Foundations of FinTech

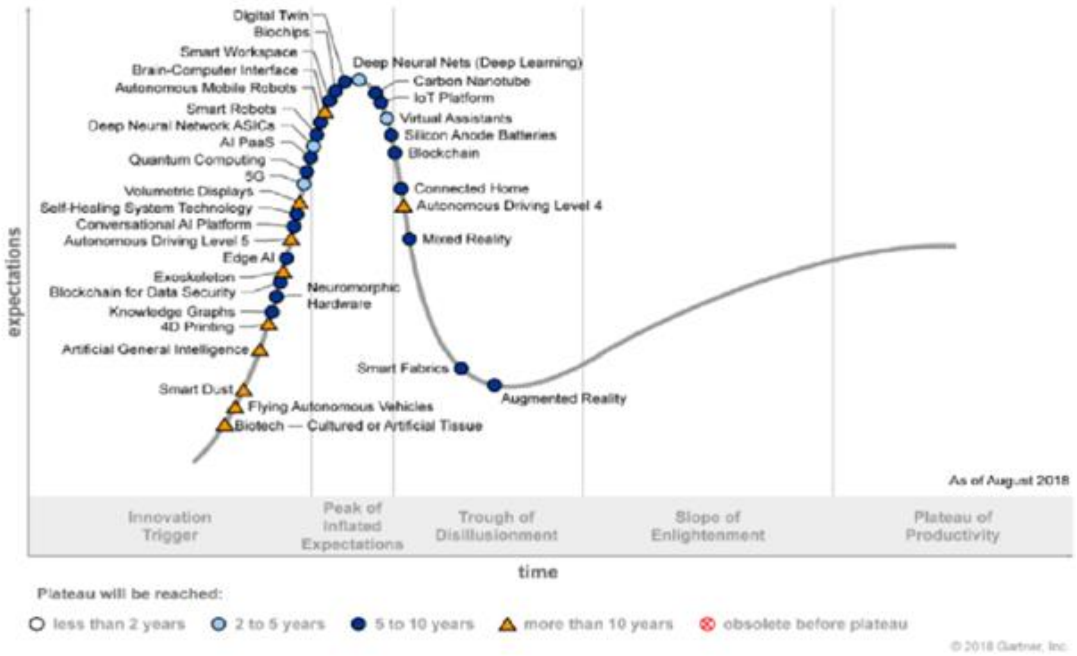
Blockchain

Eshwar Venugopal

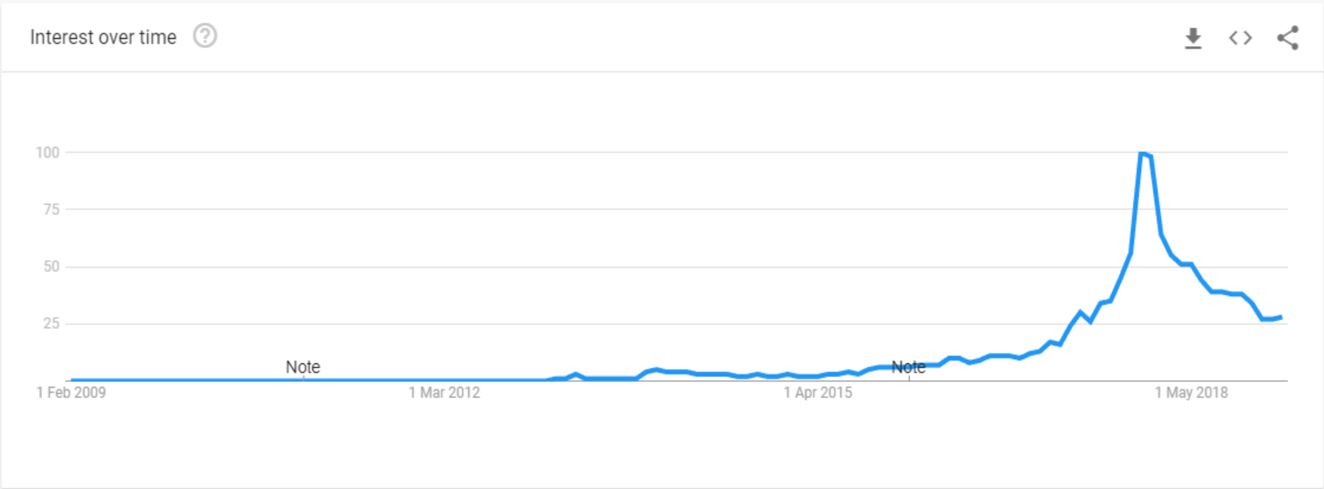
Blockchain 101

Hype around Blockchain

Figure 1. Hype Cycle for Emerging Technologies, 2018

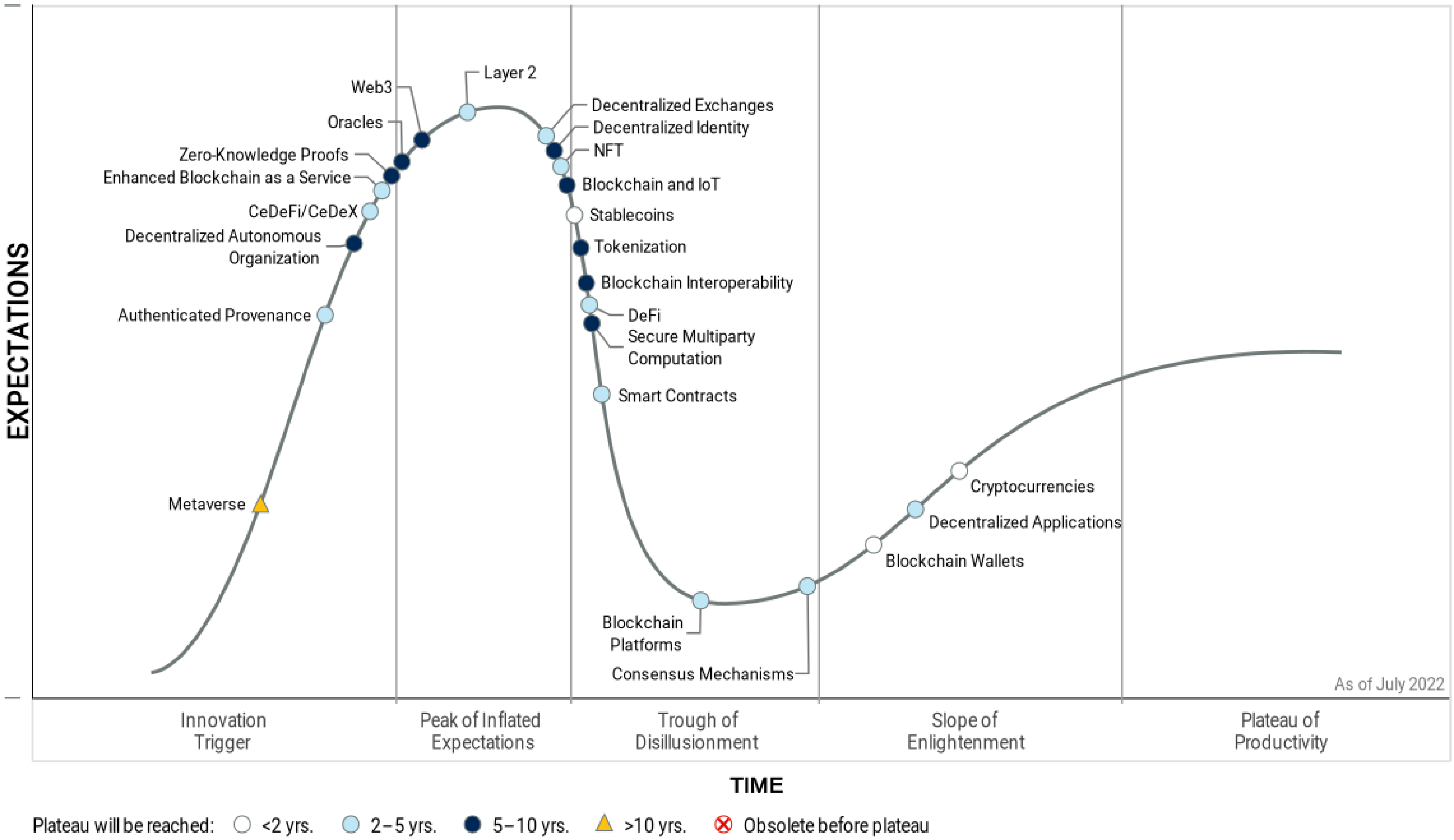


Source: Gartner (August 2018)



Source: Google Trends with search term "Blockchain"

Hype Cycle for Blockchain and Web3



Source: Gartner (August 2022)

The problem

- Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust-based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the minimum practical transaction size and cutting off the possibility for small casual transactions, and there is a broader cost in the loss of ability to make non-reversible payments for non-reversible services. *With the possibility of reversal, the need for trust spreads.* Merchants must be wary of their customers, hassling them for more information than they would otherwise need. *A certain percentage of fraud is accepted as unavoidable.* These costs and payment uncertainties can be avoided in person by using physical currency, but no mechanism exists to make payments over a communications channel without a trusted party.

-- *Satoshi Nakamoto (2009)* [emphasis and colors added]

The problem: Centralization

- Practically all of these machines have architectures that were designed to be controlled by a single person or a hierarchy of people who know and trust each other.... they can read, alter, delete, or block any data on that computer at will.... With current web services we are fully trusting, in other words we are fully vulnerable to, the computer, or more specifically *the people who have access to that computer, both insiders and hackers, to faithfully execute our orders, secure our payments*, and so on. If somebody on the other end wants to ignore or falsify what you've instructed the web server to do, no strong security is stopping them, only fallible and expensive human institutions, which often stop at national borders.

  -- *Nick Szabo, developer of "Smart Contracts" concept, accused of creating Bitcoin* [emphasis and colors added]

- *Problem in a video: https://www.youtube.com/watch?v=s4g1XFU8Gto*

The Solution: Distributed Ledger Technology

- Blockchain, a form of Distributed Ledger Technology (DLT) is dubbed as the solution to the problem in previous slides.

- A blockchain is:
    - A database (ledger of activities)
    - Distributed across multiple servers (transparent)
    - Immutable & Consensus based (tamper-proof to a certain extend)

- Bitcoin is NOT Blockchain!
    - Bitcoin is based on a flavor of blockchain
    - Nakamoto (2009) Bitcoin paper
        - Solved double-spending problem (more in Cryptocurrency lecture)
        - Popularized distributed systems and the term blockchain
    - I use Bitcoin for illustration since it is the most popular implementation

A Database

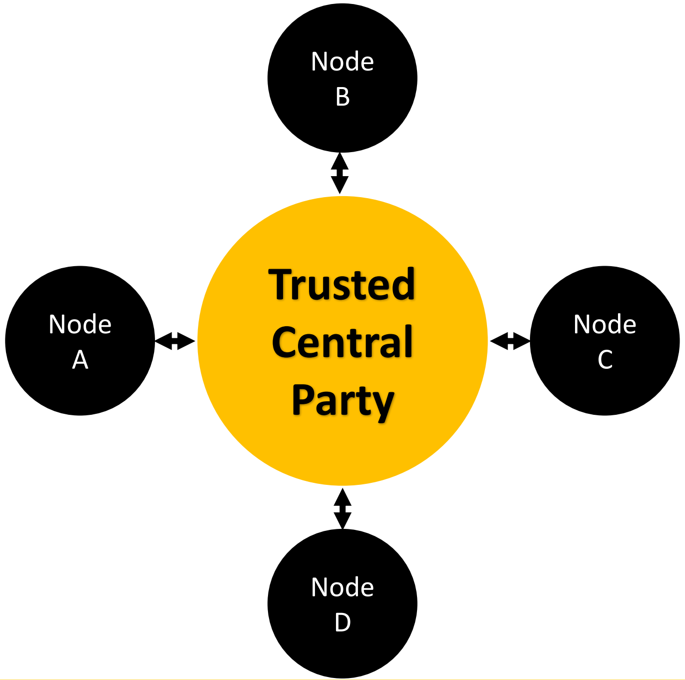
- “Fundamentally, [Blockchain] an improvement over the way that, traditionally, databases have been designed and used in the past.” (Morgan Stanley (2016), Global Insight)

- A traditional database is a large collection of data organized for rapid search and retrieval.
    - Most databases are relational and store data in tables that users can search and update.
    - Relational databases are centralized and controlled by a trusted central authority.
    - Users must trust the central authority to keep the records accurate and maintain the technological infrastructure necessary to prevent data loss from equipment failure or cyberattacks.
    - The central authority represents a single point of failure

A Distributed Ledger

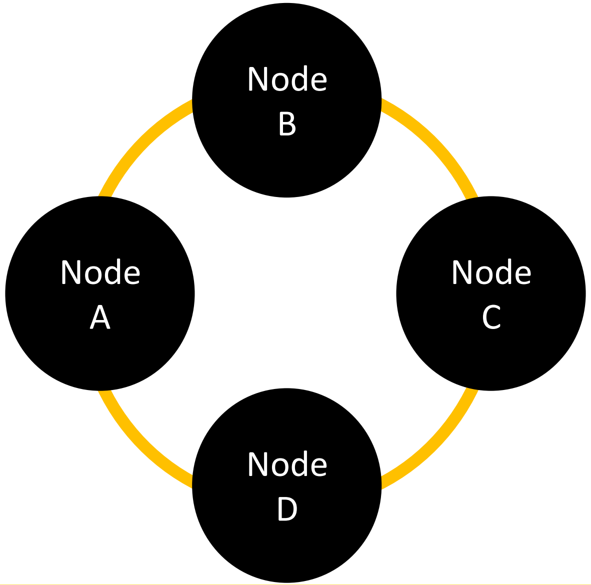
Centralized Ledger

- The trusted party holds the “true” record.
- Nodes have to reconcile their copies against the trusted party.
- Discrepancies need to be mediated
- Single point of contact is highly susceptible to breaches



Distributed/Decentralized Ledger

- Copies of the same ledger are held across all nodes
- Based on a consensus protocol, nodes determine the “true” record
- Takes a lot of computing power and coordination to breach
- Trust is NOT place on a single entity rather on the entire system



Blockchain

- Blockchain = DLT + additional conditions
- A DLT can be called as a blockchain when:
  - A specific data structure (chain-like) is enforced.
    - This ensures that every record is tied to its predecessor.
  - Every record is encrypted
  - Every change must be verified and confirmed (consensus requirement)
- Every Blockchain is a DLT. The opposite is not true.

Immutable record

- A blockchain is essentially a record of messages/transactions between buyers and sellers
- Immutability requires that once a transaction has been processed, that record cannot be modified/reversed.
- But the effects of a record can be reversed by another transaction.
- Example 1:
    - Suppose, I owe you 10BTC.
    - Instead of sending 10BTC, I send 20BTC by mistake in transaction T1
    - I cannot modify T1
    - You have to return 10BTC in a new transaction T2
    - *Remember tuples from python?*

(Distributed) Consensus driven

- Without trusted central parties, foul play could be widespread.
- In order to avoid foul play, blockchain *assumes that everyone is a crook and relies on self-interest* (i.e., Adam Smith's "invisible hand") to make them follow the rules.
- "It is not from the benevolence of the butcher, the brewer, or the baker, that we expect our dinner, but from their regard to their own interest. We address ourselves, not to their humanity but to their self-love, and never talk to them of our own necessities but of their advantages."

  -- Adam Smith (The Wealth of Nations (1776)), Father of Economics

- Technical details will have to wait for now.

History Lesson

- The idea of digital cash was first introduced in early ’80s by David Chaum
- Institutions have attempted to introduce and commercialize cryptocurrencies in the past. Ex., e-cash and E-gold.
- e-cash protocols of the 1980s and 1990s were mostly reliant on a cryptographic primitive known as Chaumian blinding.
    - Their underlying protocols largely failed to gain traction because of their reliance on a centralized intermediary.
- In 1998, Wei Dai’s b-money became the first proposal to introduce the idea of creating money through solving computational puzzles as well as decentralized consensus
    - But the proposal was scant on details as to how decentralized consensus could actually be implemented.

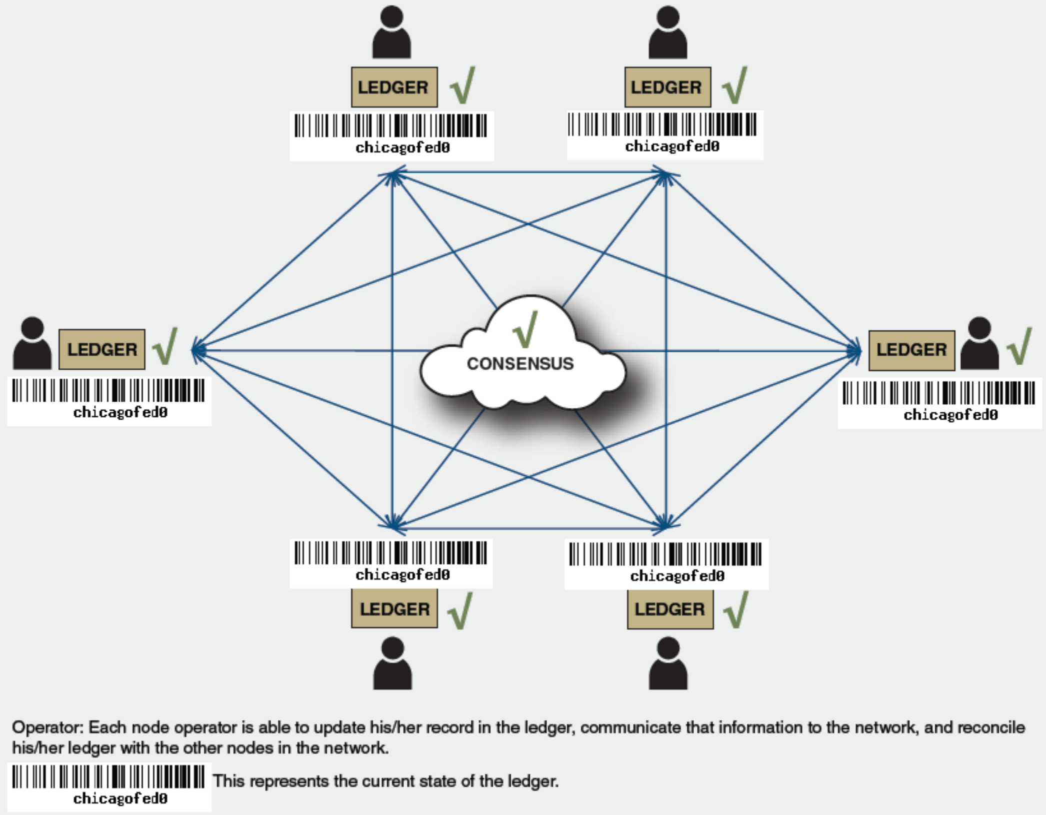
History Lesson

- In 2005, Hal Finney introduced a concept of “reusable proofs of work,” a system that uses ideas from b-money together with Adam Back’s computationally difficult Hashcash (http://hashcash.org) puzzles to create a concept for a cryptocurrency
  - Once again fell short of the ideal by relying on trusted computing as a backend.
- All efforts, until Bitcoin, failed due to different reasons, like lack of legal compliance, bad business management or network centralization.
- Blockchain technology can be conceived as the fifth paradigm of computing after (i) mainframe, (ii) personal computer, (iii) Internet and (iv) mobile and social network revolution.

DLT/Blockchain: Components

- DLT is a network of users, each of which stores its own copy of the data.
- Components of a DLT:
  - Distributed Ledger/Blockchain data structure
  - Consensus mechanism: to confirm transactions and update the ledger
  - Peer-to-peer network of nodes: to perform validation & approach consensus (miners)
  - Participants: who perform transactions (need not be part of P2P nodes)

Distributed ledger (DL) network–All records are updated



Operator: Each node operator is able to update his/her record in the ledger, communicate that information to the network, and reconcile his/her ledger with the other nodes in the network.

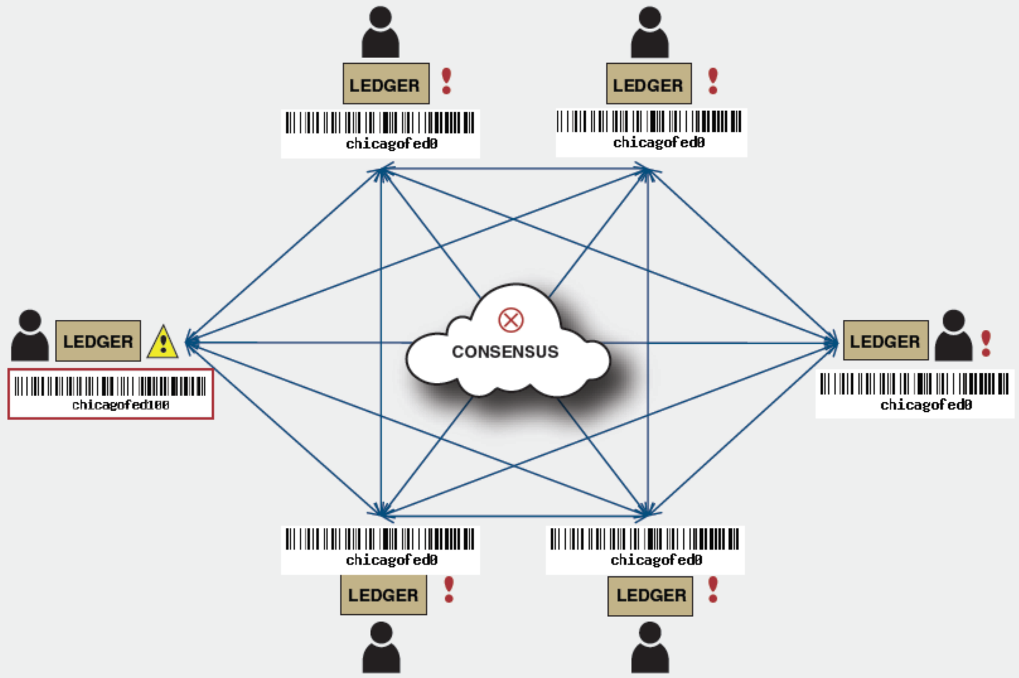
chicagofed0 This represents the current state of the ledger.

Source: Financial Markets Group, Federal Reserve Bank of Chicago.

DLT: Mechanism Overview

- The submission of the new transaction changes the state of the ledger (chicagofed100)
- Which is in conflict with the state of other copies of the ledger.
- The consensus breaks, forcing other operators to either validate and update their records with the latest change or reject the new addition to the ledger.

Distributed ledger (DL) network–New record added and state changes



⊗ When a node operator updates his/her records and digitally signs the ledger, it will invoke a reconciliation/consensus fail alert.

⚠ Represents a change in the state of the ledger. In this example, the state of the ledger changes from chicagofed0 to chicagofed100.

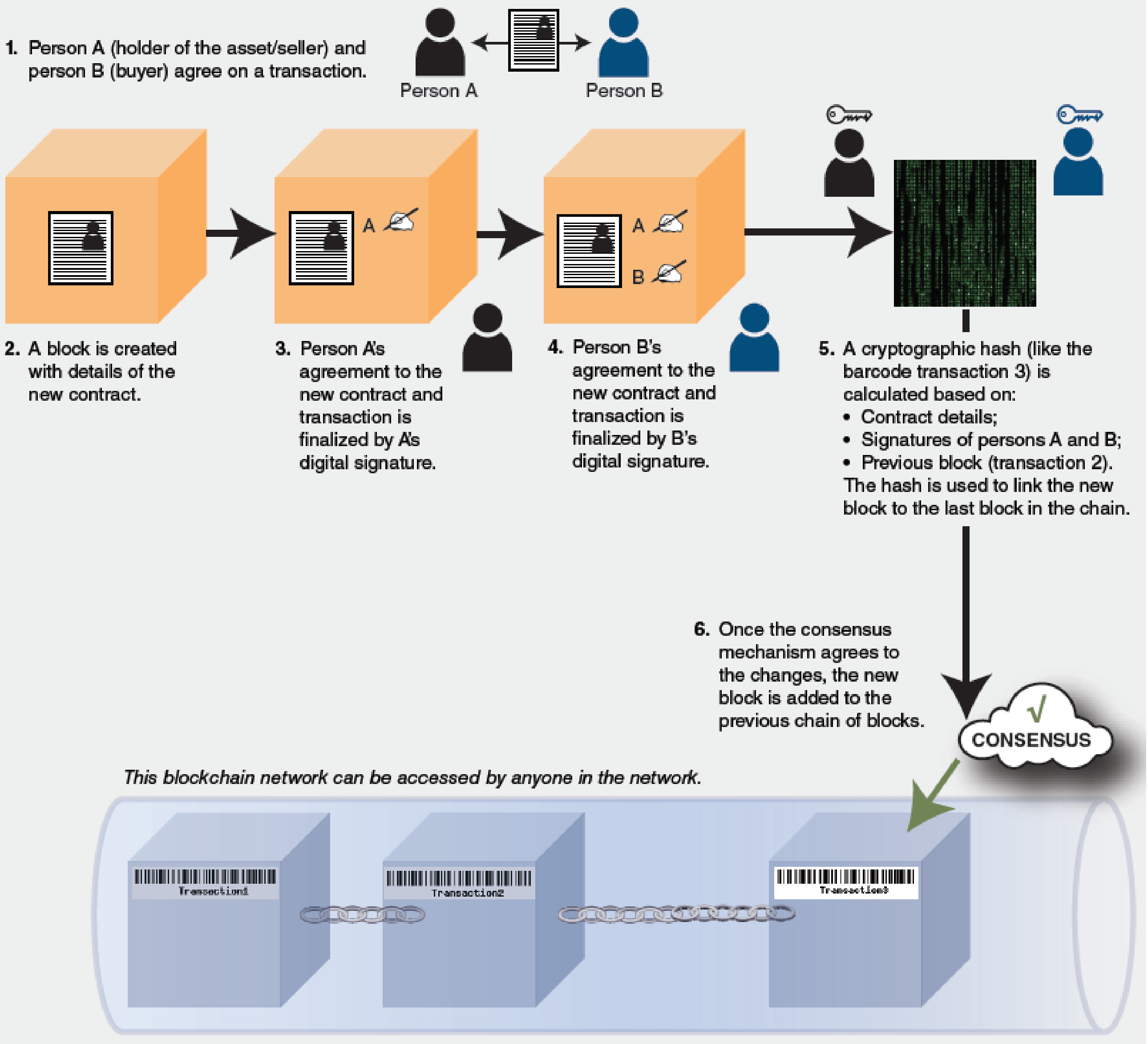
! When the states of the ledgers do not match, there is an alert that notifies node operators about the change.

Source: Financial Markets Group, Federal Reserve Bank of Chicago.

DLT: Mechanism Overview

- A & B agree to transact
    - Title of the asset is verified by looking at past records
- Block with new contract is created; A & B digitally sign
- Data in the block passes through a cryptographic hash and link to previous block is added
- If consensus is reached, the block is added to the chain
- More the number of consensuses after the block, higher the probability of successful transaction.

Blockchain (DL) network–Stylized example of a transaction



1. Person A (holder of the asset/seller) and person B (buyer) agree on a transaction.

2. A block is created with details of the new contract.

3. Person A's agreement to the new contract and transaction is finalized by A's digital signature.

4. Person B's agreement to the new contract and transaction is finalized by B's digital signature.

5. A cryptographic hash (like the barcode transaction 3) is calculated based on:
    - Contract details;
    - Signatures of persons A and B;
    - Previous block (transaction 2).

   The hash is used to link the new block to the last block in the chain.

6. Once the consensus mechanism agrees to the changes, the new block is added to the previous chain of blocks.

This blockchain network can be accessed by anyone in the network.

Source: Financial Markets Group, Federal Reserve Bank of Chicago.

Bitcoin Mechanism

Overview

Transaction Flow: Bitcoin State Transition

- The ledger of any cryptocurrency can be conceptualized as a state transition system
  - State S consists of the ownership status of all existing coins
  - EXECTX is an API that takes in state S and a transaction TX and outputs a new state S’
    - *Definition: EXECTX(S,TX) results in S’ (new state) or ERROR and S (no change to state)*

  - Assume Alice is sending coins to Bob:
    - *EXECTX({ A:$1000, B:$500},"send $500 :A to B") results in { A:$500, B:$1000 } (Alice has req. coins)*
    - *EXECTX({ A:$1000, B:$500 },"send $1001 from A to B") results in ERROR (Alice doesn’t have req. coins)*

Bitcoin State Transition

- The state in a blockchain is the “consensus view” of all transactions at any given moment, which is the result of the existing authenticated ledger distributed among all nodes
  - For Bitcoin, it is the collection of all unspent transaction outputs (UTXOs) that have been minted and not yet spent.
  - Each UTXO has a denomination and an owner defined by a 20-byte address, which is essentially a cryptographic public key
  - The outputs of all transactions are categorized as either unspent transaction outputs or spent transaction outputs
  - For a payment to be valid, it must only use UTXOs as inputs

Bitcoin State Transition

- A transaction contains:
  - One or more inputs, with each input containing a reference to an existing UTXO and a cryptographic signature produced by the private key associated with the owner’s address,
  - And one or more outputs, with each output containing a new UTXO for addition to the state.

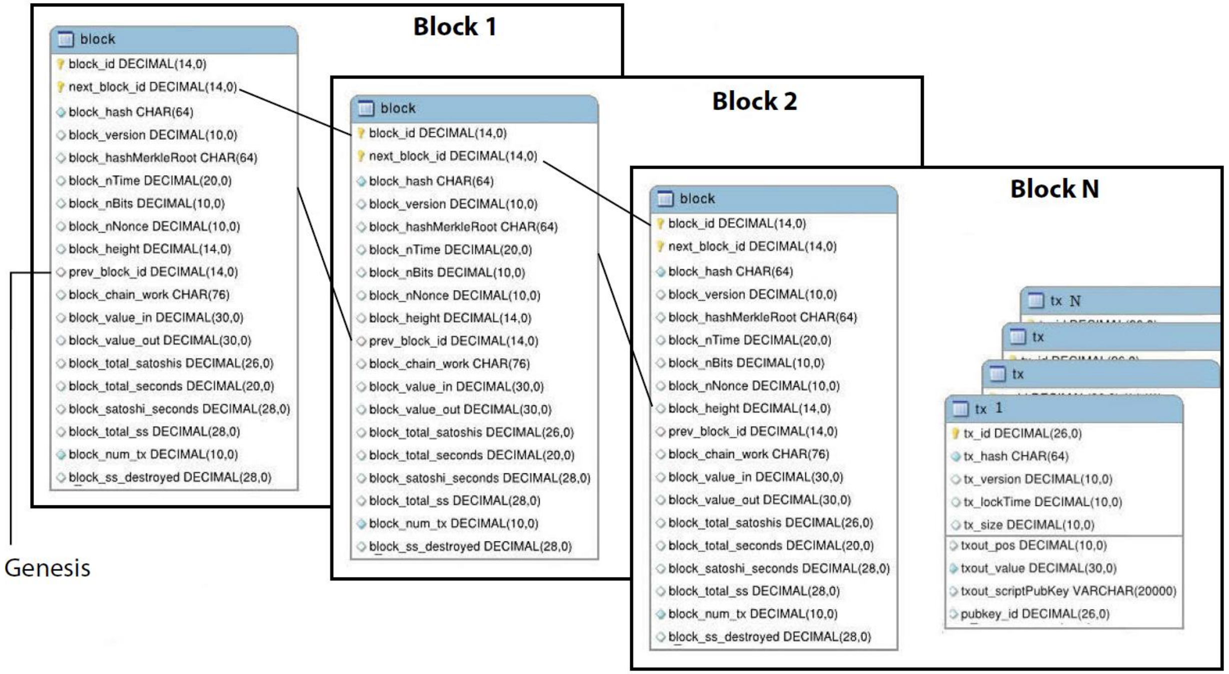
- State transition function EXECTX(S,TX) -> S’ is defined as: For each input in TX:
  - If the referenced UTXO is not in S, return an error (*this prevents transaction senders from spending coins that do not exist*).
  - If the provided signature does not match the owner of the UTXO, return an error (*this prevents transaction senders from spending other people’s coins*).
  - If the sum of the denominations of all input UTXO is less than the sum of the denominations of all output UTXO, return an error.
  - Return S' with all input UTXO removed and all output UTXO added.

Bitcoin Mining

- Bitcoin combines the state transition system with a consensus system in order to ensure that everyone agrees on the order of transactions.
- Bitcoin's decentralized consensus process requires nodes in the network to continuously attempt to produce blocks (each containing 1 to N transactions).
- The Bitcoin network is intended to create one block approximately *every 10 minutes*, with each block containing a timestamp, a nonce, a reference to (i.e., hash of) the previous block, and a list of all transactions that have taken place since the previous block.
- Each newly created block is "chained" to the last added block of the blockchain and stores its digital fingerprint (hash).

Bitcoin Blockchain Schema

- *Block identifier*: This is an identifier for the Blockchain network
- *Next block identifier*
- *Block size*: Indicates how large the block is. Since the very beginning, each block has been fixed to 1 MB. This will be increased to 2 MB in 2016 (BIP 109). Segwit further increased it to 2-4MB.



Block 1

block
block_id DECIMAL(14,0)
next_block_id DECIMAL(14,0)
block_hash CHAR(64)
block_version DECIMAL(10,0)
block_hashMerkleRoot CHAR(64)
block_nTime DECIMAL(20,0)
block_nBits DECIMAL(10,0)
block_nNonce DECIMAL(10,0)
block_height DECIMAL(14,0)
prev_block_id DECIMAL(14,0)
block_chain_work CHAR(76)
block_value_in DECIMAL(30,0)
block_value_out DECIMAL(30,0)
block_total_satoshis DECIMAL(26,0)
block_total_seconds DECIMAL(20,0)
block_satoshi_seconds DECIMAL(28,0)
block_total_ss DECIMAL(28,0)
block_num_tx DECIMAL(10,0)
block_ss_destroyed DECIMAL(28,0)

Genesis

Block 2

block
block_id DECIMAL(14,0)
next_block_id DECIMAL(14,0)
block_hash CHAR(64)
block_version DECIMAL(10,0)
block_hashMerkleRoot CHAR(64)
block_nTime DECIMAL(20,0)
block_nBits DECIMAL(10,0)
block_nNonce DECIMAL(10,0)
block_height DECIMAL(14,0)
prev_block_id DECIMAL(14,0)
block_chain_work CHAR(76)
block_value_in DECIMAL(30,0)
block_value_out DECIMAL(30,0)
block_total_satoshis DECIMAL(26,0)
block_total_seconds DECIMAL(20,0)
block_satoshi_seconds DECIMAL(28,0)
block_total_ss DECIMAL(28,0)
block_num_tx DECIMAL(10,0)
block_ss_destroyed DECIMAL(28,0)

Block N

block
block_id DECIMAL(14,0)
next_block_id DECIMAL(14,0)
block_hash CHAR(64)
block_version DECIMAL(10,0)
block_hashMerkleRoot CHAR(64)
block_nTime DECIMAL(20,0)
block_nBits DECIMAL(10,0)
block_nNonce DECIMAL(10,0)
block_height DECIMAL(14,0)
prev_block_id DECIMAL(14,0)
block_chain_work CHAR(76)
block_value_in DECIMAL(30,0)
block_value_out DECIMAL(30,0)
block_total_satoshis DECIMAL(26,0)
block_total_seconds DECIMAL(20,0)
block_satoshi_seconds DECIMAL(28,0)
block_total_ss DECIMAL(28,0)
block_num_tx DECIMAL(10,0)
block_ss_destroyed DECIMAL(28,0)

tx N

tx

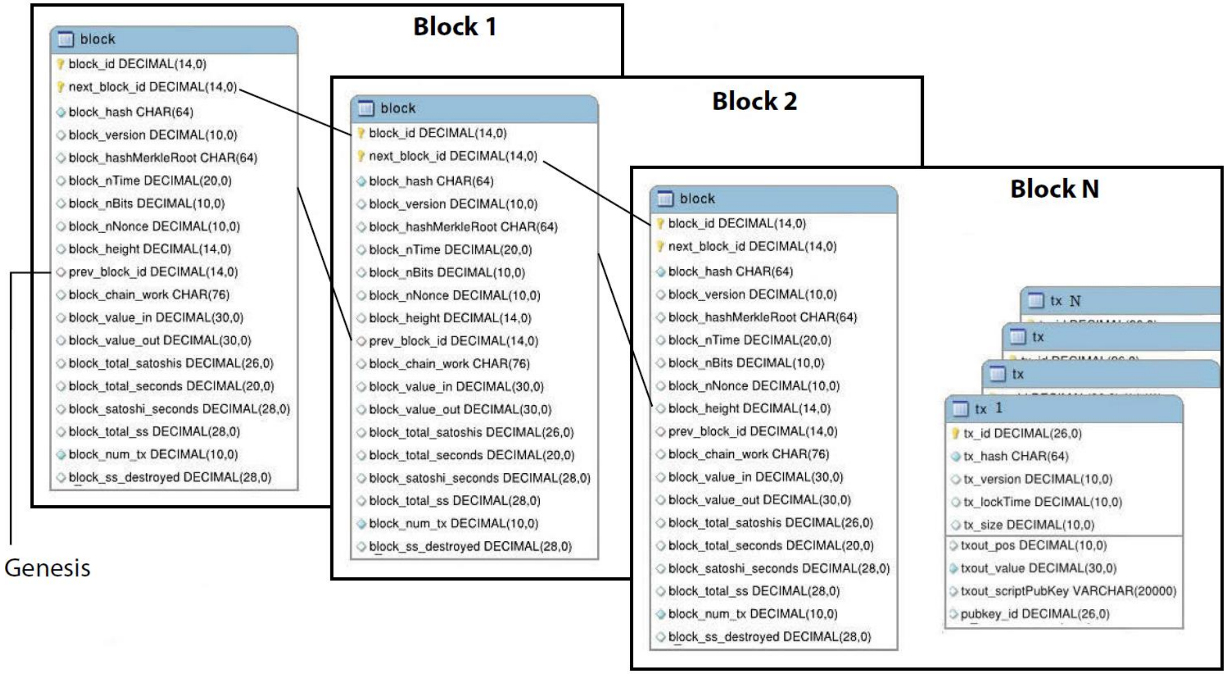
tx

tx 1
tx_id DECIMAL(26,0)
tx_hash CHAR(64)
tx_version DECIMAL(10,0)
tx_lockTime DECIMAL(10,0)
tx_size DECIMAL(10,0)
txout_pos DECIMAL(10,0)
txout_value DECIMAL(30,0)
txout_scriptPubKey VARCHAR(20000)
pubkey_id DECIMAL(26,0)

Bitcoin Blockchain Schema

- *Block version*: Each node running the Bitcoin protocol has to implement the same version and it is mentioned in this field.
- *Previous block hash*: This is a digital fingerprint (hash) of the block header of the previous (last added) block of the blockchain. It is calculated by taking all the fields of the header (version, nonce, etc.) together and applying a cryptographic function (SHA-256) twice by rearranging the bytes of the individual fields.
- *Block Merkle root*
- *Block timestamp*
- *Nonce*

The above fields together form a block's header



Block 1

block
block_id DECIMAL(14,0)
next_block_id DECIMAL(14,0)
block_hash CHAR(64)
block_version DECIMAL(10,0)
block_hashMerkleRoot CHAR(64)
block_nTime DECIMAL(20,0)
block_nBits DECIMAL(10,0)
block_nNonce DECIMAL(10,0)
block_height DECIMAL(14,0)
prev_block_id DECIMAL(14,0)
block_chain_work CHAR(76)
block_value_in DECIMAL(30,0)
block_value_out DECIMAL(30,0)
block_total_satoshis DECIMAL(26,0)
block_total_seconds DECIMAL(20,0)
block_satoshi_seconds DECIMAL(28,0)
block_total_ss DECIMAL(28,0)
block_num_tx DECIMAL(10,0)
block_ss_destroyed DECIMAL(28,0)

Genesis

Block 2

block
block_id DECIMAL(14,0)
next_block_id DECIMAL(14,0)
block_hash CHAR(64)
block_version DECIMAL(10,0)
block_hashMerkleRoot CHAR(64)
block_nTime DECIMAL(20,0)
block_nBits DECIMAL(10,0)
block_nNonce DECIMAL(10,0)
block_height DECIMAL(14,0)
prev_block_id DECIMAL(14,0)
block_chain_work CHAR(76)
block_value_in DECIMAL(30,0)
block_value_out DECIMAL(30,0)
block_total_satoshis DECIMAL(26,0)
block_total_seconds DECIMAL(20,0)
block_satoshi_seconds DECIMAL(28,0)
block_total_ss DECIMAL(28,0)
block_num_tx DECIMAL(10,0)
block_ss_destroyed DECIMAL(28,0)

Block N

block
block_id DECIMAL(14,0)
next_block_id DECIMAL(14,0)
block_hash CHAR(64)
block_version DECIMAL(10,0)
block_hashMerkleRoot CHAR(64)
block_nTime DECIMAL(20,0)
block_nBits DECIMAL(10,0)
block_nNonce DECIMAL(10,0)
block_height DECIMAL(14,0)
prev_block_id DECIMAL(14,0)
block_chain_work CHAR(76)
block_value_in DECIMAL(30,0)
block_value_out DECIMAL(30,0)
block_total_satoshis DECIMAL(26,0)
block_total_seconds DECIMAL(20,0)
block_satoshi_seconds DECIMAL(28,0)
block_total_ss DECIMAL(28,0)
block_num_tx DECIMAL(10,0)
block_ss_destroyed DECIMAL(28,0)

tx N

tx

tx

tx 1
tx_id DECIMAL(26,0)
tx_hash CHAR(64)
tx_version DECIMAL(10,0)
tx_lockTime DECIMAL(10,0)
tx_size DECIMAL(10,0)
txout_pos DECIMAL(10,0)
txout_value DECIMAL(30,0)
txout_scriptPubKey VARCHAR(20000)
pubkey_id DECIMAL(26,0)

Bitcoin Block size and Segregated Witness

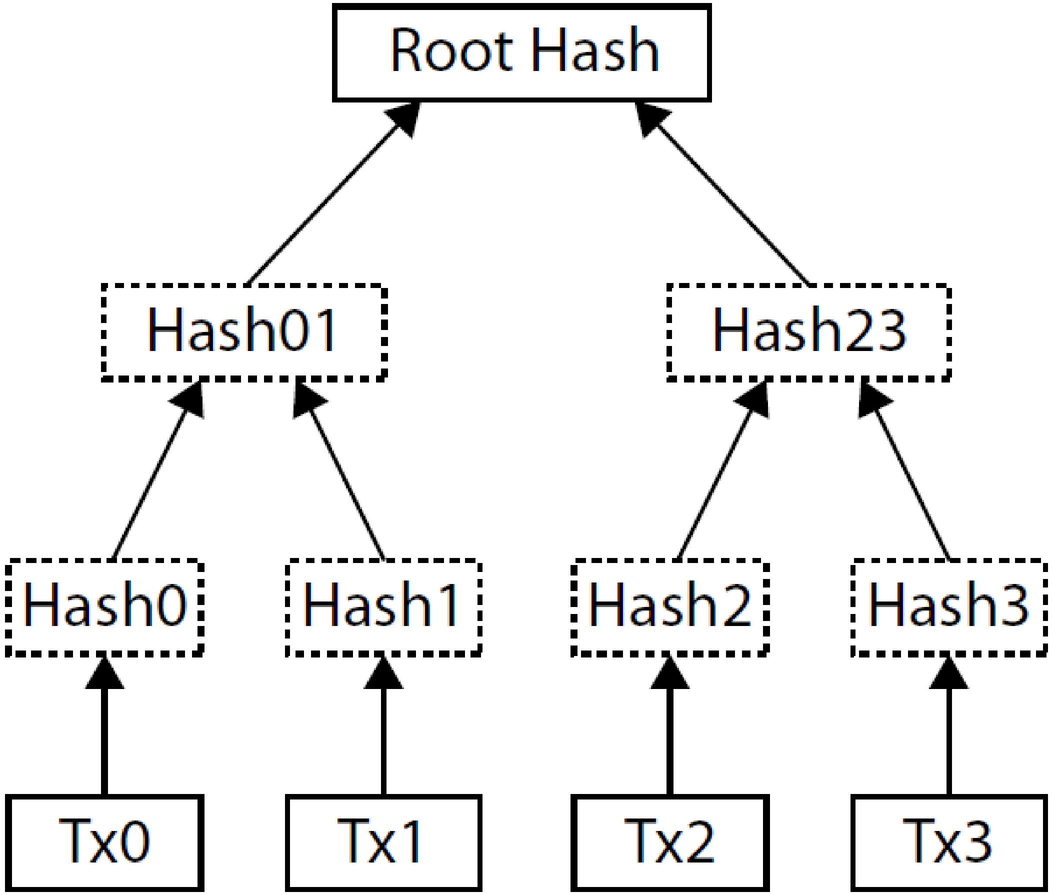
- Ever since Bitcoin gained popularity, the number of transactions awaiting processing has increased
    - Each transaction contains: (i) Input (sender) (ii) Output (receiver), (iii) Digital Signature (witness)
    - Given the size of a block is restricted (1,2,2-4MB), the number of transactions that can be placed in a block is limited
    - This means that miners have to solve more hash puzzles and expend a lot of resources
    - Also means more transaction costs for users
- Raising the size limit has been a point contention since 2015 among developers

Bitcoin Block size and Segregated Witness

- Segregated Witness (SegWit) was the proposed solution
  - Digital signatures take the maximum space for each transaction.
  - SegWit removes the signatures from the transaction and moves it to a new block – *Extended Block*
  - This allows for more space for transaction inputs and outputs
  - This called for removing/segregating the witness.
- SegWit will improve the Bitcoin network scaling ability.
- Consensus is not required to make SegWit work. SegWit works even where users do not upgrade their software versions to the newest version.
- SegWit affords the following advantages:
  - It will reduce the file size of transactions,
  - There will be faster confirmation of transactions
  - Transaction fees will be lower

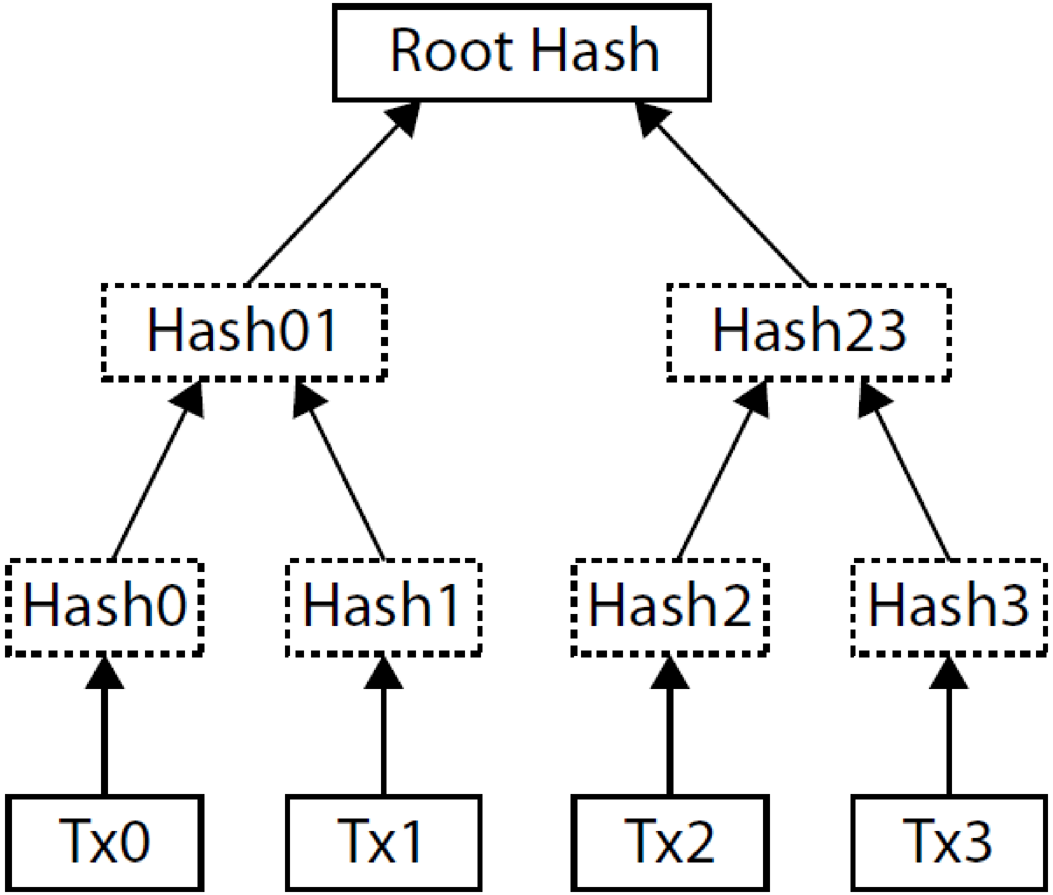
Bitcoin and Merkle Root

- The transactions are listed as Merkle tree or a binary hash tree
- The root of the tree is the topmost node. The nodes at the bottom are called leaf nodes.
- Each node is simply a cryptographic hash of a transaction.
- The Merkle tree does not contain a list of all the transactions, but rather a hash (digital fingerprint) of all transactions as a tree structure
- Hash of Transaction 0 = Hash[Tx(0)] = SHA256 (SHA 256 (Transaction A))
- Parent node Hash(01): the 32-byte Hash[Tx(0)] and 32-byte Hash[Tx(1)] is concatenated as a 64-byte hash string and then SHA-256 is applied twice to give a 32-byte Hash(01)



Bitcoin and Merkle Root

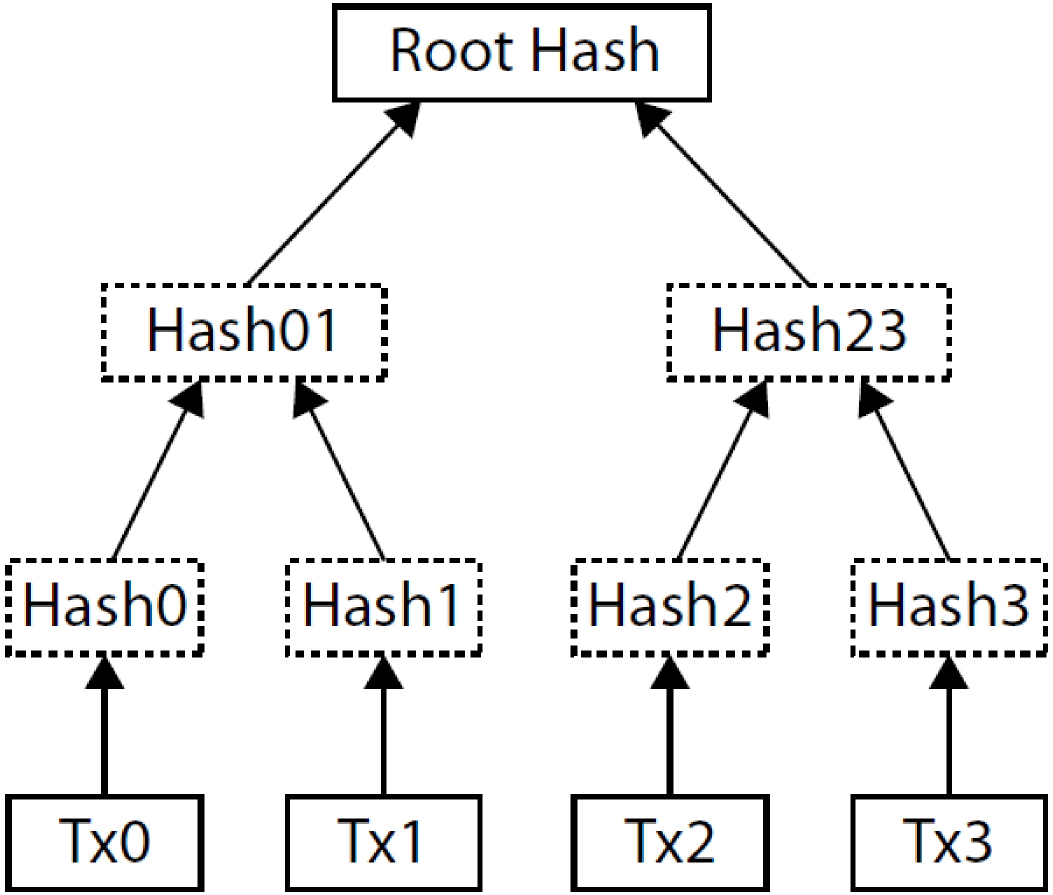
- An important scalability feature of Bitcoin is that the block is stored in a multilevel data structure.
- The "hash" of a block is the hash of the block header that contains the timestamp, nonce, previous block hash, and the root hash of the Merkle tree storing all transactions in the block.
- *The purpose of the Merkle tree is to allow the data in a block to be delivered piecemeal:* a node can download the header of a block from one source, the small part of the tree relevant to them from another source, and still be assured that all of the data is correct.



Bitcoin and Merkle Root

- *Merkle Trees are essential for security and long-term sustainability of blockchains*

- If a malicious user attempts to swap in a fake transaction to the bottom of a Merkle tree:
  - This change will cause a change in the node above, and then a change in the node above that, finally changing the root of the tree
  - Therefore will change the hash of the block, causing the protocol to register it as a completely different block and resulting almost certainly with an invalid proof-of-work.



Bitcoin and Secure Hashing (SHA 256)

- SHA stands for secure hash algorithm.
- It is used to prove data integrity.
- The same inputs will always produce the exact same output.
- The output is always 256 bits or 32 bytes in length regardless of the length of the input (even if input is millions of bytes).
- Any change in the inputs will result in a change of output.
- The same output can never be derived from different inputs.
- However, from the output we can never determine the inputs, which is why this is highly secure.

- Example:
    - This is a test for FinTech class at UCF
        - Output: db1793a70838f7aec96b52fec1578d39692357430cc989998c5d9deddce8b044
    - This is a test for finTech class at UCF
        - Output: 09005cad4898515eeee1510c04b827cbbbf3a042eb108d49509754a9ffe7c852

Bitcoin and Secure Hashing (SHA 256)

- The algorithm for checking if a block is valid is as follows:
  - Check if the previous block referenced by the block exists and is valid.
  - Check that the timestamp of the block is greater than that of the previous block.
  - Check that the proof-of-work on the block is valid.

- Let S[0] be the state at the end of the previous block and suppose TX is the block's transaction list with n transactions.
  - For all i in 0...n-1, set S[i+1] = EXECTX({ (S[i],TX[i]) }.
  - If any application returns an error, exit and return false.
  - Return true, and register S[n] as the state at the end of this block.
- Essentially, each transaction in the block must provide a valid state transition from what was the canonical state before the transaction was executed to some new state.
- Order of placing transactions is important!

Bitcoin and Secure Hashing (SHA 256)

- Proof-of-Work is essentially computing the double SHA-256 hash of every block (a 256-bit number) that is less than a dynamically adjusted target.
  - *The target is adjusted every 2016 blocks to make sure on average a block is added every 10 mins*

- The purpose of this is to make block creation computationally hard
  - Helps in preventing Sybil attackers from remaking the entire blockchain in their favor.

- Because SHA-256 is designed to be a completely unpredictable pseudorandom function, the only way to create a valid block is simply trial and error, repeatedly incrementing the nonce and seeing if the new hash matches.

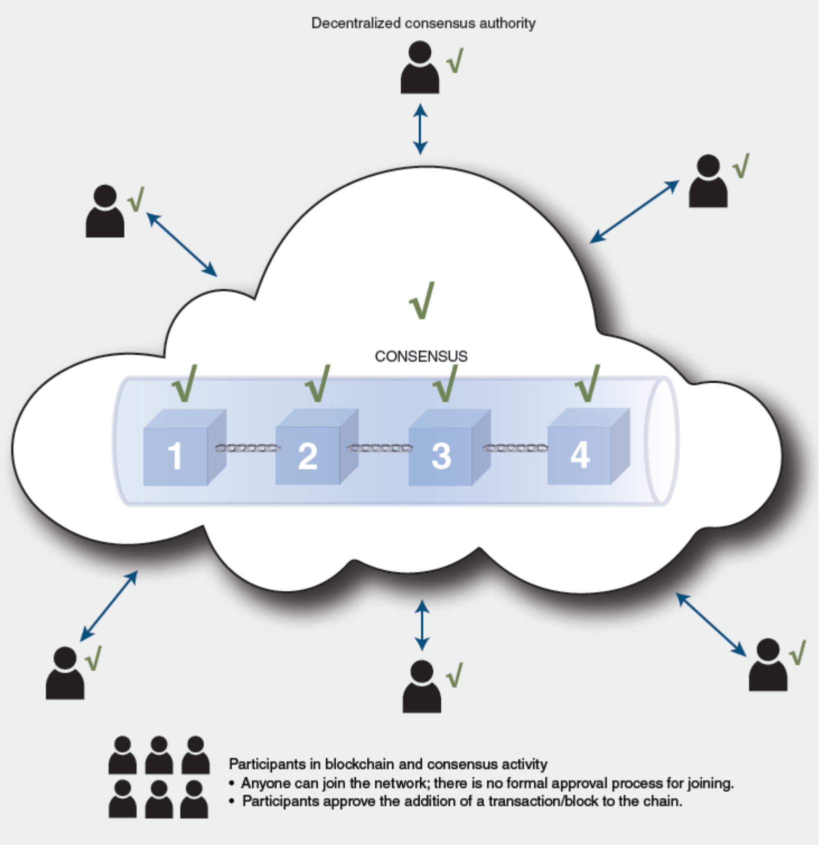
- Video: https://www.youtube.com/watch?v=mMxkxwPSfvo
- Hash target/Difficulty
  - Definition: https://en.bitcoin.it/wiki/Difficulty
  - History: https://data.bitcoinity.org/bitcoin/difficulty/5y?t=l

Types of Blockchain

Types of Blockchain: Permission-less

- Public/Permissionless: Anyone can read or submit transactions (submissions will be committed if valid), and anyone can participate in the consensus process.
- These trustless platforms are secured by mechanisms such as proof of work or proof of stake, a.k.a. cryptoeconomics.
- In other words, influence in the consensus process is proportional to the quantity of economic resources that entity can bring to bear.

Permissionless/public blockchain network



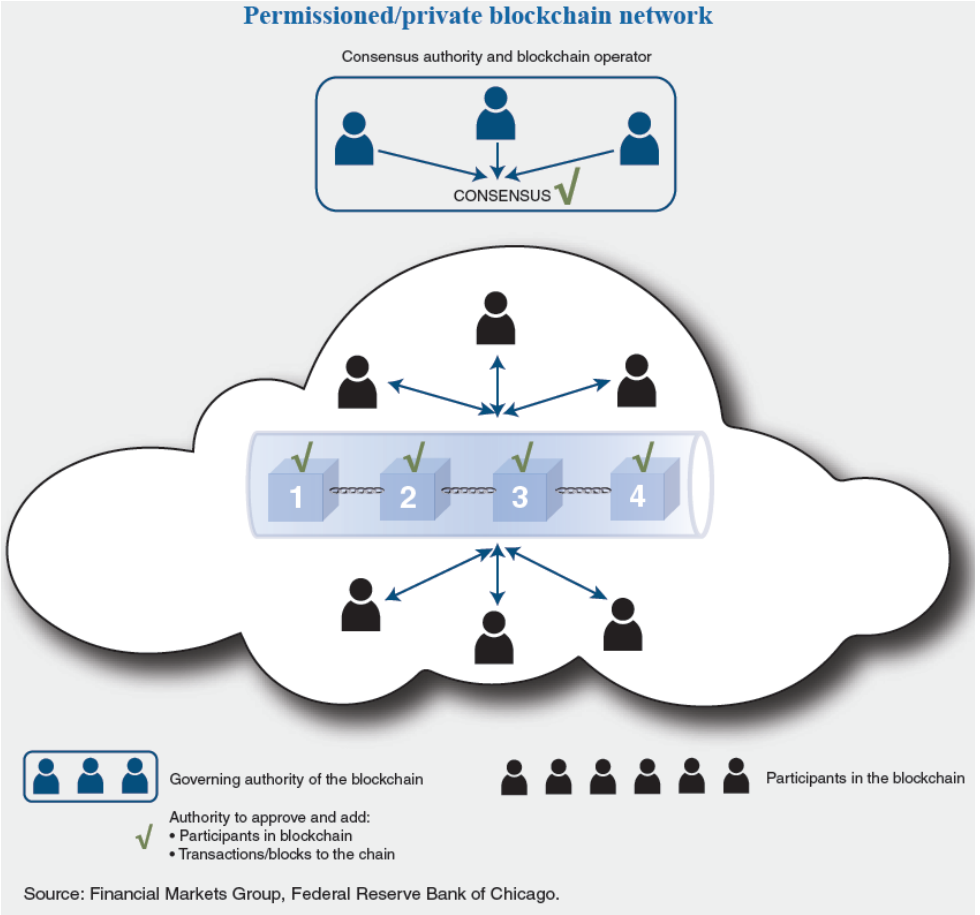
Participants in blockchain and consensus activity

- Anyone can join the network; there is no formal approval process for joining.
- Participants approve the addition of a transaction/block to the chain.

Source: Financial Markets Group, Federal Reserve Bank of Chicago.

Types of Blockchain: Permissioned

- Private/Permissioned: Write permissions are kept centralized to a single organization or part of it.
- Read permissions may be public or restricted to a set of known participants.
- Private blockchains could provide solutions to financial enterprise problems, including compliance agents for regulations such as the Health Insurance Portability and Accountability Act (HIPAA), anti–money laundering (AML), and know-your-customer (KYC) laws.
- The Hyperledger project from the Linux Foundation and the Gem Health network are private blockchain projects under development.



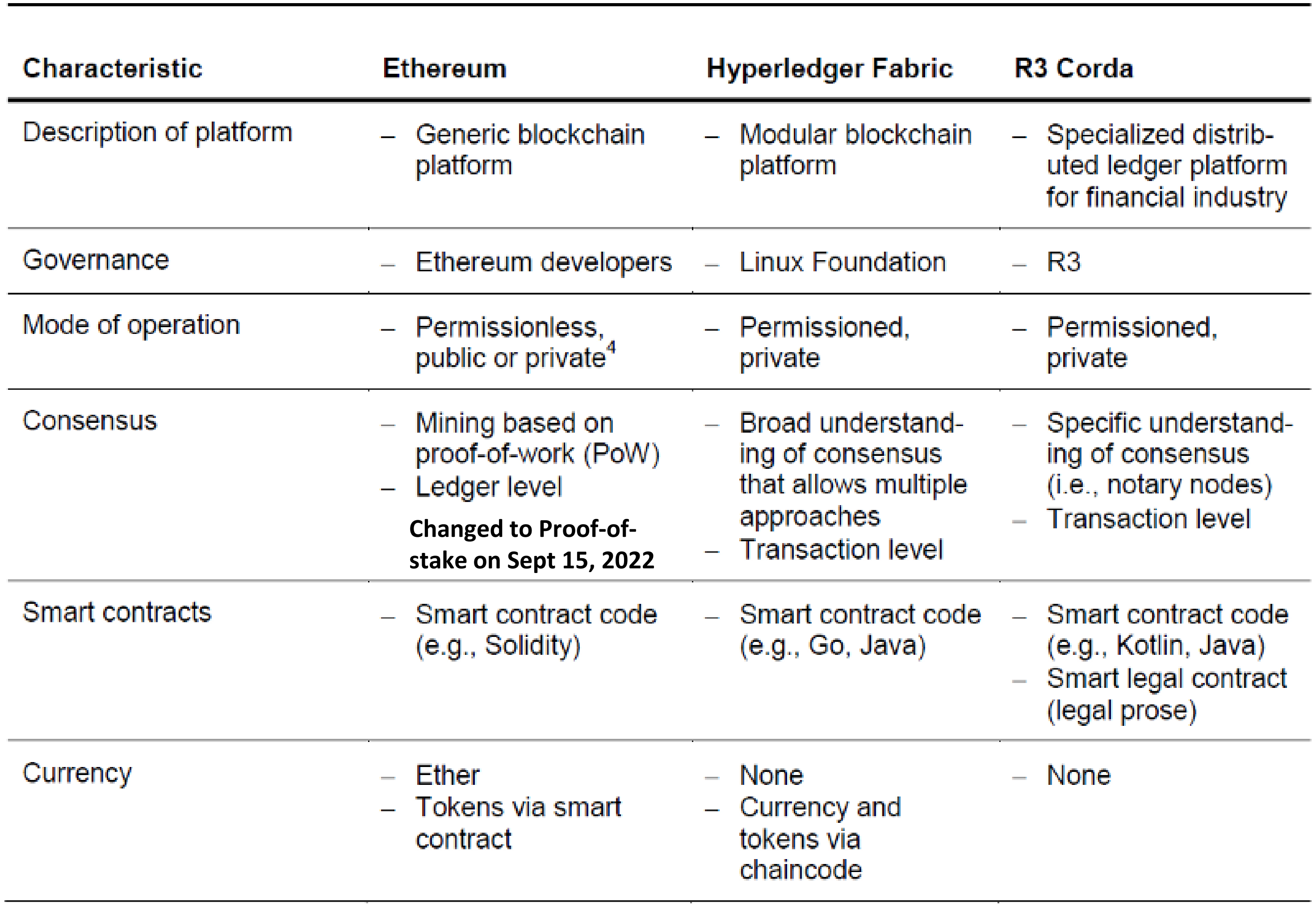
Permissioned/private blockchain network
Consensus authority and blockchain operator
CONSENSUS √
1
2
3
4
Governing authority of the blockchain
Participants in the blockchain
√
Authority to approve and add:
• Participants in blockchain
• Transactions/blocks to the chain

Source: Financial Markets Group, Federal Reserve Bank of Chicago.

Types of Blockchain: Consortium

- Consortium/Mixed: Consensus is controlled by a preselected set of nodes and rules for achieving consensus.
- The right to read the blockchain can be open to the public, or it can also be restricted to a set of known participants
  - For example, 10 banks in a consortium that agree to the consensus rule that 7 of 10 banks must sign (approve) a block for it to be considered a valid representation of the truth.
- Or, hybrid routes such as the root hashes of the blocks being public together with an API that allows members of the public to make a limited number of queries and get back cryptographic proofs of some parts of the blockchain state.
  - These sort of blockchains are distributed ledgers that may be considered "partially decentralized."

Types of Blockchain: Comparison



Characteristic	Ethereum	Hyperledger Fabric	R3 Corda
Description of platform	– Generic blockchain platform	– Modular blockchain platform	– Specialized distributed ledger platform for financial industry
Governance	– Ethereum developers	– Linux Foundation	– R3
Mode of operation	– Permissionless, public or private[4]	– Permissioned, private	– Permissioned, private
Consensus	– Mining based on proof-of-work (PoW) – Ledger level **Changed to Proof-of-stake on Sept 15, 2022**	– Broad understanding of consensus that allows multiple approaches – Transaction level	– Specific understanding of consensus (i.e., notary nodes) – Transaction level
Smart contracts	– Smart contract code (e.g., Solidity)	– Smart contract code (e.g., Go, Java)	– Smart contract code (e.g., Kotlin, Java) – Smart legal contract (legal prose)
Currency	– Ether – Tokens via smart contract	– None – Currency and tokens via chaincode	– None

Source: Comparison of Ethereum, Hyperledger Fabric and Corda, Medium (2017)

Blockchain Examples

Ripple

- Ripple (www.ripple.com) is considered as one of the most advanced DLT companies in the industry. It focuses on the using of blockchain-like technology for payments.
- Ripple has obtained a virtual currency license from the New York State Department of Financial Services, making it one of the few companies with a BitLicense.
- As of 2017, Ripple is the third-largest cryptocurrency by market capitalization, after bitcoin and ether.
- *Ripple is a financial real-time gross settlement solution, currency exchange, and remittance network using distributed ledger technology.*
- Ripple is built upon a distributed open-source Internet protocol, consensus ledger, and native currency called XRP (ripples) enabling (cross-border) payments for retail customers, corporations, and other banks.

Ripple

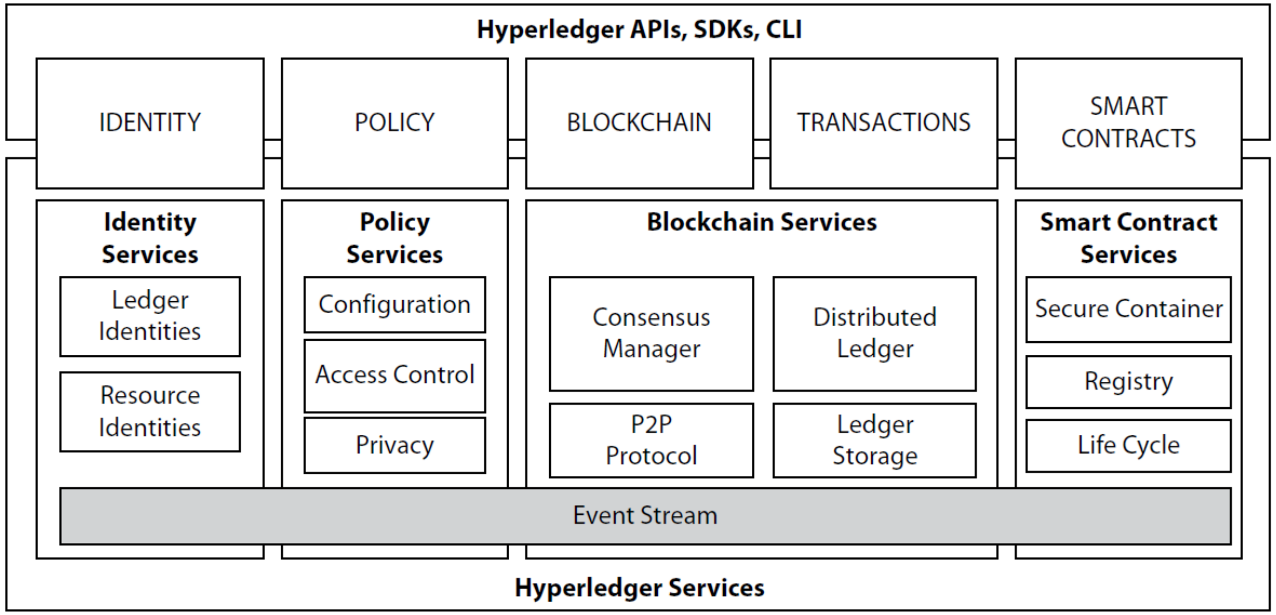
- The Ripple protocol, described as “basic (settlement) infrastructure technology for interbank transactions,” enables the interoperation of different ledgers and payment networks and brings together three aspects of modern payment solutions: *messaging, settlement, and FX management.*
- It allows banks and non-bank financial services companies to incorporate the Ripple protocol into their own systems and thereby allow their customers to use the service. The protocol enables the instant and direct transfer of money between two parties. As such the protocol can circumvent the fees and wait times of the traditional correspondent banking system.
- Many financial companies are experimenting and integrating with Ripple. A host of major banks have adopted Ripple to improve their cross-border payments, and many have completed trial blockchain projects. These banking institutions— including Santander, UniCredit, UBS, Royal Bank of Canada, Westpac Banking Corporation, CIBC, and National Bank of Abu Dhabi,

Ethereum

- Ethereum (https://ethereum.org) is a decentralized platform that runs smart contracts: applications that run exactly as programmed without any possibility of downtime, censorship, fraud, or third-party interference.
- The project was bootstrapped via an ether presale in August 2014 by fans all around the world. It is developed by the Ethereum Foundation, a Swiss nonprofit, with contributions from developers across the globe.
- *Bitcoin was blockchain 1.0, Ethereum is considered as blockchain 2.0 and beyond.*
- Ethereum applications run on a custom-built blockchain, a shared global infrastructure that can move value around and represent the ownership of property.
- This enables developers to create markets, store registries of debts or promises, and move funds in accordance with instructions given long in the past (such as a will) or a futures contract without a middleman or counterparty risk.
- In March 2017, the Ethereum Enterprise Alliance was formed. Participating organizations include Microsoft, Intel, J.P. Morgan, BNY Mellon, and a host of others.

Hyperledger

- An open-source collaborative effort created to advance cross-industry blockchain technologies.
- It is a consortium of companies working together to develop standardized blockchain protocols.
- The project aims to develop open protocols and standards by providing a modular framework that supports different components for different uses.
- This would include a variety of blockchains with their own consensus and storage models, and services for identity, access control, and contracts.
- *It is a global collaboration hosted by the Linux Foundation*
- The famous Hyperledger Fabric is the IBM contribution, an implementation for the enterprise, with capabilities including network security, scalability, confidentiality, and performance, in a modular blockchain architecture.



Source: www.hyperledger.org

R3 Corda

- Corda is a distributed ledger platform designed to record, manage, and automate legal agreements between business partners.
- *It is a collaborative effort by R3, a group of more than 100 financial companies.*
- Corda is a distributed ledger made up of mutually distrusting nodes that would allow for a single global database that records the state of deals and obligations between institutions and people.
- Transactions may execute in parallel, on different nodes, without either node being aware of the other's transactions.
- Nodes are arranged in an authenticated peer-to-peer network.
- All communication is direct.

R3 Corda

- There is no blockchain.
- Corda could accurately be described as a messaging protocol.
- Transaction races are deconflicted using pluggable notaries. A single Corda network may contain multiple notaries that provide their guarantees using a variety of different algorithms. Thus, Corda is not tied to any particular consensus algorithm.
- *Data is shared on a need-to-know basis.*
- Nodes provide the dependency graph of a transaction they are sending to another node on demand, but there is no global broadcast of all transactions.

Blockchain: Use Cases

Blockchain: Use cases - Finance

- Traditional trade processes within asset management can be slow, manual, cumbersome, and filled with risk when reconciling and matching

- Each party in the trade life cycle (e.g., broker dealers, intermediaries, custodians, clearing and settlement teams) currently keeps their own copy of the same record of a transaction, creating significant inefficiencies and room for error.
- A fair amount of trades have errors, requiring manual intervention and extending the time required to settle trades.

Blockchain: Use cases - Finance

- Blockchains do not require an exchange to verify, clear, and settle security transactions
- Blockchain can eliminate significant fees across FX, commodities, and OTC derivatives.
- Blockchain technology *could simplify and streamline this entire process, providing an automated trade life cycle* where all parties in the transaction would have access to the exact same data about a trade.
- This would lead to substantial infrastructural cost savings, effective data management and transparency, faster processing cycles, minimal reconciliation, and the potential removal of brokers and intermediaries altogether
- *Financial derivatives are the most common application of a smart contract, and one of the simplest to implement in code.*
- Companies currently working in this space are: Digital Asset Holdings, Chain.com, and Ripple.

Blockchain: Use cases – Resource Sharing

Ride-share:

- La`Zooz is a blockchain-based ride-sharing solution that rewards its users, developers, and drivers with tokens called zooz. Unlike Uber, La`Zooz has no central authority and cannot be blocked or shut down by governments. Your identity could be linked to reviews in the “sharing economy” on the marketplace. People can check out your review as a trusted individual by checking your ID number.

Electricity:

- With solar and high-capacity battery technology, individuals can potentially act as distributed power providers.
- Blockchain could be used to facilitate secure transactions of power between individuals on a distributed network who do not have an existing relationship.
- The fact that all transactions are verified by a consensus network means you are protected from customers who claim the transaction did not happen.
- The current applications are Sun Exchange, TransActive Grid, and Grid Singularity.