Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра информатики

Отчет по предмету:

«Технология блокчейн»

По лабораторной работе №3

«Build your own network»

Выполнил: Зюсько Кирилл Дмитриевич

магистрант кафедры информатики

группы №858642

Проверил: Прудник Александр Михайлович

доцент, кандидат технических наук

Минск 2020

**Оглавление**

[1 Цель работы](#_vs5uoaxwhcth) 2

[2 Ход работы](#_sfyyx6qpw1mr) 3

[2.1 Выбор и подготовка среды](#_wam8b8ljkthq) 3

[2.2 Запуск и обзор сети. Установка и вызов chaincode](#_7t4ffogpglfo) 3

[2.3 Авторизация в сети](#_9jlq7w3rxd16) 6

[Вывод](#_yrvexwyxdbtp) 8

# 1 Цель работы

Создать тестовую сеть с Hyperledger Fabric.

# 2 Ход работы

## 2.1 Выбор и подготовка среды

Для выполнения работы нужно установить Go версии 1.13.x. После этого установим новые требуемые компоненты Hyperledger Fabric:

curl -sSL https://bit.ly/2ysbOFE | bash -s

Здесь я хочу отметить, что я проверил обе версии (2.0.0 и 1.4.6), но они не заработали, также chaincode, который описывается в работе отсутствовал. Ввиду этих обстоятельств, а также того, что в введение есть ссылка на новую статью – я решил выполнить шаги согласно новому туториалу.

## 2.2 Запуск и обзор сети. Установка и вызов chaincode

Проверим работоспособность сети:

./network.sh up

Данный скрипт создает две ноды, одна из которых это участник сети (каждый в своей организации), их identities (с помощью инструмента cryptogen) и genesis блок (для системного канала с помощью configtxgen). createChannel также создает канал для связи между участниками и добавляет в него созданных участников обоих организаций.

В тестовой сети находятся две ноды участников и одна нода по обработке транзакций. Каждый пользователь сети Fabric должен быть частью организации (несколько организаций вместе представляют собой консорциум). В данном примере две организации: Org1 и Org2. Ordering сервис формирует блоки из одобренных транзакций и отправляет их к участникам, коммитающим изменения (у каждого участника копия блокчейна). Каждый участник валидирует транзакции, исполняет смарт контракты. В статье два участника: один из организации Org1 - peer0.org1.example.com, второй Org2 - peer0.org2.example.com. Ordering сервис состоит из одной ноды orderer.example.com (в идеале ordering сервис должен состоять из нескольких нод, которые принимают решение о порядке транзакций с помощью алгоритма консенсуса Raft).

Организации могут создавать приватные каналы связи, для каждого канала отдельный ledger, доступ к каналу имеют только организации сети, получившие инвайты на джойн. В тестовой сети канал создается следующим образом:

./network.sh createChannel -c channel1

Использование сети происходит следующим образом: участники канала вызывают chaincode из смарт контрактов. Chaincode оперирует ассетами. Ключевая часть блокчейна: определенные участники должны провалидировать транзакцию. Эти участники указаны в полиси канала.

Chaincode устанавливается сначала на участников организации, а потом и на канал, после чего его можно использовать. Опять же: именно когда организации из полиси принимают соглашение, только тогда chaincode может быть использован.

В нашей тестовой сети установить chaincode можно с помощью вызова:

./network.sh deployCC

Chaincode сначала будет установлен на оба участника организации, а затем и на канал (по умолчанию mychannel).

Убедимся, что chaincode работоспособен. Для начала, установим несколько переменных среды, чтобы вызывать chaincode от имени участника организации Org1:

export PATH=${PWD}/../bin:${PWD}:$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

Проверим chaincode:

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

[{"Key":"CAR0", "Record":{"make":"Toyota","model":"Prius","colour":"blue","owner":"Tomoko"}},{"Key":"CAR1", "Record":{"make":"Ford","model":"Mustang","colour":"red","owner":"Brad"}},{"Key":"CAR2", "Record":{"make":"Hyundai","model":"Tucson","colour":"green","owner":"Jin Soo"}},{"Key":"CAR3", "Record":{"make":"Volkswagen","model":"Passat","colour":"yellow","owner":"Max"}},{"Key":"CAR4", "Record":{"make":"Tesla","model":"S","colour":"black","owner":"Adriana"}},{"Key":"CAR5", "Record":{"make":"Peugeot","model":"205","colour":"purple","owner":"Michel"}},{"Key":"CAR6", "Record":{"make":"Chery","model":"S22L","colour":"white","owner":"Aarav"}},{"Key":"CAR7", "Record":{"make":"Fiat","model":"Punto","colour":"violet","owner":"Pari"}},{"Key":"CAR8", "Record":{"make":"Tata","model":"Nano","colour":"indigo","owner":"Valeria"}},{"Key":"CAR9", "Record":{"make":"Holden","model":"Barina","colour":"brown","owner":"Shotaro"}}]

Следующий вызов chaincode меняет собственника автомобиля:

peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls true --cafile ${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C mychannel -n fabcar --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":"changeCarOwner","Args":["CAR9","Dave"]}'

Проверить нового собственника можно, сконфигурировав терминал для использования chaincode от имени участника второй организации Org2 аналогичным образом и получив список автомобилей еще раз:

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="Org2MSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt

export CORE\_PEER\_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org2.example.com/users/Admin@org2.example.com/msp

export CORE\_PEER\_ADDRESS=localhost:9051

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

## 

## 2.3 Авторизация в сети

Каждый участник валидирует транзакции, а, значит, должен иметь свою пару ключей. network.sh использует специальный инструмент, поставляющийся с Fabric - cryptogen.

В production среде используются CA в качестве root of trust, у каждой организации он свой. С помощью скрипта network.sh можно запустить сеть и с 3 CA по одному на каждую организацию и на ordering service:

./network.sh up -ca

Creating network "net\_default" with the default driver

Creating ca\_org2 ... done

Creating ca\_org1 ... done

Creating ca\_orderer ... done

Fabric поставляется с CA клиентом, его и использует рассматриваемая тестовая сеть. Клиент используется для создания identity для участника, включающего в себя роль, пару ключей, членство в организации. Хранится это в папке msp.

# Вывод

В ходе лабораторной работы я запустил тестовую сеть test network, изучил базовые компоненты сети и их роль в коммуникации между собой, также разобрался в основах коммуникации и ее правил для участников.