To perform a thorough study of blockchain development on Hyperledger Fabric using Composer

Practical 7

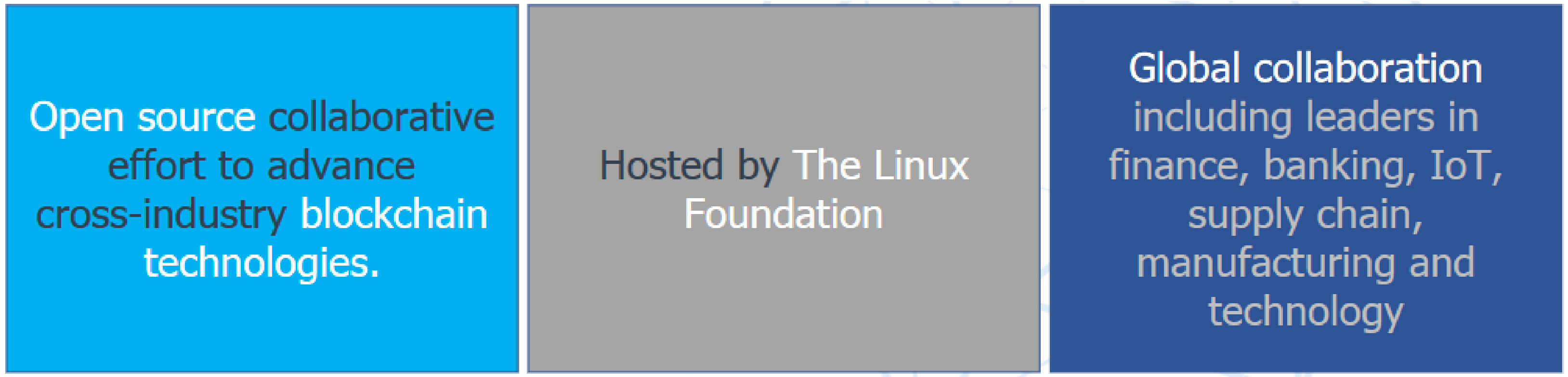
HYPERLEDGER
FABRIC

The architecture of the permissioned ledger

Why Hyperledger?

- Members of a network work together, but because businesses need some of their data to remain private, they often maintain separate relationships within their networks.
- **For example**, a purchaser may work with different sellers, selling the same product. The transactional relationship between the purchaser and each of the sellers should remain private and not visible across all sellers.
- This is made possible via **Hyperledger Fabric** if you need total transaction isolation, and the “private data” feature

Hyperledger



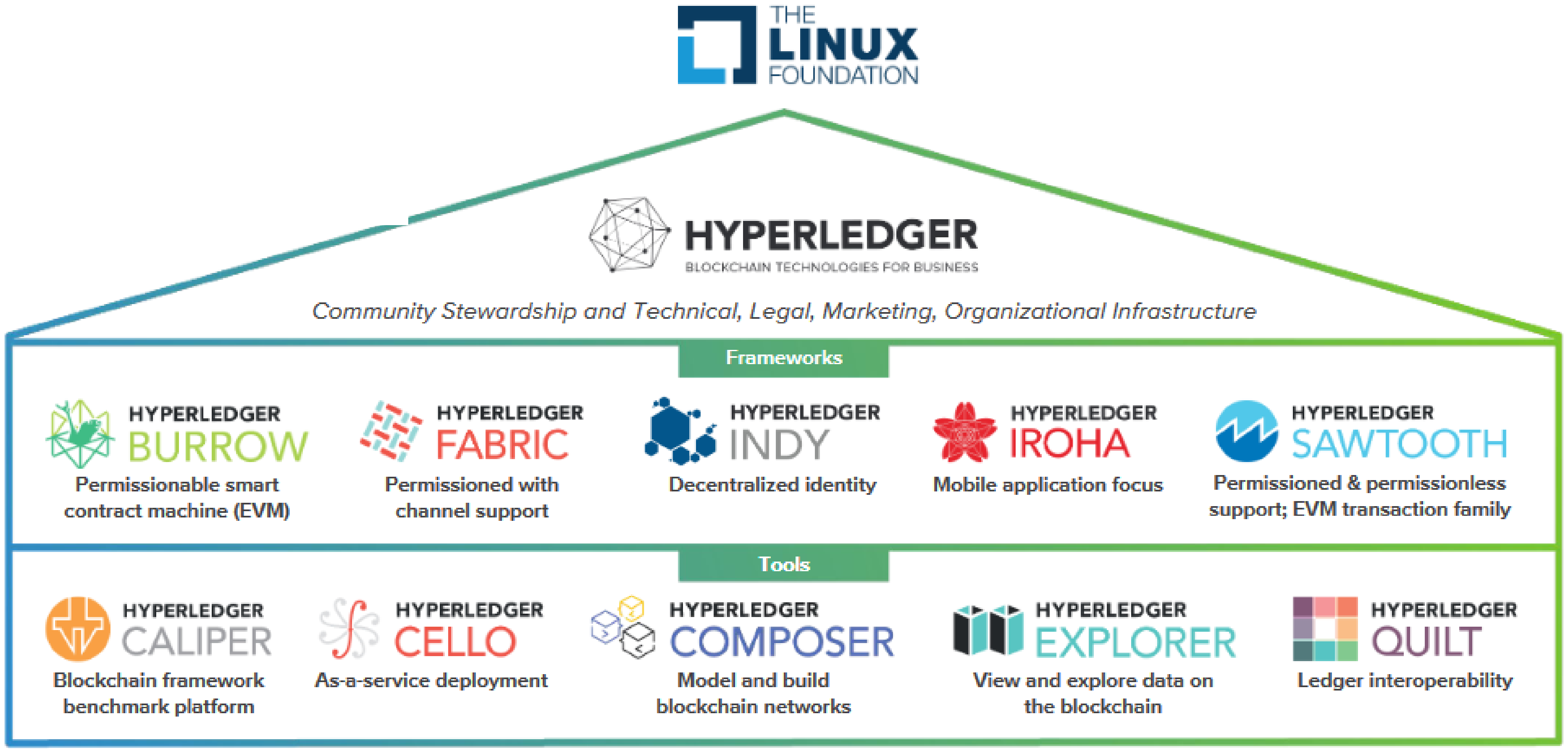
Open source collaborative effort to advance cross-industry blockchain technologies.

Hosted by The Linux Foundation

Global collaboration including leaders in finance, banking, IoT, supply chain, manufacturing and technology

- Fabric offers a scalable and secure platform that supports private transactions and confidential contracts.
- Its modular and versatile design satisfies a broad range of industry use cases.
- There is no mining, just order system do it.
- Operational power: 0.5 million operations per minute, whereas blockchain does only 1000.

Hyperledger


THE LINUX FOUNDATION
HYPERLEDGER
BLOCKCHAIN TECHNOLOGIES FOR BUSINESS
Community Stewardship and Technical, Legal, Marketing, Organizational Infrastructure
Frameworks
HYPERLEDGER BURROW
Permissionable smart contract machine (EVM)
HYPERLEDGER FABRIC
Permissioned with channel support
HYPERLEDGER INDY
Decentralized identity
HYPERLEDGER IROHA
Mobile application focus
HYPERLEDGER SAWTOOTH
Permissioned & permissionless support; EVM transaction family
Tools
HYPERLEDGER CALIPER
Blockchain framework benchmark platform
HYPERLEDGER CELLO
As-a-service deployment
HYPERLEDGER COMPOSER
Model and build blockchain networks
HYPERLEDGER EXPLORER
View and explore data on the blockchain
HYPERLEDGER QUILT
Ledger interoperability

Hyperledger

Hyperledger embraces the **full spectrum** of industry use cases, especially enterprise scenarios with widely varied requirements for *decentralization, trust, continuity and confirmation times*. Each represents a potentially unique optimization point for the technology.

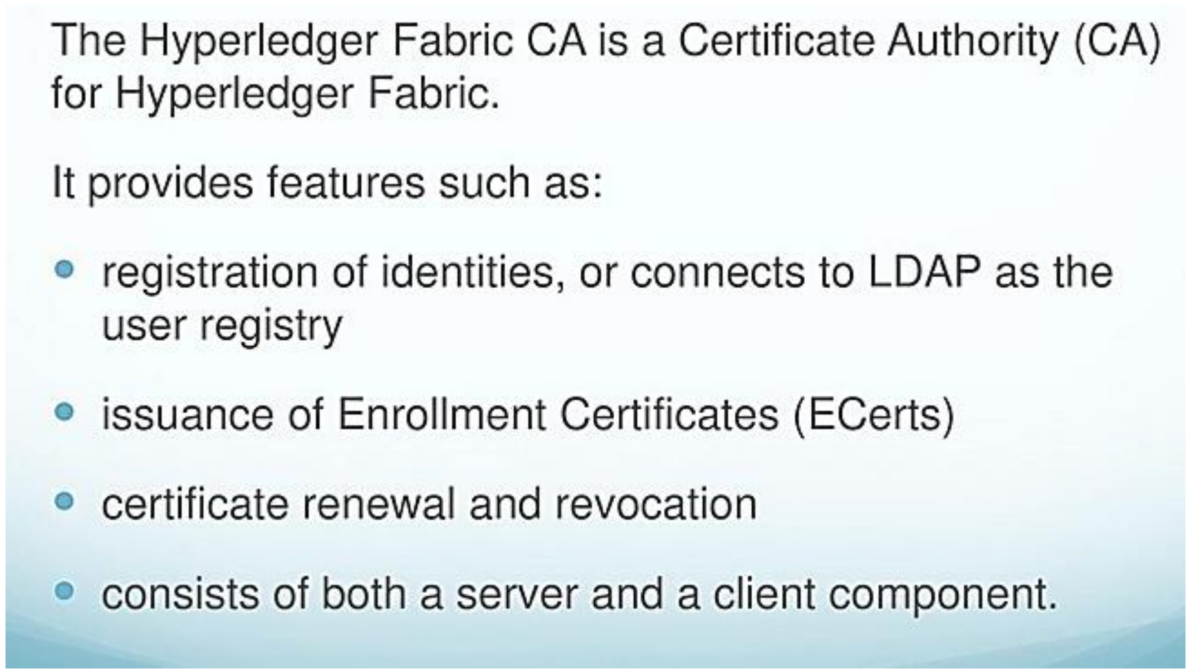
Key Features of Hyperledger

- Permissioned architecture
- Highly modular
- Pluggable consensus
- Low latency of finality/confirmation
- Flexible approach to data privacy
- Support for EVM and Solidity
- Multi-language smart contract support

Hyperledger Components

- Fabric CA
- Peer
- Ordering Service
- Channel
- Chaincode

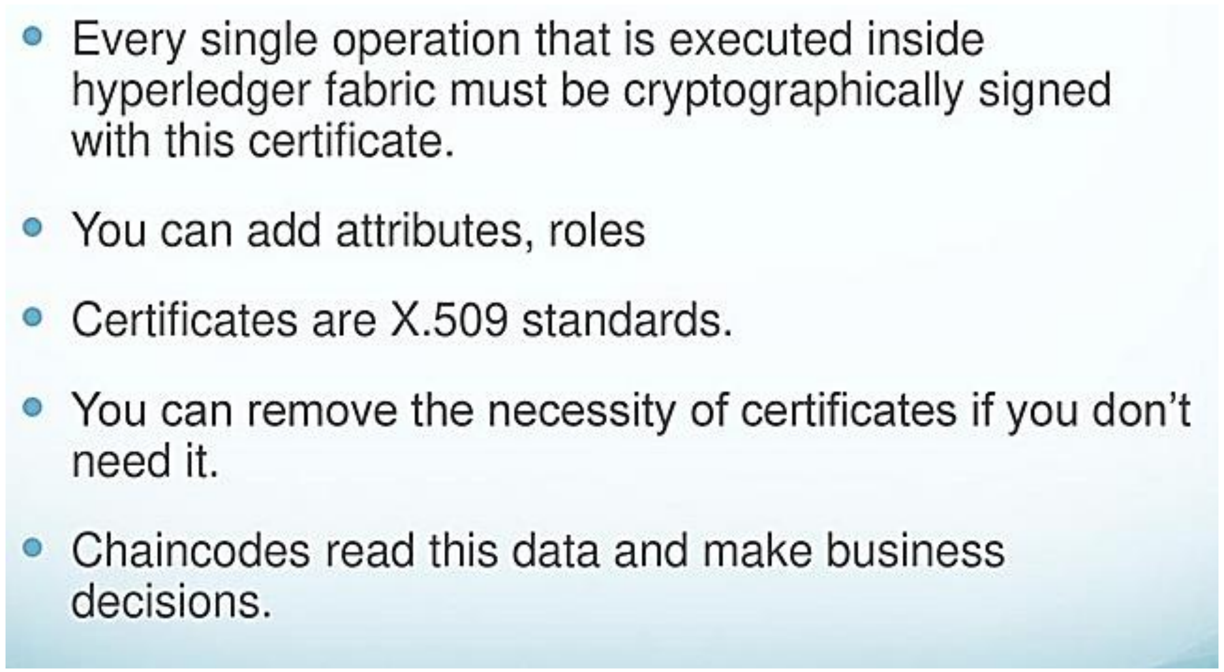
Fabric CA



The Hyperledger Fabric CA is a Certificate Authority (CA) for Hyperledger Fabric.

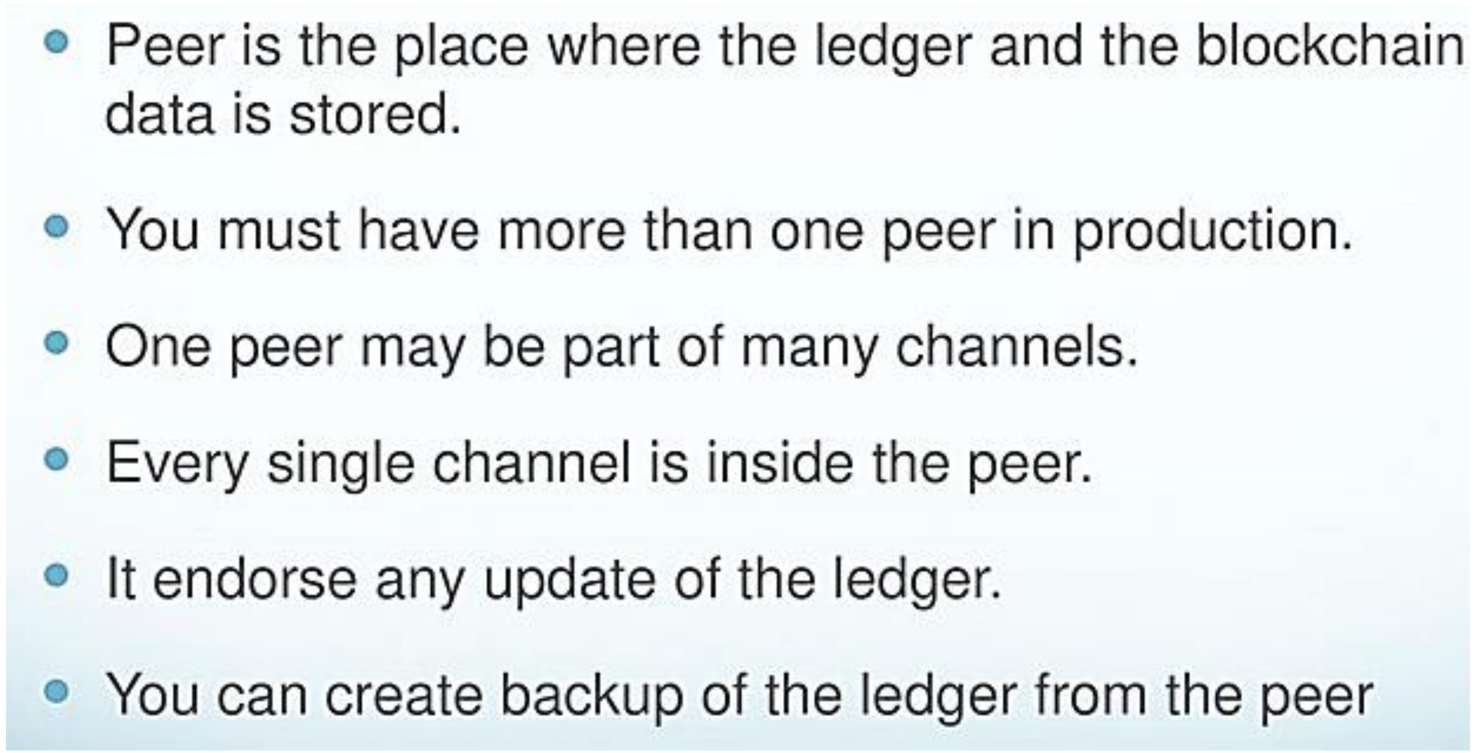
It provides features such as:

- registration of identities, or connects to LDAP as the user registry
- issuance of Enrollment Certificates (ECerts)
- certificate renewal and revocation
- consists of both a server and a client component.



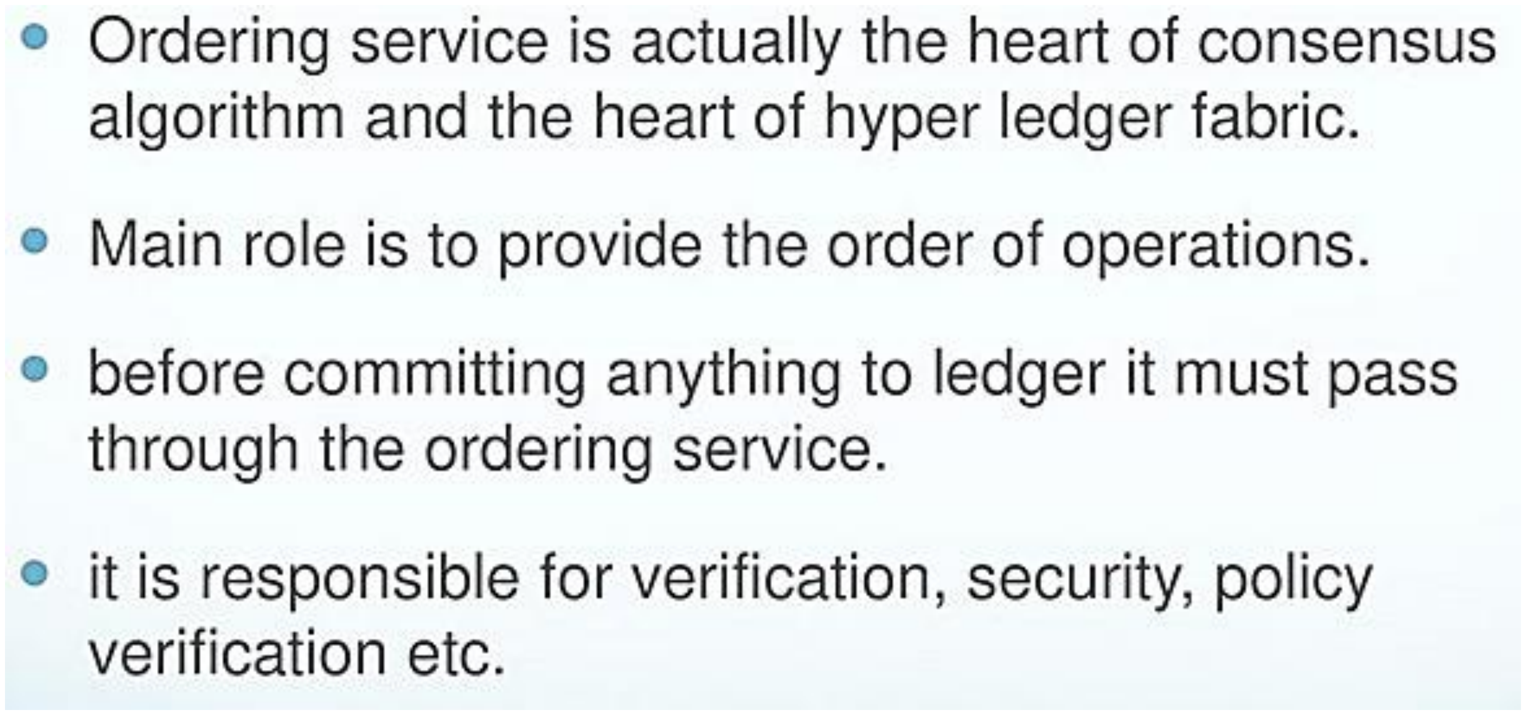
- Every single operation that is executed inside hyperledger fabric must be cryptographically signed with this certificate.
- You can add attributes, roles
- Certificates are X.509 standards.
- You can remove the necessity of certificates if you don't need it.
- Chaincodes read this data and make business decisions.

Peer



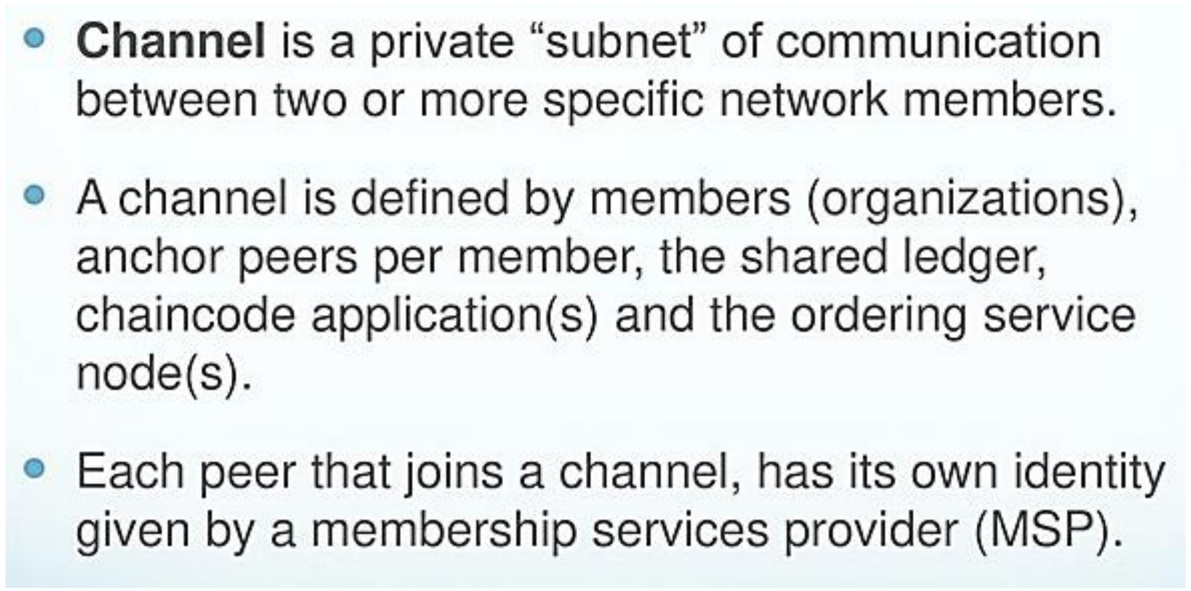
- Peer is the place where the ledger and the blockchain data is stored.
- You must have more than one peer in production.
- One peer may be part of many channels.
- Every single channel is inside the peer.
- It endorse any update of the ledger.
- You can create backup of the ledger from the peer

Ordering Service

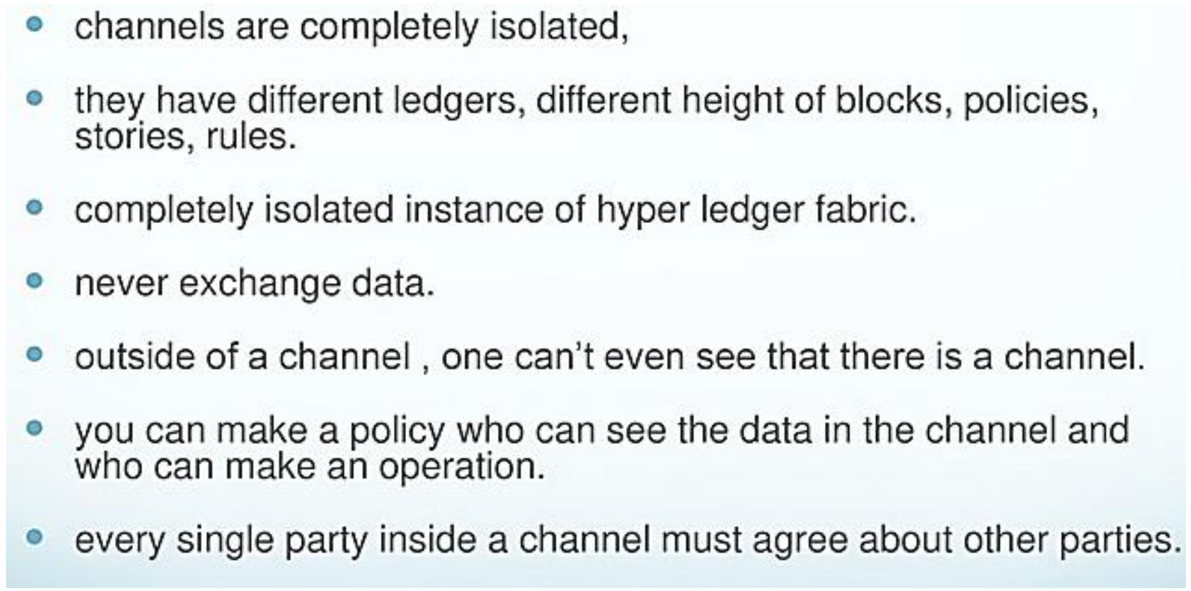


- Ordering service is actually the heart of consensus algorithm and the heart of hyper ledger fabric.
- Main role is to provide the order of operations.
- before committing anything to ledger it must pass through the ordering service.
- it is responsible for verification, security, policy verification etc.

Channel

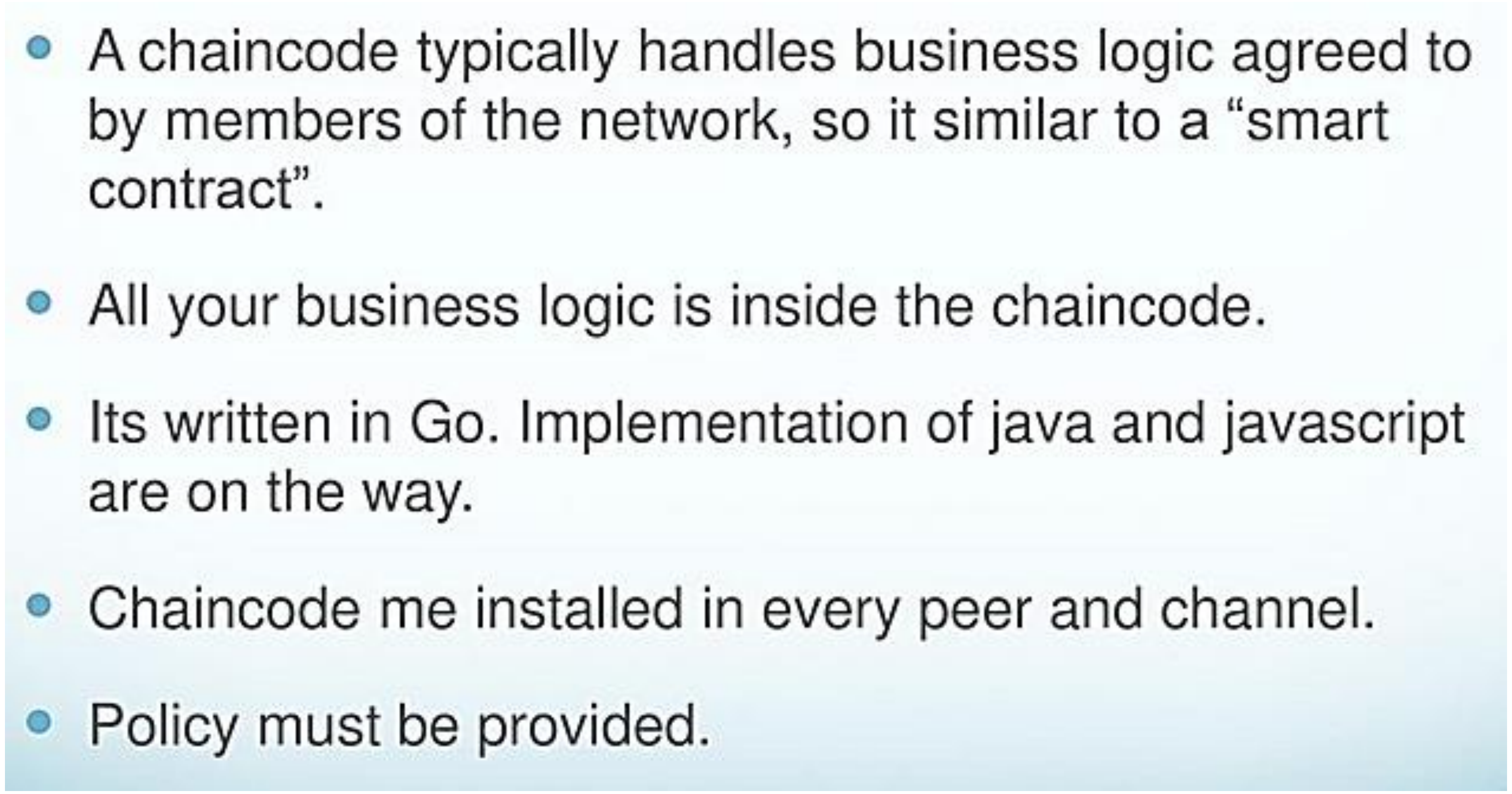


- **Channel** is a private "subnet" of communication between two or more specific network members.
- A channel is defined by members (organizations), anchor peers per member, the shared ledger, chaincode application(s) and the ordering service node(s).
- Each peer that joins a channel, has its own identity given by a membership services provider (MSP).



- channels are completely isolated,
- they have different ledgers, different height of blocks, policies, stories, rules.
- completely isolated instance of hyper ledger fabric.
- never exchange data.
- outside of a channel , one can't even see that there is a channel.
- you can make a policy who can see the data in the channel and who can make an operation.
- every single party inside a channel must agree about other parties.

Chaincode



- A chaincode typically handles business logic agreed to by members of the network, so it similar to a “smart contract”.
- All your business logic is inside the chaincode.
- Its written in Go. Implementation of java and javascript are on the way.
- Chaincode me installed in every peer and channel.
- Policy must be provided.

HYPERLEDGER FABRIC INSTALLATION

- 1. https://www.soawork.com/2020/
- 1.1 Node js installation "curl -fsSL https://deb.nodesource.com/setup_14.x | sudo -E bash -"
- 1.2 sudo apt-get install -y nodejs
- 1.3 GIT installation
- 1.4 Python installation (also install python 3 "sudo apt-get install python3")
- 2. Open GoLang website and download for linux (https://golang.org/dl/)
- 2.1 cd downloads
- 2.2 use this command "sudo tar -xvf go1.17.2.linux-amd64.tar.gz"
- 3. Install using commands mentioned in (1)
- 4. Install upto lib tools mentioned in (1)
- 5. Install docker from "https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-18-04" upto Step 4 "docker run hello-world" this command.
- 6. Install Docker compose installation and hyperledger fabric in (1).

Install Fabric Test-Network

1. Open this link "https://hyperledger-fabric.readthedocs.io/en/release-2.2/test_network.html"
2. **curl -sSL https://bit.ly/2ysbOFE | bash -s -- 2.2.2 1.4.9**
3. **cd fabric-samples/test-network**
4. From inside the test-network directory, run the following command to remove any containers or artifacts from any previous runs: "**./network.sh down**"
5. You can then bring up the network by issuing the following command. You will experience problems if you try to run the script from another directory: "**./network.sh up**"
6. Examine the components of the test network "**docker ps -a**"

Deploy Smart contract

1. open this “https://github.com/hyperledger/fabric-samples/blob/main/token-erc-20/README.md”
2. ./network.sh up createChannel -ca
3. ./network.sh deployCC -ccn token_erc20 -ccp ../token-erc-20/chaincode-javascript/ -ccl javascript