BLOCKCHAIN

https://www.investopedia.com/terms/b/blockchain.asp

Hyperledger

Permissioned vs permissionless: https://cointelegraph.com/blockchain-for-beginners/permissioned-blockchain-vs-permissionless-blockchain-key-differences

The key permissioned blockchains include **Quorum, Corda, and Hyperledger Fabric**. These are the top-notch solutions when it comes to permissioned solutions

https://www.geeksforgeeks.org/blockchain-hyperledger-vs-ethereum/

https://aws.amazon.com/blockchain/what-is-hyperledger-fabric/

https://www.coding-bootcamps.com/blog/intro-to-hyperledger-architecture-projects-tools-and-libraries.html

Hyperledger Architecture Pg 117 Intro to Hyperledger

https://www.devteam.space/blog/pros-and-cons-of-hyperledger-fabric-for-blockchain-networks/

We can see that **a smart contract is a domain specific program which relates to specific business processes, whereas a chaincode is a technical container of a group of related smart contracts**. Chaincode, also referred to as smart contracts, is **software that you can use to read and update data on the blockchain ledger**. Chaincode can turn business logic into an executable program that is agreed to and verified by all members of the blockchain network. Walmart, Visa, FedEx user Hyperledger fabric

https://www.investopedia.com/terms/h/hyperledger.asp#:~:text=Hyperledger%20was%20set%20up%20with,and%20transactions%20of%20the%20various

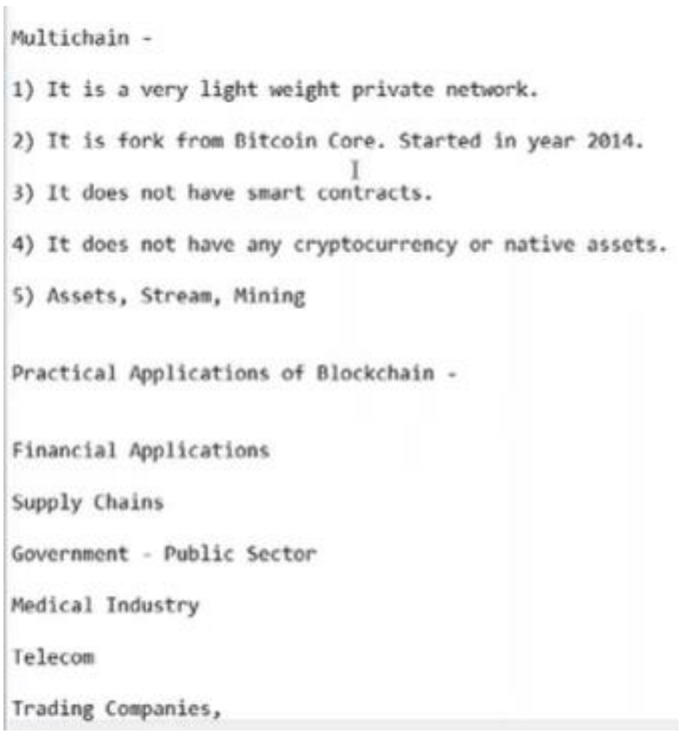
Multichain

https://www.youtube.com/watch?v=m145YBm_nI0

https://www.youtube.com/watch?v=14ivHRu-jfo

Disadvantage: Cannot implement any logic or write smart contract. Applications are very limited

Advantage: Very easy and fast to setup


Multichain -

1) It is a very light weight private network.

2) It is fork from Bitcoin Core. Started in year 2014.

3) It does not have smart contracts.

4) It does not have any cryptocurrency or native assets.

5) Assets, Stream, Mining

Practical Applications of Blockchain -

Financial Applications

Supply Chains

Government - Public Sector

Medical Industry

Telecom

Trading Companies,

Data streams: https://www.multichain.com/developers/data-streams/

Here's a basic architectural description:

- The two main subsystems of MultiChain are the node (which tracks the chain's global state) and the wallet (which tracks transactions of specific interest to the node and holds private keys). Each of these has separate mechanisms for storing and retrieving information.
- There's no distinction between Bitcoin Core and MultiChain since MultiChain is an extended version of Bitcoin Core.
- You can have multiple instances of MultiChain conneced to the same chain.
- The chain contains information permissions, assets and streams, although nodes also read and write these.
- Each node has its own API which can be connected to from an application.

https://medium.com/xord/multichain-architecture-explained-c83c3fd5edb6

Distributed consensus between identified block validators. It's close in spirit to something like PBFT (Practical Byzantine Fault Tolerance), but instead of multiple validators per block, there is one validator per block, working in a round-robin type of fashion.

Corda

https://www.coding-bootcamps.com/blog/introduction-to-corda-blockchain-development.html

BLOCKCHAIN

Components of a Blockchain

The components of an open, public blockchain are (usually): • A peer-to-peer (P2P) network connecting participants and propagating transactions and blocks of verified transactions, based on a standardized “gossip” protocol.

• Messages, in the form of transactions, representing state transitions

• A set of consensus rules, governing what constitutes a transaction and what makes for a valid state transition

• A state machine that processes transactions according to the consensus rules

• A chain of cryptographically secured blocks that acts as a journal of all the verified and accepted state transitions

• A consensus algorithm that decentralizes control over the blockchain, by forcing participants to cooperate in the enforcement of the consensus rules

• A game-theoretically sound incentivization scheme (e.g., proof-of-work costs plus block rewards) to economically secure the state machine in an open environment

• One or more open-source software implementations of the above (“clients”)

ETHEREUM

About Ethereum:
Ethereum is **a decentralized blockchain platform that establishes a peer-to-peer network that securely executes and verifies application code, called smart contracts**. Smart contracts allow participants to transact with each other without a trusted central authority.

Consensus Model

When launched in 2015, Ethereum too had an underlying proof-of-work consensus model, just like Bitcoin; however, it was a bit faster than its predecessor. Ethereum blocks were validated and committed to the Blockchain network in nearly every 12 seconds, whereas a Bitcoin approximately took 10 minutes. Also, unlike Bitcoin that has a fixed supply of 21,000,000 coins, Ethereum had no upper cap in supply. In year 2021, Ethereum started its process of moving from Proof of Work to Proof of Stake with the help of sharding. This transition would occur in many stages and ultimately would lead to a very high scalability and low transaction fees.

Transaction Fees:

The Ethereum network runs on its native cryptocurrency called Ether that is available in most of the crypto exchanges in the world. The current average transaction fee on the Ethereum network is close to 0.0008 Ethers per transaction (i.e., almost $3 USD) which is pretty high. Most traders and investors communicate in terms of Ethers, whereas when it comes to the Ethereum network, the consumption happens in terms of gas that determines how expensive a transaction is from its computation requirements.

Solidity Features:

Solidity is a statically typed programming language. In Solidity, the variables have to be declared before they can be assigned values which is very similar to what we observe in Java, C, C++, FORTRAN, Pascal, and Scala.

Solidity is an object-oriented language and supports all the related features such as encapsulation, inheritance, polymorphism, abstraction, function overloading, etc.

Once compiled, Solidity is translated to bytecodes and can run on the Ethereum Virtual Machine.

Application Binary Interface:

Every contract comes with a binary interface for integration with the front-end code as well as the other contracts that can interact with it. The binary interface is way similar to API or Application Programming Interface that the high level languages use to expose their services to the external world. ABI has the information of all the function names, input and output parameters, and Event names and their parameters.

Interfaces:

An interface in Solidity is an abstract type that specifies the behaviour of the contract. The interface declares the function signatures which are external in nature, that an inheriting contract has to implement. It can also not declare the constructor or state variables.

Events:

Events are an abstraction on top of the Ethereum Virtual Machine's logging functionalities. They are especially useful for integration with the front end code. The front end code can work as an observer with the call back functions waiting for a particular event to have occurred. In the decentralized applications, events play a major role in knowing the status of a transaction.

The following is the proper order of elements:

The pragma statement – which handles the associated compiler version.

Import statements – that imports the other files to the current.

Interfaces – which are the abstract types for a contract.

Libraries – mostly utilities or common codes.

Contracts – every Solidity file can have one or more contracts within them.

Constructor:

Constructor is a function with the same name as a contract. Every contract can optionally have a maximum of one constructor in it. However, unlike Java, the constructor can't be overloaded. The constructor is invoked only once when the contract is created.

Gas and operation cost:

Similar to Bitcoin, on the Ethereum network, we need to allure and engage the miners to validate the transactions by paying a fee called gas. The other advantage of the gas is that it keeps the spammers at bay, as with each transaction, they are spending the money from their pocket in the form of gas. Hence, it's well understood that more complex the transaction is, more would be the gas price that the requester of the smart contract has to pay. Similarly, the sooner the transaction has to be executed, the more would be the price of the gas to be paid and vice versa.

Transaction fees, Tx Fees = Gas Limit * Gas Price

Ether:

Ether is a crypto currency that is associated with a value which keeps on fluctuating, depending upon the demand and supply in the market. One can always buy, retain, and sell the Ethers in a crypto exchange.

Gas limit

It's the maximum value that the user is ready to pay to run an operation. The minimum price of the Gas limit is 21,000. If the entire value is not consumed, then the rest is returned to the user's account after the execution of the transaction. The Gas limit helps the user to safeguard his money that he is paying on the transaction with a maximum limit.

Gas Price:

It's the per unit price, which is calculated in Gwei. This final transaction fee is calculated in Ether. Gas is a very small fraction of Ether (the crypto currency associated with Ethereum).

Ethereum Virtual Machine:

The EVM uses four different types of memory locations to save the associated data in a smart contract. They are Storage, Memory, Calldata, and Stack.

Calldata is a non-modifiable, non-persistent area with almost a memory type of a temporary storage, and therefore, only stores the function arguments.

Stack is the data area for performing the calculations and it can store a maximum of up to 1024 elements with a maximum value of 256 bits. the Stack uses the least gas cost for saving the data, and yet can hold a limited amount of data.

Advantages:

1. It is not only a digital currency, but also provides a way to create applications on top of its network. Decentralized Applications (DApps), Decentralized Finance (DeFi),

Decentralized Autonomous Organizations (DAOs) are some of the projects that can be developed on Ethereum.
2. Processing cost & time is much less as compared to its competitor blockchain, i.e Bitcoin. Ethereum takes about 10-15 seconds for Block validation, while Bitcoin takes about 10 minutes
3. It has large community support built over years of trust wherein the blockchain has been able to manage billions of dollars worth of transactions, and hence would lead to constant improvements and development.
4. It removes the need of intermediaries like banks in Finance, lawyers in case of contracts, or third-party web hosting providers.

Disadvantages:

1. With more features to serve, more breakdowns and threats.
2. Can't be considered as beginner friendly to start creating apps on it, since learning Solidity is a task.
3. There's no limit to the amount of Ether tokens that are created. Though every year 18 million of them can get created. Thus it will be difficult for Ether's value to rise as much as Bitcoin's did.
4. With the chain getting bigger, the cost of processing a transaction calculated in Gas has been continuously increasing.
5. Ethereum is not scalable and hence layer 2 solutions like Matic(Polygon) are coming in the picture.
6. Turing completeness is very dangerous, particularly in open access systems like public blockchains, because of the halting problem

Ethereum Features

Ether: This is the digital token of the Ethereum blockchain.

Smart contracts: Ethereum allows the development and deployment of smart contracts. A smart contract is a simple computer program that facilitates the exchange of any valuable asset between two parties. It could be money, shares, property, or any other digital asset that people want to exchange.

Ethereum Virtual Machine: Ethereum provides the underlying technology, the architecture, and the software that understands smart contracts and enables people to interact with it.

Decentralized applications (Dapps): Ethereum allows people to create consolidated applications, called decentralized applications. A decentralized application is called a Dapp.

Decentralized autonomous organizations (DAOs): Ethereum enables users to create these for democratic decision-making. DAOs operate entirely transparently and independently of any intervention, with no single leader.

Characteristics:

- It is decentralized.
- It allows anonymous identities through pseudonyms.
- It is immutable.
- It is highly divisible.

Need:

- Ethereum enables deployment of smart contracts and decentralised apps (dapps) to be built and run without any downtime, fraud, control or interface of 3rd party.

- Smart contracts used from issuing tokens to creating entire decentralised autonomous organization. (DAO)

Applications: Cryyptocurrency transfer apps, IOT, security of personal identity, logistics, digital media

HYPERLEDGER

Hyperledger was set up with the aim of accelerating industry-wide collaboration for developing high-performance and reliable blockchain and distributed ledger-based technology framework that could be used across the various industry sectors to enhance the efficiency, performance, and transactions of the various business processes.

Hyperledger Fabric is an open source private permissioned DLT, introduced by Linux foundation whose main contribution came from IBM. It has a modular blockchain framework. The Hyperledger family also comes with a benchmark tool called Caliper to measure the performance of many Hyperledger Blockchain protocols, including Fabric

About Hyperledger fabric:

Identity is paramount in any Blockchain network. Fabric has many different actors such as orderers, peers, and administrators, and each of them have separate roles and access rights associated with their identities.

MSP is the trusted authority that has the power to issue, govern, or revoke the certificates of all the actors in the Fabric Blockchain network.

Policies are a set of rules that determine which organization has access to what resources. Policies can be implemented in the system channel (primary channel) level or application level or smart contract and access control level. Policies make sure of the access by matching the signature associates with each identity in the transactions.

Hyperledger Fabric introduced a concept of security zones with a new feature called channel that enables us to create kinds of Blockchains within a Blockchain. A channel is a zone within which, only the organizations affiliated to that channel can share the information. Of course, there is a base channel on which all the organizations in the network are registered, but then, on the basis of the business requirements, the channels facilitate the organizations to create different channels with a smaller group of selected organizations to share the information within that group.

Orderers are nodes responsible for creating an order of transactions after they are endorsed by endorsing the peers. The group of orderers collectively create the ordering service. Each orderer is bootstrapped with an initial channel known as the system channel. Orderers maintain a list of organizations or the consortium that can create the channels. The orderer service can configure the orderer nodes in three different ways, i.e. Solo, Kafka, and Raft.

Transactions may be of two types:

- *Deploy transactions* create new chaincode and take a program as parameter. When a deploy transaction executes successfully, the chaincode has been installed "on" the blockchain.
- *Invoke transactions* perform an operation in the context of previously deployed chaincode. An invoke transaction refers to a chaincode and to one of its provided

functions. When successful, the chaincode executes the specified function - which may involve modifying the corresponding state, and returning an output.

There are three types of nodes:

- Client or submitting-client: a client that submits an actual transaction-invocation to the endorsers, and broadcasts transaction-proposals to the ordering service.
- Peer: a node that commits transactions and maintains the state and a copy of the ledger (see Sec, 1.2). Besides, peers can have a special endorser role. https://hyperledger-fabric.readthedocs.io/en/release-2.2/peers/peers.html#:~:text=Peers%20are%20a%20fundamental%20element,%2C%20more%20on%20this%20later
- Ordering-service-node or orderer: a node running the communication service that implements a delivery guarantee, such as atomic or total order broadcast.
- https://hyperledger-fabric.readthedocs.io/en/release-1.4/arch-deep-dive.html

Chain Code:

Chaincode is the term for programs that run on top of the blockchain to implement the business logic of how applications interact with the ledger. When a transaction is proposed, it triggers chaincode that decides what state change should be applied to the ledger. For example, chaincode for the sale of bitcoin would check that the seller actually has bitcoin to sell. Once the smart contract is packaged, it is called Chain code that has to be deployed to the Blockchain network.

Chaincode implements higher-level functionality on top of the blockchain ledger. There are two kinds of chaincode, both running under a Peer's control. System chaincode is part of the Peer process. Normal chaincode is a separate container managed by the Peer. This arrangement works with plain Docker, but not in Kubernetes

- System Chaincode

There's a special kind of chaincode that follows a different lifecycle. System chaincode runs as part of the Peer process, not in a separate container. It's used to implement low-level ledger features like the endorser system, query system, and validation system. Because these features are so fundamental, the system chaincodes that implement them are deployed when the Peer starts (instead of being dynamically added later like other chaincodes).

Consensus Process:

- The consensus process of Hyperledger Fabric consists of the following components: Client SDK, Endorsing Peers, Committing Peers, Orderer(s). The process can be divided into three different phases, i.e., proposal and endorsement, ordering, and validation and commit that occurs in a chronological order.
- In Kafka, only the leader does the ordering and only the in-sync replicas can be voted as leader. This provides crash fault-tolerance and finality happens in a matter of seconds. While Kafka is crash fault tolerant, it is not Byzantine fault tolerant, which prevents the system from reaching agreement in the case of malicious or faulty nodes.

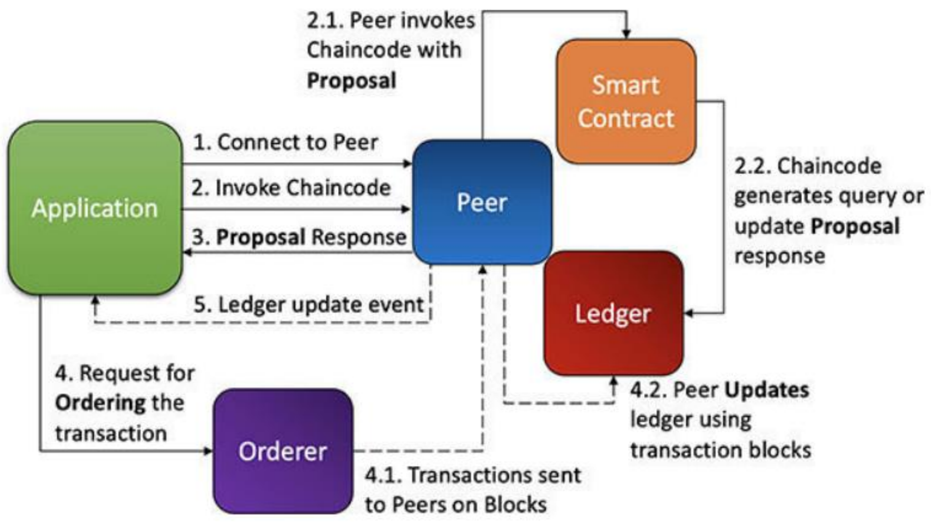

2.1. Peer invokes Chaincode with **Proposal**
Smart Contract
1. Connect to Peer
2. Invoke Chaincode
3. **Proposal** Response
Application
Peer
2.2. Chaincode generates query or update **Proposal** response
Ledger
5. Ledger update event
4. Request for **Ordering** the transaction
Orderer
4.2. Peer **Updates** ledger using transaction blocks
4.1. Transactions sent to Peers on Blocks

Figure 13.3: *Consensus Process in Hyperledger Fabric*

Pg 497 – Blockchain from concept to execution

<u>Consensus Properties</u>

Consensus must satisfy two properties to guarantee agreement among nodes: safety and liveness.

Safety means that each node is guaranteed the same sequence of inputs and results in the same output on each node. When the nodes receive an identical series of transactions, the same state changes will occur on each node. The algorithm must behave identical to a single node system that executes each transaction atomically one at a time.

Liveness means that each non-faulty node will eventually receive every submitted transaction, assuming that communication does not fail.

Features:

1. Here smart contracts are known as chain codes which are generally hosted using the technology of containers.
2. It makes use of channel technology for all transactions that are confidential.
3. It offers various ordering services that deliver various transactions to nodes in a blockchain network.
4. It endorses transaction validation policy.
5. It has database services like couch DB.

Advantage:

1. Hyperledger Fabric can be fast and highly scalable if configured properly.
2. Performance, scalability and levels of trust.
3. Permissioned membership (known identities)
4. Data on a need-to-know basis (data partitioning using channels)

5. Rich queries over an immutable distributed ledger.
6. Modular architecture supporting plug-in components
7. Protection of digital keys and sensitive data

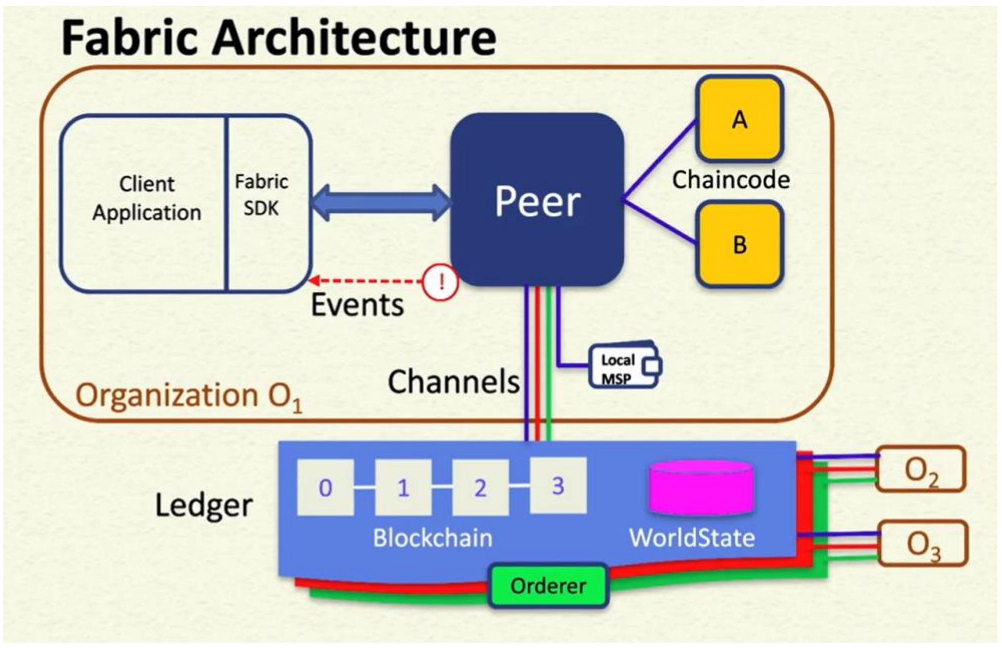
Disadvantage:

1. It does not have much-skilled programmers.
2. It has shown a lack of use cases.
3. It has got a complex architecture.
4. It has got minimum APIs and SDKs.
5. It is not a Network fault-tolerant.

Applications:

1. Digital Payments, Diamond tracing, B2B contracts, digital identity
2. https://www.upgrad.com/blog/hyperledger-fabric-features-applications/#Hyperledger_Fabric_Applications

Architecture:



Hence, every smart contract is associated with an endorsement policy that finds out the organization that must approve the transactions before they are considered valid.

MULTICHAIN

Fork of bitcoin used in private network. Used for lightweight financial systems (loyalty points/crown funding), provenance tracking (begin-to-end of asset), inter-organizational record keeping (healthcare/legal sectors). Multichain now offers three areas of high-level functionality:

- **Permissions** to control who can connect, transact, create assets/streams, mine/validate and administrate.

- **Assets** including issuance, reissuance, transfer, atomic exchange, escrow and destruction.
- **Streams** with APIs for creating streams, writing, subscribing, indexing and retrieving.

About Multichain:

1. Platform for creation and deployment of private blockchain, either within or between organizations.
2. Very lightweight private network.
3. Fork of bitcoin core. Started in 2014. Hence compatible with bitcoin ecosystem. Bitcoin's protocol, transaction, architecture but changes in handshaking process.
4. It doesn't have smart contracts.
5. It doesn't have any cryptocurrency or native assets.
6. Assets, Data Streams(key-value), Mining
7. Supports Windows, Linux and Mac servers.
8. Provides a simple API and CLI that makes it easy to maintain and deploy.

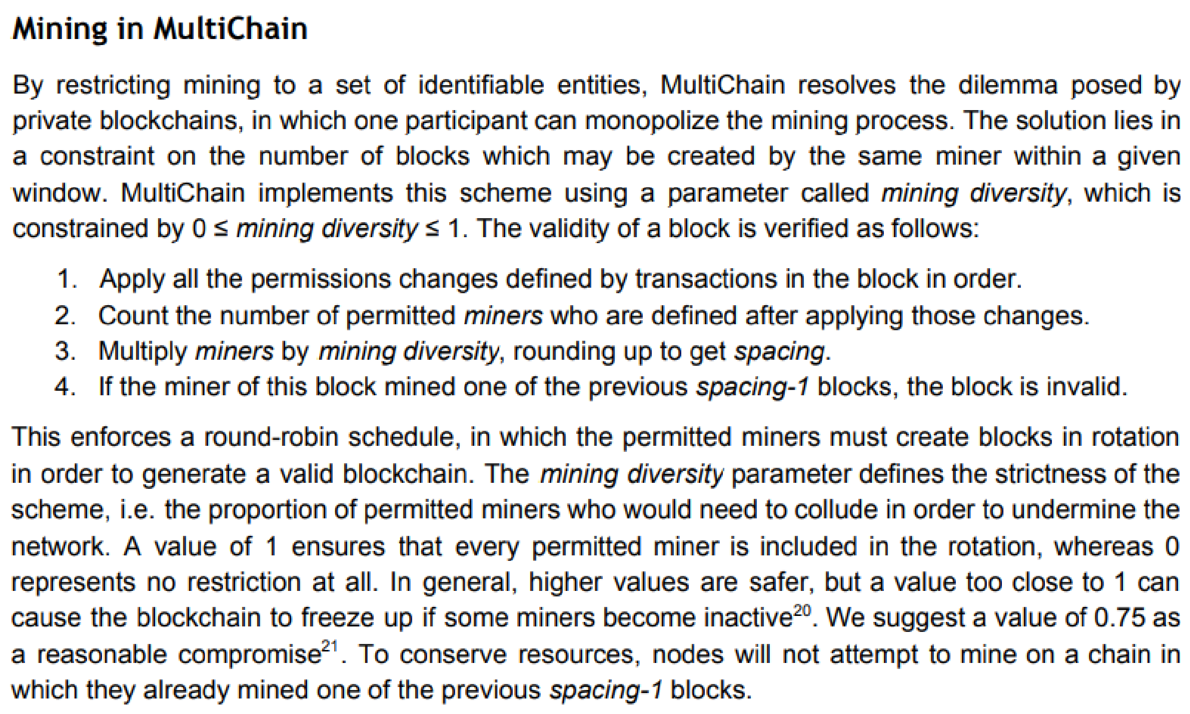
Multichain Streams:

- Key-value pairs, provides a natural abstraction for general data retrieval, time stamping and archiving
- Any number of streams created, acts as independent append only collection of items
- Nodes choose which streams to index
- Used to implement databases (Key-value db/ time series db/ identity driven db)
- Each item in a stream has the following characteristics:
    - One or more publishers who have digitally signed that item.
    - An optional key for convenient later retrieval.
    - Some data, which can range from a small piece of text to many megabytes of raw binary.
    - A timestamp, which is taken from the header of the block in which the item is confirmed.
- https://www.multichain.com/blog/2016/09/introducing-multichain-streams/

Consensus:

- PBFT (Practical Byzantine Fault Tolerance)
- Distributed consensus, one validator per block working in round robin fashion.
- Block digitally signed by creator, only permissioned parties can mine. No POW or cryptocurrency.
- Uses the concept of 'mining diversity'. No incentive for mining.
- In a private blockchain, you don't need "mining" in the same sense as a public blockchain like bitcoin. Instead, the network is protected by the need for distributed

consensus (or "mining diversity" as we call it) and no difficult computation is required at all.



Mining in MultiChain

By restricting mining to a set of identifiable entities, MultiChain resolves the dilemma posed by private blockchains, in which one participant can monopolize the mining process. The solution lies in a constraint on the number of blocks which may be created by the same miner within a given window. MultiChain implements this scheme using a parameter called *mining diversity*, which is constrained by 0 ≤ *mining diversity* ≤ 1. The validity of a block is verified as follows:

1. Apply all the permissions changes defined by transactions in the block in order.
2. Count the number of permitted *miners* who are defined after applying those changes.
3. Multiply *miners* by *mining diversity*, rounding up to get *spacing*.
4. If the miner of this block mined one of the previous *spacing-1* blocks, the block is invalid.

This enforces a round-robin schedule, in which the permitted miners must create blocks in rotation in order to generate a valid blockchain. The *mining diversity* parameter defines the strictness of the scheme, i.e. the proportion of permitted miners who would need to collude in order to undermine the network. A value of 1 ensures that every permitted miner is included in the rotation, whereas 0 represents no restriction at all. In general, higher values are safer, but a value too close to 1 can cause the blockchain to freeze up if some miners become inactive[20]. We suggest a value of 0.75 as a reasonable compromise[21]. To conserve resources, nodes will not attempt to mine on a chain in which they already mined one of the previous *spacing-1* blocks.

-

Privacy and permissions:

Streams provide a natural way to support encrypted data on a blockchain, as follows:

1. One stream is used by participants to distribute their public keys for any public-key cryptography scheme.

2. A second stream is used to publish data, where each piece of data is encrypted using symmetric cryptography with a unique key.

3. A third stream provides data access. For each participant who should see a piece of data, a stream entry is created which contains that data's secret key, encrypted using that participant's public key.

This provides an efficient way to archive data on a blockchain, while making it visible only to certain participants.

Advantage:

1. Difference is you can create your own private or permissioned network.
2. Easy to use and fast to setup.

Disadvantage:

1. Doesn't allow you to write smart contracts. Thus, limited applications.
2. Doesn't work on docker container model.

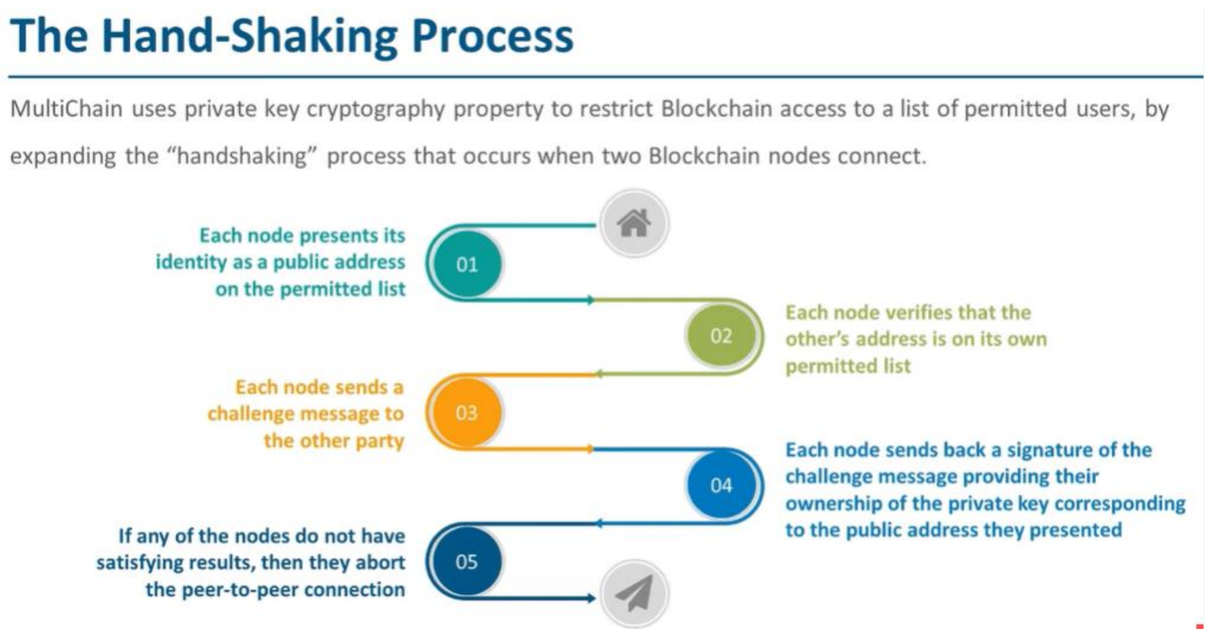
3. Doesn’t have any emulator. Will have to create actual network to use it.

Applications:

1. Financial, supply chain, government – public sector, medical industry, Telecom, Trading companies

Architecture:

1. Same as bitcoin



The Hand-Shaking Process

MultiChain uses private key cryptography property to restrict Blockchain access to a list of permitted users, by expanding the “handshaking” process that occurs when two Blockchain nodes connect.

01
Each node presents its identity as a public address on the permitted list
02
Each node verifies that the other’s address is on its own permitted list
03
Each node sends a challenge message to the other party
04
Each node sends back a signature of the challenge message providing their ownership of the private key corresponding to the public address they presented
05
If any of the nodes do not have satisfying results, then they abort the peer-to-peer connection

The simplest way to resolve this problem is for each participant to transact under many different addresses. When sending or receiving transactions, they can use a different address depending on the identity of the counterparty. This prevents either party from gaining a full picture of their counterparty’s activities, since they do not know which other addresses the counterparty is using. Participants can move assets between their addresses as and when is required, taking care to ensure that those transactions are indistinguishable from payments to other entities. Alternatively, a trusted central party could provide a “coin mixing” service, allowing assets to deposited and subsequently withdrawn using different addresses, while ensuring there is no visible connection between the deposit and withdrawal transactions. A more fundamental approach to privacy is promised by cryptographic techniques such as homomorphic encryption and zeroknowledge proofs. In a general sense, these enable specific computations to be performed, with their accuracy publicly proven, without revealing the inputs and outputs of those computations. In a blockchain the techniques can be applied to hide a transaction’s asset quantities from all but the sender and recipient of that transaction, while still enabling all network participants to verify that the transaction is valid . If blockchain participants cannot see the 31 quantities in each other’s transactions, it becomes trivial to obscure genuine activity behind a smokescreen of transactions which move negligible amounts.

Multiple configurable blockchains:

Rather than supporting a single blockchain like Bitcoin Core, MultiChain is easy to configure and can work with different blockchains at the same time. The immediate benefit for institutional users is enabling private blockchains to be configured and deployed by system administrators rather than specialized developers. An analogy is the way in which relational database management systems such as Oracle or SQL Server allow databases to be created and used with a few SQL commands. A further benefit of supporting multiple blockchains is the ability for a server to create connections between the activity in different chains. For example, an institution may want the arrival of funds on one blockchain to trigger a corresponding transfer of funds on another.

BLOCKCHAINS FOR ENTERPRISE:

1. Designated miners
2. Private shared database
3. Collective admin
4. Blockchain as tool not ideology
5. No cryptocurrency
6. Hide the details

Multichain benefits:

1. Managed permissions
2. Rapid deployment
3. Unlimited assets
4. Data streams

Hyperledger Benefits:

1. Modular Design
2. Security and privacy
3. High fidelity and efficient transactions (peer and ordering nodes)
4. Ledger (immutable record of transactions)
5. Rich querying capability and on-demand data