Distributed Ledger (Blockchain)

From Ground UP↑↑

Shashank Rai

What we may have heard / read about....

Cryptocurrency

Bitcoin (mining)

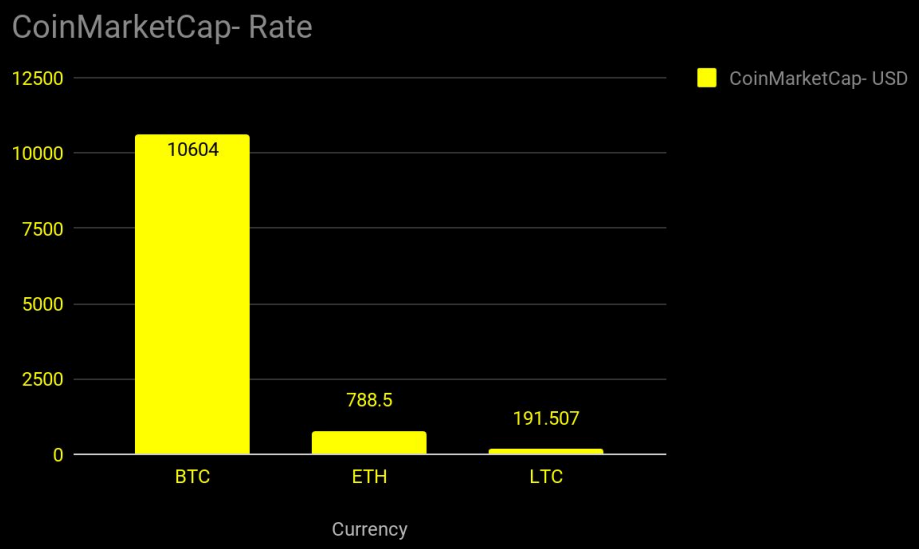
Blockchain

Satoshi Nakamoto :)

Ethereum

Distributed Ledger??

$$$ (beware of investment decisions :)



How does it all add up….

- Blockchain is a form of ‘Distributed Ledger’ (DL)
  - We’ll talk about DL Technology (DLT) in a few slides from now
- Bitcoin is a specific cryptocurrency based on blockchain; the core concepts of the currency conceived by yet to be identified person(s): Satoshi Nakamoto:
  (https://www.mail-archive.com/cryptography@metzdowd.com/msg09959.html)
- Ethereum is a technology platform for creating *blockchain* based distributed ledgers
  - ETH is a cryptocurrency based on the Ethereum platform
- As for the $$$ …. We are not here to talk about crypto trading

Topics

- ★ Housekeeping
- ★ About:yourHost
- ★ Common Concepts (#7)
- ★ Bitcoin Dissection (#9)
- ★ Ethereum
- ★ Hyperledger Project
  - Fabric (w/ examples)
- ★ Business Application

Time Permitting

Great Expectations

- ★ We'll be covering topics from an entry level….
- ★ ….based on the RSVP form feedback
- ★ End of the show - walk away with clear understanding of DLT/Blockchain and some non-cryptocurrency use.
- ★ Techies - get lead into different toolsets & languages available
- ★ AND a lot of leads into reading on Internet and self-experimenting

*apologies to Charles Dickens

Housekeeping points

- ★ Q&A + Voting: URL at the top
- ★ Remote attendees
  - Can you please mute your microphone
- ★ Assembly Point:
  - Straight out of the main entrance, turn left.
  - Walk along the building - to big parking lot.
  - Assembly Point Board - across the parking on your left
- ★ Facilities - across the door, down the stairs.
- ★ Coffee break
  - At your own leisure - Cafe closes at 1600

Housekeeping points

- ★ All material will be made available along with recording
- ★ Will also provide links to sources of information:
  - Nos esse quasi nanos gigantum humeris insidientes.
    - Please do chip in
- ★ Shared under CC BY-NC-SA 4.0: https://creativecommons.org/licenses/by-nc-sa/4.0/
  - Creative Commons, Attribute, Non-Commercial, Share Alike:
- ★ CHATHAM HOUSE RULE: When a meeting, or part thereof, is held under the **Chatham House Rule**, participants are free to use the information received, but neither the identity nor the affiliation of the speaker(s), nor that of any other participant, may be revealed (https://www.chathamhouse.org/about/chatham-house-rule)
- ★ Interchangeable use of some terms: DL / DLT / Blockchain - makes talking easier.

About:yourHost

International Computing Centre (ICC)

www.unicc.org

ICC

- ★ Providing ICT Services to the the United Nations family and not-for-profit international organizations for 45 years
- ★ Hosted by the World Health Organization (WHO)
- ★ Serving clients globally from
  - Geneva, Switzerland
  - Valencia, Spain
  - New York, USA
  - Rome and Brindisi, Italy
- ★ Strong team comprising of 250 regular staff and nearly 180 consultants covering multiple ICT disciplines

What We Do

Client Advisory Services

- Strategic Consulting
- Subject Matter Expertise
- Access to IT Advisory Services
- Information Security
- Training

Software-as-a-Service

- Application Development
- Unified Communications
- Identity and Secure Access Management

Platform-as-a-Service

- Business Intelligence
- Enterprise Resource Planning (ERP)
- Web Applications
- Database and Middleware
- Directory, Resources and Domains

Infrastructure-as-a-Service

- Managed Computing
- Managed Network
- Monitoring

Public Cloud Integration

- Software Integration and Management
- Platform Integration and Management
- Infrastructure Integration and Management

Professional Services

What we do…in our spare time

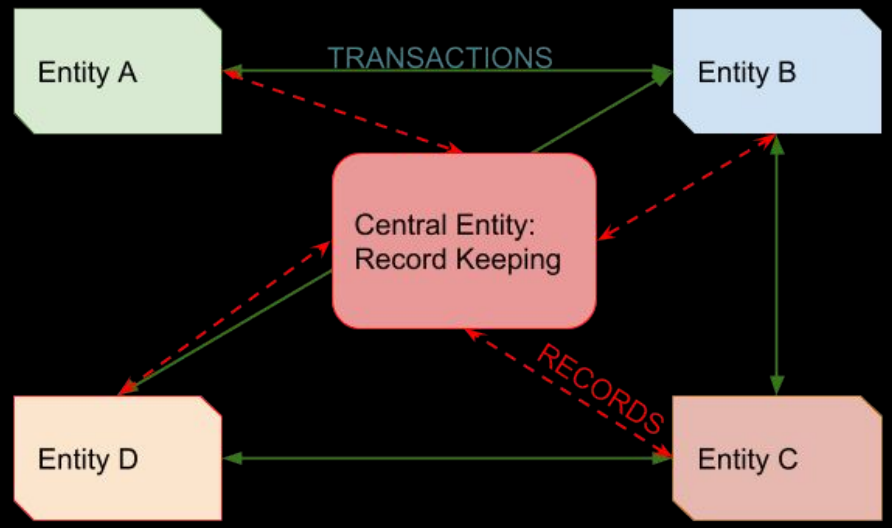
- ★ Geeks who love to play with technology ;)
- ★ “devOPS- from Git to Prod” (*topic for our next gathering* - PLEASE VOTE)
- ★ big data (spark on hadoop garnished with Python)
- ★ IoT (particle.io)
- ★ flash LineageOS on android phones….
- ★ ….and integrating Tesla with Amazon Echo

Common Concepts

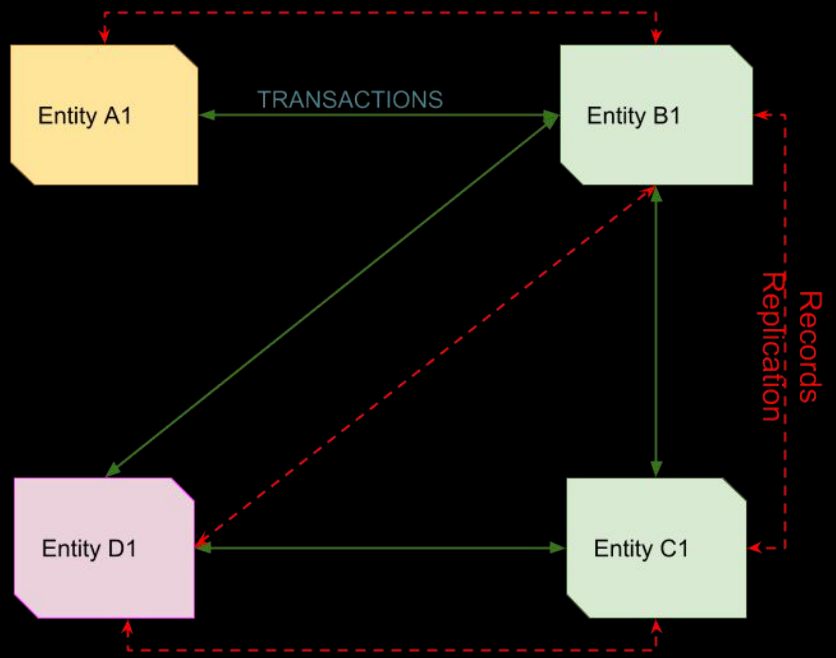
Context for Distributed Ledger

- ★ Record keeping - one of the *many* use-cases for distributed ledger
- ★ Next couple of slides are a very simplified view of business transactions...
- ★ ...to set-up the context for DLT
- ★ More use-cases discussed later

Record Keeping: Functional View



Through a central authority

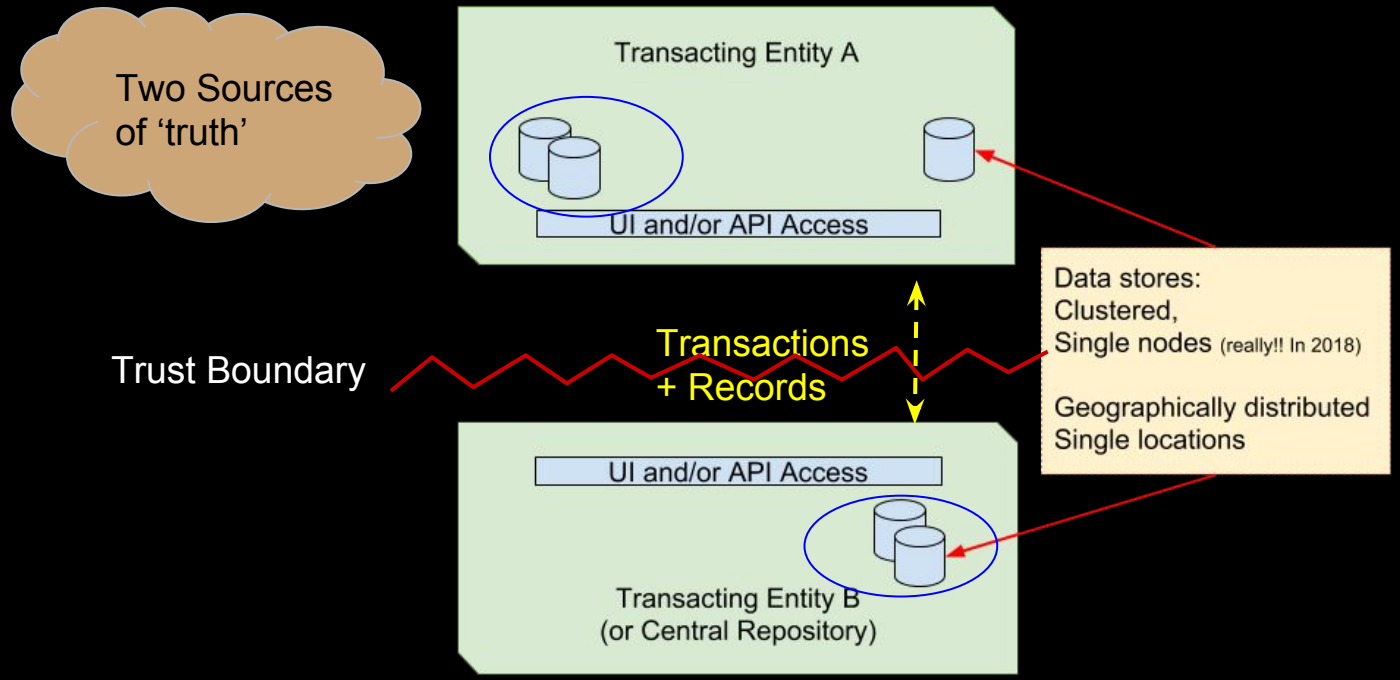


Without a central authority

https://www.theguardian.com/business/2018/feb/27/hawk-or-dove-bitcoin-is-forcing-central-banks-to-take-sides

Record Keeping: Technical View

Entity A: Keeps copy of records - its own copy is treated as authoritative. Data can be distributed across multiple geographies



Entity B: Keeps copy of records - its own copy is treated as authoritative.
In Case of a 'central repository' - it may have precedence over Entity A's copy

Distributed Ledger

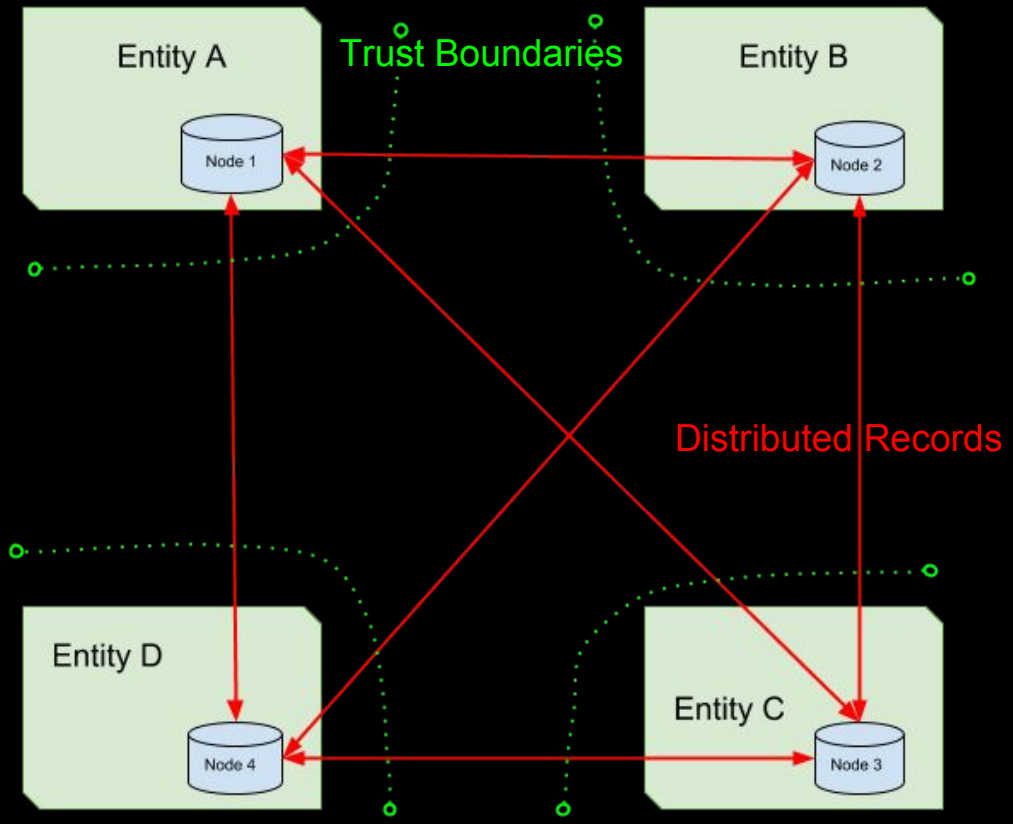
Inter-entity transactions (as usual)

Common view (**same copy**) of the records through:

- ❑ Consensus on transactions
- ❑ Peer-to-peer network for data exchange
- ❑ Agreement on who can participate in this P2P network

Certain Transactions can lead to automatic actions or vice-versa

Side effect: All nodes store all data



DL: Terms we picked up

- Peer-to-peer Network + participating ‘nodes’:
  - Node: End-point (computing device) participating in the DL network
  - P2P: Different protocols used to establish such a network (think Bittorrent).
    - Ethereum: ÐΞVp2p* (which in turn uses RLPx for P2P - based on Kademlia for node discovery). TCP Port 30303 by default
- Permissioned: Single Central OR a set of delegated members decide which node can participate in the DL (and a few other nuances in between)
- Permissionless Network: Anyone and everyone can join - Hola Bitcoin!!

* Ð: Modern Icelandic (and a few other lang) sound ‘eth’ Ξ: Greek Uppercase Xi

DL: Terms we picked up

- Consensus: Methods and means to ensure that the ledger is consistent and all parties agree to the state of the ledger
  - Including - avoiding ‘double-spending’ for crypto currencies
  - Proof of Work* (aka mining)
  - Proof of Stake
  - Practical Byzantine Fault Tolerance(PBFT)
  - A bit of deep dive a few slides later
- Smart Contracts** : Programs (lines of code) - running on top of the DLT; allows transactions between two un-trusted entities.
  - Chaincode (Go & Java implementations) in Hyperledger/Fabric
  - Solidity (quite close to C++ and Python) in Ethereum

* Very high energy consumption for computing (mining) ** https://www.investopedia.com/terms/s/smart-contracts.asp

How is this different from Distributed Databases

Distributed Database

Multiple nodes coordinate to keep a consistent view of the data

Nodes trust each other

There is a logical Central control

Examples:

- Traditional SQL databases (MariaDB, Postgresql, Oracle)
- NoSQL (Mongo, Cassandra, Redis, Google's BigTable, Couch)
- NewSQL (Google Spanner, VoltDB, Clustrix)
- Hadoop

Distributed Ledger

Multiple nodes build 'consensus' to keep a consistent view of the data.

Nodes do not trust each

No logical central control

Example:

- Blockchain (Fabric, Ethereum, Corda)
- Directed Acyclic Graph (DLT technology behind IOTA)

Source: https://gendal.me/2016/11/08/on-distributed-databases-and-distributed-ledgers/

Still waiting for….

…. What has all this got to do with mining coins and getting VERY rich….

Patience is a virtue

....mining part I'll get into more details ....

.....getting rich - well, best of luck!!!

Mapping to Bitcoin:

- Node: Head over to: https://bitcoin.org/en/full-node

  It's common for full nodes on high-speed connections to use 200 gigabytes upload or more a month. Download usage is around 20 gigabytes a month, plus around an additional 140 gigabytes the first time you start your node.
- Permissionless Network: Anyone and everyone can join - become a node.
- Consensus: Uses Proof Of Work- Algorithm Hashcash
- Smart Contracts: Not really - only one type of transaction. Not Turing Complete

  (does not compute any algorithm)
- Don't Need to be a node for transactions:
  - Need 'wallet' + 'address'.... Don't think we'll get time to talk about these :(..but there is hope

Bitcoin Dissection

Got some Hash ;)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam elit tellus, blandit sed hendrerit in, blandit ut lorem. Aliquam erat volutpat. Sed risus enim, congue eu maximus nec, vehicula id ligula. Donec in ipsum eget tortor posuere blandit. Cras mattis condimentum odio a viverra.

Pass through

Hashing Algorithms

A unique value. Changes even if 1 bit in the original input changes*

sha256sum <above text> = e09c704e1dbd19c398643110cc3de3e29e9dff9b8dd6ce5f798ed3caca7bd292

SHA256 Double (SHA256d): run SHA256 algo again on hash

echo <text> | openssl dgst -sha256 -binary | openssl dgst -sha256

32edd8acf2fa5df4d565c2321890474edc0d2dc4f40e2922e8cca50d36632a0c

* Collision is possible in hashing algos such as MD5 etc.

Bitcoin dissection

- *What's a block*: A bunch of transactions (transfer of Bitcoin value) from last 10 minutes* + 4 other fields:

*Arbitrary Number: Ethereum 1 min

Magic Number	0xD9B4BEF9
Blocksize	Size of block in bytes
Blockheader	*6 items*
Transaction Counter	# of transactions in the block
Transactions List	SHA256d of transactions

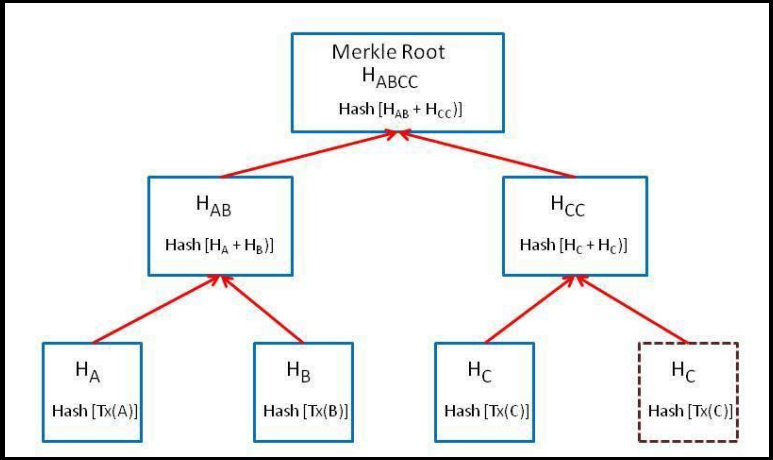
- Block Header

Version	Block version number
hashPrevBlock	256 bit hash of previous BLOCK HEADER… ***THIS IS THE CHAIN PART***
hashMerkleRoot	256-bit hash of all TRANSACTIONS in THIS block
Time	Timestamp (epoch - # of seconds since 00:00:00 1/Jan/1970 - leap seconds)
Bits	Current <target> in compact format *< more in a couple of slides...patience ;) >*
Nounce	32 bit number

Keep an eye on these two

Merkle Tree - not be confused w/ other Merkel

SHA256***d*** hash of leaf transactions ➡ SHA256***d***(*hash pair*) ➡ climb up the tree to end with one hash



Source: https://medium.com/all-things-ledger/bitcoins-implementation-of-blockchain-2be713f662c2

The mining part

- Proof of Work (hashing for cashing): Build YOUR BLOCK
  - Take transactions from the transaction pool (some nuances re transaction priority and age etc).
  - Generate the coinbase transaction - the MOST important one :)
  - Take the most recent block header - the 'agreed state of the chain'
  - Build a complete block
  - Pick a random number or start with Zero - **NOUNCE**!!!
- Find SHA256d of the YOUR BLOCK *HEADER*
- Is the HASH <= current ***target*** ?
- If NO (i.e. hash>target), well - change the nounce and burn some more electricity!!!
- If YES, broadcast the new block to the network
  - ***Network agrees*** ?
  - Yes, you get coins (currently 12.5) from the coinbase transaction + transaction fee (collected from spender of the embedded transactions)
  - No, someone else 'blocked' the transactions - tough luck!!

Some other titbits

- Blocks generated circa 10 min
- The difficulty (or target) is adjusted to maintain this time
- Creation of a block generates ‘x’ # of bitcoins to reward the miner
- ‘x’ reduces every 210,000 blocks (circa 4 years).
  - Initially 50 BTC
  - Nov 28, 2012 - reduced to 25
  - July 9, 2016 - reduced to 12.5
  - Next reduction expected in 2020
- Bitcoin has an upper limit: 21×10^6 : 21 Million can be mined

Bitcoin- Target & Difficulty

- Difficulty - as the word means “the fact of not being easy to do or understand”*
- Number of Zeros the output hash has to start with
- Difficulty changes every 2016 blocks (circa 2 weeks)
  - At Genesis: 10 leading zeros
  - Currently, hovers between 17 to 19 zeros
- Current difficulty: https://blockexplorer.com/api/status?q=getDifficulty
- New Difficulty = Current Difficulty * (20160 minutes / <actual minutes to produce last 2016 blocks>)
- Target is inversely proportional to Difficulty
  - When difficulty increases, target decreases.
  - Target stored in ‘packed representation’ called Bits
- Target = TargetMax / <Current Difficulty>
- TargetMax = 0x00000000FFFF0000000000000000000000000000000000000000000000000000

http://learnmeabitcoin.com/guide/difficulty * Cambridge English Dictionary

How does it all add up?

Why type & waste when you can copy & paste.....

Show: https://visual.ly/community/infographic/technology/bitcoin-infographic

How a ฿itcoin transaction works

Bob, an online merchant, decides to begin accepting bitcoins as payment. Alice, a buyer, has bitcoins and wants to purchase merchandise from Bob.

WALLETS AND ADDRESSES

Bob and Alice both have Bitcoin "wallets" on their computers.

Wallets are files that provide access to multiple Bitcoin addresses.

An address is a string of letters and numbers, such as 1HULMwZEPkjEPeCh43BeKJL1ybLCWrfDpN.

CREATING A NEW ADDRESS

Bob creates a new Bitcoin address for Alice to send her payment to.

Each address has its own balance of bitcoins.

SUBMITTING A PAYMENT

Alice tells her Bitcoin client that she'd like to transfer the purchase amount to Bob's address.

Private key — Public key

Alice's wallet holds the private key for each of her addresses. The Bitcoin client signs her transaction request with the private key of the address she's transferring bitcoins from.

Anyone on the network can now use the public key to verify that the transaction request is actually coming from the legitimate account owner.

Private key — Public key

Public Key Cryptography 101

When Bob creates a new address, what he's really doing is generating a "cryptographic key pair," composed of a private key and a public key. If you sign a message with a private key (which only you know), it can be verified by using the matching public key (which is known to anyone). Bob's new Bitcoin address represents a unique public key, and the corresponding private key is stored in his wallet. The public key allows anyone to verify that a message signed with the private key is valid.

It's tempting to think of addresses as bank accounts, but they work a bit differently. Bitcoin users can create as many addresses as they wish and in fact are encouraged to create a new one for every new transaction to increase privacy. So long as no one knows which addresses are Alice's, her anonymity is protected.

VERIFYING THE TRANSACTION

Gary, Garth, and Glenn are Bitcoin miners.

Gary — Garth — Glenn

b4056df6 69f8ad7 2e56302d dad345d6

Their computers bundle the transactions of the past 10 minutes into a new "transaction block."

The miners' computers are set up to calculate cryptographic hash functions.

Hash value* + Nonce → New hash value

* Each new hash value contains information about all previous Bitcoin transactions.

The mining computers calculate new hash values based on a combination of the previous hash value, the new transaction block, and a nonce.

Cryptographic Hashes

Cryptographic hash functions transform a collection of data into an alphanumeric string with a fixed length, called a hash value. Even tiny changes in the original data drastically change the resulting hash value. And it's essentially impossible to predict which initial data set will create a specific hash value.

The root of all evil	6d0a 1899 086a... (56 more characters)
The root of all euil	486c 6be4 6dde...
The root of all veil	b8db 7ee9 8392...

Nonces

To create different hash values from the same data, Bitcoin uses "nonces." A nonce is just a random number that's added to data prior to hashing. Changing the nonce results in a wildly different hash value.

The root of all evil ??? → 0000 0000 0000 ...

Creating hashes is computationally trivial, but the Bitcoin system requires that the new hash value have a particular form—specifically, it must start with a certain number of zeros.

The miners have no way to predict which nonce will produce a hash value with the required number of leading zeros. So they're forced to generate many hashes with different nonces until they happen upon one that works.

Each block includes a "coinbase" transaction that pays out 50 bitcoins to the winning miner—in this case, Gary. A new address is created in Gary's wallet with a balance of newly minted bitcoins.

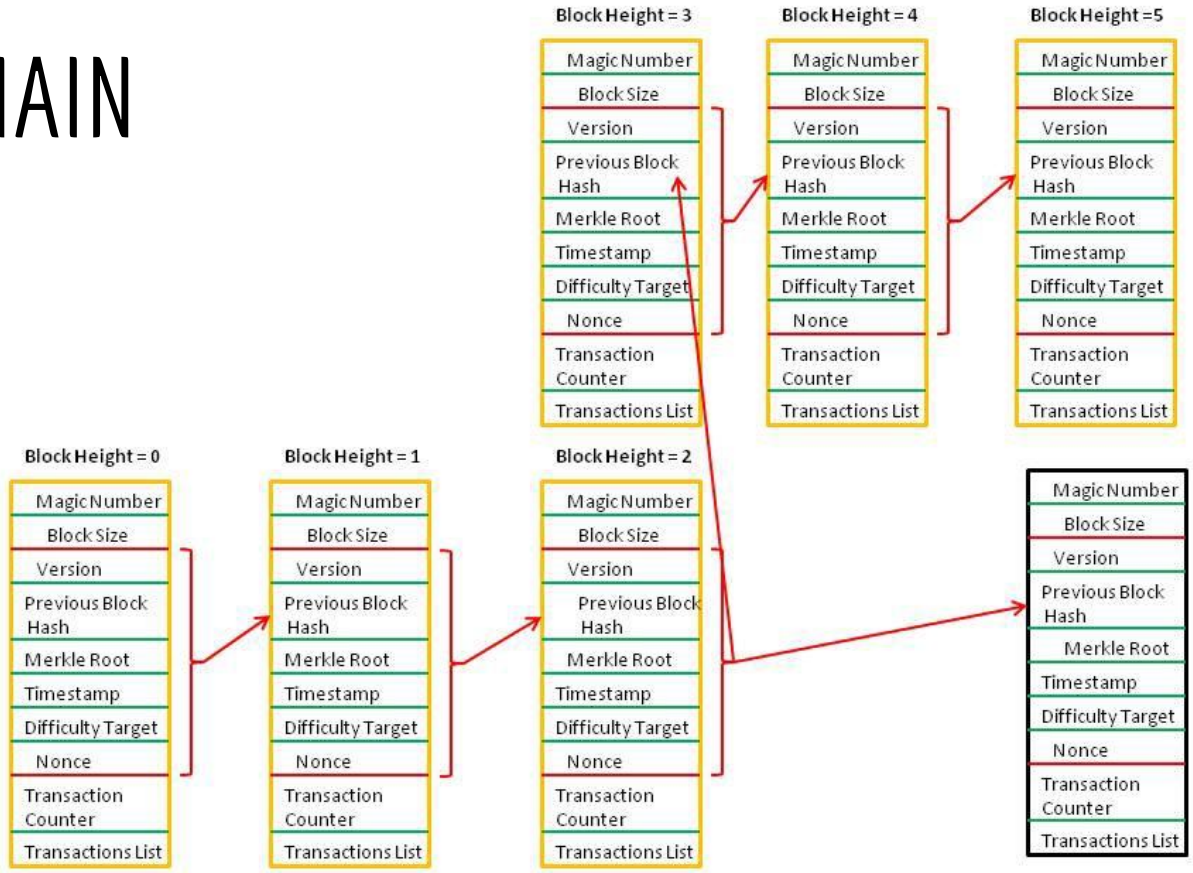
TRANSACTION VERIFIED

As time goes on, Alice's transfer to Bob gets buried beneath other, more recent transactions. For anyone to modify the details, he would have to redo the work that Gary did—because any changes require a completely different winning nonce—and then redo the work of all the subsequent miners. Such a feat is nearly impossible.

Bob & Alice

JOSHUA J. ROMERO, BRANDON PALACIO & MARISSOWNUSER INC.

BLOCKCHAIN



Source: https://medium.com/all-things-ledger/bitcoins-implementation-of-blockchain-2be713f662c2

Ethereum

Ethereum

- Ethereum Project(s): Open source code for blockchain based DLT
- Ethereum Public Blockchain: An instance of the ethereum code; running as a public blockchain and open to all (permissionless) - similar to Bitcoin chain.
- ETH (hard-fork) & ETC (classic) - coins on two different public chains... long story, short: Read about the DAO fiasco for the reason: https://blockgeeks.com/guides/what-is-ethereum-classic/
  - Similar to Bitcoin hard fork (BTC & new BCH- bitcoin cash).
- ETH: https://github.com/ethereum
- ETC: https://ethereumclassic.github.io/

Ethereum: DLT Characteristics

- Permission: Public Chains are permissionless
- Consensus: Proof Of Work for public chain
  - In the upcoming release (Casper) of Ethereum - public chain will also have Proof of Stake
  - PoW: Algorithm is Ethash - specifically created for Ethereum to avoid ‘mining centralisation’.
- Can be used to implement Private (permissioned) chain
- Consensus on Private chain:
  - Proof of Authority: One client w/ private key adds blocks to the chain
  - PBFT: Practical Byzantine Fault Tolerant**
  - DPOS: Delegated Proof of Stake**
  - Proof of Stake

Ethereum: EThash

- Fundamental concept is the same as hashcash:
  - Guess a nonce
  - Run some computation
  - Check if result is <= pre-specified value
- A very good explanation: https://www.vijaypradeep.com/blog/2017-04-28-ethereums-memory-hardness-explained/

Ethereum: Casper- Proof of Stake

- Set of validators take turns ***proposing*** & voting (in one variation of PoS) on the next block.
- Weighted votes based on size of validator's Ether deposit
- Anyone can be a validator - provided they have enough Ether
  - Send a special transaction that locks their ether into a deposit - bonding
- Bonded Validators - make money if block is added to the chain
- Lose money if the block is discarded
  - Same as losing all the electricity cost if the mining does not produce a successful block
- Forfeit of bond: In event of validator misbehaviour
- Variation in Private Chain:
  - Start with agreed trusted validators
  - Validators can agree to add new validators

Ethereum: Beyond Bitcoin...

- EVM: Ethereum Virtual Machine
  - Software running on Ethereum Nodes; that is capable of executing Ethereum bytecode from programs written in high level programming languages
  - Runtime environment (think jre)
- Smart Contracts (programs/scripts):
  - Execute Transactions under specific conditions
  - Programs written in Solidity (Current forerunner) - other options life Serpent, Viper etc
  - Compiled into EVM bytecode and deployed on the ethereum network.
- GAS: Fuel for running operations (computation tasks) on the Ethereum network
  - to ensure they have a finite state
  - The network is not misused
  - Each Operation in EVM has a amount of gas associated with it- determined by market

Ethereum: Beyond Bitcoin...

- Gas Price: Initiator offers a price (in ether) for per gas unit.
- Ether - the currency for ethereum network (like pence and pounds):
  - Basic unit is 'wei'
  - 10^{18} wei make 1 ether (yes, that's 1000 trillion)
  - 1 billions wei (10^{9}) is 1 gwei
- Median price of gas is 20gwei: https://ethgasstation.info/ *(at least last night when I was preparing these slides)*
- Setting Gas price in Ether - allows the cost of computing to be independent of the market value of Ether.
- DApps: Distributed Apps - Business applications that use Smart Contracts and the distributed and decentralized computing power of Ethereum
  - uPort - Identity Management (PoC w/ Canton of Zug)
  - CryptoKitties - seriously!!

Ethereum: Adding it all up...

- Initiator may want to run a simple transaction *(transfer from A to B)* or a complex DApp *(generate a kitty for me)* or a smart contract *(upon certain conditions being met; different transfers are initiated)* on ETH network
- Sets the Gas it will take (startgas) & a price per unit gas
- Miners take up the transaction - capitalism at work...lower gas price means transaction may not be picked up quickly
- Miners run the transactions
  - If transaction runs out of gas - execution is aborted; but initiator loses all money
  - If transaction is completed - miner keeps the burn cost and returns the excess back

Ethereum: Private Blockchain - Linux

- *geth is go-language based ethereum (ETH) client.*
  - Mac Users: https://medium.com/mercuryprotocol/how-to-create-your-own-private-ethereum-blockchain-dad6af82fc9f (uses homebrew)
  - Windows: https://geth.ethereum.org/downloads/ (ZIP archive has just the exe. Installer sets the path as well)
- This is for Fedora based distros - similar instructions (replace the dnf command with apt-get) for Debain/Ubuntu distros
- dnf install automake make gcc gcc-c++ git gmp-devel kernel-devel **golang**
- git clone https://github.com/ethereum/go-ethereum
- cd go-ethereum
- make geth
- export PATH=$PATH:$HOME/go-ethereum/build/bin
- **Don't start geth w/o any parameters - starts to join the public chain!!**

Ethereum: Private Blockchain

- *geth account new*
  - Passpharse “the woods are lovely dark and deep”
- Address: <PUBLIC KEY>
- Create genesis file & add the PUBLIC Key to it
- *geth init myGenesis_1984.json* - See sample Genesis File
- Show keystore & geth directories
- Keystore -> has the public/private keypair. Private is encrypted using the passphrase supplied at creation.
- geth -> has the chaindata
- *geth --networkid 1984 console 2>>1984logs.log* - Starts the javascript console
- *eth.coinbase*
- *eth.getbalance(eth.coinbase)*

Ethereum: Private Blockchain

- 2nd Console - *tail -f <log file>*
- *miner.start()*
- Will take time to generate the DAG file in ~/.ethash: https://github.com/ethereum/wiki/wiki/Ethash-DAG
- *eth.getbalance(eth.coinbase)*
- ←- we are getting RICH!!! --->
- *miner.stop()*

Ethereum: Tokens

- Usage Tokens: Native currency of a DApp - analogy of chips in a casino. Allow certain functions to be easily executed in a smart contract
- Work Tokens: Shareholder of a DApp. ICOs offer Work Tokens to raise capital

Ethereum: Smart Contracts Tools

- Remix Solidity IDE: https://remix.ethereum.org/
- Truffle Ethereum (github)

Hyperledger: Light Touch Intro

Hyperledger

- A bunch of projects under the umbrella of The Linux Foundation:
  - The same people who among other stuff manage the Cloud Native Computing Foundation (CNCF) - interesting projects for devOPS
- 5 Projects in Hyperledger:
  - **Indy**: Identity Management using DLT (donated by Sovrin)
  - **Fabric**: (donated by IBM) - Container based platform for developing blockchain based applications
  - Sawtooth - Honestly, I wasn't able to figure out the difference between this & Fabric (have played w/ Fabric only) except the origin is Intel
  - Burrow: EVM implementation
  - Iroha: Yet another blockchain implementation

Hyperledger: Fabric

- Permissioned Blockchain Only!
- Consensus:
  - Concept of MSP: Managed Service Provider
  - PBFT: Practical Byzantine Fault Tolerance
  - Solo
  - Kafka
- Fabric has an overlaying tool: Fabric Composer
  - WebUI to create business networks and deploy them
- Smart Contracts using ‘chaincode’ - let’s just see it in action!

Hyperledger: Fabric

- https://blockchaindevelop.mybluemix.net/editor
- Deploy new business network
- “Test-commodity-transfer” Empty Business Network
- Add model.cto & script.js - refer to separate files
- Click update
- Test the network:
  - Add traders
  - Add Commodity
- Submit Transaction
  - Type: Trade
- Commodity Owner has changed
- View All Transactions

DLT: Business Use-Cases

Business Use-Cases

- Ideas from the Floor
- Removing any central exchange system-
  - Clearing House: 3 examples from my professional life: Telco / ATI / IPPC
- Identity Management
  - Projects - Sovrin / uPort
  - ...a whole host of other initiative including within UN (UNHCR/ID2020)
  - Can be viewed as a part of eliminating a central registry
- Distributed Registry:
  - Land ownership records
- Trading Platforms:
  - Carbon Trading: Climate Chain Coalition
  - Futures: http://www.augur.net/

Challenges

- Volume of Data:
  - BigChainDB: https://www.bigchaindb.com/
- Foolproof consensus in permissionless DL without Proof of Work
-

Questions?