Client Application

What we have learned

- Defined a Network
- Bootstrapped a Network
- Wrote a Chaincode
- Deployed a Chaincode

The Blackbox

"All the things that we have done so far can be described as the **blackbox** part of the application"

Chaincode Interaction

There are a couple of ways to interact with chaincode:

- Using standard Fabric CLI (Developer way)

- Using Hyperledger Fabric Client SDK (User way)

Fabric CLI

- **Peer chaincode invoke**

peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.auto.com --tls --cafile $ORDERER_CA -C $CHANNEL_NAME -n KBA-Automobile --peerAddresses localhost:7051 --tlsRootCertFiles $MANUFACTURER_PEER_TLSROOTCERT --peerAddresses localhost:9051 --tlsRootCertFiles $DEALER_PEER_TLSROOTCERT --peerAddresses localhost:11051 --tlsRootCertFiles $MVD_PEER_TLSROOTCERT -c '{"function":"CreateCar","Args":["Car-101", "Tata", "Nexon", "White", "Factory-1", "22/07/2024"]}'

- **Peer chaincode query**

peer chaincode query -C $CHANNEL_NAME -n KBA-Automobile -c '{"Args":["GetAllCars"]}'

Issues with CLI

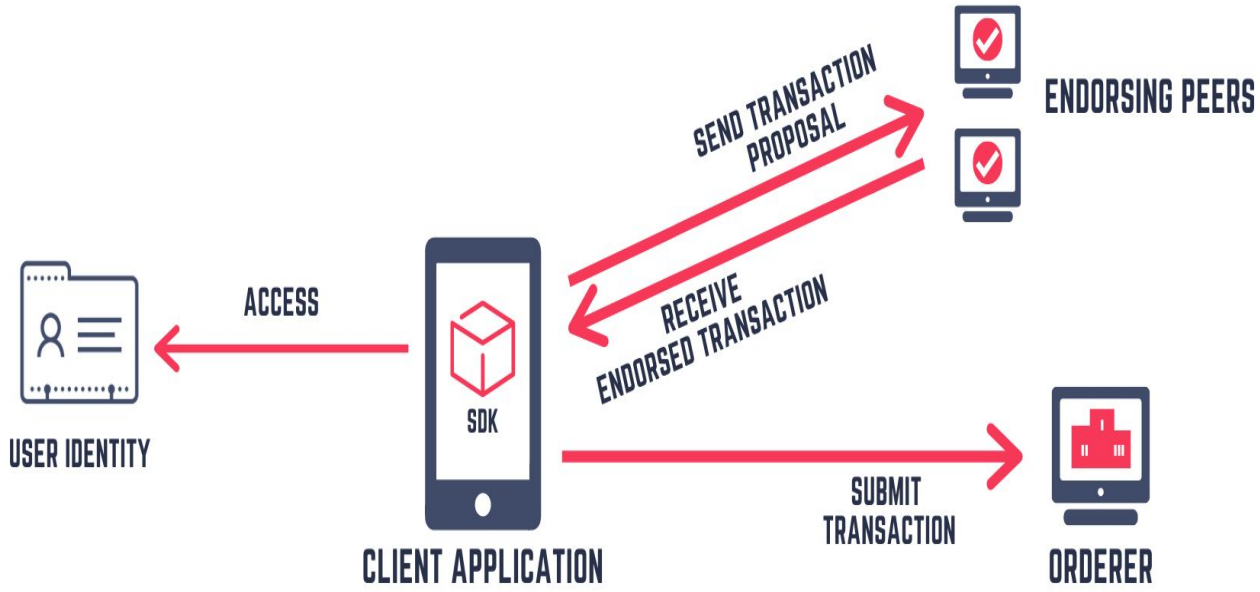
- Using a CLI to connect with the network can be quite hard on the user
- The size of the Commands tends to increase gives chances for errors
- Also, the user should remember all the commands that are required to interact with the network

Building the Client Application

- We use the client applications to submit transactions to the fabric network.
- They provide easier interfaces that user can use to interact with the network

Duties of Client SDK

- Providing Identification Material
- Connecting to the Network
- Submitting the transaction
- Processing the Response



Connection Profile

- A connection profile describes a set of components including peers, orderers and certificate authorities in a Hyperledger Fabric blockchain network.

- It is usually written in YAML or JSON file

- Essential for client using Fabric version prior v2.4

Fabric Gateway

- Service running on peer nodes that manages transaction submission and processing for client applications
- Introduced in Hyperledger Fabric v2.4 peers
- Provides a simplified, minimal API for submitting transactions to a Fabric network.
- Client application can simply delegate transaction submission to a trusted peer
- Manages the collection of transaction endorsements and submission to the ordering service
- Determines what endorsements are required for a given transaction

Client application APIs

Fabric Gateway client API supports developing applications in

- Go
- Node (Typescript/Javascript)
- Java

API uses the Gateway peer capability introduced in Fabric v2.4 to interact with the Fabric network

Steps to build Client Application

- Establish a gRPC connection to the Fabric Gateway
  - Fabric Gateway's endpoint address
  - TLS certificates
- Create a Gateway connection
  - gRPC connection
  - Client identity
  - Signing implementation
- Access the smart contract to be invoked
  - Access the Network
  - Specify the smart contract
- Submit the transaction

Fabric Gateway Module

github.com/hyperledger/fabric-gateway

newGrpcConnection(tlsCertPath, gatewayPeer, peerEndpoint)

```
client.Connect(id, client.WithSign(sign),
        client.WithClientConnection(clientConnection),)
```

gateway.GetNetwork(channelName)

network.GetContractWithName(chaincodeName, contractName)

```
contract.SubmitTransaction(args);
contract.EvaluateTransaction(args);
```

THANK YOU