BLOCKCHAIN

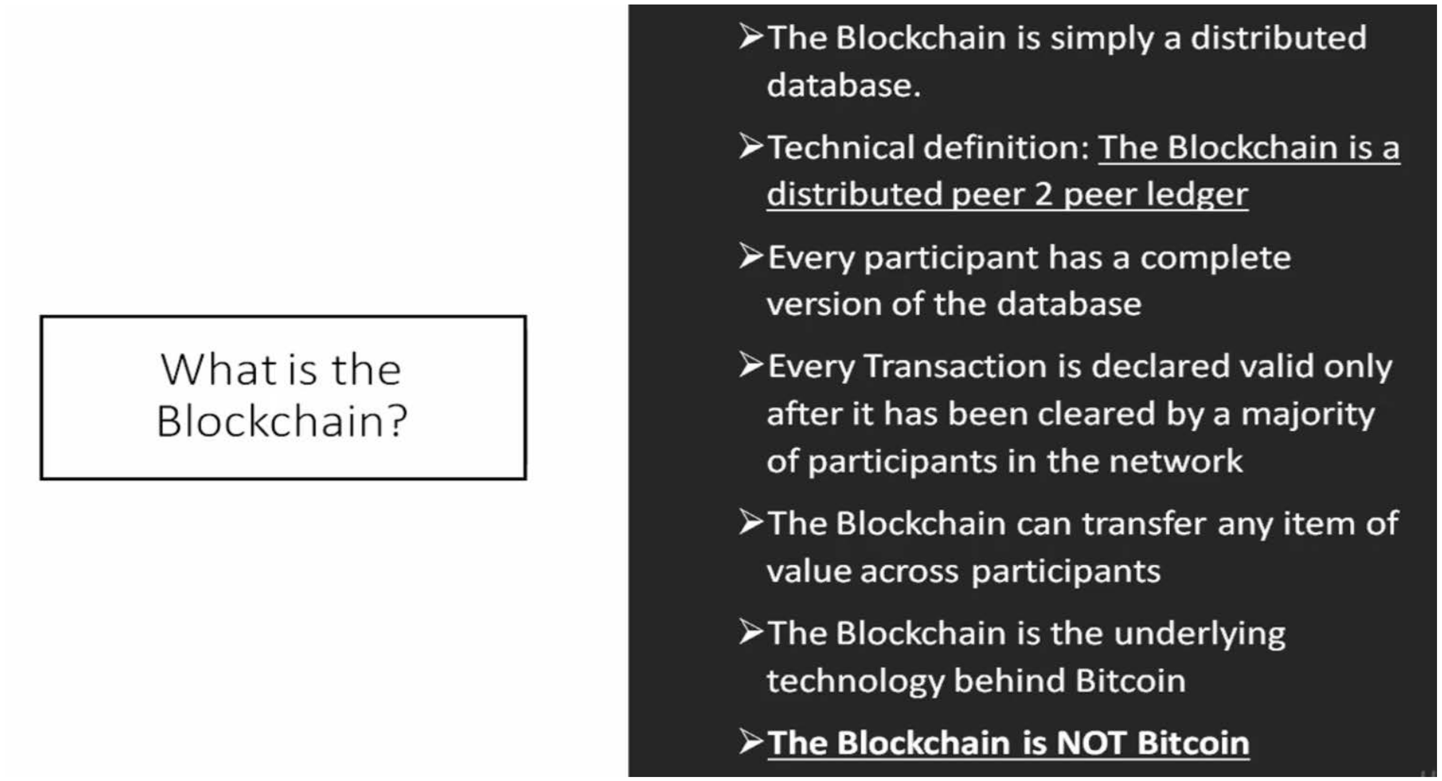
TRAINING ALLIANCE

BLOCKCHAIN
TRAINING ALLIANCE

BLOCKCHAIN
Blockchains and Transactions Part II - A Deeper Dive

www.blockchaintrainingalliance.com

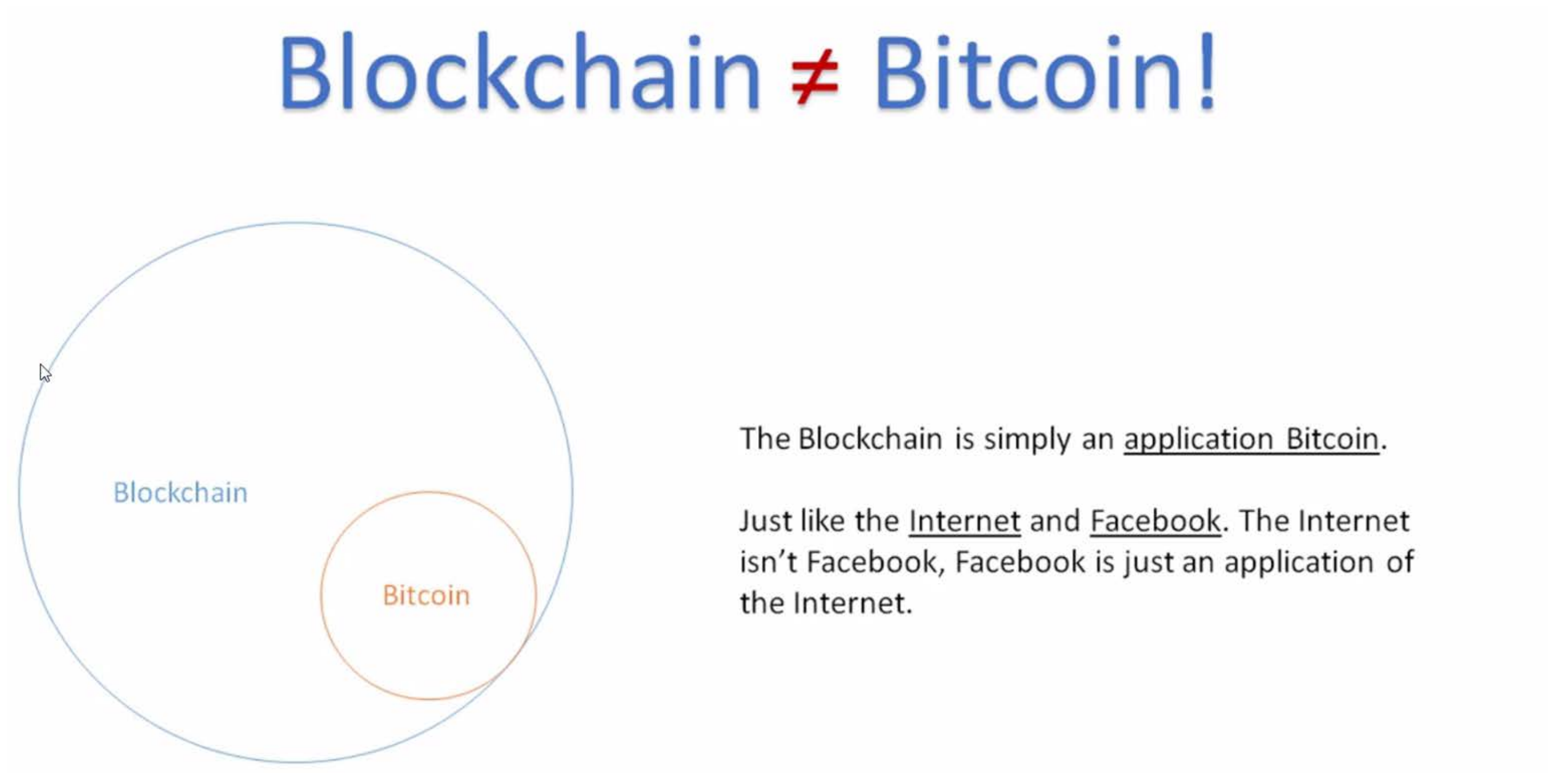
Blockchain



What is the Blockchain?

- The Blockchain is simply a distributed database.
- Technical definition: The Blockchain is a distributed peer 2 peer ledger
- Every participant has a complete version of the database
- Every Transaction is declared valid only after it has been cleared by a majority of participants in the network
- The Blockchain can transfer any item of value across participants
- The Blockchain is the underlying technology behind Bitcoin
- **The Blockchain is NOT Bitcoin**

Blockchain is NOT Bitcoin

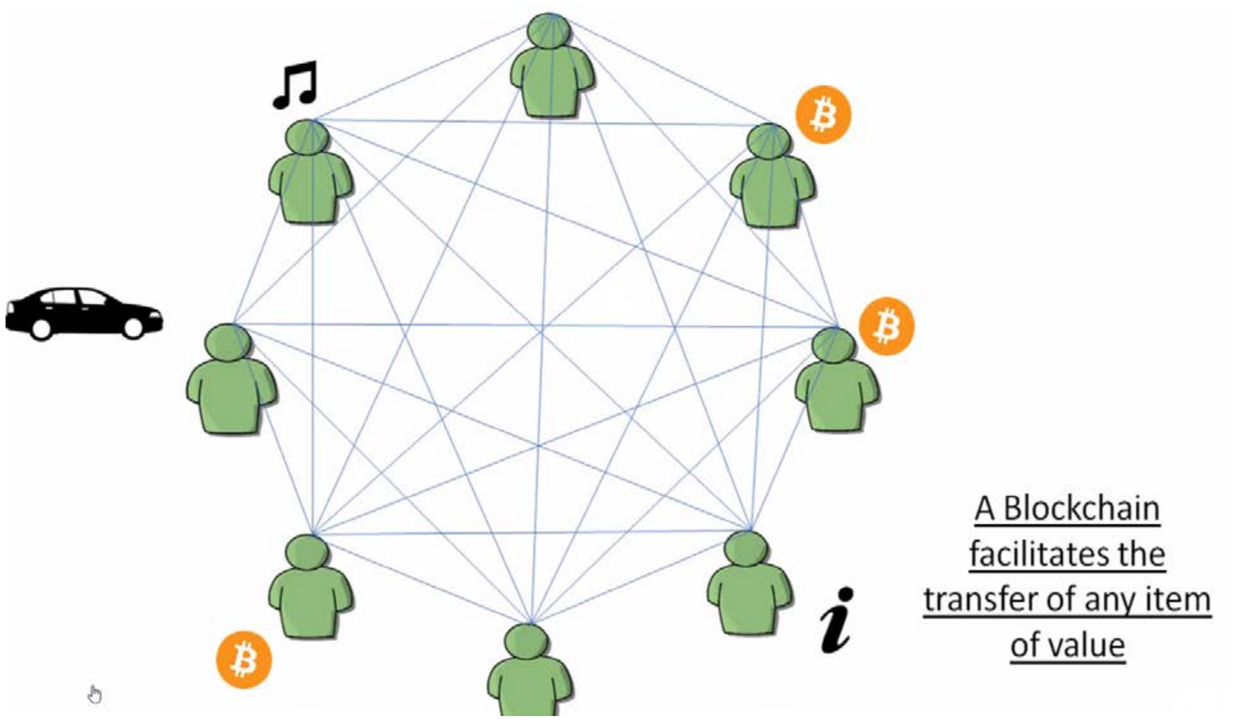


Blockchain ≠ Bitcoin!

The Blockchain is simply an application Bitcoin.

Just like the Internet and Facebook. The Internet isn’t Facebook, Facebook is just an application of the Internet.

Transactions



Multi-Signature Addresses

- An Address that requires multiple people to sign Transactions from it
- Known as m of n where n<16
- The public keys of all the signers are combined to create a special multi-sig Address. Anyone can send coins to this Address
- To spend these coins you have to create a Transaction without broadcasting it, then pass it round the signers until you have the required number of signatures

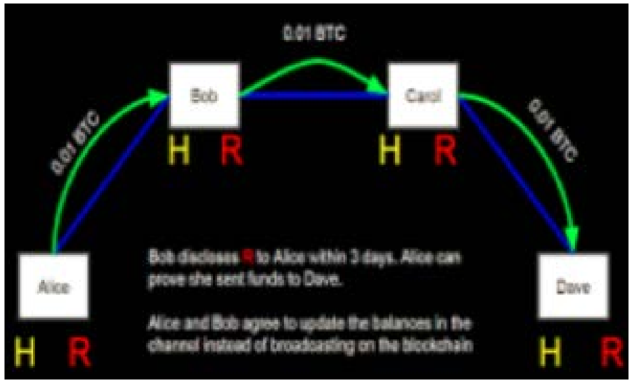
```
OP_2 [A's pubkey] [B's pubkey] [C's pubkey] OP_3 OP_CHECKMULTISIG
```

Micropayment Channel

- A way to allow many mini-transactions to be rolled up into two Transactions
- Buyer places a bond Transaction and a time-bound refund Transaction (which isn't broadcast yet)
- As the buyer consumes the service, new refund Transactions are created but not broadcast, changing the amount of the refund
- When the service is complete, the most recent refund Transaction is broadcast

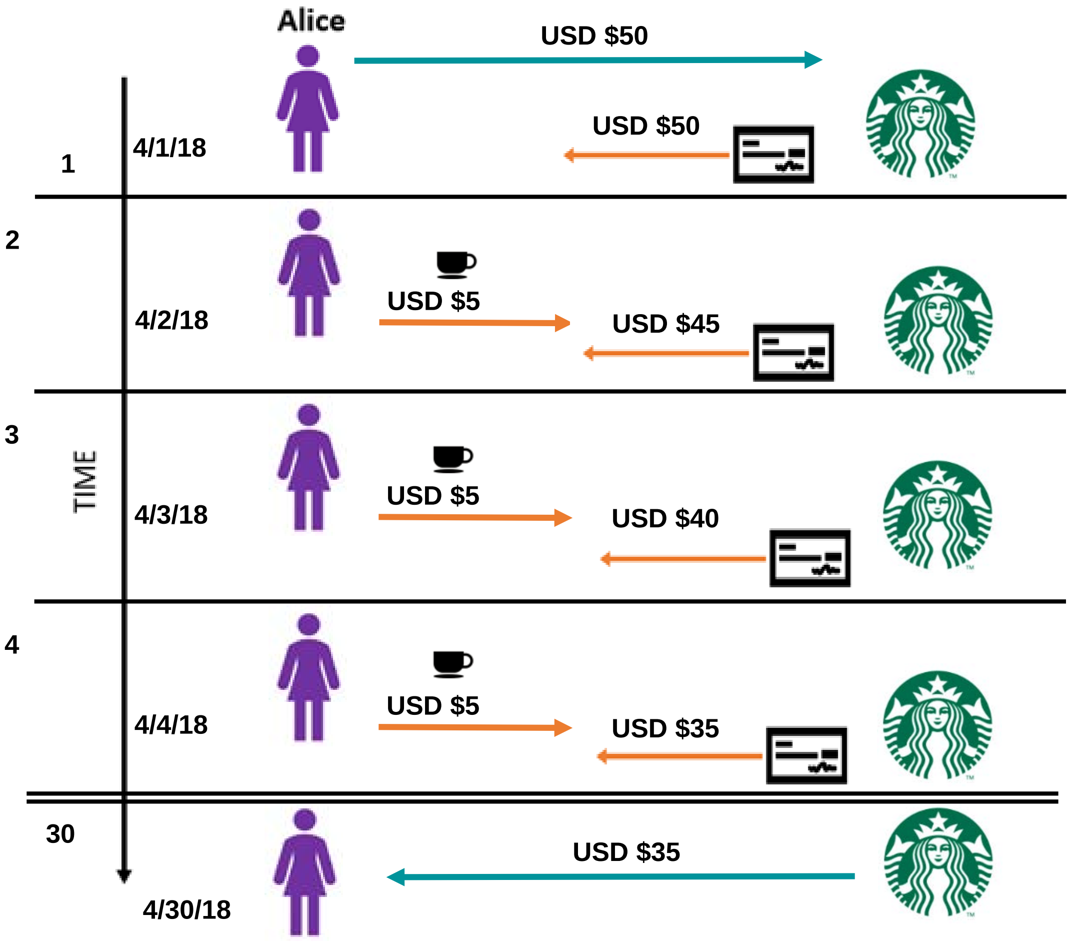
Payment Channels: the concept

- Contractually-obligated relationship
- 3-step economic contract executed over time
  1. Party A opens a payment channel with Party B and posts a pre-paid escrow balance
  2. Party A consumes against the escrow credit over the given time period (activity is tracked)
  3. At the end of the period, cumulative activity is booked in one net transaction to close the contract
- Lightning Network - Bitcoin
- Raiden - Ethereum


0.01 BTC
0.01 BTC
0.01 BTC
Alice
Bob
Carol
Dave
H R
H R
H R
H R
Bob discloses R to Alice within 3 days. Alice can prove she sent funds to Dave.
Alice and Bob agree to update the balances in the channel instead of broadcasting on the blockchain

1. Unilateral Payment C

Starbucks Coffee

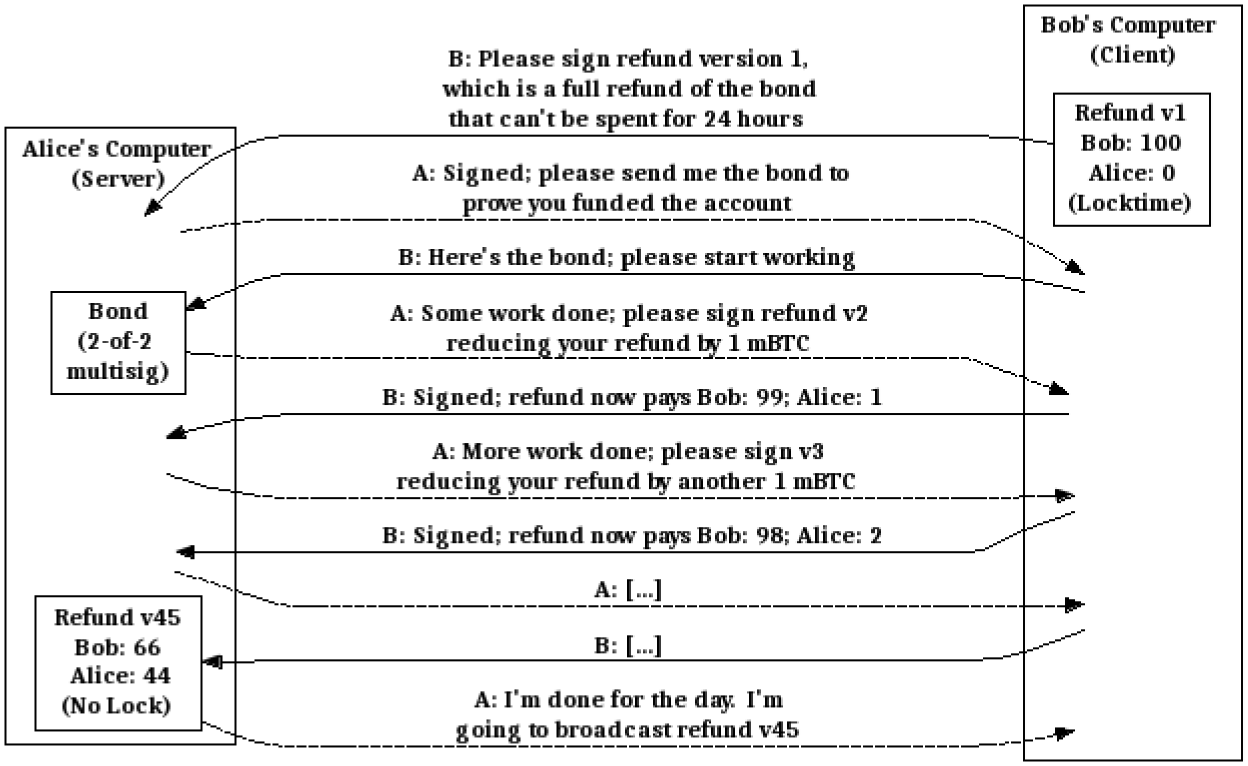


Broadcast to Network

Signed
between Wallets
not
Broadcast to Network

Broadcast to Network

Signature Scripts are used in Payment Channels


Bob's Computer (Client)
Refund v1
Bob: 100
Alice: 0
(Locktime)
Alice's Computer (Server)
Bond (2-of-2 multisig)
Refund v45
Bob: 66
Alice: 44
(No Lock)
B: Please sign refund version 1, which is a full refund of the bond that can't be spent for 24 hours
A: Signed; please send me the bond to prove you funded the account
B: Here's the bond; please start working
A: Some work done; please sign refund v2 reducing your refund by 1 mBTC
B: Signed; refund now pays Bob: 99; Alice: 1
A: More work done; please sign v3 reducing your refund by another 1 mBTC
B: Signed; refund now pays Bob: 98; Alice: 2
A: [...]
B: [...]
A: I'm done for the day. I'm going to broadcast refund v45

Alice broadcasts the bond to the Bitcoin network immediately. She broadcasts the final version of the refund when she finishes work or before the locktime. If she fails to broadcast before refund v1's time lock expires, Bob can broadcast refund v1 to get a full refund.

Bitcoin Micropayment Channels (As Implemented In Bitcoinj)

Blockchain Summary

- Blocks store records of transactions
- Ordered in time, immutable historical records
- Measured in block ‘height’ identified by id (hash of metadata)
- Tamper evident through hash of hashes, each block contains hash from the previous block
- Blocks are created by Miners
- New blocks are added to the chain

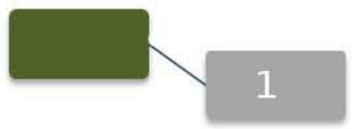
Simultaneously Broadcasting Blocks

Like the blockchain miners are distributed, so who gets to make the block?

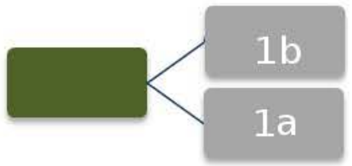
- Miners are distributed and are in competition with each other
- First to publish/broadcast a block wins
- At the top of the chain multiple miners could create a block at roughly the same time
- Blocks take time to propagate their way round the network
- Different nodes receive blocks at different times
- The network needs a way to decide which block it will use as it’s official record of what happened
- Mining doesn’t stop while all this is figured out

Recap: How To Build A Block

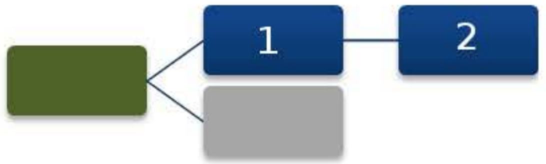
- Receive the transaction broadcast
- Verify the crypto in the transaction
- Add it to the unconfirmed pool
- Do some hard maths on all the transactions in the pool
- Broadcast the Block to the network
- The Block is added to the blockchain



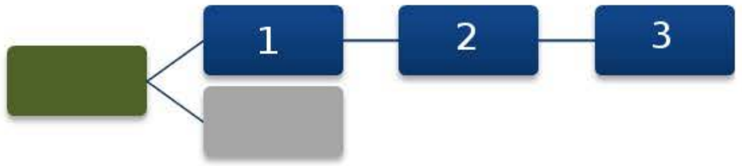
- Example: the green block is at height 0
- All miners try to solve the next one…
- A miner solves one at height 1



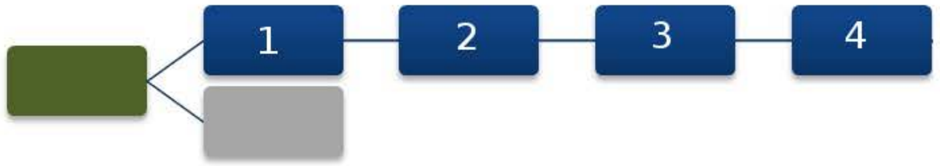
- But so does another miner. We don’t know which is official.
- Block 1a may contain different transactions from 1b.
- We don’t know which is the accepted block yet
- So, mining continues, with half the network working on 2a, half on 2b
- Good idea not to treat any transactions as final yet



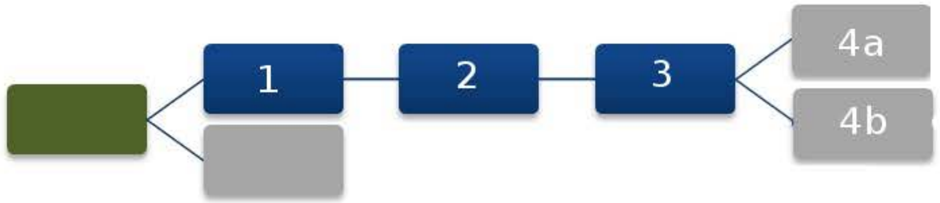
- A miner finds 2b
- All miners working on 2a stop work, they must work at the highest height
- All miners work on finding a block at height 3
- Transactions that were only in 1a, are now back to not-being-in-a-block



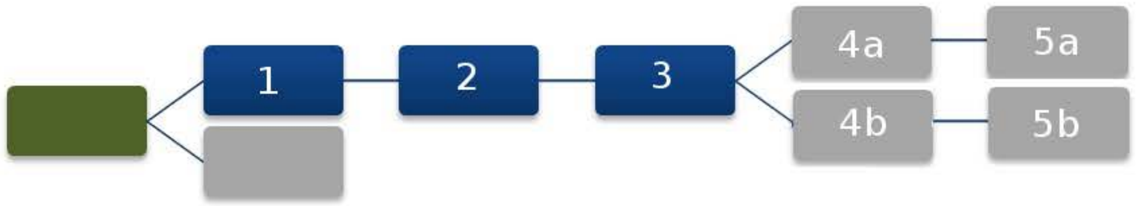
- A miner solves one and broadcasts to the network
- Other miners abandon their work and start trying to solve a block at height 4



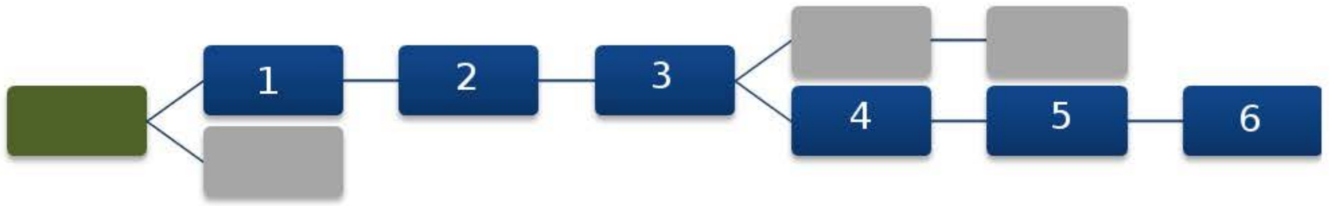
- A miner solves one at height 4, all is well with the world
- All miners try to solve a block at height 5



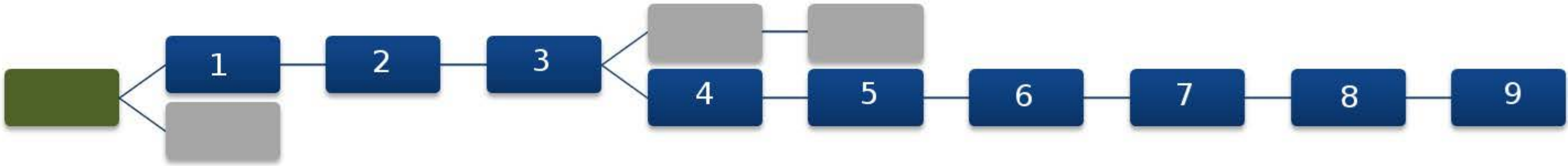
- But, while this is sorting itself out, another miner solves a block at height 4
- We now have a fork at height 4.
- Half the miners will try to solve 5a, half will try 5b



- We have a race condition, lets pretend that both forks of the chain solve another block at the same time



- The miners working on 5b solve a block first. All miners stop what they're working on, and try to solve for height 7
- Blocks 4a and 5a are now accepted as the longest chain



- Smooth sailing from here on out

Implications

- Transactions take time to ‘confirm’
- Each transaction, once it’s in an accepted block has a height
- Each increase in blockchain height is called a confirmation
- A transaction 5 blocks below the top of the chain is said to have ‘6 confirmations’
- The merchant can decide how many confirmations is sensible to wait for (cup of coffee, perhaps no confirmations; for a meal, 1 or 2 confirmations; for a car at least 6!)
- Wait for 6 confirmations for anything of value

Other Types of Forks

- Upgrades to the protocol can cause problems – but can be managed
- Blocks that are created have a version number
- New blocks using the new protocol use a different version number
- If the upgrade is backwardly compatible, it's a soft fork
- If the upgrade isn't backwardly compatible, it's a hard fork
- Hard forks are much harder and we try to avoid them

Any Questions?

Transactions Summary

- Transactions transfer control of coins from inputs to outputs
- Control is enforced by cryptography
- Refresh on public/private keys
  - Two keys, one private one public
  - Encryption by one, decryption by the other and vice versa
  - If you can encrypt a known value, which is then decrypted by the public key, you must have the private key

To Construct a Transaction

- You need the address of the person you're going to pay
- Find the Unspent Transaction Outputs (UTXOs) that exceed the amount you wish to pay (that you control)
- Calculate the transaction fee (optional but recommended)
- Create the outputs with the correct scriptPubKey
- Sign the transaction details
- Broadcast the transaction, see if it works

ScriptPubKey Is A Mini-Program

- scriptPubKey defines who can spend the coin by specifying a small verification program that is run in order to perform that verification
- 40 bytes of instructions
- Forth like scripting language, deliberately not Turing complete
- Elements are pushed onto a stack, if the end of the stack is true then the Transaction is valid and works

ScriptPubKey Is A Mini-Program

- You can get quite clever with it
  - Multi-signatures required where m of n are required
  - Verifications that don’t need private keys
  - Time bound, escrow services
  - The beginnings of smart contracts
  - Deliberately make coins unspendable

Signature Scripts

- In order for the network to validate and relay your transaction you have to prove you have that private key

- You do this through the SignatureScript
  - Full unhashed public key
  - A secp256k1 signature of most of the transaction data
    - The transaction ID & output index of the input
    - Previous transaction's scriptPubKey
    - The output's scriptPubKey
    - The value of the transaction

- The signed transaction is then sent to the network for relaying

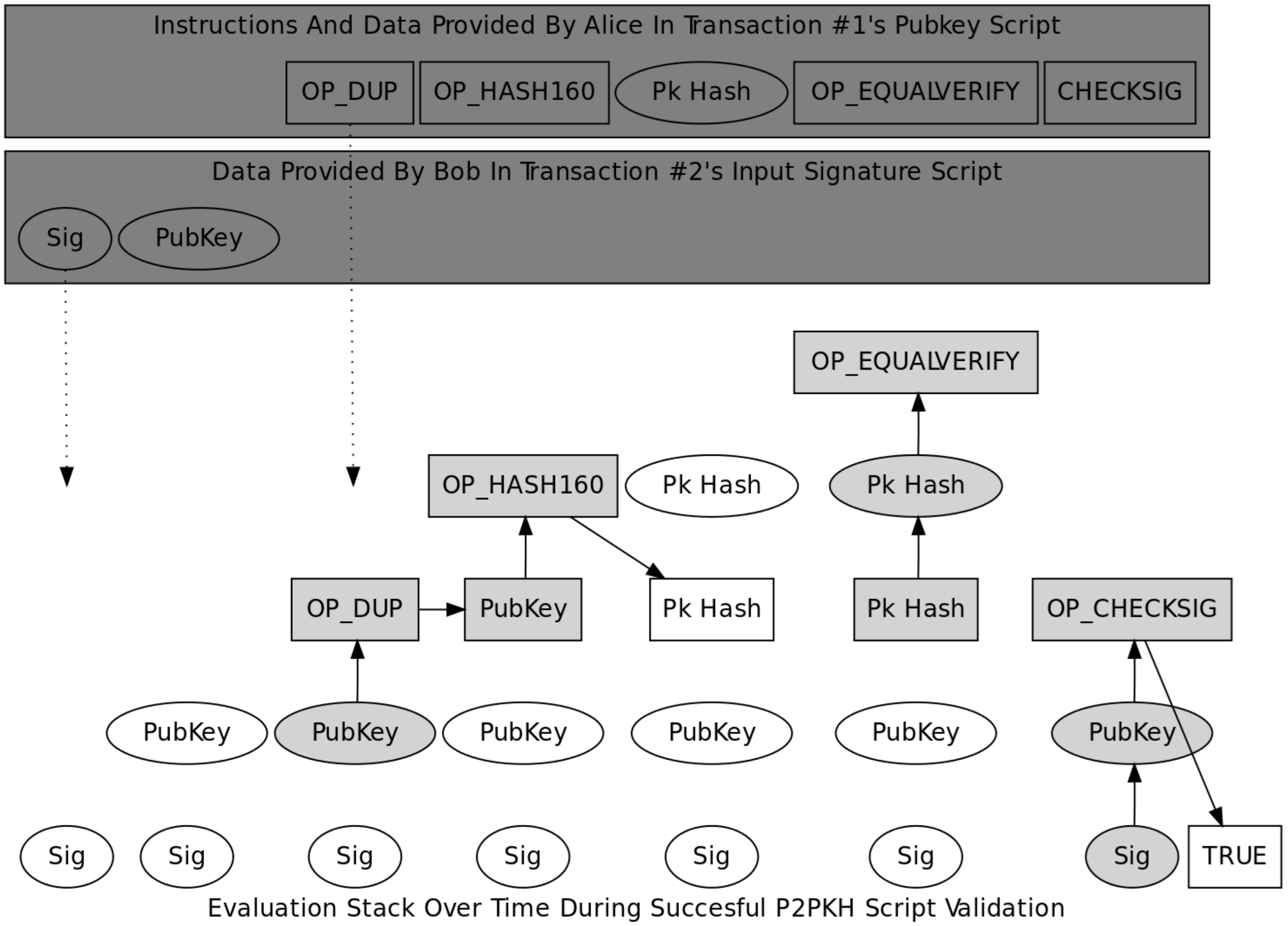
P2PKH

- When decoded the scriptPubKey says this:

```
OP_DUP OP_HASH160 <PubkeyHash> OP_EQUALVERIFY OP_CHECKSIG
```

- You append the SignatureScript to the front, then evaluate LtoR

<Sig> <PubKey> OP_DUP OP_HASH160 <PubkeyHash> OP_EQUALVERIFY OP_CHECKSIG


Instructions And Data Provided By Alice In Transaction #1's Pubkey Script
OP_DUP
OP_HASH160
Pk Hash
OP_EQUALVERIFY
CHECKSIG
Data Provided By Bob In Transaction #2's Input Signature Script
Sig
PubKey
OP_EQUALVERIFY
OP_HASH160
Pk Hash
Pk Hash
OP_DUP
PubKey
Pk Hash
Pk Hash
OP_CHECKSIG
PubKey
PubKey
PubKey
PubKey
PubKey
PubKey
Sig
Sig
Sig
Sig
Sig
Sig
Sig
TRUE
Evaluation Stack Over Time During Succesful P2PKH Script Validation

This is Complicated

- ➔ Use a library or toolset to do this encryption - if it’s wrong you don’t get error messages, it just doesn’t work
- ➔ … and you could lose your coins!
- ➔ Start on the testnets
  - work just like the real networks
  - coins are free
  - playground for experimentation

Store Data in the Blockchain

- ➔ Space is limited - 79 bytes in the testnet
- ➔ If your data is larger than that, use an external reference
  - · URL to object
  - · Magnet link for torrentable files
- ➔
- ➔ People store things on the blockchain
  - · Messages
  - · Prayers
  - · Proof of existence
  - · URLs/email addresses/auth codes
  - · Marketing
- ➔ FlorinCoin blockchain provides more storage space

Any Questions?

Store Data In The Blockchain

- Burn 0 satoshi as one output as part of a bigger transaction, and pay the mining fee
- First command says ‘these coins can’t be spent’ - takes 1 byte
- Known as OP_RETURN, always evaluates to false
- What you do with the rest of the data is up to you
- You’ll need a special script that looks for this data and processes it independently
- Obviously, read only once the Transaction is sent