HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Introduction

This module covers the basic components of the Hyperledger Fabric Network. This module will also help to understand to setup the development environment to be able to build the network. After completing this module, you will have the basic understanding of how a Hyperledger Fabric Network works; the various components and services included in the network; how to install and have a development environment ready to be able to work in the local environment.

The topics covered in this session are as follows:

- Distributed ledger technology
- Introduction to Hyperledger and Hyperledger Fabric
- The architecture of Fabric along with the transaction flow
- Important concepts of Hyperledger Fabric such as gossip protocol and private data
- Installation of the Fabric environment on a local machine using a virtual box to build a network

The prerequisites for this module are as follows:

- NodeJS programming language
- NPM tool
- Docker and Docker Compose (please refer to the Additional resources session in the module)

All the resources required to run the Hyperledger Fabric inside of Docker is provided in this module.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Distributed Ledger Technology

The fundamental building blocks of a Distributed Ledger Technology (DLT) are:

- Data model: how the data is stored in the network -> **Ledger**
- Language of transactions: logic on the data to be able to change its state -> **Smart Contracts**
- Consensus protocols: define the sequence of transactions which will be replicated across all the peers on the network -> **Proof of Work (PoW)**

Contrasting Ethereum and Hyperledger:
Ethereum is a permisionless, open, public blockchain. This means that anybody can join the network at any point if they wish to. These types of networks cannot fully be able to meet the requirements of enterprises in terms of scale and throughput. Also, enterprises would require access restrictions into the network.

Requirements of enterprises:

- ✔ Speed: eg. Speed required for running an application like Facebook
- ✔ Scalability: eg. Payment processing system like Visa
- ✔ Latency: Time taken between initiation and commitment of a transaction. Eg. Train tracking solution for railways
- ✔ Privacy: eg. Applications like Paytm will require KYC and access controls
- ✔ Restricted access: eg. Paytm user balances are required to be kept private

The above requirements are unable to be solved by the public, permisionless blockchain.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Introduction to Hyperledger

Hyperledger is an Open Source project which was built to foster cross industry blockchain technologies with a purpose to build enterprise ready blockchain solutions. Hyperledger is hosted under the Linux Foundation.

There are numerous projects/frameworks under the Hyperledger umbrella that provide blockchain frameworks for different use cases. These frameworks were designed to solve specific use cases. Some of these frameworks are as follows:

Fabric: It is an open-source, enterprise-ready solution for blockchain.
Sawtooth: It is a variation in the blockchain environment and is similar to Fabric, but differs with regard to the consensus processes and the architecture around which it is built.
Indy and Iroha: They are built for different purposes or have different consensus processes or the architecture on which it is built.

Apart from the frameworks mentioned above, there are a few other frameworks in Hyperledger. Under its umbrella, Hyperledger also hosts tools for specific purposes. Some of the Hyperledger tools are as follows:

Composer: This tool helps to easily build applications over the Hyperledger Fabric network.
Calliper: It allows measuring the activities and performance of some of the frameworks hosted on the Hyperledger consortium.
Explorer: It allows viewing of the data hosted on the Hyperledger network.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Hyperledger Fabric:

- Is a Distributed Ledger Technology (DLT) platform
- Is an Open Source solution
- Is enterprise grade
- Is a permissioned network

Distinguishing features from other blockchain networks:

- Is a highly modular blockchain network: all services and platform within the Hyperledger network can be changed with other openly available platforms. Hence it is a highly configurable platform for enterprise usage
- It supports openly used programming languages: commonly used programming languages like Go, NodeJS, Java can be used to develop Smart Contracts for Hyperledger
- It allows for privacy: transactions that happen on the networks can be private and it allows for confidentiality

Fabric was first introduced as an open source project inside the Hyperledger consortium using a combination of three projects:

- Libconsensus project by Blockstream
- Digital Assests holdings
- OpenBlock by IBM

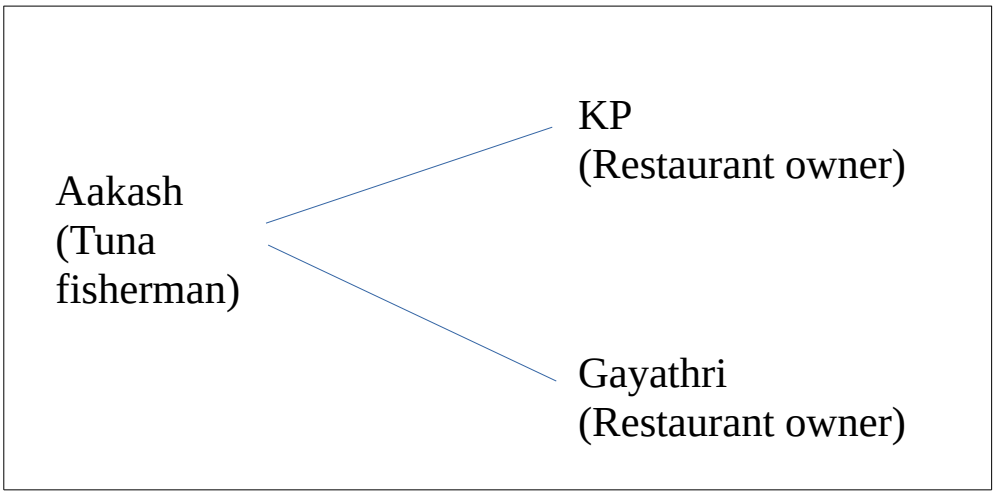
Since then Fabric has grown immensely and become a widely popular and used platform within the Hyperledger consortium.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Hyperledger Fabric: Architecture

Case Study -1: Tuna supply chain

We will understand the architecture using this case study. Aakash is a Tuna fisherman. Aakash has to sell this to restaurant owners to be used as a part of their menu. Aakash sells this to two restaurants: KP and Gayathri.

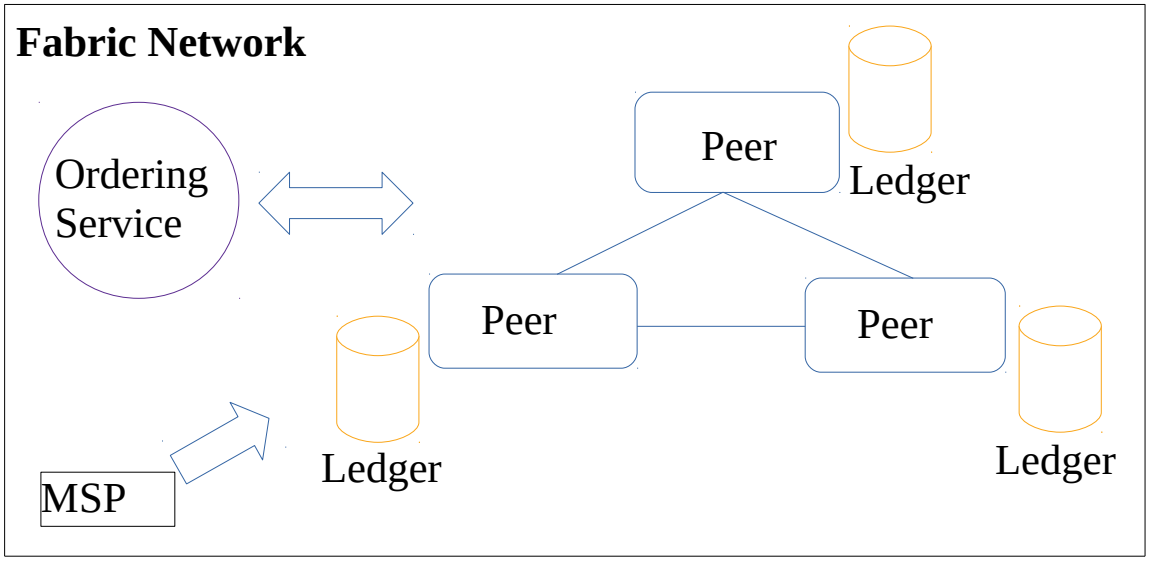


Elements involved in Hyperledger Network:

- **Channel:** Allows for compartmentalizing data between stakeholders in the network. In this use case, Aakash and KP have a specific deal over the pricing of the Tuna. But Aakash would like to sell it at a different price for Gayathri. Hence Aakash would want to have these details kept completely separate and private between his customers. This would enable him to sell his products at differentiating prices to different customers.
- **Assets:** This is the data that gets tracked and stored on the network: Tuna, in this case. Asset is a key-value pair that gets represented in network with value equivalent to that of the physical product or service. It also can track the changes to this value (examples like ownership, location etc). In this use case, the Tuna changes ownership from Aakash to KP or Gayathri as the sale transaction goes through.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

- **Transaction:** A transaction enables the change of state of the asset. In this case, the change of ownership from Aakash to KP is enabled using a transaction.
- **Ledger:** List of transactions and the state of the assets forms the ledger. In other words, a ledger is a series of assets being tracked along with the transactions that have led to the change of state of these assets to its current state.
- **World state:** Current state of all the assets and its transactions is called as the world state.
- **Smart contract (Chaincode):** Logic to derive the transactions and the changes in state of the assets.
- **Peer:** A computing resource which is part of the network. In this case, Aakash, KP and Gayathri have come together to form this network. Hence each one would contribute a peer and hence the ownership of the network is split between the three contributors.
- **Ordering service:** Takes transactions coming in from the peers and puts them in a block in a sequence to be written on the ledgers of each peer. Every peer will maintain a ledger.
- **MSP (Membership Service Provider):** Provide the credentials or ID's required by the applications run by the peers to interact with the network. Since Hyperledger is a permissioned network, each element of the network including applications will need identification.



HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Case Study - II

As Aakash the fisherman catches a new Tuna, he would record that information on the network. This will be an asset being tracked on the network. The Tuna asset would have information like unique id, boat used to catch, weight of tuna, when and by whom it was captured etc. All these information will be recorded as key-value pairs which together forms the assets. The value for each of the assets becomes the current state of the network. As the sale transaction takes place, the state of the assets like ownership will change and that becomes the new state of the asset.

Aakash will use the smart contract or chaincode to record the asset information on the network. These transactions and assets will form the data on the ledgers. The data will be stored as key-value pairs on the ledgers and will be shared across the peers on the network. The ordering service will capture the transactions and puts them in Blocks. These will then be distributed and synchronized across all the ledgers on the network.

Tuna:

{
id: value
boat: value
kg: value
when: value
who: value
}

Problem statement:

- Aakash the Tuna fisherman has recorded the data about the Tuna he has caught as assets on the network
- Aakash wants his Tuna to be bought and paid for it by the restaurants
- Restaurant owners KP and Gayathri want to buy Tuna sustainable. But they don't know much about the Tuna that was caught, for example they don't know how much is the weight and whether it was caught legally

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Solving the problem using the Hyperledger Fabric network:

- Information about each Tuna gets recorded on the network
- This data is distributed across the network for everyone to access
- The restaurant owners are able to buy this securely and this sale will be recorded privately on the network

After the Tuna is caught and recorded on the network, KP or Gayathri would not want to buy this Tuna which is now available in the list of assets available on the network.

Aakash decides to sell the Tuna at Rs.1000/box to KP and at Rs.1500/box go Gayathri. And Aakash would want to keep these deals private and not known to each other. This is where the concept of a Channel would help. Here Aakash can cut a deal and perform the transactions on different channels:

Channel CA – deal and transactions with KP. Here Gayathri cannot see the details of this deal or the transactions since she is not a part of CA
Channel CB – deal and transactions with Gayathri. Here KP cannot see the details of this deal or the transact since he is not a part of CB
Channel CC – Aakash will want all the network participants to see the details of the Tuna he has caught. Hence he will use this channel to put the information about the Tuna which will be available for both KP and Gayathri to see.

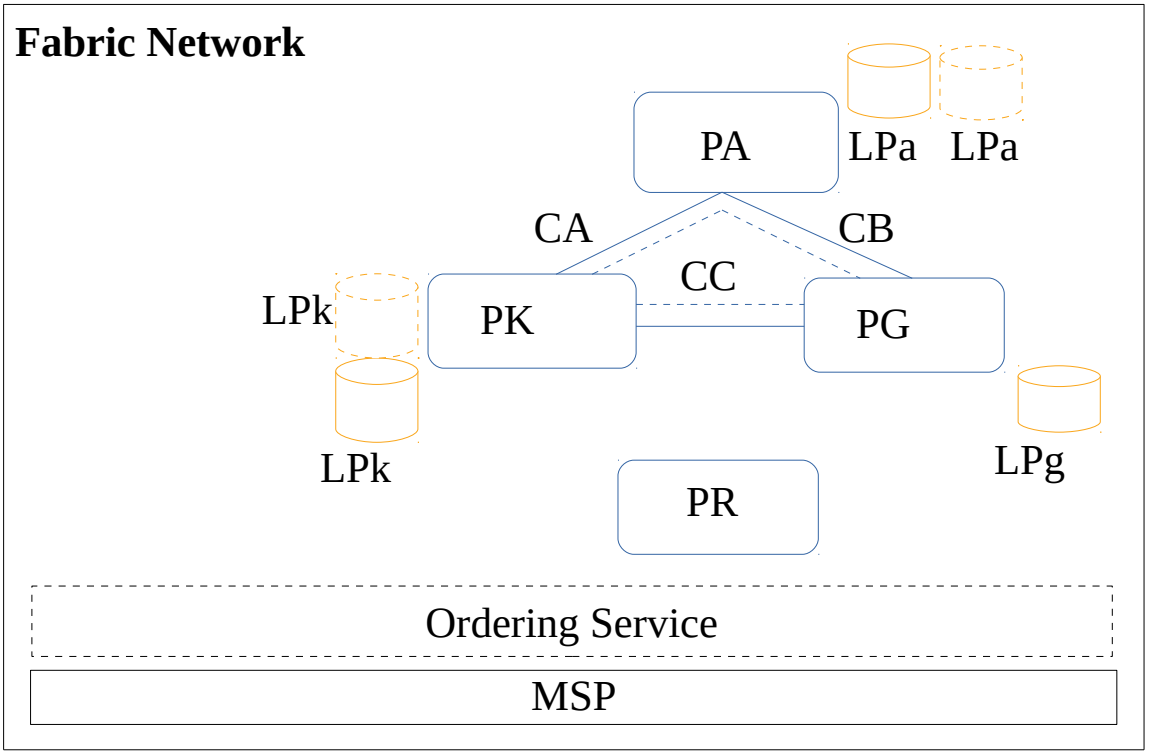
Regulator:
A regulator is another Peer to put some governance and decide if some network participants have been acting unethically or illegally. The regulator will have the properties set to penalize any of the participants in the network. Also, the regulator will not have any assets or perform transactions on the network. Hence this peer will only need to query the ledger and need not have any write access. This level of role and control will be provided by the MSP.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Summary:

- We have built a network where each of the network participants Aakash, KP and Gayathri have contributed a peer PA, PK, PB
- Each of these peers know each other and also their owner through the identities issued by the MSP (Membership Service Provider)
- Another peer PR is contributed by the regulator whose role is to control the activities taking place on the network
- We have three channels on the network: CA, CB, CC. These allow Aakash to have separate deals between KP and Gayathri
- The transactions in these channels are kept private. This is enabled by another separate ledger that is maintained by the channel which will be unique and available only to the peers who have joined the channel
- Channel CC is used by Aakash to record data about all the Tuna that is caught. This channel is joined by everyone and hence all are able to have access to this information
- We have 2 chaincodes to list the Tuna and also to perform sale transactions. One chaincode will enable recording of the Tuna information and the sale transactions. This is available on channel CC
- The second chaincode will enable recording of the deal separately between KP and Gayathri. This is available on channel CA and CB. This is used to privately make the deals between the restaurant owners
- MSP allows the peers to be identified and also allows new participants. It issues credentials to applications that act on behalf of the owners from outside of the network
- Flow:
  - Aakash will record the details/parameters of the Tuna caught and this becomes the Tuna asset
  - This asset will be put on the ledger using the chaincode
  - The ordering service will captures this transaction, creates a block and broadcasts this to all the peers on the network
  - These ledgers then become synchronized and the blockchain continues further

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS



PA: Peer contributed by Aakash
PK: Peer contributed by KP
PG: Peer contributed by Gayathri
PR: Peer contributed by the regulator
CA: Channel in which only PA and PK have joined
CB: Channel in which only PA and PG have joined
CC: Channel in which all peers PA, PK, PG have joined
LPk: Ledger maintained by peer PK and one separately for channel CA
LPa: Ledger maintained by peer PA and one separately for channel CA
Lpg: Ledger maintained by peer PG
MSP: Membership Service Provider

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Components of Hyperledger Fabric – I

The components of the Hyperledger Fabric network are follows:

Assets:

An Asset in Hyperledger Fabric is an important entity that is stored on the ledger. They are stored in the form of key-value pairs.
For example, in the Tune Supply Chain problem, the Tuna fish is the asset. Suppose Aakash wants to add a tuna fish to the network named “Fishy”. This tuna is represented in a key-value pair and the representation of the same is as follows:

```
{
 "ID" : "0001"

 “Name”: “Fishy”
 “Weight”: “2kg”
 “Owner”: “Aakash”
 “Source”: “Narmada River”

 } version='0'
```

Note: The tuna asset example defined above is just for illustration purpose.

Transaction:

A transaction is responsible for the change in the state of the asset.
Suppose the tuna named “Fishy” is bought by the restaurant owner KP. This change in the ownership is referred to as the transaction. The version of the asset “Fishy” will now be updated to version=”1” and the owner will be changed from Aakash to KP. It is represented as-

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

{
"ID" : "0001"

“Name”: “Fishy”
“Weight”: “2kg”
“Owner”: “KP”
“Source”: “Narmada River”

} version='1'

Ledger:

The information about an asset or a transaction of the network is stored in the data structures.
The ledger is a data structure which stores all the changes in the assets and transaction.
It also stores the current value of the assets as well as the history of the transactions which lead to the current state of the asset.

World State:

The world state is the data structure which stores the current state of the asset.
Revisiting the asset tuna fish with the name ‘Fishy’ currently stored in the network.
Thus in the world state, the version of the tuna ‘Fishy’ will be ‘0’.
It is represented as-

{
"ID" : "0001"

“Name”: “Fishy”
“Weight”: “2kg”
“Owner”: “Aakash”
“Source”: “Narmada River”

} version='0'

After a sale transaction, the version of this tuna will be updated to version=”1”.

Channel:

A Channel is a distinguishing factor that helps compartmentalise the data of the network.
These are the private communication sub-networks between different peers in a blockchain.
Only the members of the channel can see the specifics of the transactions in that channel.
In the tuna supply chain, Aakash will put his tuna on a channel so that all the stakeholders of the channel can view the details of the tuna “Fishy”.

Chaincode:

Chaincode is the business logic for performing and validating transactions in a Hyperledger Fabric network.
It enables users to create transactions in a Hyperledger Fabric network’s shared ledger and update the state of the assets.
Chaincode is invoked by triggering transactions that result in a change of the state of the ledger.

Peers:

Each of the stakeholders of the network contributes a computing resource called a peer to become part of the network.
Peers are the primary communication entities in the network.
In the tuna supply chain, Aakash, KP, and Gayathri contribute a peer to the network, PA, PK and PG respectively. They will control their own respective peers.
This way, the ownership of the network will be divided among the different stakeholders.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Ordering Service:

The ordering service is responsible to put the transactions into a block which are coming from various peers.
The ordering service then decides the sequence in which the blocks should be placed in the ledger of each peer.
The ordering service will be explained in-depth in upcoming segments.

MSP:

It defines the rules using which identities are validated, authenticated and allowed access to a network.

Note:

- Each peer maintains its own version of the ledger.
- The sequence of the blocks kept inside these ledgers is maintained by the ordering service.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Components of Hyperledger Fabric – II

This covers the roles that are defined for different elements of the Fabric network and the process of consensus.

Client: It is an application that acts on behalf of any stakeholder in the network. In our example, Aakash, KP, and Gayathri are the stakeholders, who use the application to access the different functions on the network.

Peers: These are the computing resources that are owned by the members of the organizations in the network. Each member can have one or more peers in the network. In the tuna supply chain problem, Aakash, KP, and Gayathri contribute one peer each in the network, and a regulatory body also contributes a peer in the network.

These peers can further have two roles as follows:

Committer: It takes the blocks that have been advertised by the ordering service and puts them on its ledger. Every peer that is part of the network will be a committer for a transaction as long as it belongs to the channel where the transaction was initiated.

For example, if in this case, each of the organizations contribute 2 peers, then there will be a total of 8 peers and all of them will be committers.

Endorser: The job of the endorser is to take an incoming transaction from the application, simulate the transaction and pass it to the ordering service. A particular peer can be an endorser in the network if it has a chaincode installed on it.

Again in this example, we can decide to install the chaincode only on 4 of the 8 peers and hence these become the endorsers. The role of these peers will then be to test out the transactions and pass it on to the ordering service.

Ordering service: An ordering service puts the transactions in the right sequence of their occurrence; converts these transactions into a block and then disseminates this block in the network. It is built out as a network of several nodes. We don't want the ordering nodes which puts the transactions together as blocks to be owned by any one of the network participants. Hence this network of ordering nodes will be owned by the different organizations. Each node is called an Orderer.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Consensus:

A key challenge in a distributed network is to decide the ordering of the Blocks. In a network, multiple peers work together to validate transactions and try to create Blocks which look similar but have some variations. One of these Blocks get chosen to be put as the next Block on the chain. This is the process of ordering of the Blocks. This process of deciding the ordering of the Block and the selection of the winner Block to be included in the chain is called as Consensus.

To achieve consensus, Ethereum uses process of Mining. Whereas Hyperledger uses the Ordering Service.

The consensus inside Fabric is a three-step process:

Endorsement: Every time a transaction is initiated by any of the peers of the network, that transaction is first endorsed by the peers of the network.
Ordering: Once endorsed, the transaction moves to order. Ordering decides which transactions are put on the block and in what sequence the blocks will be put onto to the ledger of the different peers.
Validation: After the blocks have been ordered in a particular sequence by the ordering service, before being committed on to the ledger by each of the peers, they go through a process of validation.

Transaction Flow in Fabric:

The following takes place when Aakash tries to record his Tuna catch through the client application:

1. The client (the application) makes a transaction proposal
    a) The transaction proposal which has the details of the Tuna asset will hit the Tuna Record function in the chaincode on the network. This chaincode will require the details that needs to be updated about the Tuna asset
    b) The application will send this to all the peers on the network

2. The transaction proposal reaches the endorsers. These endorsers simulate the transaction
    a) The peers will run the transaction proposal on their local CPU's using the current state of the ledger
    b) The peers will project the expected output which will only be committed later on the network
3. Once the transaction is executed, each endorser generates a read set and a write set. A read-write set is the current state of the ledger (read) and the expected output (write) of the transaction
4. The transaction proposal with the read-write set along with the signature of the peer forms the endorsed proposal response
5. An application has to receive a certain amount of endorsement proposal response among the peers of the network based on the endorsement policy
6. As part of setting up a network, the organizations in the network can decide for each of the channels in the network what their endorsement policy will be
7. If the application does not receive the right amount of endorsements that are required by the endorsement policy from the peers, it will send a communication to the client application saying that the transaction does not fulfill the endorsement policy and, hence, cannot go on to the network
8. If this transaction fulfills the endorsement policy and receives the endorsed proposal responses from the peers (based on the endorsement policy), the transaction is then passed to the ordering service

Ordering Service:

After the transaction is sent to the ordering service, ordering mechanism is applied to order the transaction into a block. The ordering service will group them into a Block and deliver it to the peers. The ordering service does not run the transactions, it does not have the smart contracts to simulate these transactions and also does not perform any processing on those transactions. It merely decides the order in which these transactions have to be sequenced on the ledger.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Since Fabric is modular, it allows the enterprises to choose which ordering mechanism they want for their network:

Solo: used mostly in the development mode. It is a single node ordering service

Kafka: it is a more complicated cluster of nodes owned by different organizations and it allows building of the transactions into the Blocks

BFT: currently not implemented in Hyperledger. It is available as a roadmap

Block validation:

After the ordering service groups the transactions into Blocks and delivers them to the peers, the peers perform the validation.

- The peer after receiving a block from the ordering service looks at the ledger for that particular transaction. The peer matches the state of the ledger with the read state of the block
- If it checks out, the reading process is validated. It then commits the write value on top of its ledger and updates the current state of its ledger to the particular write value of the transaction
- As part of the validation process, all the endorsements that are required for the transaction to be valid are checked to see whether they are present or not. This is different from the process of verification that happens on the side of application. This is the second check at this point to ensure that any unethically passed transactions by the applications are not recorded on the blockchain. Once this check is passed, it will commit the write value to the ledgers.

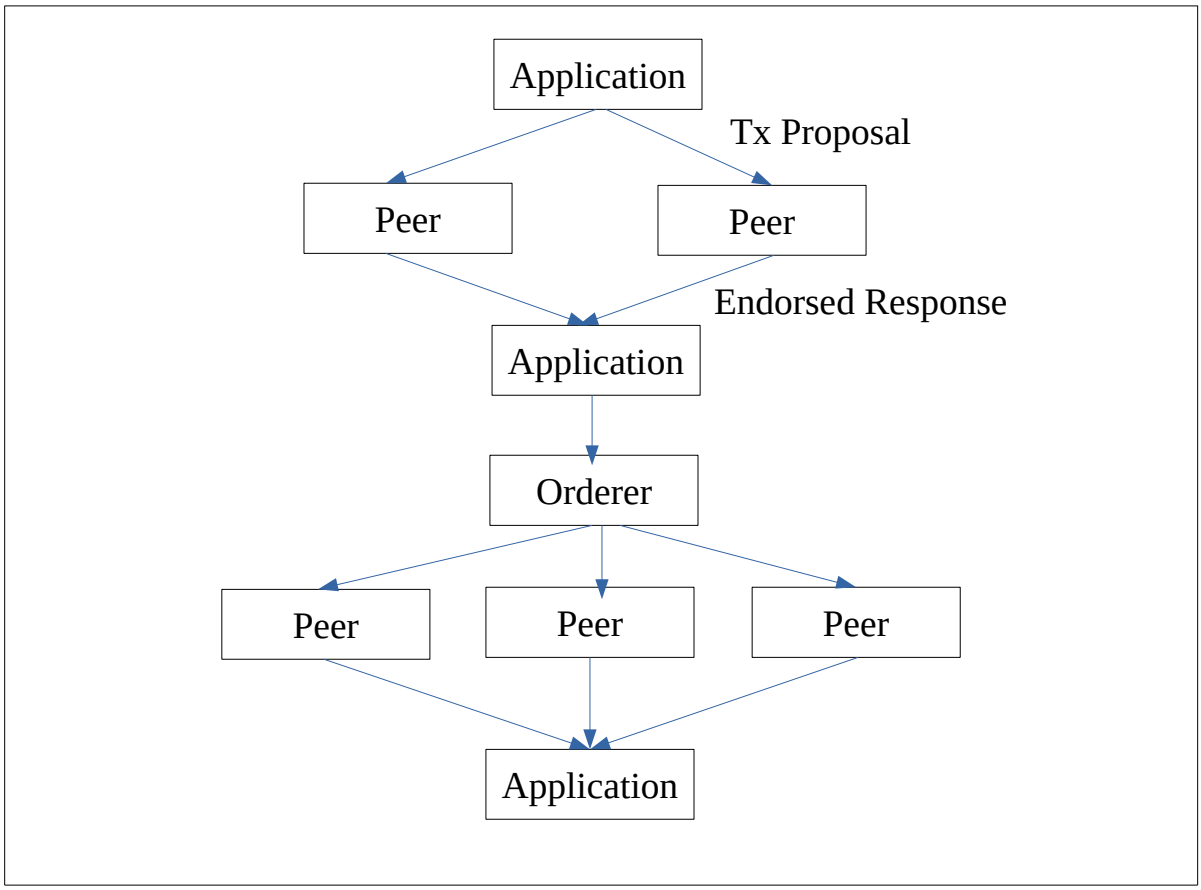
What after validation?

Finally, after the values have been written to the local ledgers of the peers, each of the committing peer will send asynchronous response back to the application notifying the update to the ledger.

This will end the entire transaction flow for this particular update to the state of the ledger.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Summary of the transaction flow:



- Aakash would use an application and initiate a transaction proposal
- The transaction proposal will be sent to the peers in the network which has the smart contract installed in them. These peers are the endorsers
- The endorser peers will simulate the transaction and create the read-write set
- The peers will sign these read-write sets with their credentials and send back the endorsed response to the application
- The application will wait for the right number of endorsed responses from the peers which are required based on the endorsement policy of the channel
- Once the application receives all the required number of endorsed responses, it sends the read-write sets and the endorsements received to the ordering service

- The ordering service would put these transactions into the Block and sequence the block and deliver it to the committers which are also peers that have a local copy of the ledger
- The peers will go through the validation. They go through the read set to check if they match with the current state of the ledger. They will also check if the endorsement policy has been met by the transaction
- Once the validation is complete, the committer peers will send asynchronous response back to the application

Important points to remember:

- In a transaction flow, the same peers can act as both endorsers and committers.
- However, an endorser will also always be a committer, which means that every peer in the network will be a committer. But a committer need not necessarily be an endorser.
- The only way a peer ends up being an endorser is if it has a smart contract installed on it. Since an endorser has to simulate a transaction, it needs a smart contract code to be able to simulate it.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Highlights of Hyperledger Fabric

Important Concepts of Fabric

Channels:

- There can be any number of channels in a network
- A peer can join and be a part of more than one channel
- The same chaincode can be installed on more than one peer
- Channels enable compartmentalizing data and hence ensure privacy among a group of peers within the network

State Database / World State:

- Transactions are used by the peers to update their ledgers
- This updated state or current state of the ledger is maintained as the World State in each peer for each channel they have joined
- For every channel, a peer has a separate ledger
- Ledgers data is a combination of the current state of the network (World State) and a history of transactions
- When a new peer joins a network, the history of transactions is used to calculate the current state of the network before accepting new transactions
- The default implementation of the world state is LevelDB
- LevelDB is a document database that stores data as key-value pairs
- Peers can also choose to implement CouchDB. It is very similar in structure and functionalities. The key difference is that CouchDB allows to store data as JSON objects. This enables running of rich data queries

MSP (Membership Service Provider):

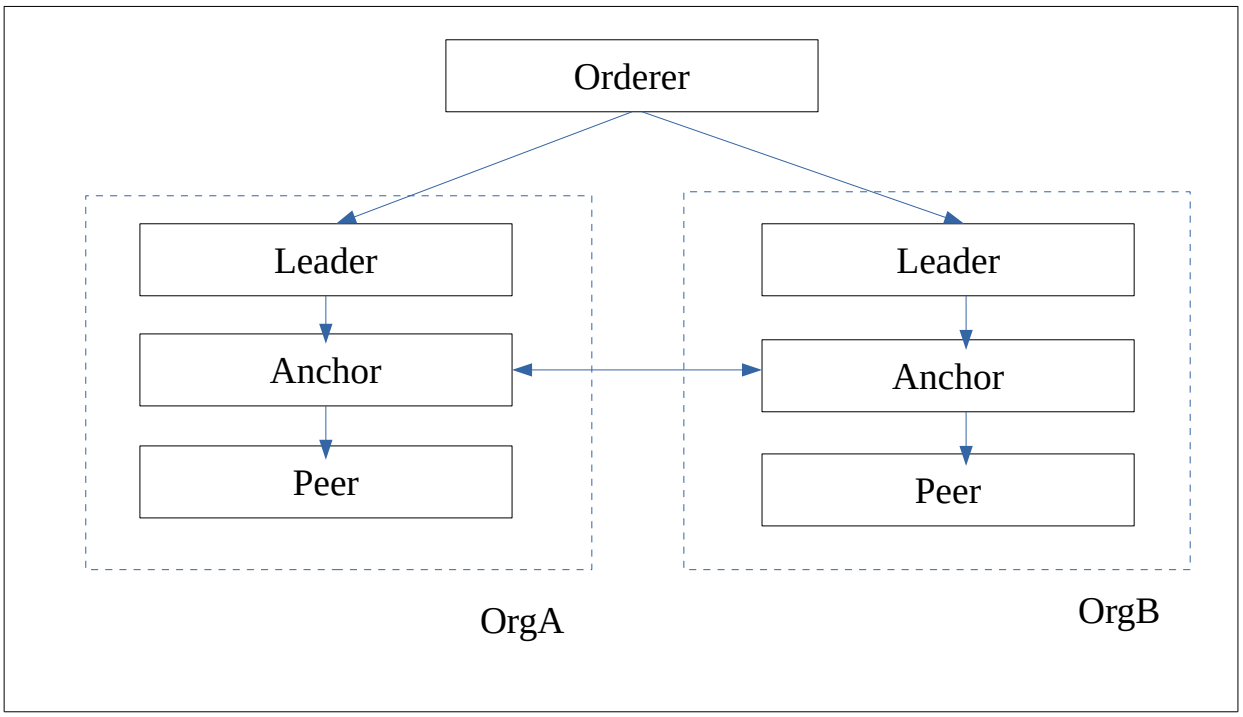
- Purpose of MSP is to issue identities / credentials to each element in the network and also to manage the process of authorization
- MSP issues credentials to all the roles in the network including the peers, client application or the orderers
- MSP achieves this using a CA – Certificate Authority

- CA issues the certificates to identify the different elements in the network
- The default implementation of CA is the Fabric-CA which is included in the network
- However, enterprises are free to implement their own version of the CA which is called as an External-CA implementation
- The External-CA can be any open source, third party CA service to issue the identification to the entities in the network
- The Fabric network is plugable, the enterprise can choose MSP using its own implementation of the certificate authority
- There can be more than one MSP in the network, one for each organization that belong to the network

Gossip Protocol:

- Allows communication between peers
- Primarily responsible for peers to be able to exchange data directly between themselves
- Constantly, the peers push out and receive ledger data between them
- This also enables the discovery of new peers in the network
- In a distributed network, there is no common list of peers and hence peers can discover the new peers using this gossip protocol
- This enables a constant visibility of the members in the network and the channels that they operate on
- Leader selection: All peers would select a peer within themselves to be the leader of the gossip process. This leader will be responsible for communicating with the ordering service. This will also take the Blocks coming from the ordering service and pass it to the other peers in the network
- Leader can be defined by static way by the admin during the network setup or it can be a dynamic process which means that the peers will decide the leader among themselves. If this peer goes down, another peer takes its position as the leader
- Anchor peer: Each organization will need to defined one of its peers as the Anchor peer. The purpose of the Anchor peer is to communicate to the Anchor peers of other organizations and exchange information

related to the peers in each of their organizations. This allows for cross communication between the organization. If a new peer joins one organization, the other organizations get to know this through the communication between the Anchor peers



To understand the gossip protocol using the example above, we have OrgA and OrgB and one Ordering Service:

- The ordering service would communicate to the organizations and the peers within them through the Leader of that organization
- Between the peers of the organization, one of them is a Leader, another the Anchor and the others are the regular peers
- Orderer would communicate to the Leader of both organizations to transfer the Blocks
- The Leaders in each organization would communicate this across to the other peers in the organization
- Anchor peers communicate between two organizations. This helps to update activities pertaining to the peers within the organizations. The Anchor peers will disseminate the information across to the other peers in the organization

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Private data collections:

In situations where some transactions on a channel needs to be shared with all the peers who have not joined the channel, some details of those transactions such as total value needs to be shared only between the channel participants. In the case of Tuna example, Aakash would like to share the sale of Tuna to KP, using channel CA, across to all the network participants. However, he would not like to share some details such as total value of the sale etc and likes to keep it only between himself and KP. This is achieved using Private Data Collections otherwise known as collections.

The difference between a channel and a collection is while a channel allows peers or a subset of peers in a network to share transactions among themselves, a collection allows sharing of data within some of the channel participants. For example within the Tuna network, channel CA is used to share data between Aakash and KP. However, to share data only between Aakash and KP but using channel CC, we can use collections.

The private data within a transaction is sent only to the required peers using gossip protocol. However, the same private data is also hashed and sent across to all the other peers on the network. This way, the proof of the private data is captured in the network at the same time not revealing the details of the transaction. Any tampering with the private data can be evidenced using this method and corrected if needed.

HYPERLEDGER FABRIC & COMPOSER
MODULE 1
HYPERLEDGER FUNDAMENTALS

Characteristics of Fabric:

1. Modularity:

   Hyperledger supports different ordering service implementations (Solo, Kafka); pluggable membership services; different databases (LevelDB, CouchDB); pluggable endorsement policies. This makes Hyperledger a very modular platform for a variety of enterprise uses
2. Permissioned Network:

   MSP within the Hyperledger allows each entity to be identified using credentials before they can join the network. It is mandatory for every network element such as peers, applications or users to have identities issued by the MSP
3. Common programming languages:

   Hyperledger supports writing smart contracts in common programming language such as NodeJS, Go, Java.

   Some blockchains like Ethereum operate under a consensus based on Order-Execute fashion. In these cases, the transactions are first ordered in a Block and shared to all the peers in the network. The peers then execute the transactions locally. This requires that the transactions be deterministic. Hence there is a need for special programming language such as Solidity to ensure determinism.

   Fabric however, works on Execute-Order-Validate model. Here the transactions are first executed on the endorsers, achieve consensus and only then sent to the orderer. If there is no consensus, the transactions are not sent to the ordering service.

   This key difference makes Fabric more versatile to use any open programming languages
4. Privary/confidentiality:

   Privacy is achieved using channels which allow for data to be privately exchanged between a subset of peers within the network. Also, using private data collections, certain set of data can be privately shared between only a few peers within the channel. This makes Hyperledger a privacy focused blockchain framework for enterprises

5. Speed, cost and scalability:

   Fabric uses a consensus protocol which does not employ mining, the cost of additional mining resources on the peers is eleminated. Hence the cost of implementing it is nearly the same as any legacy platform. There is also no incentives for mining activity and the Blocks does not have to wait for acceptance on the network. Hence the latencies are low and transactions are faster. Network is also highly scalable since any organization within the network can support any number of peer nodes