Case Study on Hyperledger Fabric

The steps to create a Blockchain network with 1 organization and 1 peer per organization, create a channel with the name "mychannel", and implement chaincode for the following functions:

* Create / Record a vehicle
* Query a vehicle
* Query all vehicles
* Change the owner of the vehicle

These steps assume that you have already installed Hyperledger Fabric and set up the network.

1. Implement the chaincode Create a new file named **vehicle.go** with the following chaincode implementation:

package main

import (

"fmt"

"github.com/hyperledger/fabric/core/chaincode/shim"

"github.com/hyperledger/fabric/protos/peer"

)

type VehicleChaincode struct {

}

type Vehicle struct {

VIN string `json:"vin"`

Make string `json:"make"`

Model string `json:"model"`

Year string `json:"year"`

Owner string `json:"owner"`

}

func (t \*VehicleChaincode) Init(stub shim.ChaincodeStubInterface) peer.Response {

return shim.Success(nil)

}

func (t \*VehicleChaincode) Invoke(stub shim.ChaincodeStubInterface) peer.Response {

fn, args := stub.GetFunctionAndParameters()

if fn == "createVehicle" {

return t.createVehicle(stub, args)

} else if fn == "queryVehicle" {

return t.queryVehicle(stub, args)

} else if fn == "queryAllVehicles" {

return t.queryAllVehicles(stub)

} else if fn == "changeVehicleOwner" {

return t.changeVehicleOwner(stub, args)

}

return shim.Error("Invalid function name")

}

func (t \*VehicleChaincode) createVehicle(stub shim.ChaincodeStubInterface, args []string) peer.Response {

if len(args) != 4 {

return shim.Error("Incorrect number of arguments. Expecting 4")

}

vin := args[0]

make := args[1]

model := args[2]

year := args[3]

vehicle := Vehicle{VIN: vin, Make: make, Model: model, Year: year, Owner: ""}

vehicleAsBytes, \_ := json.Marshal(vehicle)

stub.PutState(vin, vehicleAsBytes)

return shim.Success(nil)

}

func (t \*VehicleChaincode) queryVehicle(stub shim.ChaincodeStubInterface, args []string) peer.Response {

if len(args) != 1 {

return shim.Error("Incorrect number of arguments. Expecting 1")

}

vin := args[0]

vehicleAsBytes, \_ := stub.GetState(vin)

return shim.Success(vehicleAsBytes)

}

func (t \*VehicleChaincode) queryAllVehicles(stub shim.ChaincodeStubInterface) peer.Response {





