1) Steps for Installing and Instantiating the Chaincode on HLF 2.2

Commands for installing and running Fabric:

- Download the script file `install-fabric.sh` from the Hyperledger Fabric GitHub repository. Make it executable using `chmod +x.`

curl -sSLO https://raw.githubusercontent.com/hyperledger/fabric/main/scripts/install-fabric.sh && chmod +x install-fabric.sh

- Install Hyperledger Fabric version 2.2.2 and Fabric-CA version 1.4.9:

./install-fabric.sh -f '2.2.2' -c '1.4.9'

- Navigate to the `fabric-samples` folder, which contains the files needed for the sample network setup:

cd fabric-samples/test-network/

- Start the network and create a channel for communication between organizations using the `cryptogen` tool:

./network.sh up createChannel

- Deploy the basic chaincode to the channel, specifying its language and path:

./network.sh deployCC -ccn basic -ccp ../asset-transfer-basic/chaincode-javascript -ccl javascript

- Set up environment variables for paths and configurations:

export PATH=${PWD}/../bin:$PATH
export FABRIC_CFG_PATH=$PWD/../config/
export CORE_PEER_TLS_ENABLED=true
export CORE_PEER_LOCALMSPID="Org1MSP"
export CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt

export CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org1.example.com/users/Admin@org1.example.com/msp
export CORE_PEER_ADDRESS=localhost:7051

- Initialize the ledger and add assets by invoking the `InitLedger` function:

peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddresses localhost:7051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"InitLedger","Args":[]}'

- Retrieve all assets using the `GetAllAssets` function:

peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'

- Stop the network:

./network.sh down

---

2) Explain Cryptogen and Configtxgen

Cryptogen

Cryptogen is a command-line tool provided by Hyperledger Fabric to generate cryptographic materials necessary for setting up a Fabric network. These include private keys, certificates, and Membership Service Provider (MSP) files. Cryptogen simplifies the creation of network identities for participants, including organizations, peers, and orderers.

Key features:

- Uses the `crypto-config.yaml` file to define the number of peers, orderers, organizations, and users.
- Creates cryptographic artifacts by running the `cryptogen generate` command.

This ensures secure communication and proper identity management within the network.

Configtxgen

Configtxgen is a configuration transaction generator tool used in Hyperledger Fabric for creating configuration artifacts necessary to set up and manage the network. These artifacts define the network structure, policies, and operational behavior.

Key functionalities:

- Generates the **genesis block**, which bootstraps the network:

configtxgen -profile ChannelUsingRaft -outputBlock ./channel-artifacts/channel1.block -channelID channel1

- The `-profile` flag refers to the `ChannelUsingRaft` profile from the `configtx.yaml` file.
- The `-outputBlock` specifies the file where the genesis block is written.
- Creates channel configuration transactions and anchor peer updates.
- Defines channel profiles in the `configtx.yaml` file for specific configurations.

This tool streamlines the setup of essential configuration artifacts, facilitating smooth deployment and management of a Hyperledger Fabric network.