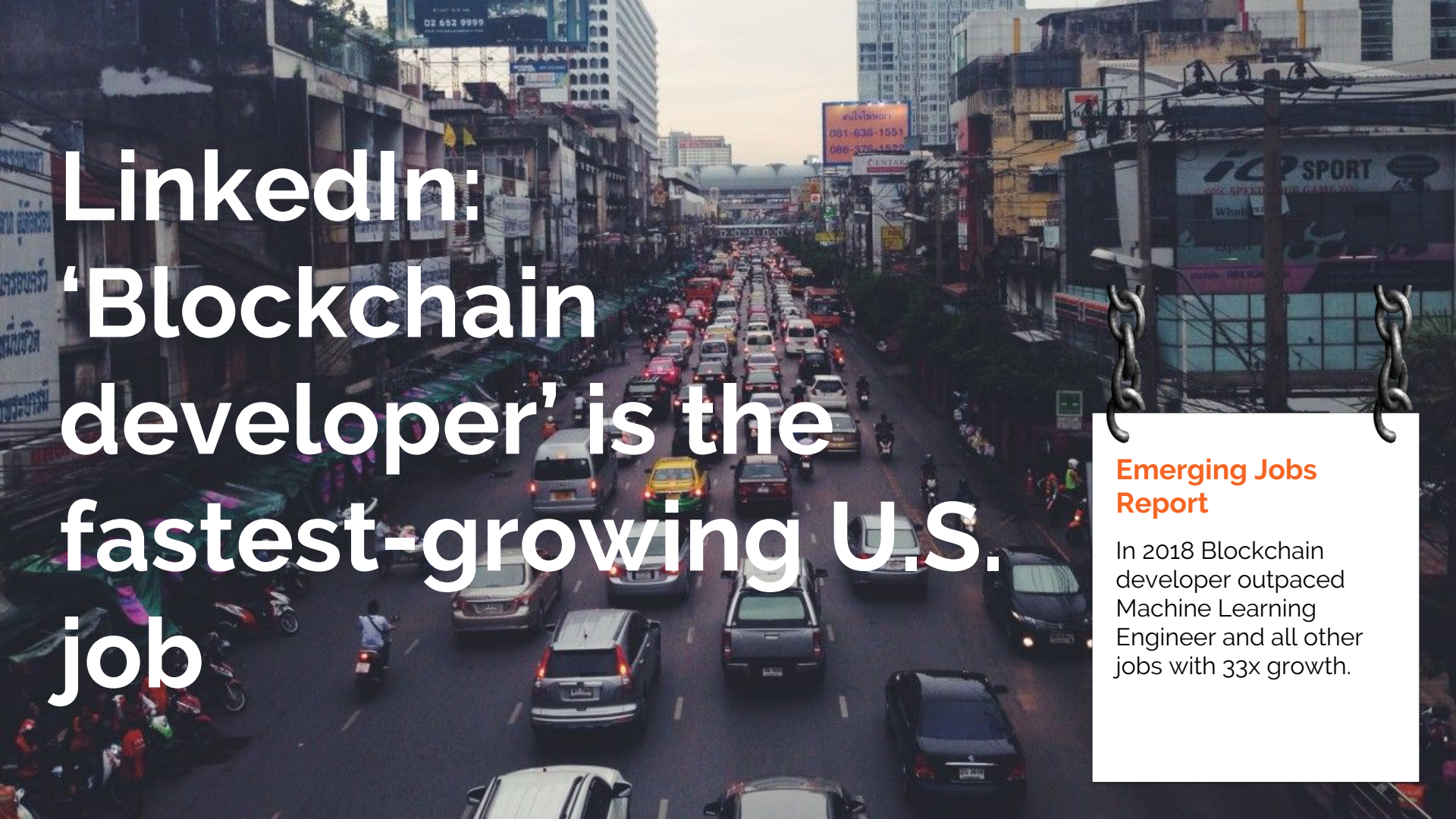

Blockchain Fundamentals:

**Creating your own
Blockchain**

Jordan Baczuk



LinkedIn: 'Blockchain developer' is the fastest-growing U.S. job

Emerging Jobs Report

In 2018 Blockchain developer outpaced Machine Learning Engineer and all other jobs with 33x growth.



Who should take
this course?

1.2 What is Blockchain?

Banking

Real Estate

Law

Healthcare

Education

Voting

Security

Government

Sharing Economy

Charities

1. Blockchain

A **data structure** of time ordered events on which nodes in a distributed network can agree based on a consensus algorithm.

→ **Data Structure**

A chain of blocks linked by references to the previous block's hash

→ **Time Order**

Order of events is established

→ **Consensus**

Network participants agree on what is a valid block. E.g. based on a proof-of-work (hash value threshold).

Block 0

Header

Version 01000000 (1)

Previous Block Hash

0x00000000000000000000000000000000
00000000000000000000000000000000
000000 (no previous block)

Transaction Merkle Hash

0x3ba3edfd7a7b12b27ac72c3e67768f6
17fc81bc3888a51323a9fb8aa4b1e5e4a

Time 1296688602 (2/2/11 11:16)

Bits 0x207fffff (PoW target)

Nonce 0x02000000

Hash

★ 0x0f9188f13cb7b2c71f2a335e3a4fc328b
f5beb436012afca590b1a11466e2206

Block 1

Header

Version 01000000 (1)

Previous Block Hash

★ 0x0f9188f13cb7b2c71f2a335e3a4fc328b
f5beb436012afca590b1a11466e2206

Transaction Merkle Hash

0x44aa82c19492f2b2bdbd379618e3aca
0af5b120df4864daaf28623bb34ec4bed

Time 1538228638 (9/29/18 1:43)

Bits 0x207fffff (PoW target)

Nonce 0x01000000

Hash

0x172ec34bd4e15ce07b48cc82b3bf4b0
3b24f66f37e7c68e61a51fbbd63fc2c92

Bitcoin for Individuals

Bitcoin is the easiest way to transact at a very low cost.



Mobile payments made easy



**Security and control over your
money**



Works everywhere, anytime

Bitcoin P2P e-cash paper

Satoshi Nakamoto [satoshi at vistomail.com](mailto:satoshi%40vistomail.com)

Fri Oct 31 14:10:00 EDT 2008

- Previous message: [Fw: SHA-3 lounge](#)
- **Messages sorted by:** [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#)

I've been working on a new electronic cash system that's fully peer-to-peer, with no trusted third party.

The paper is available at:
<http://www.bitcoin.org/bitcoin.pdf>

The main properties:
Double-spending is prevented with a peer-to-peer network.
No mint or other trusted parties.
Participants can be anonymous.
New coins are made from Hashcash style proof-of-work.
The proof-of-work for new coin generation also powers the network to prevent double-spending.

Bitcoin: A Peer-to-Peer Electronic Cash System

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without the burdens of going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted 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 honest nodes control the most CPU power on the network, they can generate the longest chain and outpace any attackers. The network itself requires minimal structure. Messages are broadcasted 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.

Full paper at:
<http://www.bitcoin.org/bitcoin.pdf>

Satoshi Nakamoto

The Cryptography Mailing List
Unsubscribe by sending "unsubscribe cryptography" to majordomo at metzdowd.com

21.871

Insights

<https://bitcoincore.org/en/download>

cryptography

 MIT

Clone or download ▾

Latest commit 5da08e0 3 hours ago

14 days ago

12 days ago

5 months ago

2 months ago

5 days ago

10 days ago

6 days ago

2 days ago

28 days ago

6 hours ago

3 hours ago

2 months ago

4 years ago

3 months ago

29 days ago

6 days ago

8 days ago

10 days ago

Section 2: Fundamentals

2: Fundamentals

2.1 Blocks

2.2 Transactions

2.3 Script

2.4 Addresses

2.5 Mining

[illegible]



Block Structure

1. Block Header (80B)

The metadata that is hashed to produce the block hash

2. Transaction Counter (var int)

The number of transactions in the block

3. Transactions

The raw transaction data for all transactions in the block

1. Block Header

→ **Version (4B)**

Block format version

→ **Previous Block Hash (32B)**

Hash of the preceding block

→ **Merkle Root Hash (32B)**

Hash of all transactions

→ **Time (4B)**

Epoch timestamp

→ **Bits (4B)**

Compact format of the hash target

→ **Nonce (4B)**

Number incremented during mining

```
$ bitcoin-cli -regtest getblockhash 0
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206
```

```
$ bitcoin-cli -regtest getblock
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206 0
0100000000000000000000000000000000000000000000000000000000000000
0003ba3edfd7a7b12b27ac72c3e67768f617fc81bc3888a51323a9fb8aa4b1e5e4ada
e5494dffff7f200200000001010000000100000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
65732030332f4a616e2f32303039204368616e63656c6c6f72206f6e206272696e6b2
06f66207365636f6e64206261696c6f757420666f722062616e6b73fffffffff0100f2
052a01000000434104678afd0fe5548271967f1a67130b7105cd6a828e03909a6796
2e0ea1f61deb649f6bc3f4cef38c4f35504e51ec112de5c384df7ba0b8d578a4c702b
6bf11d5fac00000000
```


SHA256

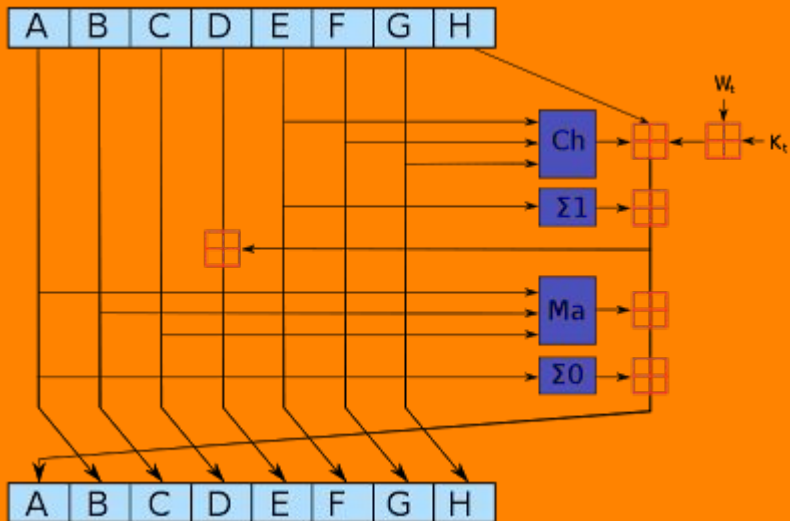


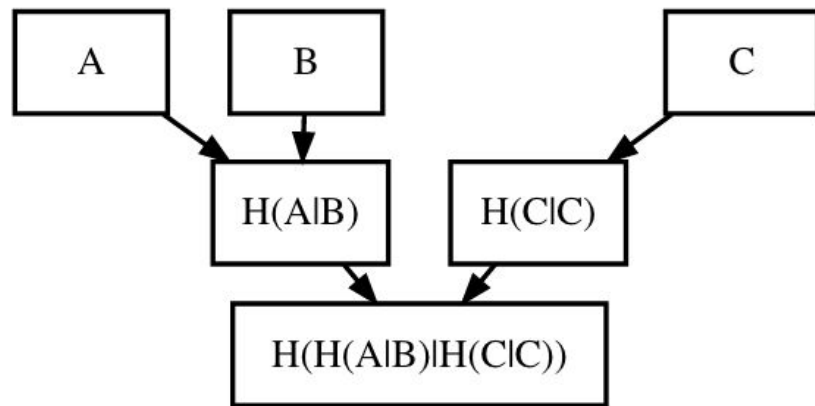
Diagram created by kockmeyer, CC BY-SA 3.0.

Merkle Trees

Row 1: Transaction hashes (TXIDs)
(A is coinbase; C can spend output from B)

Row 2: Hashes of paired TXIDs

Merkle root



Example Merkle Tree Construction [Hash function $H() = \text{SHA256}(\text{SHA256}())$]

nBits - compact format

0x181bc330 →	0x1bc330	*	256	^	(0x18	-	3)
nBits In Big-Endian Order	Significand (Mantissa)		Base		Exponent		Bytes In Significand

Result: 0x1bc33000

Converting nBits Into A Target Threshold

1. Block Header

- [illegible]

```
$ bitcoin-cli -regtest getblockhash 0
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206

$ bitcoin-cli -regtest getblock
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206 0
01000000
0000000000000000000000000000000000000000000000000000000000000000
3ba3edfd7a7b12b27ac72c3e67768f617fc81bc3888a51323a9fb8aa4b1e5e4a
dae5494d ffff7f20 02000000
0101000000010000000000000000000000000000000000000000000000000000
00000000fffffffff4d04ffff001d0104455468652054696d65732030332f4a616e2f32
303039204368616e63656c6c6f72206f6e206272696e6b206f666207365636f6e64206
261696c6f757420666f722062616e6b73fffffffff0100f2052a01000000434104678a
fdb0fe5548271967f1a67130b7105cd6a828e03909a67962e0ea1f61deb649f6bc3f4
cef38c4f35504e51ec112de5c384df7ba0b8d578a4c702b6bf11d5fac00000000
```

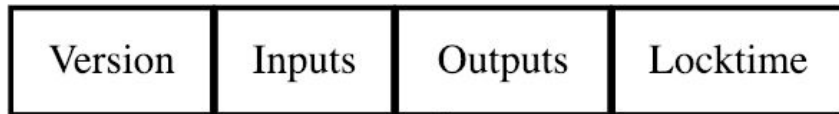
Transactions

331875e7fdc924999784467a13a96d980e04fadc7dc61859f38b279c72242de7			
Inputs		Outputs	
3aYtFTY3H6Vp1NQLTJ uBQ8tvswuKgQ82Ro	14.55036232 BTC	3PzbAbCdefgWBU6kciJ p3tPw7Hmf8QBZE7	22.4577213 BTC
3aYtFTY3H6Vp1NQLTJ uBQ8tvswuKgQ82Ro	14.55022179 BTC	3HrSWAtu1ri3tXsd5P WjP6esCd8ebZcv8y	6.642641 BTC

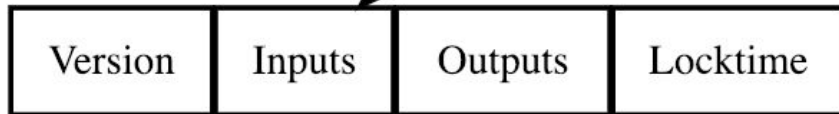
Inputs and Outputs

Each input spends a previous output

The Main Parts Of
Transaction 0



The Main Parts Of
Transaction 1



Each output waits as an Unspent TX Output (UTXO) until a later input spends it

[illegible]

Tx Structure

- **Version (4B)**
Transaction format version
- **Flag (2B Array)**
Optional, used for witness data
- **Input Counter (var int)**
Number of inputs
- **Inputs**
List of transaction inputs to be spent
- **Output Counter (var int)**
Number of outputs
- **Outputs**
List of transaction outputs to send to



Tx Structure cont.

→ **Witnesses**

Optional list of witness data (if flag is set)

→ **Lock time (4B)**

Block height or timestamp that indicates when the tx will be final.

Tx Structure

→ Version (4B)

01000000 (1)

→ Flag (2B Array)

N/A

→ Input Counter (var int)

01 (1)

→ Inputs

00000000000000000000000000000000
00000000000000000000000000000000
0000ffffffff4d...73ffffffff

→ Output Counter (var int)

01 (1)

```
$ bitcoin-cli -regtest getblockhash 0
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206

$ bitcoin-cli -regtest getblock
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206
0
0100000000000000000000000000000000000000000000000000000000000000
000000003ba3edfd7a7b12b27ac72c3e67768f617fc81bc3888a51323a9fb8aa4b
1e5e4adae5494dffff7f200200000001 01000000 01
000000000000000000000000000000000000000000000000000000000000000f
ffffffff4d04ffff001d0104455468652054696d65732030332f4a616e2f323030
39204368616e636556c6c6f72206f6e206272696e6b206f666207365636f6e64206
261696c6f757420666f722062616e6b73ffffffff 01
00f2052a01000000434104678afdb0fe5548271967f1a67130b7105cd6a828e03
909a67962e0ea1f61deb649f6bc3f4cef38c4f35504e51ec112de5c384df7ba0b
8d578a4c702b6bf11d5fac 00000000
```

Tx Structure cont.

→ Outputs

00f2052a01000000434104678afdb0fe5
548271967f1a67130b7105cd6a828e039
09a67962e0ea1f61deb649f6bc3f4cef38
c4f35504e51ec112de5c384df7ba0b8d5
78a4c702b6bf11d5fac

→ Witnesses

N/A

→ Lock time (4B)

00000000 (N/A)

```
$ bitcoin-cli -regtest getblockhash 0
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206

$ bitcoin-cli -regtest getblock
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206
0
0100000000000000000000000000000000000000000000000000000000000000
00000003ba3edfd7a7b12b27ac72c3e67768f617fc81bc3888a51323a9fb8aa4b
1e5e4adae5494dffff7f200200000001 01000000 01
000000000000000000000000000000000000000000000000000000000000000f
ffffffff4d04ffff001d0104455468652054696d65732030332f4a616e2f323030
39204368616e63656c6c6f72206f6e206272696e6b206f666207365636f6e64206
261696c6f757420666f722062616e6b73ffffffff 01
00f2052a01000000434104678afdb0fe5548271967f1a67130b7105cd6a828e03
909a67962e0ea1f61deb649f6bc3f4cef38c4f35504e51ec112de5c384df7ba0b
8d578a4c702b6bf11d5fac 00000000
```

Input Structure

- **Previous tx hash (32B)**
Tx Hash of output to be spent
- **Previous utxo index (4B)**
Index of the specific output (txs can have multiple outputs)
- **scriptSig Length (var int)**
Length of scriptSig in bytes
- **scriptSig**
Executed before the scriptPubKey, which must return true in order to spend
- **Sequence Number**
Used as relative locktime

Input Structure

- **Previous tx hash (32B)**
(all zeroes, N/A for coinbase transaction)
- **Previous utxo index (4B)**
ffffff (N/A for coinbase transaction)
- **scriptSig Length (var int)**
4d (77 Bytes)
- **scriptSig**
04ffff001d0104455468652054696d65732030332f4a616e2f32303...73
- **Sequence Number**
ffffff

```
$ bitcoin-cli -regtest getblockhash 0
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206

$ bitcoin-cli -regtest getblock
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206
0
0100000000000000000000000000000000000000000000000000000000000000
00000003ba3edfd7a7b12b27ac72c3e67768f617fc81bc3888a51323a9fb8aa4b
1e5e4adae5494dffff7f200200000001010000000100
00000000000000000000000000000000000000000000000000000000000000
ffffff 4d
04ffff001d0104455468652054696d65732030332f4a616e2f323030392043686
16e63656c6c6f72206f6e206272696e6b206f66207365636f6e64206261696c6f
757420666f722062616e6b73 fffffff
0100f2052a01000000434104678afdb0fe5548271967f1a67130b7105cd6a828e
03909a67962e0ea1f61deb649f6bc3f4cef38c4f35504e51ec112de5c384df7ba
0b8d578a4c702b6bf11d5fac00000000
```

Coinbase Message

Satoshi Nakamoto

```
$ printf
```

```
"04ffff001d0104455468652054696d65732030332f4a616e  
2f32303039204368616e63656c6c6f72206f6e206272696e6  
b206f66207365636f6e64206261696c6f757420666f722062  
616e6b73" | xxd -r -p && echo
```

```
??EThe Times 03/Jan/2009 Chancellor on brink of  
second bailout for banks
```

Output Structure

→ **Coin value (8B)**

Amount of coin being sent in satoshis
(1e8 satoshis per Bitcoin)

→ **scriptPubKey Length (var int)**

Length of scriptPubKey in bytes

→ **scriptPubKey**

Executed after the scriptSig (provided
when this output is spent in the future)

Output Structure

- **Coin value (8B)**
00f2052a01000000 (50 BTC or 5000000000 satoshis)
- **scriptPubKey Length (var int)**
43 (67 Bytes)
- **scriptPubKey**
4104678afdb0fe5548271967f1a67130b7
105cd6a828e03909a67962e0ea1f61de
b649f6bc3f4cef38c4f35504e51ec112de
5c384df7ba0b8d578a4c702b6bf11d5fa
C

```
$ bitcoin-cli -regtest getblockhash 0
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206

$ bitcoin-cli -regtest getblock
0f9188f13cb7b2c71f2a335e3a4fc328bf5beb436012afca590b1a11466e2206
0
0100000000000000000000000000000000000000000000000000000000000000
00000003ba3edfd7a7b12b27ac72c3e67768f617fc81bc3888a51323a9fb8aa4b
1e5e4adae5494dffff7f20020000000101000000010000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
455468652054696d65732030332f4a616e2f32303039204368616e63656c6c6f7
2206f6e206272696e6b206f66207365636f6e64206261696c6f757420666f7220
62616e6b73fffffffff01 00f2052a01000000 43
4104678afdb0fe5548271967f1a67130b7105cd6a828e03909a67962e0ea1f61d
eb649f6bc3f4cef38c4f35504e51ec112de5c384df7ba0b8d578a4c702b6bf11d
5fac 00000000
```

Script

OP_DUP OP_HASH160 306e2ea1eed91bf66dfe5d94f3957d4ba63bde84
OP_EQUALVERIFY OP_CHECKSIG

2.3 Script

script	stack
-----+-----	
0	
030cfcefa07af9dd6dbe770b87d7dbdd2c31ba7f4fcf8f3a1196d502f13561b046	
OP_DUP	
OP_HASH160	
306e2ea1eed91bf66dfe5d94f3957d4ba63bde84	
OP_EQUALVERIFY	
OP_CHECKSIG	
#0000 0	
btcdeb> step	

Script Example P2PKH

P2PKH is a pay-to-public key-hash script where the coins are sent to the hash of a public key. In order to spend them, the scriptSig must provide the public key and a valid signature using the private key.

```
scriptPubKey (raw): 76a914306e2ea1eed91bf66dfe5d94f3957d4ba63bde8488ac
```

```
scriptPubKey (assembly): OP_DUP OP_HASH160  
PUSHDATA(20)[306e2ea1eed91bf66dfe5d94f3957d4ba63bde84] OP_EQUALVERIFY OP_CHECKSIG
```

Addresses

3J98t1WpEZ73CNmQviecrnyiWrnqRhWNLy

Address Types

- **Pay to Pubkey Hash (P2PKH)**
Hash of a public key
- **Pay to Script Hash (P2SH)**
Hash of a script, must provide redeem script that hashes to the right value
- **Bech32 (Segregated Witness)**
Custom encoding, represents witness scripts

Bitcoin Cryptography

- Uses Elliptic Curve secp256k1
- Private keys can be any 256 bit number from 0x1 to
0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEBAAEDCE6AF48A03BBFD25E8CD0364140
- Currently uses ECDSA for transaction signing
- supports compressed public keys (33 bytes instead of 65 bytes)

Mining

0x00000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f

50 BTC



Block Validation

→ Block Hash

Must be \leq network target

→ Timestamp

- ◆ Greater than median time of last 11 blocks
- ◆ < 2 hours in the future

→ Transactions

- ◆ No double spends
- ◆ Spends are authorized
- ◆ Valid coinbase tx exists

Proof of Work Consensus

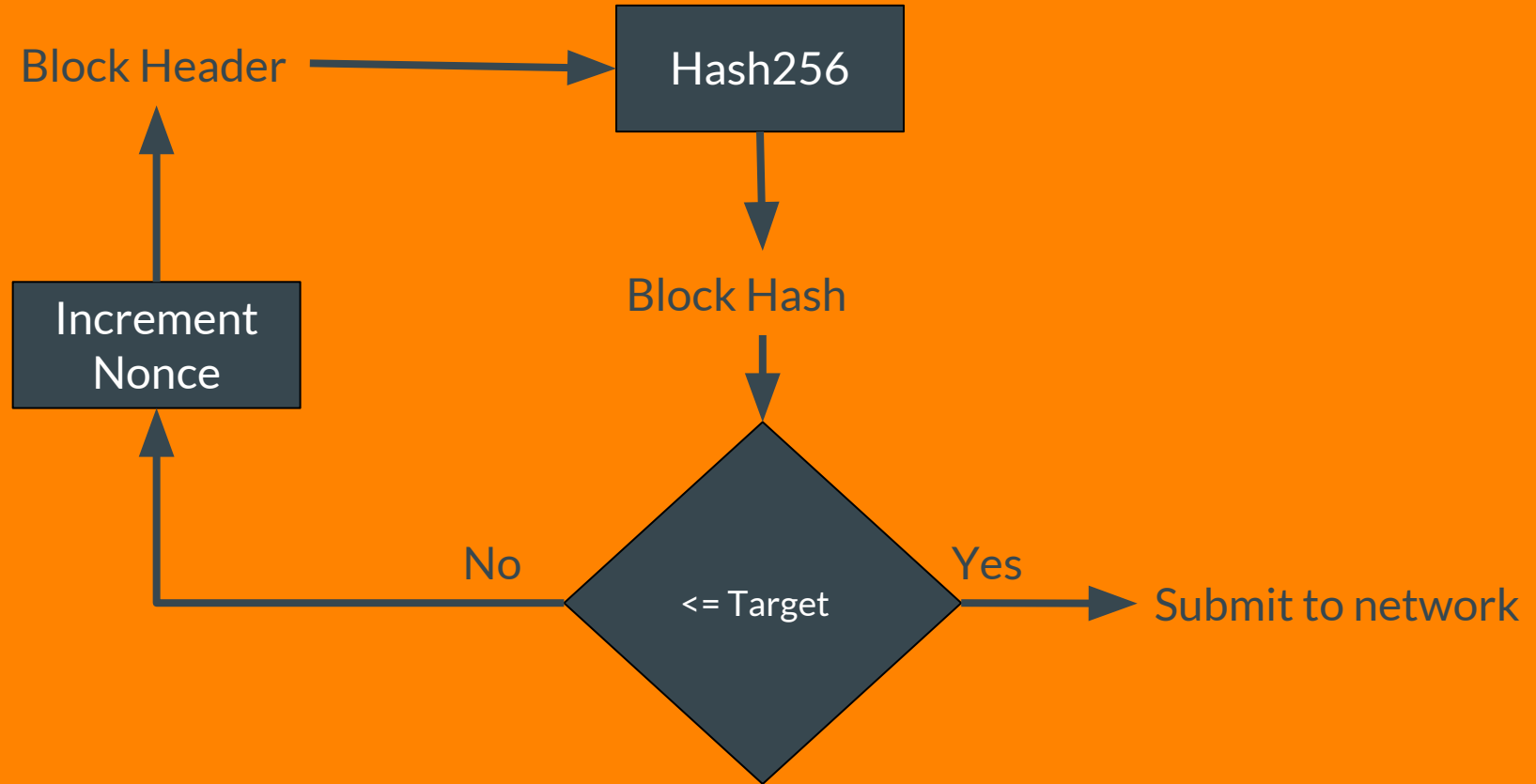
Target:

000000000fff

Example Solution:

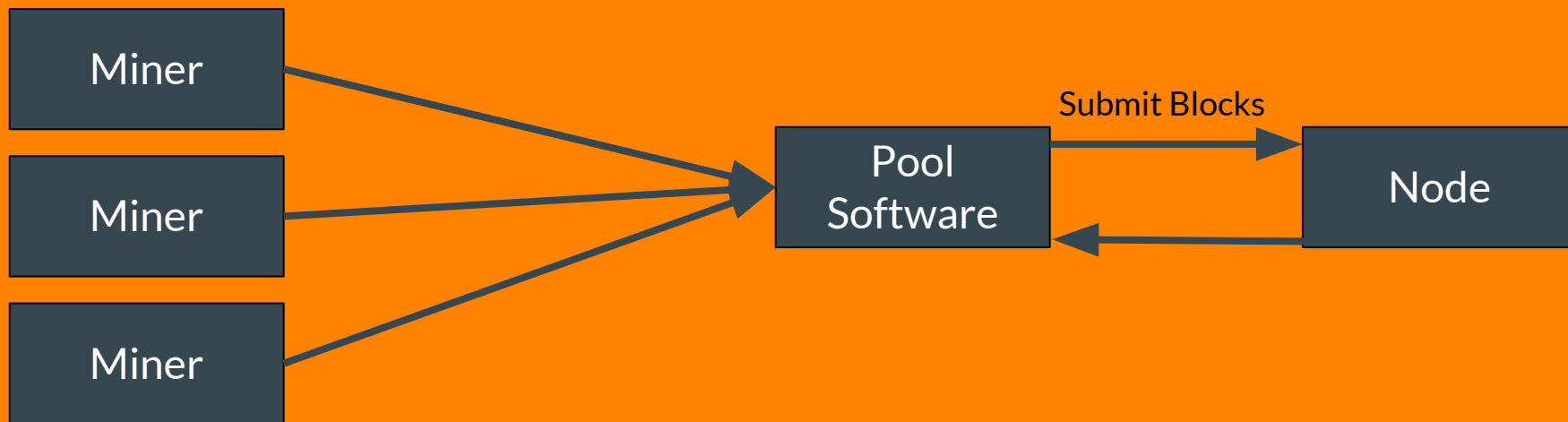
00000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f

Mining Process



Pools and Mining Optimizations

Stratum Protocol



Section 3: Design



Customizations

→ Branding

- ◆ Rename project
- ◆ Change address prefixes

→ Networking

- ◆ Ports, message prefixes, seeds

→ Consensus Rules

- ◆ Max coin supply, block rewards, halving interval
- ◆ Max block size, block time
- ◆ Activate BIPs, clear checkpoint data



Customizations

→ Consensus (cont.)

- ◆ New genesis block
- ◆ Difficulty retargeting interval

→ Standards

- ◆ OP_RETURN data limit

→ Advanced

- ◆ Premine

Resources

- Bitcoin Wiki: <https://en.bitcoin.it/wiki/>
- Bitcoin Stackexchange: <https://bitcoin.stackexchange.com>
- Bitcoin.org: <https://bitcoin.org/en/developer-documentation>
- Learncoin source code: <https://github.com/jbaczuk/learncoin>
- Course Lessons: https://github.com/jbaczuk/blockchain_fundamentals
- Course chatroom: <https://discord.gg/DaREKEP>