#ThinkVelocity

zensar
An RPG Company

Admin Guide & FAQs

Table of Contents

1. Add a new organization

To add a new organization to an existing channel in Hyperledger Fabric 2.2 version, you can follow the below steps:

1) Prepare new Org artifacts
2) Update the network to include new Org configuration
3) Bring up Peers of new Org and join my channel
4) Chaincode operations after new Org is part of the network

Can they use their own Certificate Authority (CA)?

Yes, the new organization can use their own Certificate Authority (CA) in Hyperledger Fabric. In fact, using an external CA for an organization is a best practice in Hyperledger Fabric.

1) When a new organization joins the network, it typically generates its own cryptographic material, including certificates and private keys, using a CA. This ensures that the organization has complete control over its own cryptographic material and that the CA is trusted only by the organization.
2) To use an external CA, the organization can set up its own Fabric-CA server or use a third-party CA solution that supports the Fabric CA protocol. The organization can then use the Fabric-CA client to generate cryptographic material for its peers and other components and submit enrollment requests to the external CA.
3) To add the new organization to the existing channel using its own CA, you need to make sure that the channel configuration is updated to include the root certificate of the new organization's CA.
4) The root certificate of the new organization's CA should be added to the MSP configuration of the channel, along with the root certificates of the other organizations. This will ensure that the channel trusts the new organization's certificates and can authenticate its transactions.

The difference between the Endorsement policy and the lifecycle endorsement policy:

In Hyperledger Fabric, there are two types of endorsement policies: Endorsement and Lifecycle Endorsement. The Endorsement policy determines the endorsement policy for invoking chain-code operations. It specifies the minimum number of peers that must agree on the proposed transaction before the transaction is considered valid and can be committed to the ledger. On the other hand, the Lifecycle Endorsement policy determines the endorsement policy for committing chain-code definition updates to the ledger. It specifies the minimum number of peers that must agree on the new chain-code definition before it can be committed to the ledger. These two policies have different purposes and are used in different phases of the chain-code lifecycle. The Endorsement policy is used when invoking chain-code operations, while the Lifecycle Endorsement policy is used when committing chain-code definition updates to the ledger.

2. Add a new peer

To add a new peer to an existing org in hyper ledger fabric we need to,

Step 1: generate crypto material for peer

Step 2: create a docker compose file containing this peer

Step 3: join this new peer to the existing channel

Step 4: install chaincode to that peer

1) Generate cryptographic material for the new peer. This includes generating a new private key and obtaining a digital certificate signed by the organization's Certificate Authority (CA). You can use the Fabric CA client tool to do this.
2) Configure the new peer with the necessary configuration files. This includes creating the peer's MSP (Membership Service Provider) directory and updating the core.yaml configuration file with the new peer's information.
3) Update the channel configuration to include the new peer. You will need to update the channel's configuration block with the new peer's information and then submit the updated block to the ordering service for endorsement.
4) Start the new peer by running the peer binary with the appropriate configuration files. After these steps have been completed, the new peer should be up and running and ready to participate in the network.

3. Remove an Org

Removing an organization from a Hyperledger Fabric network can be a complex and involved process, and it depends on several factors, such as the network topology, the consensus algorithm used, and the membership services provider (MSP) configuration. However, the general process involves the following steps:

1) Remove the organization's peers: Before removing the organization, all of its peers should be removed from the channels they are joined to, and their identities should be revoked from the MSP.
2) Update the channel configuration: After removing the organization's peers, the channel configuration should be updated to remove any references to the organization and its peers.
3) Update the anchor peers: If the organization has any anchor peers, they should be updated to remove any references to the organization.
4) Update the consortium configuration: If the organization is a member of a consortium, the consortium configuration should be updated to remove the organization.
5) Update the orderer configuration: If the network uses a Raft or Kafka ordering service, the configuration should be updated to remove the organization's endorsement policy.
6) Update the MSP configuration: The MSP configuration should be updated to remove the organization's root certificate and any intermediate certificates.
7) Remove the organization's certificates: Finally, the organization's certificates and keys should be removed from the local filesystem and any backup locations to ensure that they cannot be used to access the network.

It's important to note that removing an organization from a Hyperledger Fabric network can have significant impacts on the network's performance and stability. As such, it's essential to carefully plan and execute the removal process to prevent any potential disruptions to the network.

4. Cross chain-code invocation among two different channels

Cross-chain-code invocation (also known as chain-code-to-chain-code invocation) in Hyperledger Fabric allows one chain-code to invoke another chain-code on the same or a different channel within the same Fabric network. This capability enables complex business logic to be split into multiple chain-codes, each responsible for a specific set of functionalities, which can be easily managed, updated, and versioned.

To invoke another chain-code, the invoking chain-code uses the Invoke Chain-code function **provided by the Fabric SDK.** This function takes the name of the chain-code to be invoked, the arguments to be passed to the chain-code, and the channel on which the chain-code is deployed. The invoked chain-code must also be deployed on the same Fabric network and accessible to the invoking chain-code.

For chain-code to chain-code interactions using the invokeChaincode() API, both chain-codes must be installed on the same peer.

For interactions that only require the called chain-code's world state to be queried, the invocation can be in a different channel to the caller's chain-code.

For interactions that require the called chain-code's world state to be updated, the invocation must be in the same channel as the caller's chain-code.

5. Upgrade a chain-code i.e., who & how should endorsement work for upgrading a chain-code? How does a signature policy for chain-code can be changed.

Upgrading chain-code

1) Changing endorsement policy after a chain-code is deployed requires a change in chain-code definition.
2) This does not involve a new chain-code and a new chain-code package as the same source code is being used. The approval process is still governed by lifecycle chaincode endorsement policy.
3) This process does not have impact on the existing state already in the ledger. Upgrading application chain-code requires a complete process of chain-code operation, from packaging to commit chain-code definition.
4) As far as the chain-code definition keeps the same name but a different version, this is considered a chain-code upgrade, and the existing state in the ledger is kept.
5) If a different name is used, this is a complete separate deployment of application chain-code, even though one is using the same chain-code package.

6. Can we use External API call from Chaincode

- In general, it is not common to directly call external APIs from Chaincode, also known as Smart Contracts, within a blockchain network. This is because Chaincode is intended to be executed in a deterministic and decentralized manner within the blockchain network.

- In some cases, it may be necessary or desirable to use external APIs within the context of a smart contract or Chaincode in a blockchain network. However, it's important to carefully consider the implications and follow best practices to ensure secure and reliable integration of external APIs with Chaincode.

Here are some considerations to keep in mind:

- Use Case: Evaluate whether the use of external APIs is truly necessary for smart contract logic. In some cases, it may be possible to design the smart contract logic to operate solely within the blockchain network without needing to call external APIs.
- Security: Ensure that proper authentication, authorization, and data validation mechanisms are in place to secure communication between the Chaincode and the external API. Use HTTPS or other secure communication protocols to protect the integrity and confidentiality of the data exchanged.

- Privacy: Consider the privacy implications of using external APIs, as they may expose sensitive data to external systems. Handle sensitive data appropriately and in accordance with privacy requirements.
- Scalability: Consider the volume and frequency of external API calls, as they may impact the scalability and performance of the blockchain network. Optimize and manage external API calls to minimize their impact on the overall performance of the blockchain network.
- Dependencies: Be aware of potential dependencies on external APIs, as they may introduce risks related to availability, reliability, and versioning. Carefully manage and monitor external APIs to mitigate these risks.

Steps to do this external API call in Hyperledger

Integrating external API calls in Hyperledger Fabric, which is a popular blockchain framework, involves several steps. Here's a high-level overview of the typical steps:

- Identify the External API: Determine the external API that you want to call from your Chaincode. This could be an API provided by a third-party service or an internal API within your organization.
- Define the API Interface: Define the interface for interacting with the external API from your Chaincode. This may include specifying the API endpoint, request format, and expected response format.
- Implement the API Integration Logic: Write the code to invoke the external API from your Chaincode.
- Test the API Integration: Test the API integration within your Chaincode to ensure that it is functioning as expected.
- Deploy and Execute the Chaincode: Deploy the Chaincode containing the API integration logic to your Hyperledger Fabric network.
- You may need to package and install the Chaincode as per the Hyperledger Fabric's deployment process. Once deployed, you can execute the Chaincode through transactions from client applications, which can trigger the API calls to the external system.
- Regularly review and update the Chaincode and API integration logic as needed, considering any changes in the external API, security requirements, or regulatory compliance.
- It's important to follow best practices for secure development, proper authentication, authorization, error handling, and data privacy when integrating external APIs in Hyperledger Fabric Chaincode.

References:

Hyperledger Fabric documentation: https://hyperledger-fabric.readthedocs.io/en/release-2.2/

Fabric configuration documentation: https://hyperledger-fabric.readthedocs.io/en/release-2.2/configtx.html

Adding an organization to a channel: https://hyperledger-fabric.readthedocs.io/en/release-2.2/channel_update_tutorial.html

Modifying the ChannelCreationPolicy: https://hyperledger-fabric.readthedocs.io/en/release-2.2/configtx.html#modifying-the-channelcreationpolicy

Configtx.yaml file reference: https://hyperledger-fabric.readthedocs.io/en/release-2.2/configtx.html#configtx-yaml-file-reference

Endorsement policies: https://hyperledger-fabric.readthedocs.io/en/release-2.2/endorsement-policies.html

Chaincode lifecycle: https://hyperledger-fabric.readthedocs.io/en/release-2.2/chaincode_lifecycle.html

How to remove a peer from the org:
https://hyperledger-fabric.readthedocs.io/en/release-2.3/channel_update_tutorial.html#deleting-a-peer-from-an-organization

external API call in Hyperledger:

https://lists.hyperledger.org/g/fabric/topic/calling_external_api_from/17549735

https://hyperledger.github.io/composer/v0.19/integrating/call-out

https://www.edureka.co/community/30808/accessing-external-api-in-hyperledger-fabric