



Crypto

Principles of Distributed Systems

Tomáš Faltín

KSI MFF



Bitcoin

History

- Can I trust that the money is authentic?
- Can the digital money be spent only once?
- Can no one else claim my money?
- Paper "Bitcoin: A Peer-to-Peer Electronic Cash System" by Satoshi Nakamoto in 2009



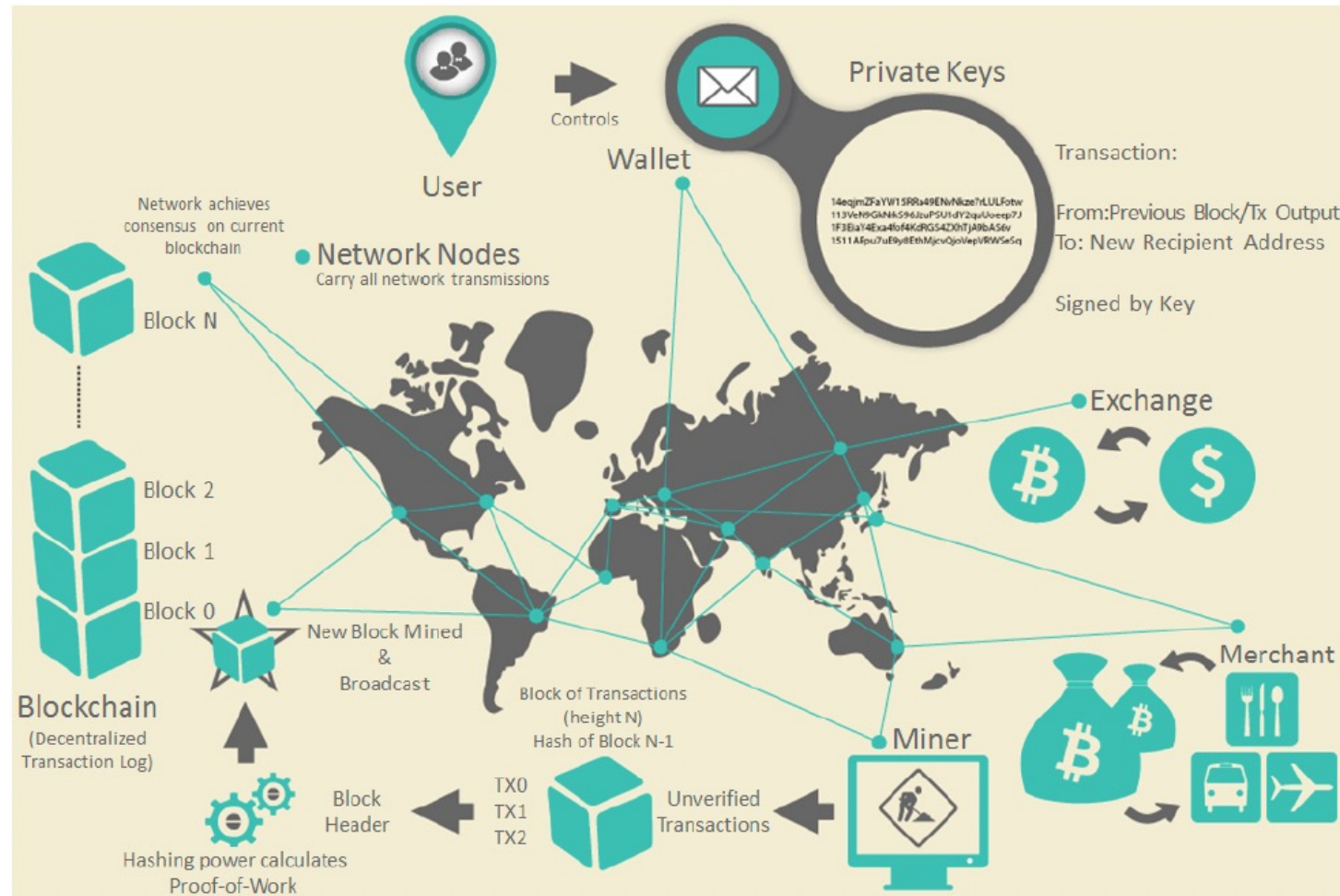
Overview

- Concepts & technologies for digital money ecosystem
- Distributed, P2P system
- bitcoins created through mining every ~10 min
 - Mining = competing to find solution to a mathematical problem while processing transactions
 - Limited to 21 mil. (by 2140)

Bitcoin

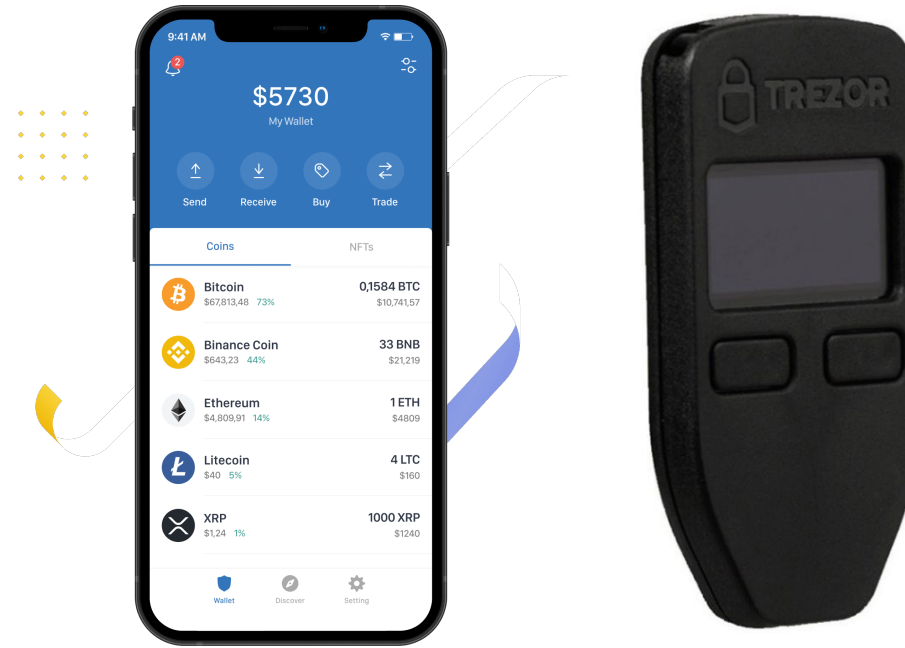
- The Bitcoin protocol - A decentralized P2P network
- Blockchain - A public transaction ledger
- Consensus Rules - A set of rules
- Proof-Of-Work algorithm A mechanism for reaching consensus

High Level



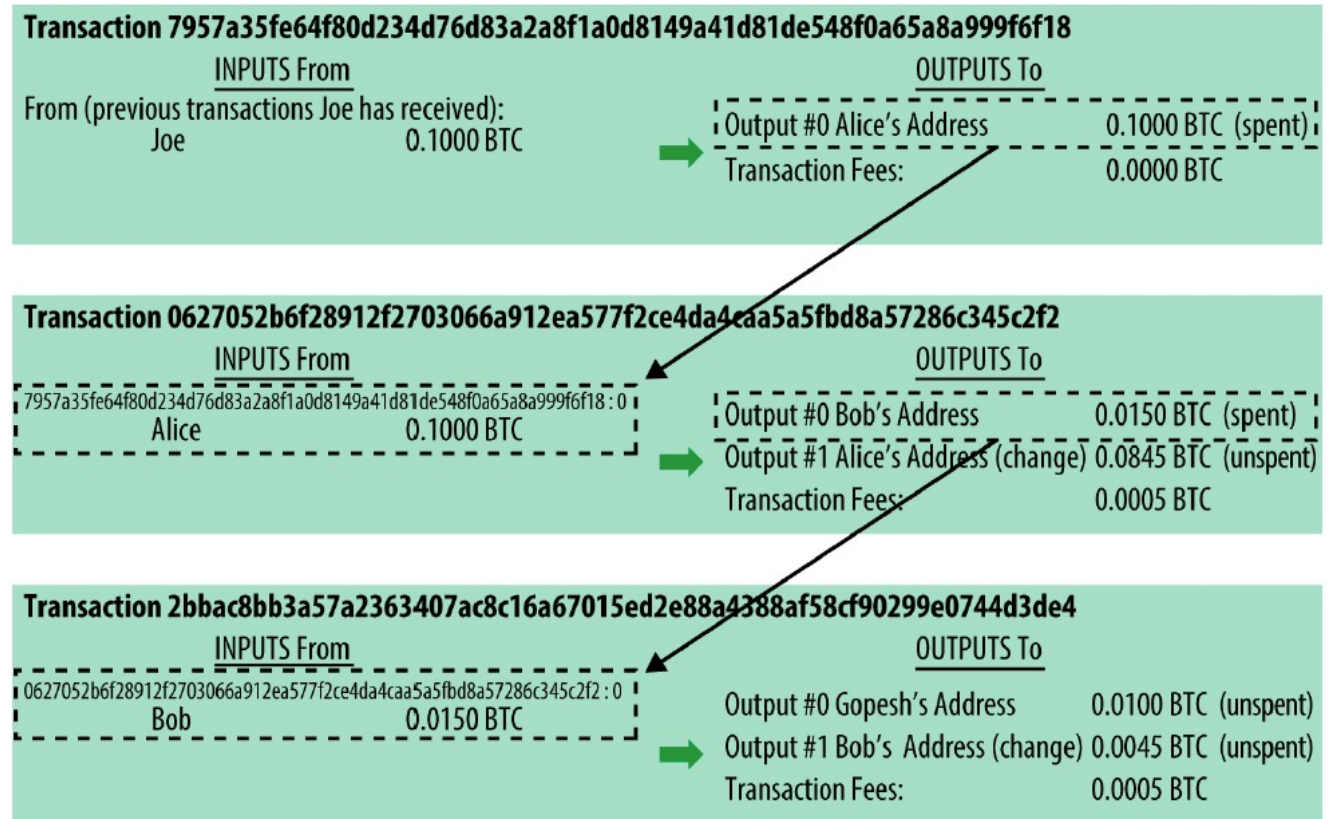
Wallet

- UI to the bitcoin interface
- Desktop, mobile, web, HW, paperwallets, ...
- A bitcoin node that stores and manages keys



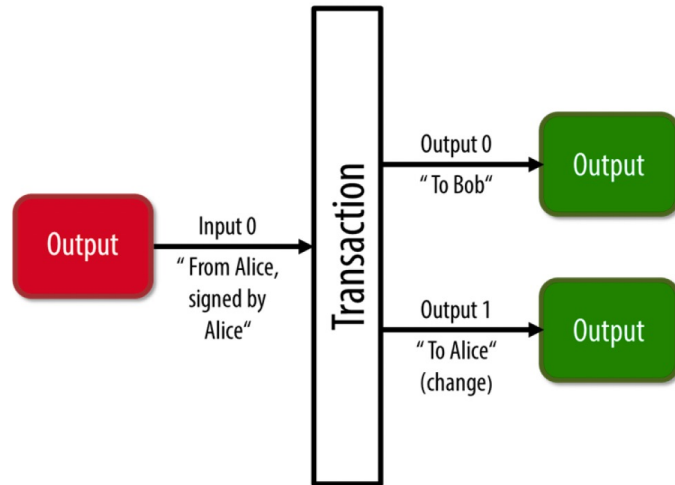
Transactions

- Bitcoin transfer from the owner to another user
- Propagation via flooding mechanism

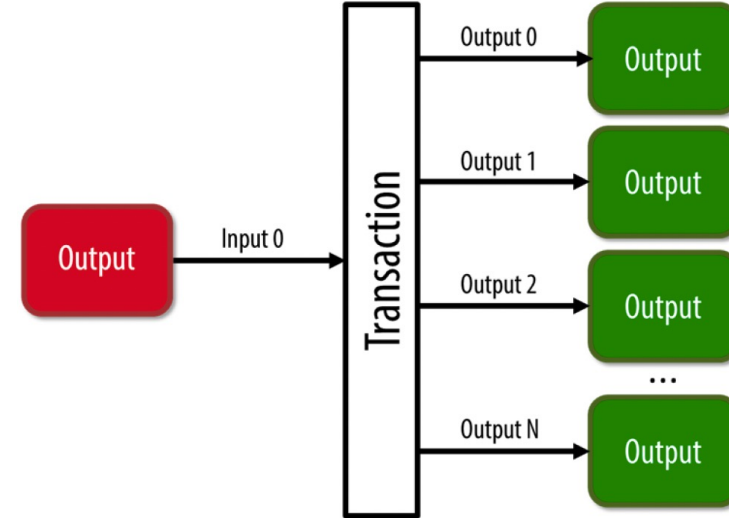


Transactions

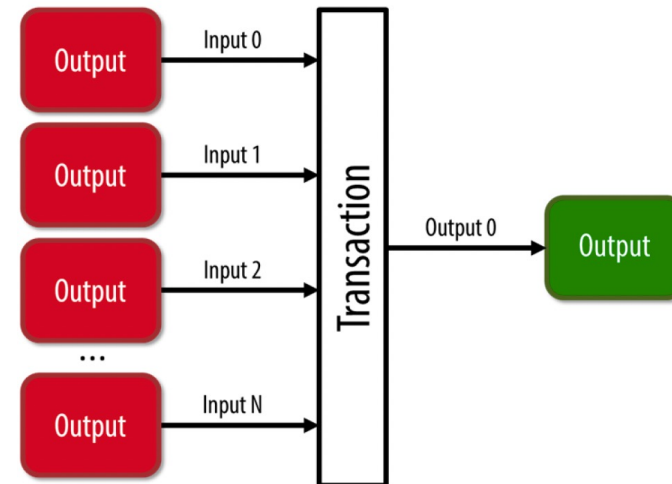
Common Transaction



Distributing Transaction



Aggregating Transaction



Mining

- Process of including transactions into the blockchain
- Validation of transactions by solving a puzzle (Proof-of-Work)
- Creation of new bitcoins (reward, currently 6.25 BTC)

Bitcoin Genesis Block

Raw Hex Version

00000000	01 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00000010	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00000020	00 00 00 00 3B A3 ED FD	7A 7B 12 B2 7A C7 2C 3E;fíýz{.²zÇ,>
00000030	67 76 8F 61 7F C8 1B C3	88 8A 51 32 3A 9F B8 AA	gv.a.È.Ã^ŠQ2:Ÿ,a
00000040	4B 1E 5E 4A 29 AB 5F 49	FF FF 00 1D 1D AC 2B 7C	K.^J)«_IŸŸ...¬+
00000050	01 01 00 00 00 01 00 00	00 00 00 00 00 00 00 00
00000060	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00000070	00 00 00 00 00 00 FF FF	FF FF 4D 04 FF FF 00 1DŸŸŸŸM.ŸŸ..
00000080	01 04 45 54 68 65 20 54	69 6D 65 73 20 30 33 2F	..EThe Times 03/
00000090	4A 61 6E 2F 32 30 30 39	20 43 68 61 6E 63 65 6C	Jan/2009 Chancel
000000A0	6C 6F 72 20 6F 6E 20 62	72 69 6E 6B 20 6F 66 20	lor on brink of
000000B0	73 65 63 6F 6E 64 20 62	61 69 6C 6F 75 74 20 66	second bailout f
000000C0	6F 72 20 62 61 6E 6B 73	FF FF FF FF 01 00 F2 05	or banksŸŸŸŸ..ð.
000000D0	2A 01 00 00 00 43 41 04	67 8A FD B0 FE 55 48 27	*....CA.gŠŸ°pUH'
000000E0	19 67 F1 A6 71 30 B7 10	5C D6 A8 28 E0 39 09 A6	.gñ q0°.\\Ö''(à9.
000000F0	79 62 E0 EA 1F 61 DE B6	49 F6 BC 3F 4C EF 38 C4	ybàê.aþ¶IÖk?Li8Ä
00000100	F3 55 04 E5 1E C1 12 DE	5C 38 4D F7 BA 0B 8D 57	óU.â.Á.þ\8M+ø..W
00000110	8A 4C 70 2B 6B F1 1D 5F	AC 00 00 00 00	ŠLp+kñ._¬....



Going Into Details

Bitcoin Ownership



Digital keys

Private secret key

Public key



Bitcoin address

Generated from the public key



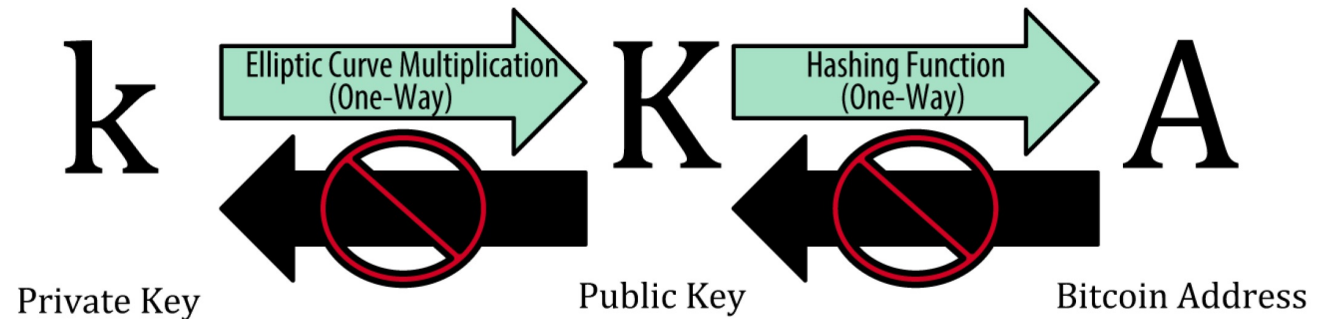
Digital signature

(Most) transactions require to be included in blockchain

Generated from the secret key

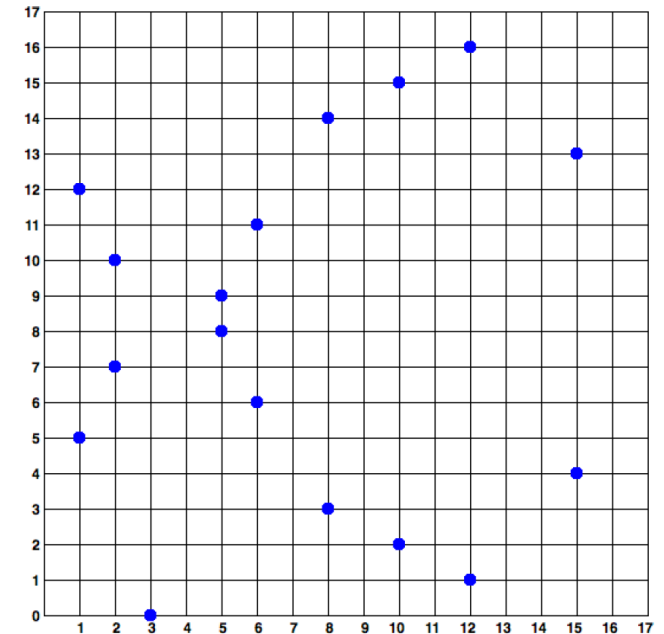
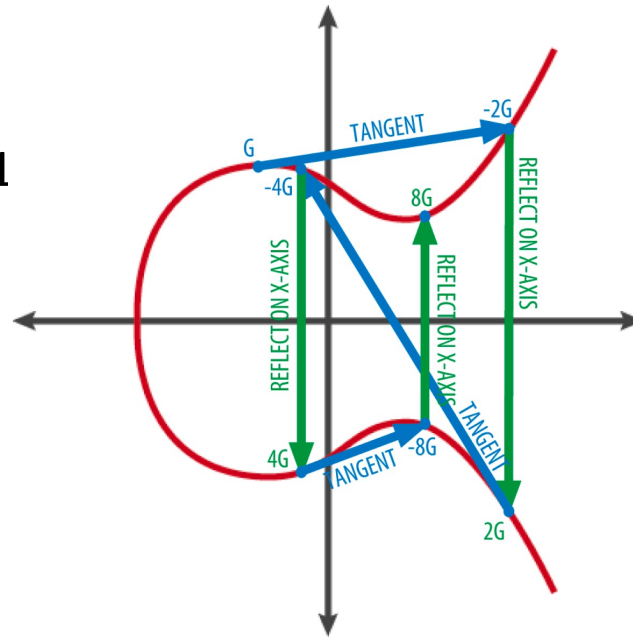
Public And Private Keys

- Need an irreversible function (**elliptic curve multiplication**, prime number exponentiations, ...)
- Digital signatures through asymmetric cryptography
- Private key = a number between $<1, 2^{256}$)



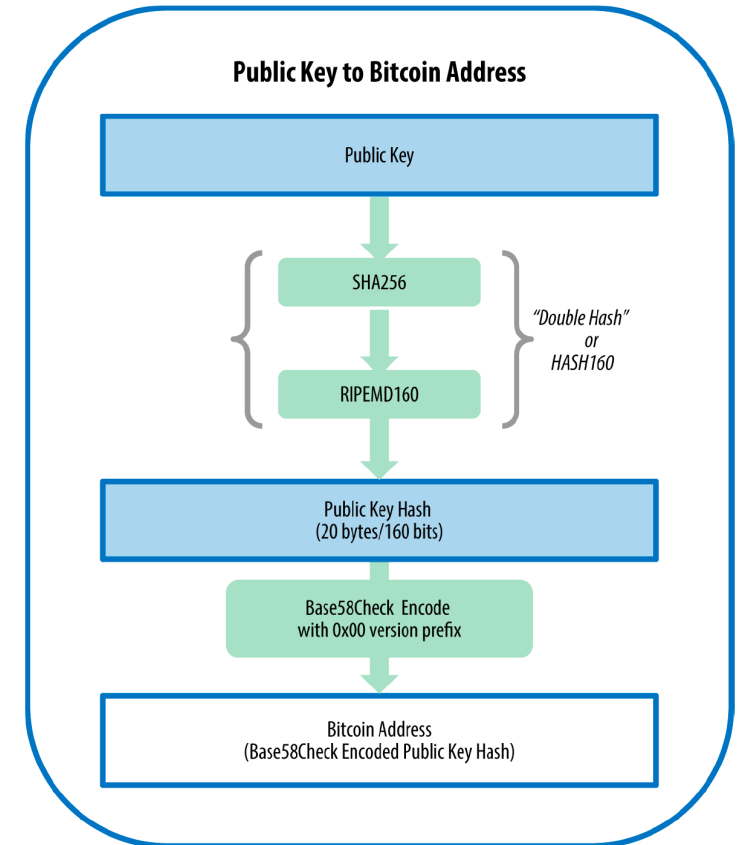
Public Keys

- Calculated from private key using elliptic curve multiplication
 - K (public key) = k (private key) * G
 - $f(x, y): y^2 \bmod p = (x^3 + 7) \bmod p$
 - $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$
 - defined by a secp256k1 standard
- Reverse operation: finding the discrete logarithm



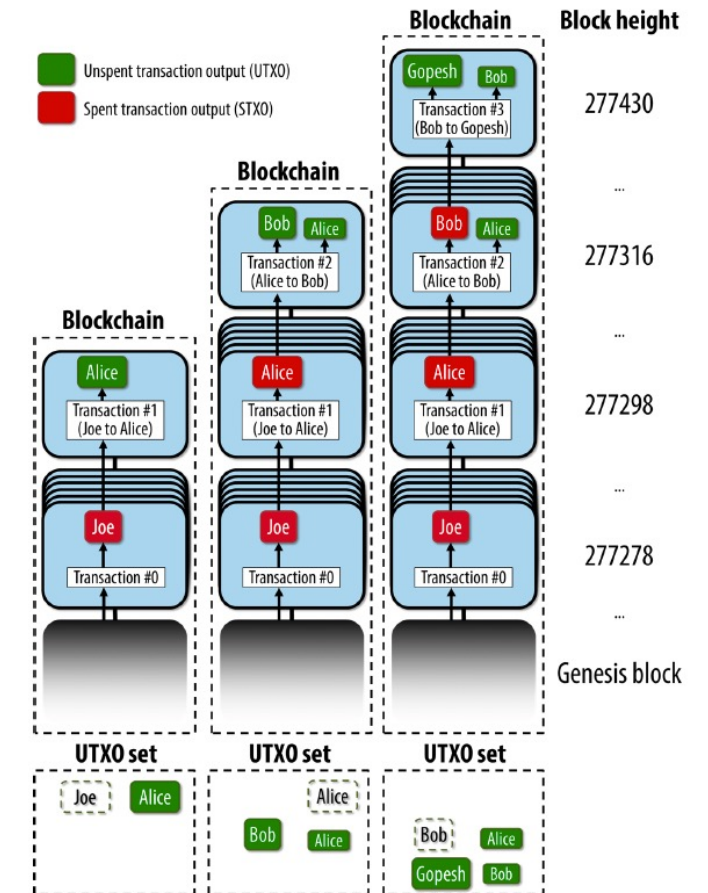
Addresses

- Produced from a public key and starts with 1
 - SHA256, RIPEMD160
 - $A = \text{RIPEMD160}(\text{SHA256}(K))$
- Encoded using Base58Check
- Use only x coordinate (y can be computed)
- Pay-to-Script addresses
 - Address created from a script
- Multisig Addresses
 - Requires M-of-N signatures



Transactions

- Transfer of value between participants
- UTXO = unspent transaction outputs
 - Full nodes track UTXO set
- Received bitcoin ~ blockchain contains an UTXO
- Change returned as another transaction output
- Coinbase transactions = reward for the miners
 - Creates new bitcoins



Transaction Outputs

- An amount of bitcoin in satoshi (0.00000001 BTC)
- scriptPubKey (locking script/witness script)
 - A cryptographic puzzle determining the requirements

```
"vout": [  
  {  
    "value": 0.01500000,  
    "scriptPubKey": "OP_DUP OP_HASH160 ab68025513c3dbd2f7b92a94e0581f5d50f654e7 OP_EQUALVERIFY  
OP_CHECKSIG"  
  },  
  {  
    "value": 0.08450000,  
    "scriptPubKey": "OP_DUP OP_HASH160 7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a8 OP_EQUALVERIFY OP_CHECKSIG",  
  }  
]
```

JSON

Transaction Inputs

- Identifier of UTXO
- Output index
- scriptSig - unlocking script
- sequence

```
"vin": [  
  {  
    "txid": "7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18",  
    "vout": 0,  
    "scriptSig" :  
    "3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298c  
ad530a863ea8f53982c09db8f6e3813[ALL]  
0484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f4938de5cc17b  
4a10fa336a8d752adf",  
    "sequence": 4294967295  
  }  
]
```

JSON

Transaction Fees



AFFECT THE PROCESSING PRIORITY



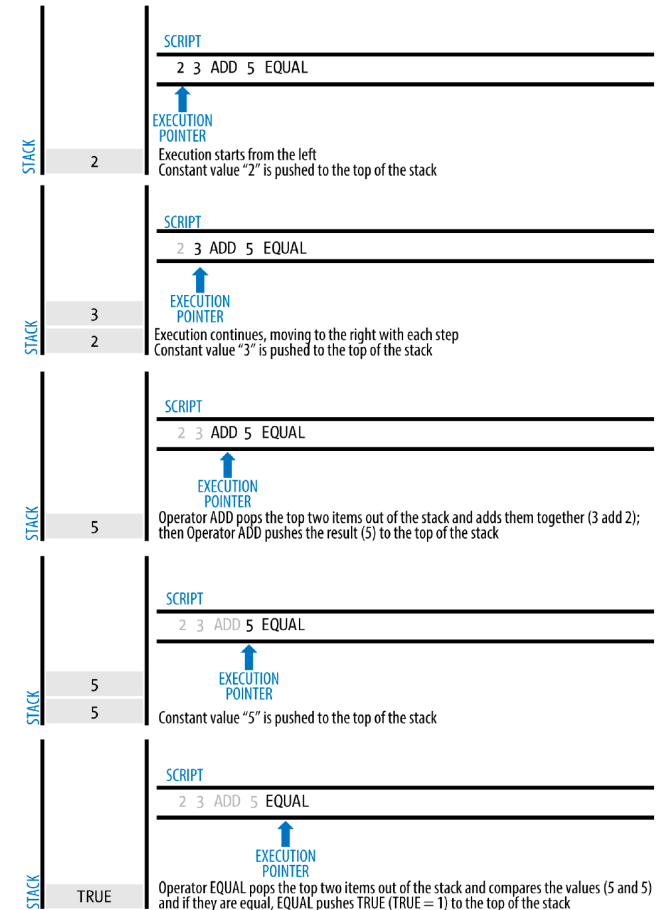
CALCULATED BASED ON THE SIZE OF
TRANSACTION (IN KB) OR
DYNAMICALLY



IMPLIED FROM THE DIFFERENCE
BETWEEN INPUTS AND OUTPUTS

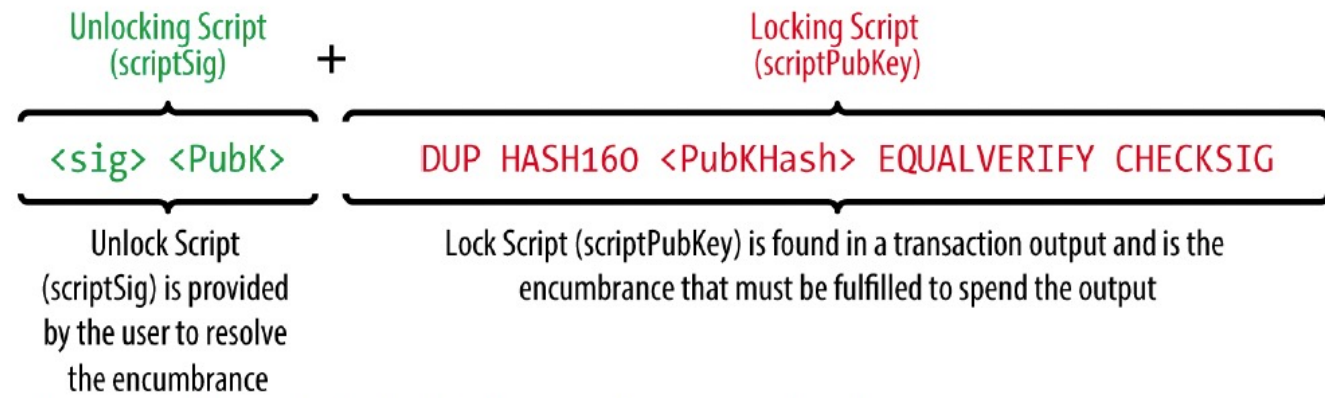
Script Language

- Script, Forth-like stack-based execution language
- Stateless, Turing incomplete



Pay-To-Public-Key-Hash (P2PKH)

- Major transaction's script



Digital Signatures

- Undeniable proof for authorization
- Transaction cannot be modified after signing
- Each transaction signed independently
- Using ECDSA
- Functions: CHECK(MULTI)SIG, CHECK(MULTI)SIGVERIFY
- SIGHASH (Signature Hash Type): ALL, NONE, SINGLE, ANYONECANPAY

```
3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298ca  
d530a863ea8f53982c09db8f6e381301
```


Elliptic Curve Digital Signature Algorithm (ECDSA)

- Creates a temporal private + public key pair
- Signing: $S = k^{-1} (\text{Hash}(m) + dA * R) \bmod n$
 - k - a temporal private key
 - R - x coordinate of the temporal public key
 - dA - the signing private key
 - m - the transaction
- Verification: $P = S^{-1} * \text{Hash}(m) * G + S^{-1} * R * Qa$
 - R, S - signature values
 - Qa - public key
 - M - the transaction
 - G - generator point

```
3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298ca  
d530a863ea8f53982c09db8f6e381301
```

Multisignature

- X-of-3 multisignature
- X-of-15 in P2SH

```
<Signature B> <Signature C> 2 <Public Key A> <Public Key B> <Public Key C> 3 CHECKMULTISIG
```

Pay-to-Script-Hash (P2SH)

- Script replaced with a hash fingerprint (redeem script)
- Script must be presented on spending
- Not recursive
- Advantages
 - Makes the transaction smaller
 - Shifts burden from output (UTXO) into input (blockchain)

```
<2 PK1 PK2 PK3 PK4 PK5 5 CHECKMULTISIG> HASH160 <redeem scriptHash> EQUAL
```

```
<Sig1> <Sig2> 2 PK1 PK2 PK3 PK4 PK5 5 CHECKMULTISIG
```

Data Recording Output (RETURN)

- Create transaction outputs to record data on the blockchain
 - E.g., proof-of-existence of a file (<https://proofofexistence.com/>)
- 80B data portion
- Create a provably unspendable output which does not need to be stored in UTXO set

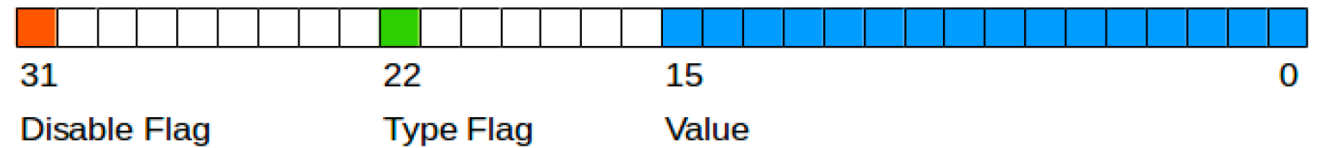
Absolute Timelocks

- nLockTime – Per-transaction timelock
 - $< 500\text{mil.}$ – not valid before the given block height
 - $\geq 500\text{mil.}$ – not valid before the given UNIX epoch
 - Double spending problem
- Check Lock Time Verify (CLTV) – Per-output timelock
 - UTXO can be spent with nLockTime set to \geq

```
<now + 3 months> CHECKLOCKTIMEVERIFY DROP DUP HASH160 <Bob's Public Key Hash> EQUALVERIFY CHECKSIG
```

Relative Timelocks

- nSequence ($< 2^{31}$)
 - Behavior based on type flag
 - 0: Not valid before given number of blocks, or
 - 1: Not valid before given number of seconds (multiples of 512s)
- CHECKSEQUENCEVERIFY (CSV)



Median Time Passed

- Timestamps set by miners
 - Potential motivation to lie
- Median-Time-Passed is a median timestamp of last 11 blocks (~2h)
 - Consensus time for timelock operations

Fee Snipping Attack

- to snipe higher-fee transaction from future to rewrite past blocks
- Fixed by limiting nLockTime to the next block

Flow Control (if-else)

- IF-ELSE-ENDIF
 - Can be nested
- VERIFY guard clause – terminates if FALSE

```
condition
IF
    code to run when condition is true
ELSE
    code to run when condition is false
ENDIF
code to run in either case
```

```
HASH160 <expected hash> EQUALVERIFY <Bob's Pubkey> CHECKSIG
```

A Complex Example

```
01 IF
02   IF
03     2
04   ELSE
05     <30 days> CHECKSEQUENCEVERIFY DROP
06     <Abdul the Lawyer's Pubkey> CHECKSIGVERIFY
07     1
08   ENDIF
09   <Mohammed's Pubkey> <Saeed's Pubkey> <Zaira's Pubkey> 3 CHECKMULTISIG
10 ELSE
11   <90 days> CHECKSEQUENCEVERIFY DROP
12   <Abdul the Lawyer's Pubkey> CHECKSIG
13 ENDIF
```

- Unlocking script A: 0 <Mohammed's Sig> <Zaira's Sig> TRUE TRUE
- Unlocking script B: 0 <Abdul the Lawyer's Sig> <Saeed's Sig> FALSE TRUE
- Unlocking script C: <Abdul the Lawyer's Sig> FALSE

Segregated Witness (segwit)

- Architectural change of separating unlocking script/scriptSig
- Data moved into a separated witness data
- Soft Fork

Pay-to-Public-Key-Hash (P2PKH)

- Old P2PKH

```
[...]
"Vin" : [
  "txid": "0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2",
  "vout": 0,
    "scriptSig": "<Bob's scriptSig>",
]
[...]
```

```
DUP HASH160 ab68025513c3dbd2f7b92a94e0581f5d50f654e7 EQUALVERIFY CHECKSIG
```

Pay-to-Witness-Public-Key-Hash (P2WPKH)

- P2WPKH
 - A witness version
 - A witness program
- Similar for P2WSH

```
[...]
"Vin" : [
  "txid": "0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2",
  "vout": 0,
    "scriptSig": "",
]
[...]
"witness": "<Bob's witness data>"
[...]
```

```
0 ab68025513c3dbd2f7b92a94e0581f5d50f654e7
```

Segregated Witness Transaction Identifiers

- Transaction signatures could have been modified
 - Potential for DoS
- 2 new fields:
 - Txid – double SHA256 hash of transaction **without** the witness data
 - Wtxid – double SHA256 hash of transaction **with** the witness data

Segregated Witness' New Signing Algorithms

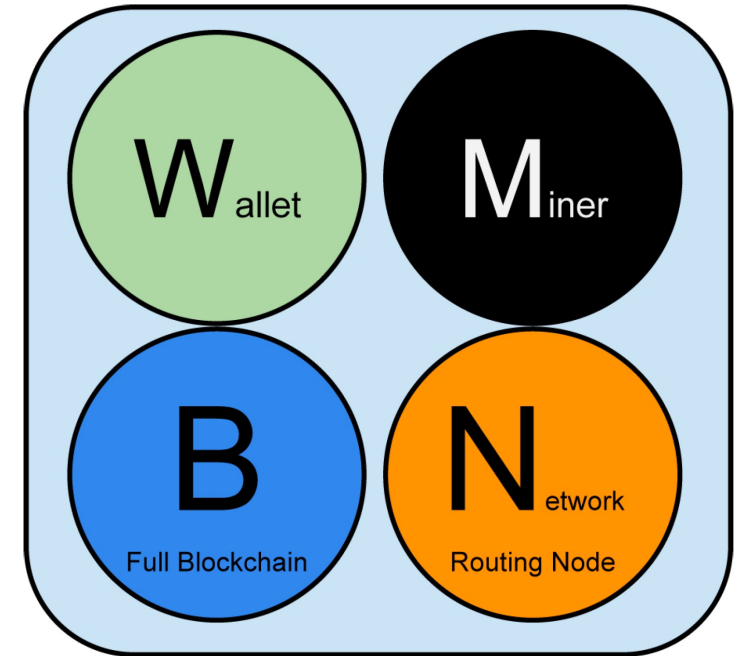
- CHECK(MULTI)SIG(VERIFY)
- Signatures applied on commitment hash
 - SIGHASH_ALL uses all inputs and outputs
- Hash operations increases in $O(n^2)$ with number of operations in Tx
 - Very large transactions can cause performance problem (DoS) of the whole network
- New algorithm increases only in $O(n)$
- Commitment hash includes the value of each input

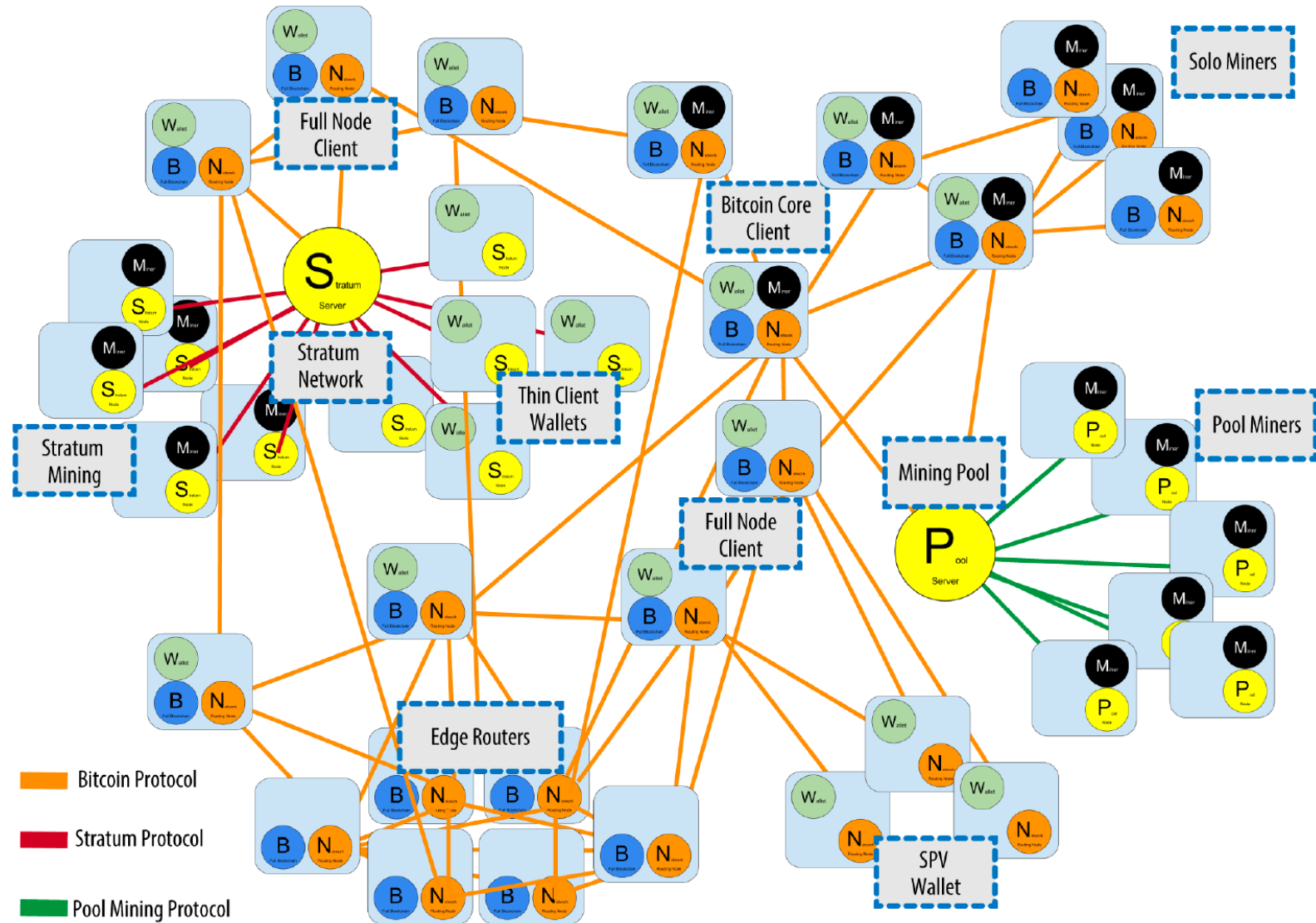


The Bitcoin Network

Network Architecture

- Bitcoin protocol = P2P protocol on top of the internet
- Stratum protocol = protocol for lightweight devices
- Node's functions: wallet, mining, blockchain, routing





Bitcoin Relay Networks

- A network minimizing latency between miners
- V1: Bitcoin Relay Network
- V2: FIBRE – UDP-based

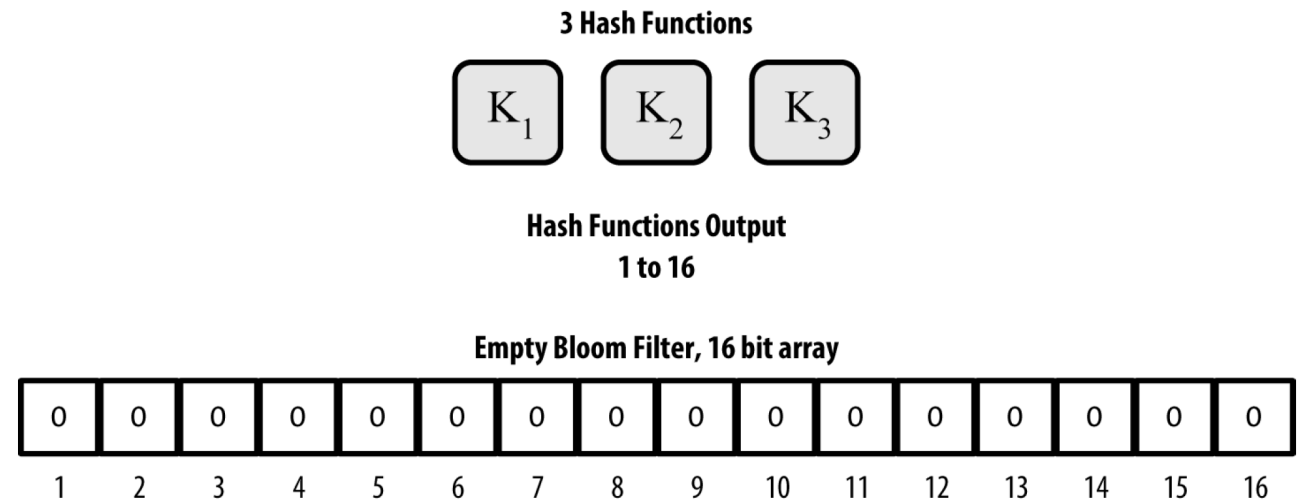


Simple Payment Verification (SPV) Nodes

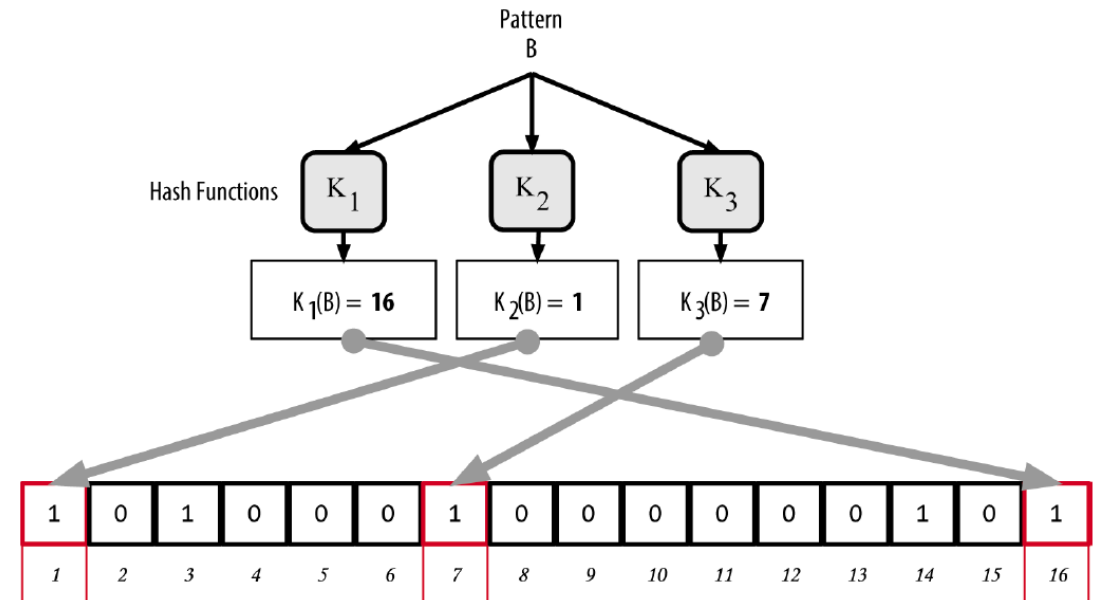
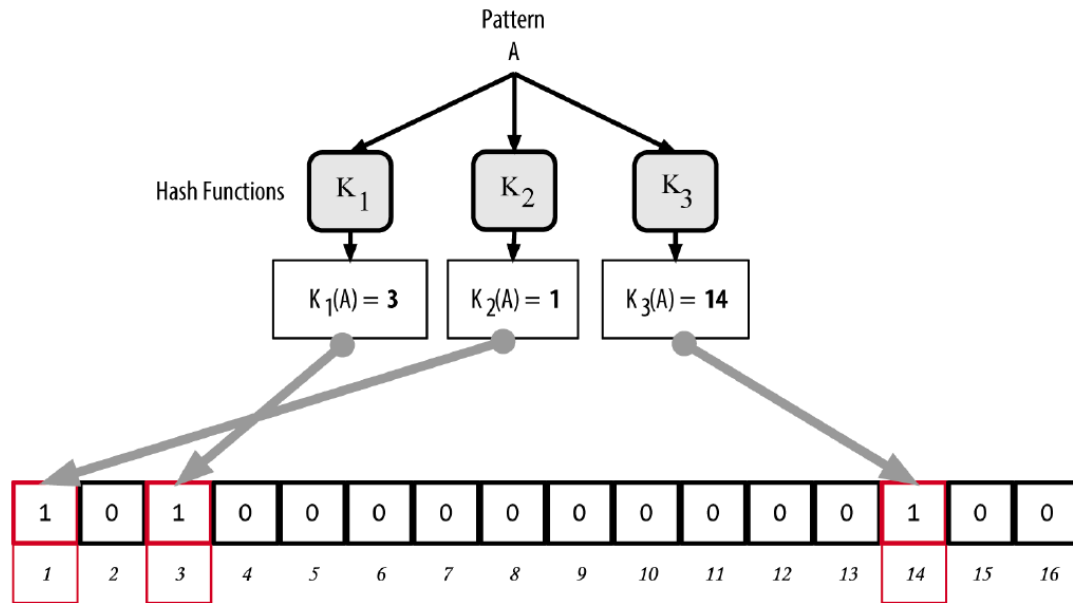
- Not storing the whole blockchain
- Download only block headers
- Verify transactions using views on demand from peers
 - SPV node establishes link between the transaction and the block containing it using merkle path
 - +6 additional blocks atop of the block with the transaction
- Cannot verify, e.g., double spending of transactions
 - Double spending, DoS, network partition or Sybil attacks
 - Connect randomly to several nodes to decrease probability

Bloom Filters

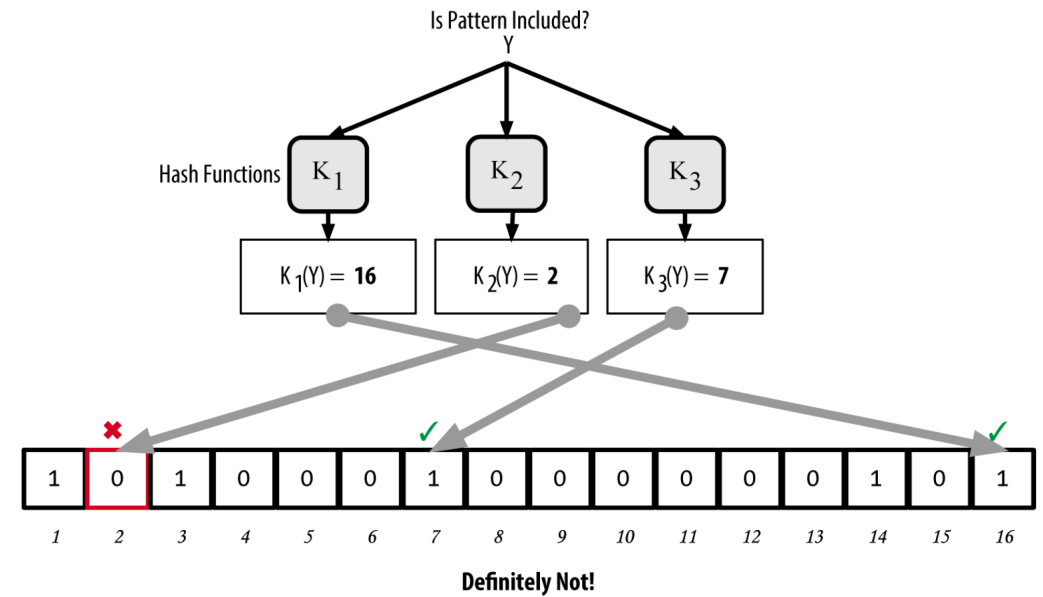
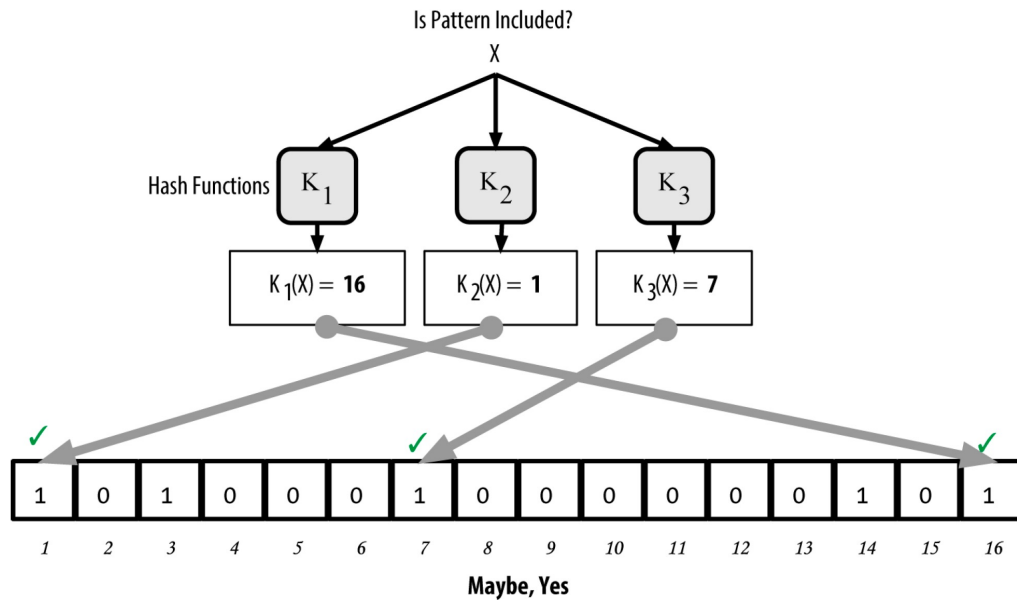
- A probabilistic filter protecting privacy
- A bitset of size N and variable number of M hash functions



Bloom Filters - Insert



Bloom Filters - Search



Bloom Filters In SPV

- To filter interesting transaction received from peers
- Extract public key hash, script hash and transaction ID from UTXO controlled by the wallet into the bloom filter
- Send filter load to a peer along with the filter
- A peer pre-filters transaction for SPV

Encrypted And Authenticated Connections

- Problem only for SPV nodes
- Tor Transport (The Onion Routing network)
 - Network offering encryption and encapsulation of data through randomized paths
- P2P Authentication & Encryption
 - Allow nodes to authenticate using ECDSA

Transaction Pool (Memory Pool/Mempool)

- A temporal list of unconfirmed transactions
- Sometimes a separated pool for orphaned transactions
 - Orphan = transaction with a missing parent
- Sometimes UTXO pool
 - Set of unspent outputs

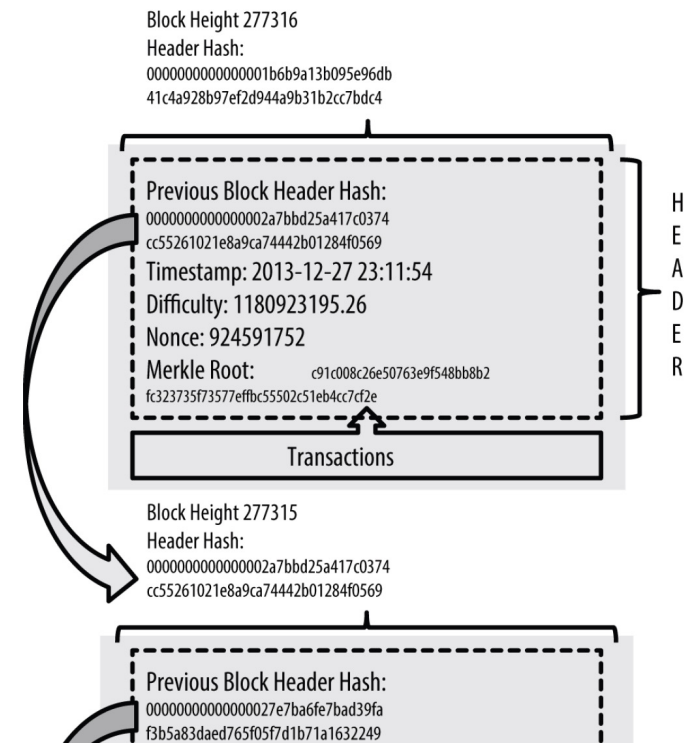




The Blockchain

The Blockchain

- An ordered back-linked list of blocks of transactions
- Each block identified by SHA256 of its header
 - The header contains a parent header and thereby affects the hash
 - (Can be referenced by the height as well)
- Each block references a parent block
 - The genesis block = the first block ever



Structure Of The Block

Size	Field	Description
4 bytes	Block Size	The size of the block, in bytes, following this field
80 bytes	Block Header	Several fields form the block header
1–9 bytes (VarInt)	Transaction Counter	How many transactions follow
Variable	Transactions	The transactions recorded in this block

Structure Of The Header

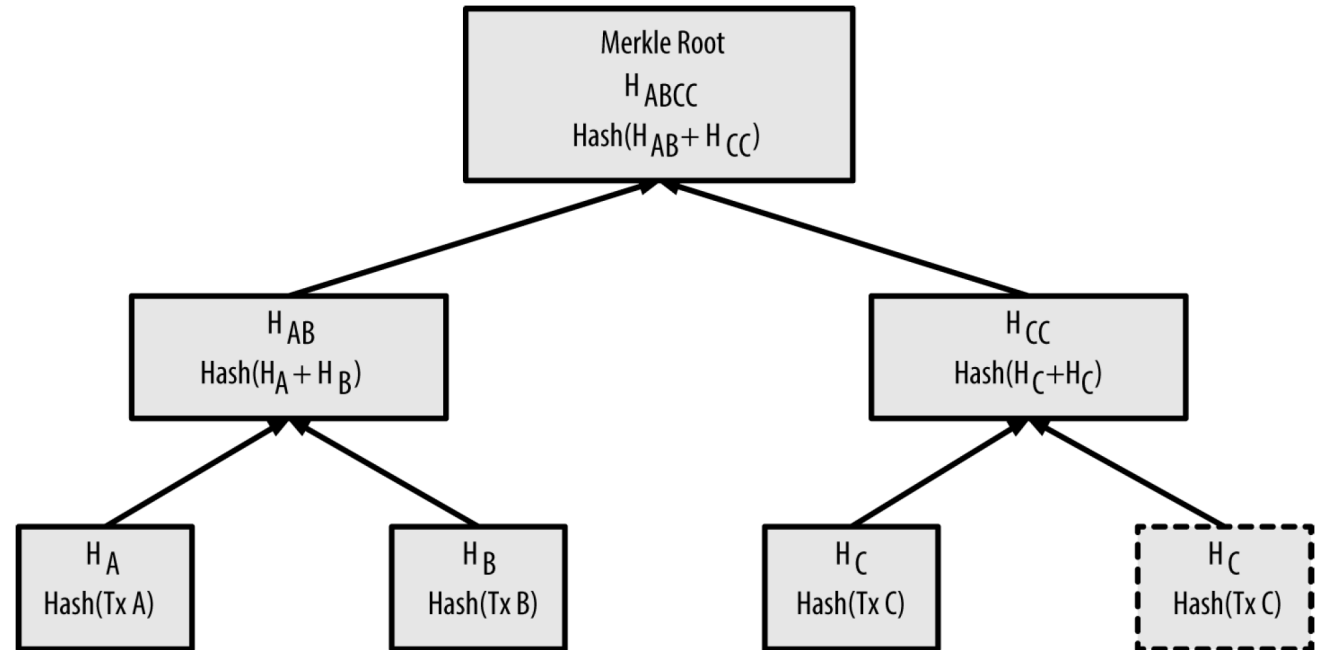
Size	Field	Description
4 bytes	Version	A version number to track software/protocol upgrades
32 bytes	Previous Block Hash	A reference to the hash of the previous (parent) block in the chain
32 bytes	Merkle Root	A hash of the root of the merkle tree of this block's transactions
4 bytes	Timestamp	The approximate creation time of this block (in seconds elapsed since Unix Epoch)
4 bytes	Difficulty Target	The Proof-of-Work algorithm difficulty target for this block
4 bytes	Nonce	A counter used for the Proof-of-Work algorithm

Merkle Trees (Binary Hash Trees)

- A data structure used to efficiently summarizing and verifying the integrity of large dataset
- A binary tree containing cryptographic hashes
- Need $2 * \log_2 N$ calculations to check for element existence

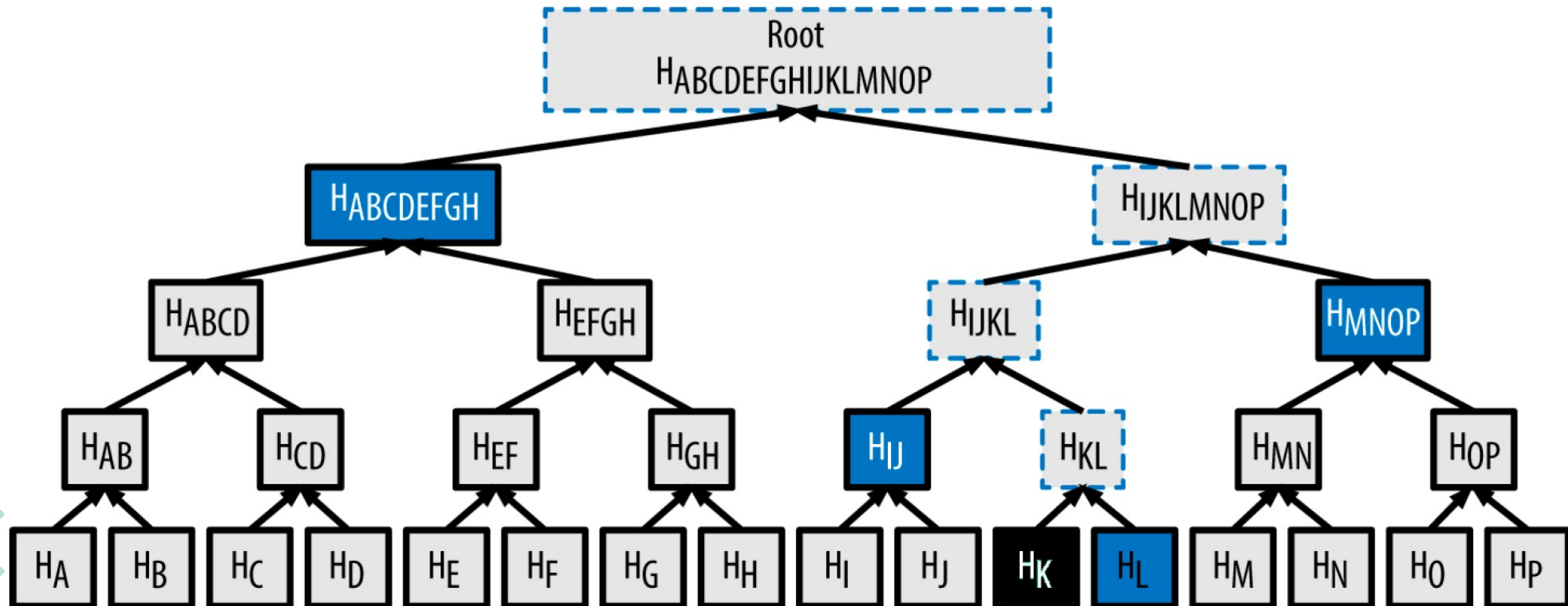
Merkle Trees - Building

- Constructed bottom-up
- $H_X = \text{SHA256}(\text{SHA256}(X))$
- $H_{X+Y} = \text{SHA256}(\text{SHA256}(H_X + H_Y))$



Merkle Trees - Search

- Merkle path = authentication path



Merkle Trees in SPV

- SPV = Simplified Payment Verification
- Merkle path used for verification of a transaction inside a block



Mining & Consensus

Mining

- A decentralized mechanism of clearing and validation of transactions
- Enables consensus without central authority
- Miners validate and record transactions on the global ledger
- Rewards for miners: A coinbase/block reward and transaction fee
- Miners compete in solving a difficult math problem based on hash
 - Proof-of-Work = solution to the problem
- The maximum amount of newly created BTCs halves every 210k blocks (~ 4 years)

Decentralized Consensus

- Bitcoin has no central authority
- The consensus is an emergent artifact of the asynchronous interaction of independent nodes all following simple rules

Independent Transaction Verification

- Each node validates forwarded transactions against many criteria
 - Syntax, non-empty inputs/outputs, values in ranges, ...



Aggregating Transactions Into Blocks

- After validation, transactions added into memory pool
- Transaction aggregated into a candidate block
 - Adding the coinbase transaction
 - Does not consume UTXO inputs

Size	Field	Description
32 bytes	Transaction Hash	All bits are zero: Not a transaction hash reference
4 bytes	Output Index	All bits are ones: 0xFFFFFFFF
1–9 bytes (VarInt)	Coinbase Data Size	Length of the coinbase data, from 2 to 100 bytes
Variable	Coinbase Data	Arbitrary data used for extra nonce and mining tags. In v2 blocks; must begin with block height
4 bytes	Sequence Number	Set to 0xFFFFFFFF

Proof-of-Work Algorithm

- Hashing the block header repeatedly, changing one parameter until the resulting hash matches a specific target
- Nonce = A number used to vary outputs of the hash function

Adjusting Difficulty

- The target determines the difficulty of the Proof-of-Work algorithm
- The target is set so the mining power result in 10-minutes intervals
- Each node retargets the Proof-of-Work algorithm every 2,016 blocks
 - Compares with expected time and adjust accordingly
 - To avoid extreme volatility, change is not more than factor of 4

Validating A New Block

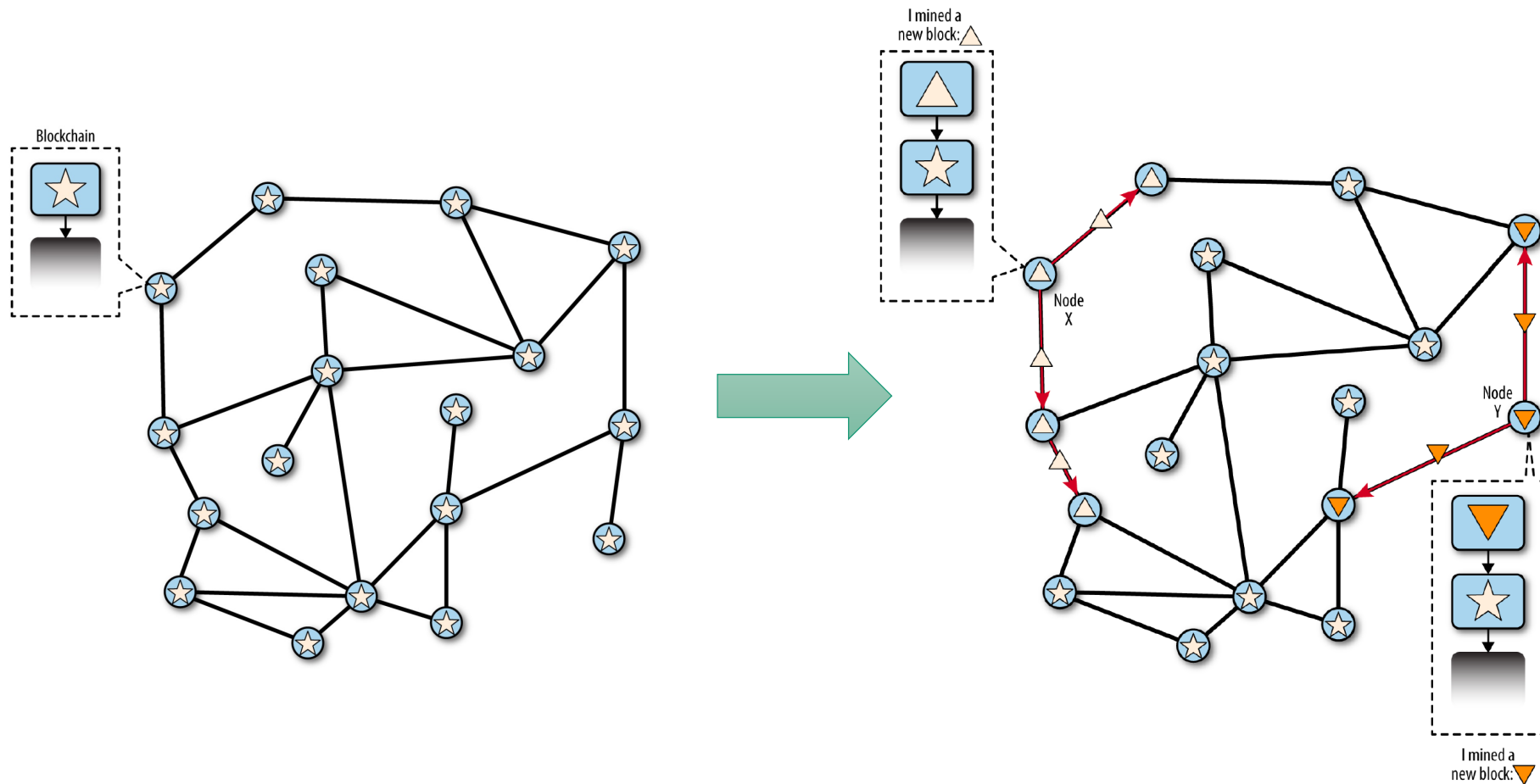
- An independent validation by every node of the network
 - Miners cannot cheat
- A lot of validation criterias
 - Syntax, block header hash, block timestamp, block size, coinbase transaction, all transactions are valid

Assembling And Selecting Chains Of Blocks

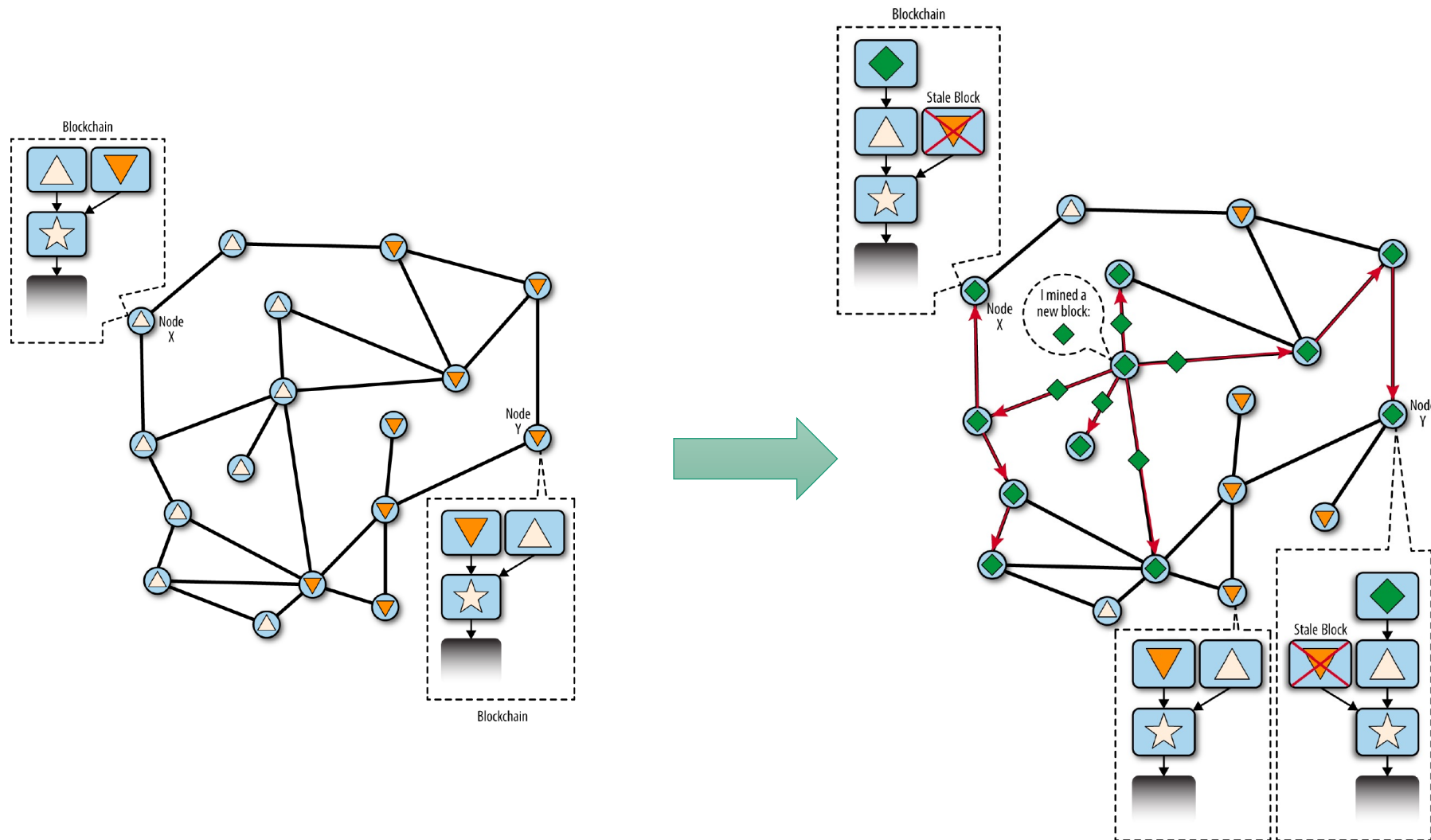
- Once block validated, attempt to assemble chain with most PoW
- Nodes maintain main blockchain, secondary blockchains and orphans
 - Orphan = block without a parent
- Received blocks connected to the main/secondary chain
 - Recorverging to secondary chain if more cumulative work
- Consensus by selecting the greatest-comulative-work valid chain
 - “Voting” which chain is the main – extending the chain with a new block is the vote

Forks

- Different copies of blockchain not always consistent, e.g., delays

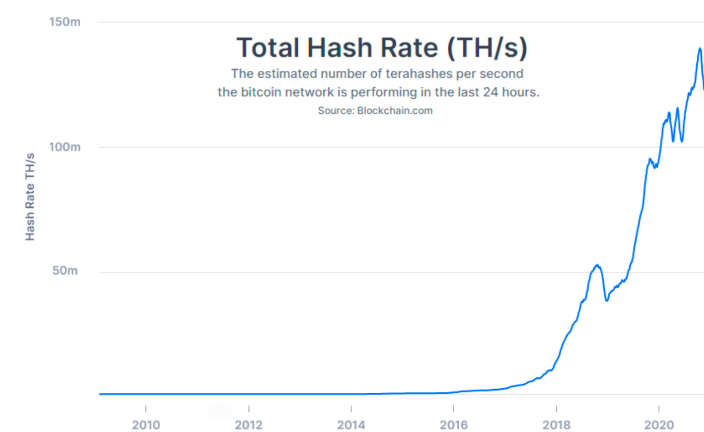
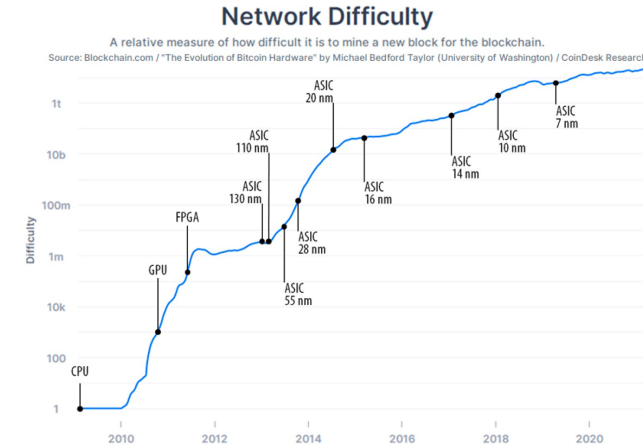


Fork Resolution



Mining Race

- Mining related to the cost of electricity
- The hashing power increased exponentially (Moore's Law)
 - CPU → GPU → FPGA → ASIC
 - The extra nonce solution

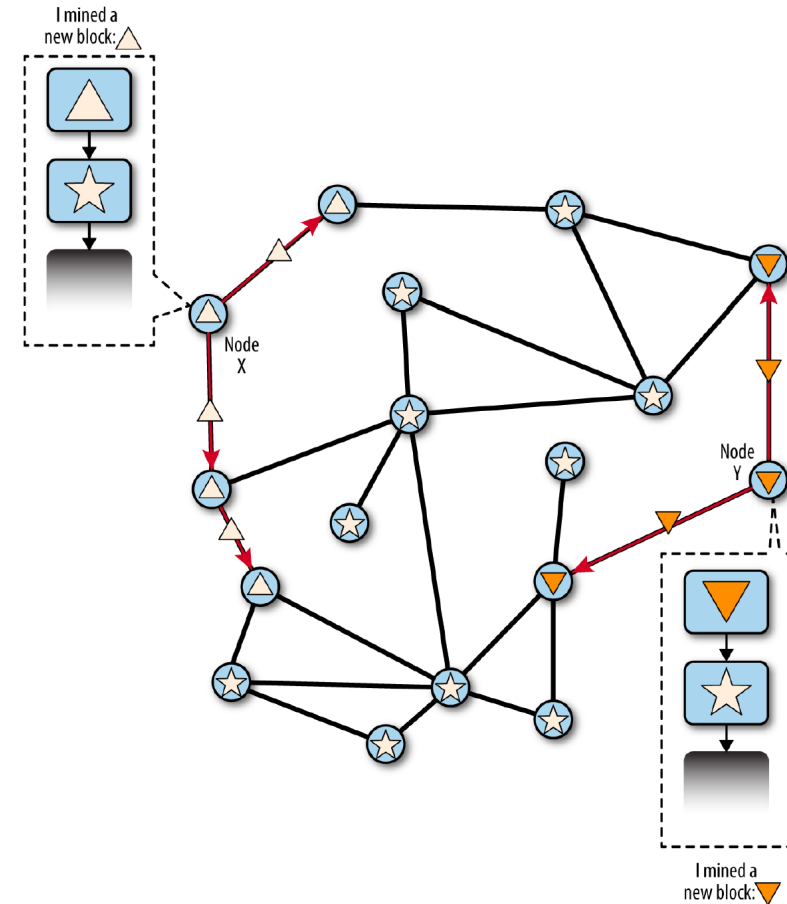


Mining Pools

- Solo miners have no chance any longer
 - E.g., $P = (14 * 10^{12} / 3 * 10^{18}) * 210,240 = 0.98$
 - HW with 14TH/s ~ \$2,500 + \$1.5/day for electricity
 - 3EH/s – network wide hashing rate
 - 210,240 – number of blocks in 4 years
- Managed pools
 - Pool operator
- P2Pool
 - No central operator
 - Blockchain like system, e.g., sharechain

Consensus Attacks

- Attack future or recent past (~ tens of blocks)
 - Large computation power needed for very deep forks
- Causes DoS, cannot steal bitcoins

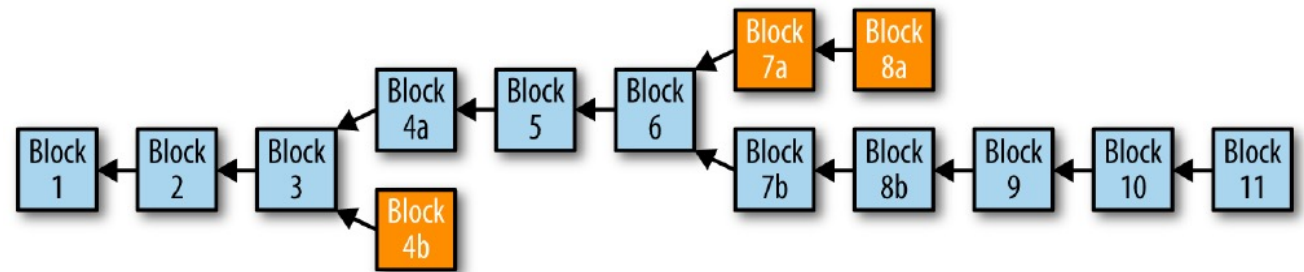


51% Attack

- Majority of miners (51%) can cause deliberate forks
 - Theoretically ~30% of hashing power should be enough
- Double spend on attackers transactions
 - Wait ≥ 6 confirmations, or
 - Multisignature account with few confirmations
- DoS of addresses
 - Ignore specific addresses

Changing Consensus Rules

- Hard forks
 - Not forward compatible changes
 - Change of consensus rules (a bug, modification)
 - Software forks (Bitcoin XT, Bitcoin Classic, Bitcoin Unlimited)
 - Forcing minority to upgrade or stay on a minority chain
 - Potentially creating two competing systems
- Soft forks
 - Forward compatible change (non-upgraded clients continue to operate)
 - E.g., redefining NOP opcodes





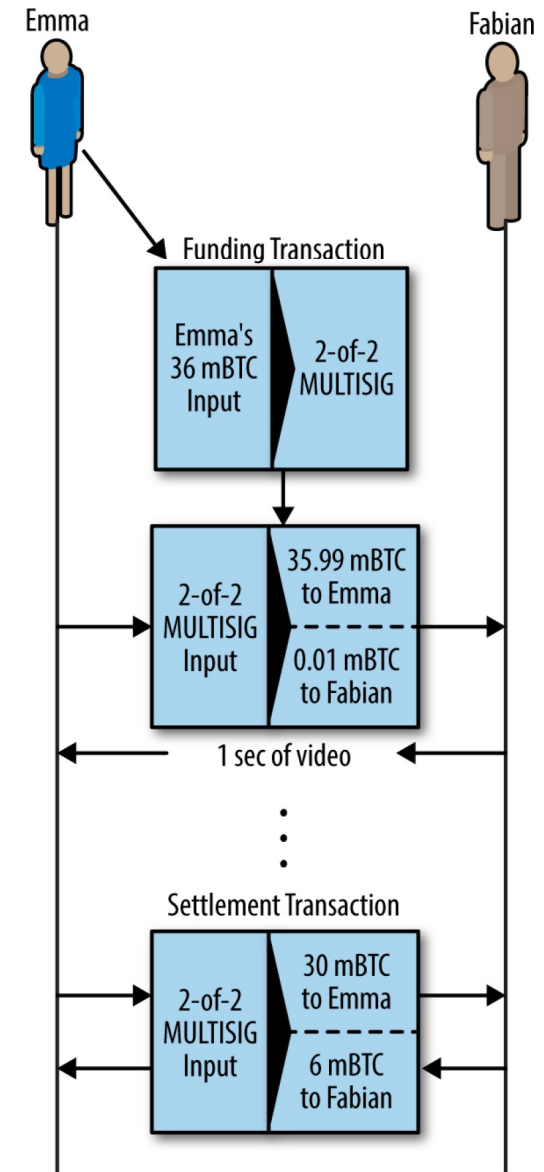
Blockchain Applications

Applications

- Digital Notary
 - Proof-of-existence - providing a proof that a document existed at the time recorded
- Lighthouse
 - Crowdfunding application
- Namecoin
 - Decentralized name registration database
- Colored coins
 - A protocol for creating your own digital currencies (tokens)
 - Publicly assigning a color to a specific UTXO
- Metacoin
 - A protocol with a different state function, i.e., $\text{APPLY}(S, \text{TX}) = S$ instead of error
- Counterparty
 - A protocol for creating/trading/exchange virtual assets and tokens
 - Embeds metadata using OP_RETURN or multisignature
 - Platform for other applications, e.g., Spells of Genesis, Folding Coin
- Payment Channels

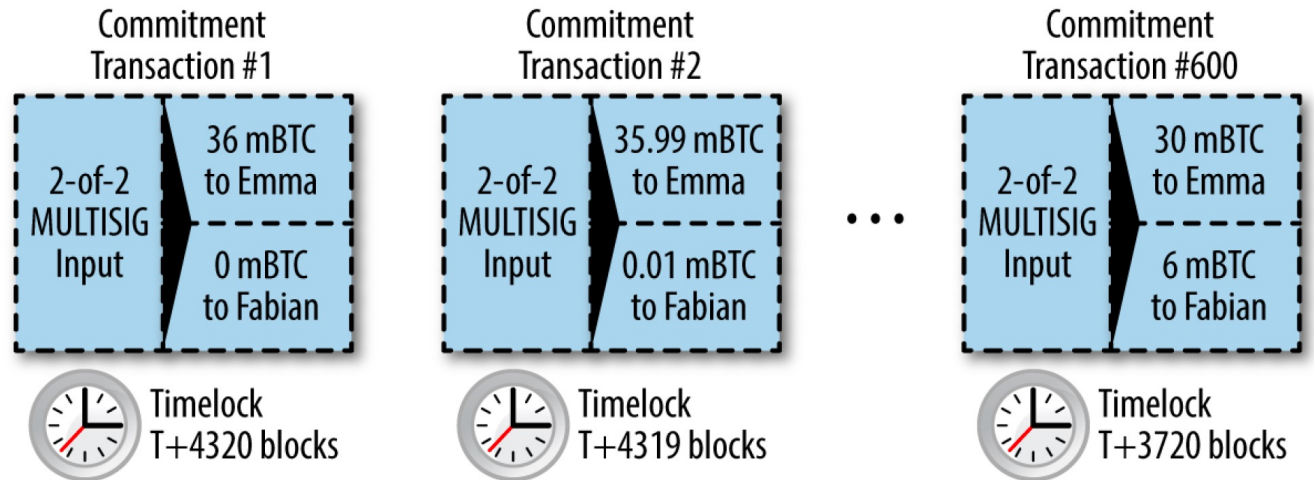
State Channels

- Mechanism for exchanging transactions between two parties held off-chain
 1. Established by *funding transaction* that locks a shared state on blockchain
 2. Exchanging *commitment transactions* off-chain
 3. Closing channel using *settlement transaction* on blockchain
- Problems:
 - What if Fabian disappears?
 - What if Emma transmits a commitment transaction into blockchain?



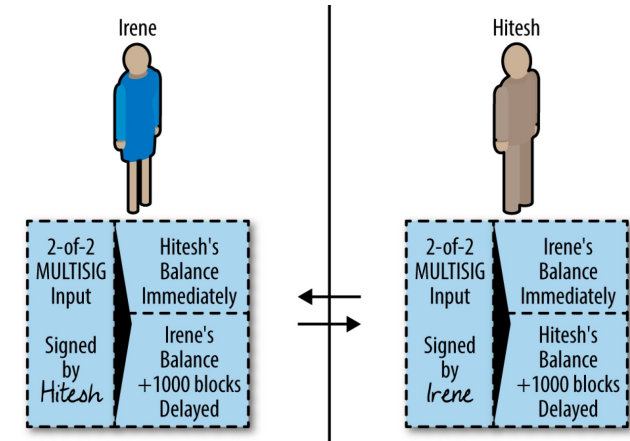
Trustless Channels

- Create funding + refund transaction
- Use refund transaction as commitment
- Set nLockTime as upper bound of the channel's life



Asymmetric Revocable Commitments

1. Establish channel with a funding transaction
2. Participants create a single asymmetric transaction each
 - Input: 2-of-2 funding output, signed by the other party
 - Output 0: return funding to the other party *immediately*
 - Output 1: return funding to yourself *delayed*
3. Include *a revocation key*
 - Allows to punish the cheater by taking the entire balance
 - Two secret halves generated by each participant + shared public key



```
Output 0 <5 bitcoin>:
  <Irene's Public Key> CHECKSIG

Output 1 <5 bitcoin>:
IF
  # Revocation penalty output
  <Revocation Public Key>
ELSE
  <1000 blocks>
  CHECKSEQUENCEVERIFY
  DROP
  <Hitesh's Public Key>
ENDIF
CHECKSIG
```

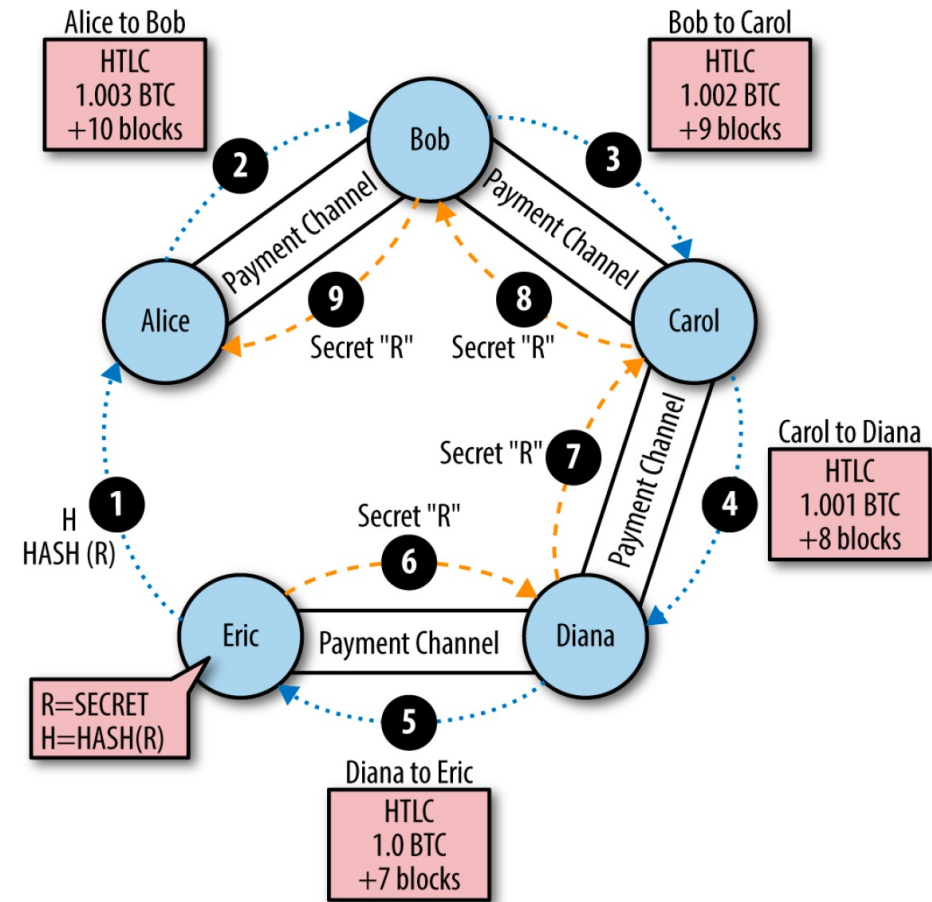
Hash Time Lock Contracts (HTLC)

- Commit funds to a redeemable secret with an expiration time
- The recipient creates a secret and hash
- HTLC can be redeemed by anyone who knows the secret

```
IF
    # Payment if you have the secret R
    HASH160 <H> EQUALVERIFY
ELSE
    # Refund after timeout.
    <locktime> CHECKLOCKTIMEVERIFY DROP
    <Payer Public Key> CHECKSIG
ENDIF
```

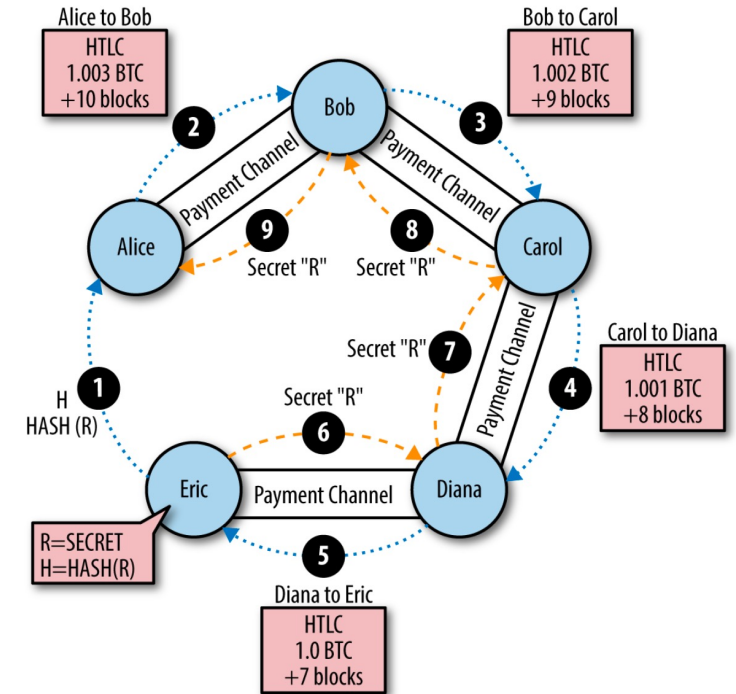
Lightning Network

- A network of bidirectional end-to-end payment channels
- To route payment from channel to channel without trusting any of the intermediaries
- Alice is committing 1.003 of her channel balance to be paid to Bob if Bob knows the secret, or refunded back to Alice's balance if 10 blocks elapse.



Lightning Network - Routing

- Route is known to Alice's node only
- Using onion-routed protocol based on Sphinx scheme
 - Intermediate nodes can verify their portion and find next hop
 - Nodes know only previous and next hop
 - Cannot identify a length of the path
 - No exit nodes, the payment updates channel balances only
- Messages are padded to 20 hops
- Message from Alice to Eric
 - wrapped in message encrypted to Diana with Eric as recipient
 - wrapped in message encrypted to Carol with Diana as recipient
 - ...



Lightning Network Benefits

- Used with any blockchain supporting multisignature, timelocks and smart contracts
- Better privacy
- Speed – transactions settled in milliseconds (HTLC are not committed to the block)