LASARIS LAB OF SOFTWARE ARCHITECTURES AND INFORMATION SYSTEMS

FACULTY OF INFORMATICS Masaryk University

Hyperledger Fabric

Bacem Mbarek, Ph.D.
Masaryk University,
Lasaris Lab (Lab of Software Architectures and Information Systems)

Hyperledger fabric first network tutorial

Installing Hyperledger Fabric:

<u>See Tutorial</u>

This is the file or directory list that we will study:

1) docker-compose-cli.yaml
2) Configtx.yaml
3) Crypto-config.yaml
4) byfn.sh
5) Channel-artifacts

docker-compose-cli.yaml

In order to develop Blockchain application we need to have a blockchain network first, right ?, This is a file related to setting up the network.
This is a Docker compose file which defines your (virtual) Fabric network, such as that nodes are in the network, their internal use domain names, order of Blocks and transactions, network to witch belong the nodes, etc.

2) Configtx.yaml

This is a file to define the configurations of the Blockchain network, though later they could be updated.

3) ***Crypto-config.yaml***

This is a file to define certifications and keys to be generated and used in the network.

In hyperledger Fabric, we have a set of certifications and keys for users and nodes, For example, peer needs to have a set of certeficates and keys to perform endorsement, prove itself as a member in the Blockchain network do signings, etc.

byfn.sh

This is a convenient script file by hyperledger Fabric to start the network, We use it this way to bring the network 9we will run tohether later).

Sudo ./bufn.sh up

1) Generate certeficates and keys based on crypto-config.yaml
2) Generate channel artifacts based on confligtx.yaml, outputs will be stored in channel-artifacts?
3) Bring up the (virtual) Fabric network based on docker-compos-cli.yaml
4) Create hyperledger Fabric channel, mychannel
5) Join peer nodes into the channel
6) Install the smart contract (chaincode) in the peer nodes.
7) Instantiate the smart contract (chincode) in one of the peer nodes

Channel

- Channel in hyperledger Fabric is a private group in Blockchain network.

- Each blockchain network could contain multiple channels, where each channel is infepependent to other channels, has its own ledger and contains multiple organizations.

What is chaincode?

Chaincode is a piece of code that is written in one of the supported languages such as Go or Java. It is installed and instantiated through an SDK or CLI onto a network of Hyperledger Fabric peer nodes, enabling interaction with that network's shared ledger.

Chaincode is a program (smart contract) that is written to read and update the ledger state. All the business logic is inside the chaincode.

In Hyperledger Fabric, **chaincode** is the ‘smart contract’ that runs on the peers and creates transactions. More broadly, it enables users to create transactions in the Hyperledger Fabric network’s shared ledger and update the world state of the assets.

So in the previous chapter we had installed Hyperledger Fabric and we started and stopped a test network on which peers and chain codes run. Now let’s dive deep into the chaincode portion of Hyperledger Fabric as that’s where business logic lies.

Here, we write the Blockchain genesis block, create the first channel transaction, and write anchor peer updates.

You may not care how exactly it is done, but this is how Fabric is built from the bottom up,

You can see that four new files are generated and stored in the channel-artifacts directory:

(1) genesis,block
(2) Channel.tx
(3) org1MSPanchors.tx
(4) org2MSPanchors.tx

(1) Genesis block: A genesis Block is the first block of a blockchain. Modern versions of Bitcoin number it as block 0, though very early versions counted ad block1. The genesis block is almost always hardcoded into the software of the applications that utilize its blockcian.

(2) The first (genesis) block in a channel, is a **channel.tx** (channel configuration, and the consortium that is allowed to use the channel,and the consortium that is allowed to use the channel,

We will go through a Sample chaincode and understand the following

- Dependencies
- Struct
- Init Method
- Invoke Method (Get & Set)
- Main Function

Right now, it can be written in Go / Node.JS.

Overview of a Chaincode Program

When creating a chaincode, there are two methods that you will need to implement:

•**Init**

Called when a chaincode receives an ***instantiate*** or ***upgrade*** transaction. This is where you will initialize any application state.

•**Invoke**

Called when the ***invoke*** transaction is received to process any transaction proposals.

As a developer, one must create both an **Init** and an **Invoke** method within your chaincode. The chaincode must be installed using the **peer chaincode install** command, and instantiated using the **peer chaincode instantiate** command before the chaincode can be invoked. Then, transactions can be created using the **peer chaincode invoke** or **peer chaincode query** commands.

<u>Initialisation of peers values</u>

{“Arg”: [‘’init,” “a”, “100”, “b”, “200”]}’

We set “a” or the first peer we created as having 100 tpkens, and “b”, the second peer as having 200 tokens

<u>Submission of data</u>

As expected we see that “a” has 100 tokens, Now let’s send 10 tokens from “a” to “b”

{“Arg”: [‘’invoke,” “a”, “b”, “10”]}’

To check value of peer

{“Arg”: [‘’Args,”: “query”, “a”]}’

- Quick High Level Overview:
- Peer can be part of one or more channel
- Every channel has a separate ledgerEvery Channel has one or more chain codes
- Every Chain code has a different endorsement policy
- Chaincode must be part of a channel. As the ledger is part of a channel. One channel can have as many chaincodes as possible.
- Chaincode must be installed in each peer that is part of the channel and instantiated.
- When a Chaincode gets instantiated a policy (endorsing) has to be defined. [consensus: before a transaction can be recorded in the ledger only if a rule is met]