BLOCKCHAIN

TRAINING ALLIANCE

BLOCKCHAIN
TRAINING ALLIANCE

BLOCKCHAIN
Technical Overview

www.blockchaintrainingalliance.com

Blockchain Summary

- Blockchains record Transactions
- Transactions are time dependent and sequential
- Once you've spent coins, you can't go back in time and spend again
- Blockchains record transactions securely in the order in which they occurred

Blockchain Rules

- Once something has happened, and we create a record of that, the fact that it happened never changes
- Data written into a blockchain is a historical record and is immutable
- Blockchains have to prove that they haven’t been tampered with
- Everyone running that blockchain has to agree on the data stored in it

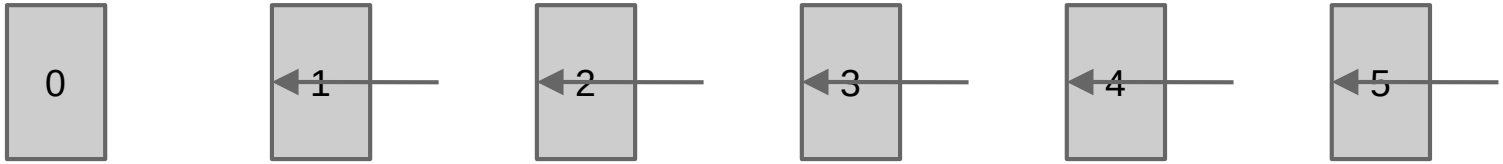
Blockchain From First Principals

What you need to know

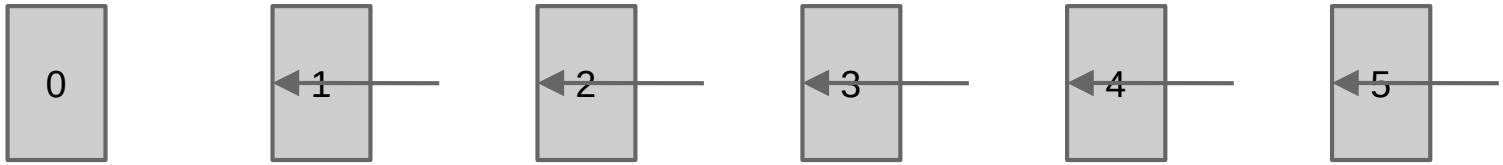
Hashing

- http://passwordsgenerator.net/sha256-hash-generator/

Transactions are grouped together into a Block

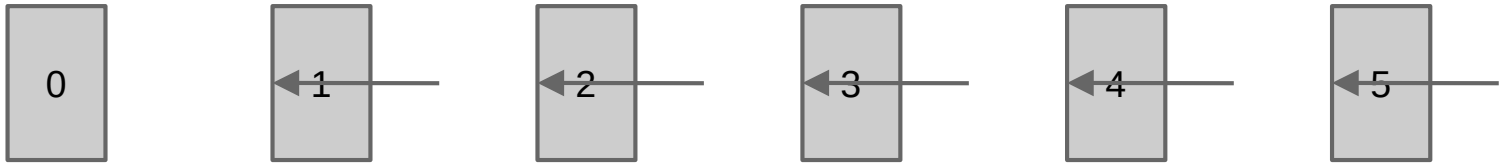


Blockchains



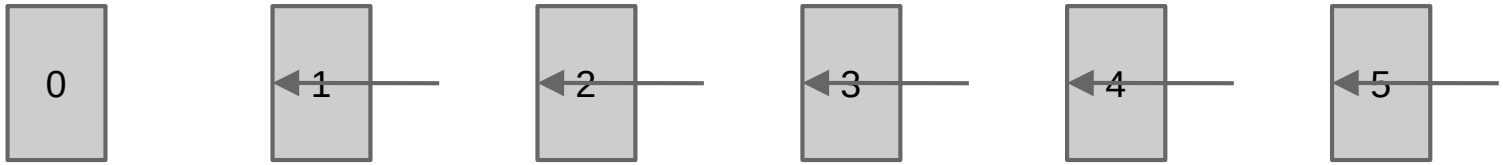
- Blocks are numbered in ascending order, 0 is first/oldest
- The number is the ‘height’ of the block
- Arrows only go from newer to older blocks - a block only directly links to the one immediately before it
- Once a block is stored, it’s read-only (which is why it doesn’t link to the ones after it - that would require you to update it)

Blockchains



- Blocks store data, in Bitcoin, it's the transactions, but it could be any digital data
- Blocks are created periodically (on average, 10mins for Bitcoin) by a process called 'mining'
- A block represents a set of events that have occurred over a particular time frame (usually, since the previous block)

Blockchains



- Blocks aren't identified by their height, but by their id
- Block id is the hash of the data in the block

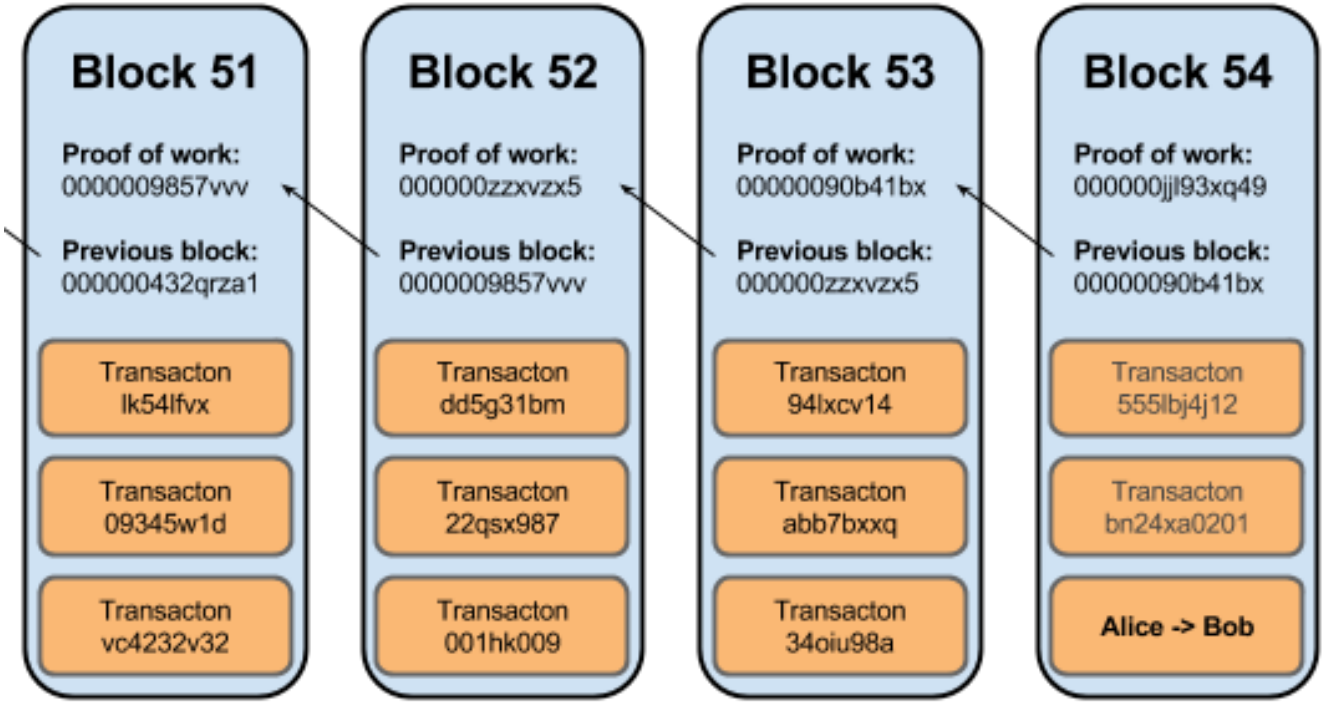
```
0=000000000019D6689C085AE165831E934FF763AE46A2A6C172B3F1B60A8CE26F
1=00000000839A8E6886AB5951D76F411475428AFC90947EE320161BBF18EB6048
2=000000006A625F06636B8BB6AC7B960A8D03705D1ACE08B1A19DA3FDCC99DDBD
```

- Block id is a digital fingerprint of that block

What is in a Block?

- A ‘magic number’ (0xD9B4BEF9) to show it’s a Bitcoin block
- A size number to specify how much data is coming next
- Some metadata:
  - A version number of the block format
  - A link to the previous block that came immediately before it
  - Merkle root of all the transactions in the block
  - Timestamp of when the block was created
  - Mining difficulty (more about this later)
  - Nonce for proof-of-work (more about this later)
- All the transaction that were recorded in this block

What is in a Block?



Tamper Evident

- Blocks are identified by their ID (hash of the metadata)
- Metadata in turn, contains Merkle Root of Transaction data
- Change the metadata, block id will change - broken chain
- Change the details of a transaction, the Merkle root will change, which in turn changes the metadata hash, which will change the block id
- Hashing data once is really quick, so easy to check the validity of each block as you go along

Why Not a Database

- Blockchains solve specific problems:
  - Fully distributed - highly fault tolerant
  - No centralized authority
  - 3rd party trust without trust
  - Low barrier to entry - computer + internet = win
  - Instant, Global transactional capability
  - No double spending
  - Very low transaction costs
- Database can’t do this yet

Blockchains as Distributed Databases

- In Bitcoin, everyone has a copy of the Blockchain
  - Everyone running the Bitcoin client is part of the network
  - New blocks are broadcast to the network
  - Everyone updates their local copy of the blockchain
  - If you're behind the current height of the chain, you can ask other nodes for copies of the Blocks needed to catch-up
  - If everyone has a copy of the blockchain, when queried, everyone gets the same answer
  - Blockchains are a bit like read-only distributed databases

How to Get Started

- Choose the blockchain you want to work with:
  - Download the reference client from their website
  - Wait for the blockchain to download
  - Obtain some coins buy/earn some
  - Start transacting

How to Get Started

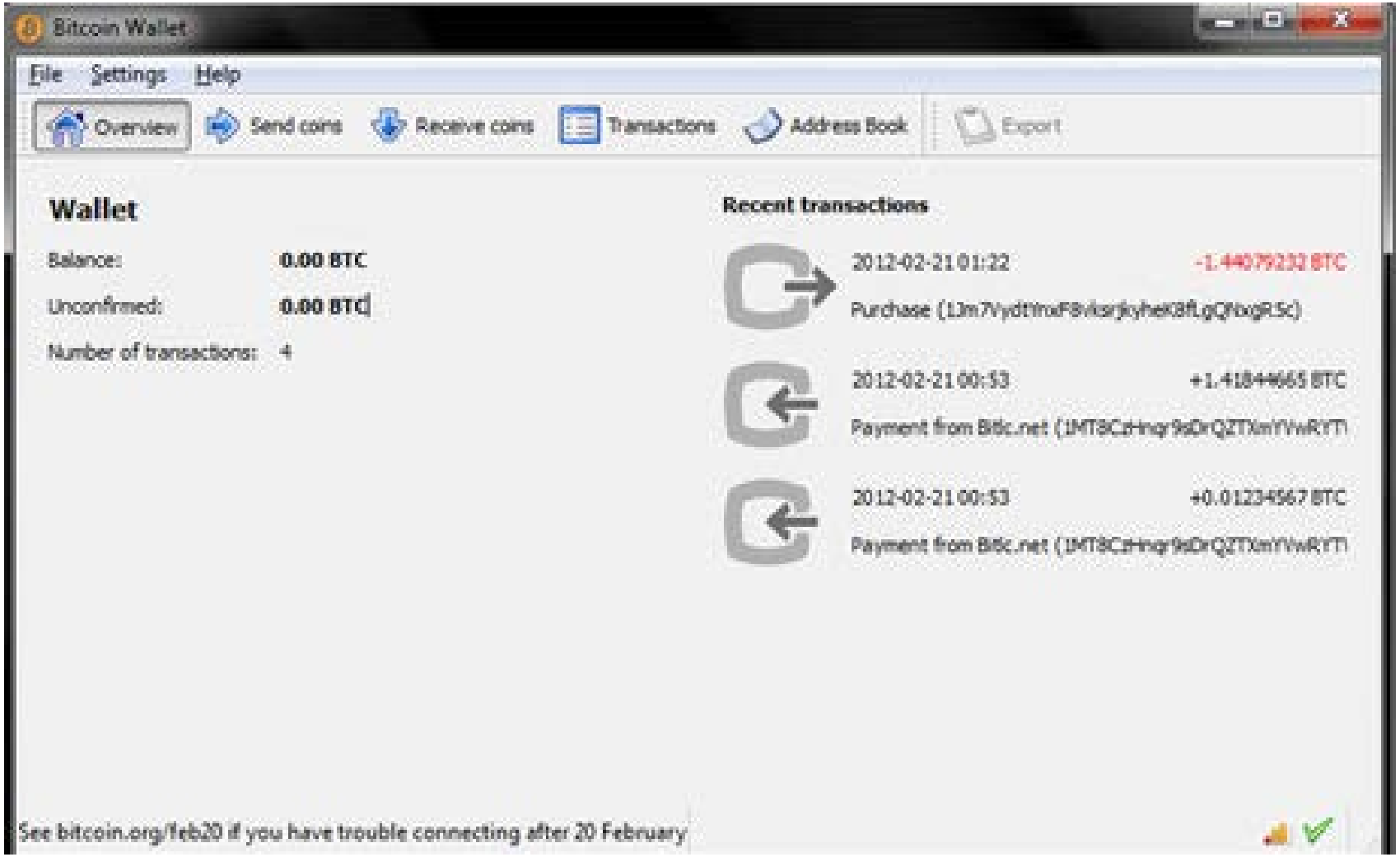
There are two types of client software:

- **Lightweight Client:** Doesn't download the entire blockchain, connects to other nodes and only collects information on transactions to it's own addresses

- **Full/Core Client:** Runs as a full node on the network, downloads entire blockchain

Smart phone software wallet app

Wallet transaction log



Any Questions?

BLOCKCHAIN
TRAINING ALLIANCE

BLOCKCHAIN
Encryption Overview

www.blockchaintrainingalliance.com

Transactions Summary

- Transactions are the focus
- Transactions transfer value between the sender and recipient
- Public/private keys (PKI Infrastructure):
  - 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

You Control Access to Assets

- Nobody actually ‘has’ bitcoins - you can’t download them, or store them on your computer
- Remember that the blockchain is a ledger - it records the transfer of bitcoin between people
- The records stay on the blockchain - what gets transferred is the ‘control’ of the bitcoin
- ‘Alice gives Bob 2btc’ becomes ‘Alice transfers control of 2 of her bitcoins to Bob’
- Control through cryptography

Control via Cryptography

- When someone sends you coins they publicly place them under the control of your public key (in the form of an Address)
- If you can prove that you have the matching public key, and the matching private key, the network lets you control the coins
- This gets super technical, super quick. Don’t panic if you don’t get all of this in your first pass

Control via Cryptography

- Because blockchain transactions are anonymous, there needs to be a way to enforcing controls on the coins
- Transactions are programmable
- Each transaction contains a program that specifies conditions that must be met in order to spend the coins

Control via Cryptography

Think of it this way…

- You don't have the amount of money you claim to have
- You only have the money that you can convince the network that you control
- You convince the network by proving you have both the public key and the private key that controls those coins
- If you can't convince the network, you don't have that money any more

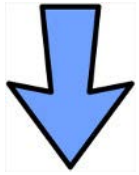
Protecting Your Private Keys

Think of it this way…

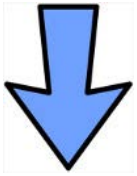
- Your private keys control your money. Most bitcoin ‘hacks’ or ‘thefts’ are in reality, crackers getting a copy of your private key. They then use this to move your coins to an address they control
- If you lose your private key, you lose your money
- Keep the offline (on paper!) or a specialised hardware device
- If they must be online, encrypt them when not in use, only use for small amounts of money, use cold/warm/hot wallets to manage risk

Addresses

Private key

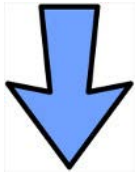


Public key



Hash

Encode



Address

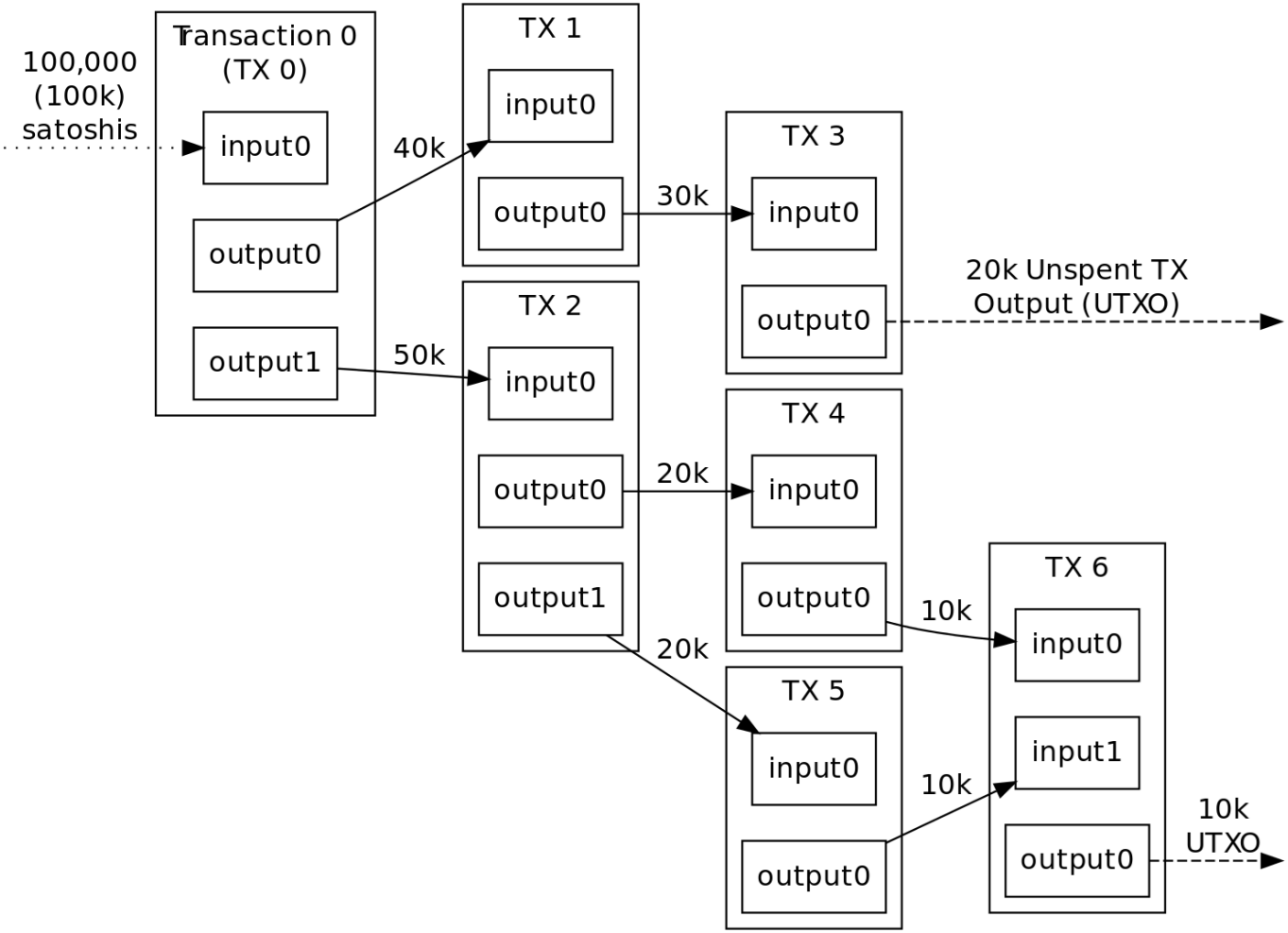
Addresses

- Hides your public key (because of the hashing), but you still have both the public and private key
- This is your bitcoin ‘Address’ - you tell people who need to send you coins this address
- You will have many Addresses
- Your wallet software keeps track of all payments made ‘to’ your Address

Transactions Control Ownership

- The transaction transfers ownership control from its inputs to its outputs
- Someone controls some bitcoin and they put them into the transaction, the recipient receives the outputs from the transaction
- The outputs from an earlier transaction form the inputs of the later one
- When we say ‘Bob has 2 bitcoin’ what we mean is ‘Bob has control of one or more unspent transaction outputs that total 2 bitcoin’

Transactions Control Ownership



Triple-Entry Bookkeeping (Transaction-To-Transaction Payments) As Used By Bitcoin

How to Spend an Input

- When you create a transaction, you transfer control to a hashed public key
- Whomever has the private key that matches the public key that matches the hash+enc of the public key (i.e. the Address) can spend these coins
- These instructions are embedded in the transaction in a field called scriptPubKey
- It looks like this: 76A9149260C8E4924720B040F20B00D7F78C0F0FDB A3C288AC

Spending an Input uses it up

- Once a transaction's outputs are used as inputs in a new transaction, they are considered 'spent'. If the owner tries to use them again, it's known as a 'double spend'
- A transaction's inputs reference the UTXO they are taken from
- Using a UTXO destroys it - you must fully spend all of the value
- If you don't want to spend that much, you send the 'spare' value back to yourself to a 'change address'