Fabric on one VM

2020年12月17日

16:04

# Virtual Machine:Ubuntu20\_64\_链码开发测试

快照：等待安装fabric：已经安装了curl，docker,docker-compose,git,go1.15

Ubuntu\_backup文件夹中包含了此快照时刻的虚拟机的完整克隆，便于后期操作

# Ubuntu16\_64\_practice跑通fabricar

部署链吗参数-ccn 不需要补全文件名，例如fabcar.go 文件只需要写明fabcar即可，在-ccl参数后面跟go

troubleshooting

go代理问题

文件权限问题chmod -R 777 …

# rsu-1-U16:rsu-2-U16:exp-server-U16

# 测试分布式fabric

hosts:

rsu-1: 192.168.31.141//p0org1

rsu-2: 192.168.31.142//p0org2

server:192.168.31.143//orderer

fabric-samples已经提前编译好各种工具如cryptogen,configtxgen,peer,order等，需要提前添加至环境变量中：

sudo vim /etc/profile

export PATH=~/fabric/scripts/fabric-samples/bin:$PATH

由于需要分发证书，所以需要提前在orderer节点电脑上安装ssh，具体链接见下：

<https://www.cnblogs.com/ryanzheng/p/11437761.html>

需要把orderer的证书拷贝到peer下，这样peer才能与orderer通信

scp -r dev1@192.168.1.108:/home/dev1/work/example/order/channel-artifacts channel-artifacts

scp -r dev1@192.168.1.108:/home/dev1/work/example/order/channel-artifacts channel-artifacts

peer1：192.168.31.142已经加入通道

peer2: 192.168.31.143已经加入通道

mkdir -p ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts

cd ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts

scp -r rain@192.168.31.143:/home/rain/work/example/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts tlscacerts

softwares needed to install: sudo apt install jq

安装链码

chmod -R 777

（二）

使用sdk

orderer的证书（设置connection-profile需要修改）：~/work/example/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts

# 重新实验分布式fabric

需要提前执行的操作：

git,curl,vim,docker,docker-compose,go

sudo vim /etc/profile

export GOROOT=/usr/local/go

export GOPATH=$HOME/go

export PATH=$GOROOT/bin:$PATH

export PATH=~/fabric/scripts/fabric-samples/bin:$PATH

export ORDERER\_TLSCA=~/exp/organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/tlsca.com-cert.pem

go env -w GO111MODULE=on

go env -w GOPROXY=https://goproxy.cn,direct

添加gopath，goroot，bin的环境变量

sudo systemctl start docker

sudo systemctl enable docker

sudo usermod -a -G docker <username>//可能需要重启才能生效

hosts:

192.168.31.144 peer0.org1.example.com

192.168.31.146 peer0.org2.example.com

192.168.31.145 orderer.example.com

重启电脑让环境变量生效

# 虚拟机上实战fabric

## 完成准备工作

## 提前在各个虚拟机上配置好host

192.168.1.124 server orderer.com peer0.rsu.com peer0.police.com peer0.vao.com

192.168.1.125 rsu-2 peer2.rsu.com

192.168.1.126 rsu-1 peer1.rsu.com

# ON THE SERVER

mkdir -p ~/exp/organizations && cd ~/exp/organizations

vim crypto-config.yaml

//复制内容

cryptogen generate --config=crypto-config.yaml --output ./

mkdir -p ~/exp/orderer && cd ~/exp/orderer

vim configtx.yaml

//复制内容

configtxgen -profile ThreeOrgsOrdererGenesis -channelID system-channel -outputBlock ./system-genesis-block/genesis.block

configtxgen -profile ThreeOrgsChannel -outputCreateChannelTx ./channel-artifacts/channel1.tx -channelID channel1

vim orderer.yaml

orderer start

# ON THE RSU-1

mkdir -p exp/organizations/peerOrganizations exp/peer

cd exp/organizations/peerOrganizations

scp -r rain@192.168.1.124:~/exp/organizations/peerOrganizations/rsu.com ./

cd ~/exp/peer

vim core.yaml

//复制内容

export FABRIC\_CFG\_PATH=$PWD

peer node start > log\_peer.log 2>&1 &

scp -r rain@server:~/exp/orderer/channel-artifacts ./

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="RsuMSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/../organizations/peerOrganizations/rsu.com/users/Admin@rsu.com/tls/ca.crt

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

export CORE\_PEER\_ADDRESS=peer1.rsu.com:7051

export ORDERER\_TLSCA=${PWD}/../organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/tlsca.com-cert.pem

mkdir -p ~/exp/organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/

scp -r rain@server:~/exp/organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/tlsca.com-cert.pem ~/exp/organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/

export ORDERER\_TLSCA=${PWD}/../organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/tlsca.com-cert.pem

peer channel create -o orderer.com:7050 -c channel1 -f ./channel-artifacts/channel1.tx --outputBlock ./channel-artifacts/channel1.block --tls --cafile $ORDERER\_TLSCA

peer channel join -b ./channel-artifacts/channel1.block

peer channel getinfo -c channel1

# ON THE RSU-2

mkdir -p exp/organizations/peerOrganizations exp/peer

cd exp/organizations/peerOrganizations

scp -r rain@192.168.1.124:~/exp/organizations/peerOrganizations/rsu.com ./

cd ~/exp/peer

vim core.yaml

//复制内容

export FABRIC\_CFG\_PATH=$PWD

peer node start > log\_peer.log 2>&1 &

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="RsuMSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/../organizations/peerOrganizations/rsu.com/users/Admin@rsu.com/tls/ca.crt

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

export CORE\_PEER\_ADDRESS=peer2.rsu.com:7051

mkdir -p ~/exp/organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/

scp -r rain@server:~/exp/organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/tlsca.com-cert.pem ~/exp/organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/

export ORDERER\_TLSCA=${PWD}/../organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/tlsca.com-cert.pem

mkdir channel-artifacts

peer channel fetch 0 ./channel-artifacts/channel.block -o orderer.com:7050 -c channel1 --tls --cafile $ORDERER\_TLSCA

peer channel join -b ./channel-artifacts/channel.block

# ON THE SERVER

## peer0.rsu.com

mkdir -p exp/peer0-rsu

cd exp/peer0-rsu

vim core.yaml

//复制文件

export FABRIC\_CFG\_PATH=$PWD

peer node start > log\_peer.log 2>&1 &

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="RsuMSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/../organizations/peerOrganizations/rsu.com/users/Admin@rsu.com/tls/ca.crt

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

export CORE\_PEER\_ADDRESS=peer0.rsu.com:7051

export ORDERER\_TLSCA=${PWD}/../organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/tlsca.com-cert.pem

mkdir channel-artifacts

peer channel fetch 0 ./channel-artifacts/channel.block -o orderer.com:7050 -c channel1 --tls --cafile $ORDERER\_TLSCA

peer channel join -b ./channel-artifacts/channel.block

## peer0.police.com

mkdir exp/peer0-police

cd exp/peer0-police

vim core.yaml

//复制文件

export FABRIC\_CFG\_PATH=$PWD

peer node start > log\_peer.log 2>&1 &

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="PoliceMSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/../organizations/peerOrganizations/police.com/users/Admin@police.com/tls/ca.crt

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

export CORE\_PEER\_ADDRESS=peer0.police.com:7153

export ORDERER\_TLSCA=${PWD}/../organizations/ordererOrganizations/com/orderers/orderer.com/msp/tlscacerts/tlsca.com-cert.pem

mkdir channel-artifacts

peer channel fetch 0 ./channel-artifacts/channel.block -o orderer.com:7050 -c channel1 --tls --cafile $ORDERER\_TLSCA

peer channel join -b ./channel-artifacts/channel.block

## peer0.vao.com

mkdir exp/peer0-vao

cd exp/peer0-vao

vim core.yaml

//复制文件

export FABRIC\_CFG\_PATH=$PWD

peer node start > log\_peer.log 2>&1 &

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="VaoMSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/../organizations/peerOrganizations/vao.com/users/Admin@vao.com/tls/ca.crt

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

export CORE\_PEER\_ADDRESS=peer0.vao.com:7154

mkdir channel-artifacts

peer channel fetch 0 ./channel-artifacts/channel.block -o orderer.com:7050 -c channel1 --tls --cafile $ORDERER\_TLSCA

peer channel join -b ./channel-artifacts/channel.block

## 设置锚节点ON THE SERVER

## new terminal

cd exp/orderer

//generate the configuration file of anchor peers

configtxgen -profile ThreeOrgsChannel -outputAnchorPeersUpdate ./channel-artifacts/RsuMSPanchors.tx -channelID channel1 -asOrg RsuMSP

configtxgen -profile ThreeOrgsChannel -outputAnchorPeersUpdate ./channel-artifacts/PoliceMSPanchors.tx -channelID channel1 -asOrg PoliceMSP

configtxgen -profile ThreeOrgsChannel -outputAnchorPeersUpdate ./channel-artifacts/VaoMSPanchors.tx -channelID channel1 -asOrg VaoMSP

## Terminal peer0-rsu

peer channel update -o orderer.com:7050 -c channel1 -f ./channel-artifacts/RsuMSPanchors.tx --tls --cafile $ORDERER\_TLSCA

## Terminal peer0-police

peer channel update -o orderer.com:7050 -c channel1 -f ./channel-artifacts/PoliceMSPanchors.tx --tls --cafile $ORDERER\_TLSCA

## Terminal peer0-vao

peer channel update -o orderer.com:7050 -c channel1 -f ./channel-artifacts/VaoMSPanchors.tx --tls --cafile $ORDERER\_TLSCA

# 使用peer channel getinfo -c channel1可以发现区块的height已经被改变

## 编写链码(Out of band)

编写好链码以后，要经过

|  |  |
| --- | --- |
| go mod init exp | //exp here is a self-defined name, and is not required to be the same as the name of the chaincode file. |

go mod tidy

go mod vendor

hyperledger fabric 的链码首字母必须大写！！！！！！！

//此时关闭所有peer节点，重新启动，为了取消掉之前设置的环境变量，（不一定需要，教程上这么写的)

# 安装链码

将编写好的链码复制粘贴到三台rsu节点目录下的./chaincode/exp文件夹下面

以**rsu-1**为例，链码位于：

~/exp/peer/chaincode/exp/exp.go

//目前所有peer节点均处于关闭状态，注意，orderer节点不需要关闭（不关闭好像也问题不大）

## ON THE SERVER

cd ~/exp/orderer

orderer start

## ON THE RSU-1

cd ~/exp/peer/chaincode/exp

go mod init exp

go mod tidy

go mod vendor

cd ../..

peer lifecycle chaincode package chaincode/exp.tar.gz --path chaincode/exp --lang golang --label exp\_1

peer node start > log\_peer.log 2>&1 &

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="RsuMSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/../organizations/peerOrganizations/rsu.com/users/Admin@rsu.com/tls/ca.crt

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

export CORE\_PEER\_ADDRESS=peer1.rsu.com:7051

peer lifecycle chaincode install chaincode/exp.tar.gz

peer lifecycle chaincode queryinstalled

## ON THE RSU-2

cd ~/exp/peer/chaincode/exp

go mod init exp

go mod tidy

go mod vendor

cd ../..

peer lifecycle chaincode package chaincode/exp.tar.gz --path chaincode/exp --lang golang --label exp\_1

//peer node start > log\_peer.log 2>&1 &

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="RsuMSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/../organizations/peerOrganizations/rsu.com/users/Admin@rsu.com/tls/ca.crt

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

export CORE\_PEER\_ADDRESS=peer2.rsu.com:7051

peer lifecycle chaincode install chaincode/exp.tar.gz

//peer lifecycle chaincode queryinstalled

# 

## ON THE SERVER

### Terminal peer0-rsu

cd ~/exp/peer0-rsu/chaincode/exp

go mod init exp

go mod tidy

go mod vendor

cd ../..

peer lifecycle chaincode package chaincode/exp.tar.gz --path chaincode/exp --lang golang --label exp\_1

//没有关闭可以不写

//peer node start > log\_peer.log 2>&1 &

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="RsuMSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/../organizations/peerOrganizations/rsu.com/users/Admin@rsu.com/tls/ca.crt

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

export CORE\_PEER\_ADDRESS=peer0.rsu.com:7051

peer lifecycle chaincode install chaincode/exp.tar.gz

peer lifecycle chaincode queryinstalled

### Terminal peer0-police

cd ~/exp/peer0-police/chaincode/exp

go mod init exp

go mod tidy

go mod vendor

cd ../..

peer lifecycle chaincode package chaincode/exp.tar.gz --path chaincode/exp --lang golang --label exp\_1

peer node start > log\_peer.log 2>&1 &

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="PoliceMSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/../organizations/peerOrganizations/police.com/users/Admin@police.com/tls/ca.crt

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

export CORE\_PEER\_ADDRESS=peer0.police.com:7153

peer lifecycle chaincode install chaincode/exp.tar.gz

peer lifecycle chaincode queryinstalled

### Terminal peer0-vao

cd ~/exp/peer0-vao/chaincode/exp

go mod init exp

go mod tidy

go mod vendor

cd ../..

peer lifecycle chaincode package chaincode/exp.tar.gz --path chaincode/exp --lang golang --label exp\_1

//没有关闭可以不写

peer node start > log\_peer.log 2>&1 &

export CORE\_PEER\_TLS\_ENABLED=true

export CORE\_PEER\_LOCALMSPID="VaoMSP"

export CORE\_PEER\_TLS\_ROOTCERT\_FILE=${PWD}/../organizations/peerOrganizations/vao.com/users/Admin@vao.com/tls/ca.crt

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

export CORE\_PEER\_ADDRESS=peer0.vao.com:7154

peer lifecycle chaincode install chaincode/exp.tar.gz

peer lifecycle chaincode queryinstalled

# 链码签名审议

*链码签名和审议是组织级别，只需要一个节点操作，最终结果会通过gossip协议传递给每个节点*

*安装完成链码后会弹出诸如*

*Chaincode code package identifier: exp\_1:75f4ebdbd767cc7da8dcbd1400e764255ef0d3ab7da06359a957937cbde8cb76*

*这样的字符串，将这个字符串中包含的package identifier作为环境变量引入，不同的节点上package值不同不影响链码审议*

## ON THE SERVER

### Terminal peer0-rsu

export CC\_PACKAGE\_ID=exp\_1:75f4ebdbd767cc7da8dcbd1400e764255ef0d3ab7da06359a957937cbde8cb76

peer lifecycle chaincode approveformyorg -o orderer.com:7050 --channelID channel1 --name exp --version 1.0 --package-id $CC\_PACKAGE\_ID --sequence 1 --tls --cafile $ORDERER\_TLSCA

//查询结果

peer lifecycle chaincode checkcommitreadiness --channelID channel1 --name exp --version 1.0 --sequence 1 --tls --cafile $ORDERER\_TLSCA --output json

*{*

*"approvals": {*

*"PoliceMSP": false,*

*"RsuMSP": true,*

*"VaoMSP": false*

*}*

*}*

### Terminal peer0-police

export CC\_PACKAGE\_ID=exp\_1:75f4ebdbd767cc7da8dcbd1400e764255ef0d3ab7da06359a957937cbde8cb76

peer lifecycle chaincode approveformyorg -o orderer.com:7050 --channelID channel1 --name exp --version 1.0 --package-id $CC\_PACKAGE\_ID --sequence 1 --tls --cafile $ORDERER\_TLSCA

//查询结果

peer lifecycle chaincode checkcommitreadiness --channelID channel1 --name exp --version 1.0 --sequence 1 --tls --cafile $ORDERER\_TLSCA --output json

*{*

*"approvals": {*

*"PoliceMSP": true,*

*"RsuMSP": true,*

*"VaoMSP": false*

*}*

*}*

### Terminal peer0-vao

export CC\_PACKAGE\_ID=exp\_1:75f4ebdbd767cc7da8dcbd1400e764255ef0d3ab7da06359a957937cbde8cb76

peer lifecycle chaincode approveformyorg -o orderer.com:7050 --channelID channel1 --name exp --version 1.0 --package-id $CC\_PACKAGE\_ID --sequence 1 --tls --cafile $ORDERER\_TLSCA

//查询结果

peer lifecycle chaincode checkcommitreadiness --channelID channel1 --name exp --version 1.0 --sequence 1 --tls --cafile $ORDERER\_TLSCA --output json

*{*

*"approvals": {*

*"PoliceMSP": true,*

*"RsuMSP": true,*

*"VaoMSP": true*

*}*

*}*

# 向channel提交定义

*由于链码已经审议通过，所以只需要一个组织的一个节点向channel提交定义即可*

*这里以位于rsu-1的peer1.rsu.com节点为例*

## ON THE RSU-1

cd ../organizations

mkdir -p peerOrganizations/police.com/peers/peer0.police.com/tls/

scp -r rain@192.168.1.124:~/exp/organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt peerOrganizations/police.com/peers/peer0.police.com/tls/

mkdir -p peerOrganizations/vao.com/peers/peer0.vao.com/tls/

scp -r rain@192.168.1.124:~/exp/organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt peerOrganizations/vao.com/peers/peer0.vao.com/tls/

cd ../peer

peer lifecycle chaincode commit -o orderer.com:7050 --channelID channel1 --name exp --version 1.0 --sequence 1 --tls --cafile $ORDERER\_TLSCA --peerAddresses peer1.rsu.com:7051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE --peerAddresses peer0.police.com:7153 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt --peerAddresses peer0.vao.com:7154 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt

*查询链码是否被布置到channel上*

peer lifecycle chaincode querycommitted --channelID channel1 --name exp --cafile $ORDERER\_TLSCA

*成功了会返回Committed chaincode definition for chaincode 'exp' on channel 'channel1':*

*Version: 1.0, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vscc, Approvals: [PoliceMSP: true, RsuMSP: true, VaoMSP: true]*

*之前用rsu-1虽然提交成功了，但是最后调用的时候不成功，这次尝试使用serve上的rsu锚节点peer0.rsu.com*

*官网对之前的问题有了更精确的解答，问题不在于锚节点，而是我安装的链码的package-id不是它本来的id，应该是和其他节点搞混了*

## ON THE SERVER

### Terminal peer0-rsu

peer lifecycle chaincode commit -o orderer.com:7050 --channelID channel1 --name exp --version 1.0 --sequence 1 --tls --cafile $ORDERER\_TLSCA --peerAddresses peer0.rsu.com:7051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE --peerAddresses peer0.police.com:7153 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt --peerAddresses peer0.vao.com:7154 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt

*这里peer0.rsu.com的--tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE内容对应的是…/rsu.com/users/Admin@rsu.com/…而不是其他peerAddress中的…vao.com/peers/peer0.vao.com/…这是因为提交链码定义必须由组织的管理者执行，所以提交者必须具有组织管理员证书，但是认证者只需要在某一个组织内即可，所以对于其他节点，我们不需要它们的管理员证书。*

*在后续的链码调用方面，我们可以把调用节点的证书也更换为普通节点证书，即使管理员证书也可以调用链码。*

*查询链码是否被布置到channel上*

peer lifecycle chaincode querycommitted --channelID channel1 --name exp --cafile $ORDERER\_TLSCA

*成功了会返回Committed chaincode definition for chaincode 'exp' on channel 'channel1':*

*Version: 1.0, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vscc, Approvals: [PoliceMSP: true, RsuMSP: true, VaoMSP: true]*

# 调用链码

*为了保证所有节点之间能相互通信，我们至少需要让它们每个人都拥有一份其他节点的普通证书*

## ON THE RSU-1

mkdir -p ../organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/

scp rain@192.168.1.124:~/exp/organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt ~/exp/organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/

mkdir -p ../organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/

scp rain@192.168.1.124:~/exp/organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt ~/exp/organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt

## ON THE RSU-2

mkdir -p ../organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/

scp rain@192.168.1.124:~/exp/organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt ~/exp/organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/

mkdir -p ../organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/

scp rain@192.168.1.124:~/exp/organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt ~/exp/organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt

## ON THE SERVER

### Terminal peer0-rsu //其他节点也可以

*初始化*

*在这里我就使用了组织管理员证书$CORE\_PEER\_TLS\_ROOTCERT\_FILE，但是从查询调用开始，我将更换为普通节点证书*

peer chaincode invoke -o orderer.com:7050 --tls --cafile $ORDERER\_TLSCA -C channel1 -n exp --peerAddresses peer0.rsu.com:7051 --tlsRootCertFiles $CORE\_PEER\_TLS\_ROOTCERT\_FILE --peerAddresses peer0.police.com:7153 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt --peerAddresses peer0.vao.com:7154 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt -c '{"function":"InitLedger","Args":[]}'

*查询*

peer chaincode query -C channel1 -n exp [ -c '{"Args":["QueryAllCars"]}'

peer chaincode query -C channel1 -n exp [ -c '{"Args":["QueryCar","car2"]}'

*Vao创建一辆车*

peer chaincode invoke -o orderer.com:7050 --tls --cafile $ORDERER\_TLSCA -C channel1 -n exp --peerAddresses peer0.rsu.com:7051 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/rsu.com/peers/peer0.rsu.com/tls/ca.crt --peerAddresses peer0.police.com:7153 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt --peerAddresses peer0.vao.com:7154 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt -c '{"function":"CreateCar","Args":["car7","au"]}'

*Rsu增减车辆信誉*

peer chaincode invoke -o orderer.com:7050 --tls --cafile $ORDERER\_TLSCA -C channel1 -n exp --peerAddresses peer0.rsu.com:7051 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/rsu.com/peers/peer0.rsu.com/tls/ca.crt --peerAddresses peer0.police.com:7153 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt --peerAddresses peer0.vao.com:7154 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt -c '{"function":"changeCarCreditRsu","Args":["car1","1","add"]}'

*Police修改车辆信誉*

peer chaincode invoke -o orderer.com:7050 --tls --cafile $ORDERER\_TLSCA -C channel1 -n exp --peerAddresses peer0.rsu.com:7051 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/rsu.com/peers/peer0.rsu.com/tls/ca.crt --peerAddresses peer0.police.com:7153 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt --peerAddresses peer0.vao.com:7154 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt -c '{"function":"changeCarCreditPol","Args":["car1","50"]}'

*Vao删除一辆车*

peer chaincode invoke -o orderer.com:7050 --tls --cafile $ORDERER\_TLSCA -C channel1 -n exp --peerAddresses peer0.rsu.com:7051 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/rsu.com/peers/peer0.rsu.com/tls/ca.crt --peerAddresses peer0.police.com:7153 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt --peerAddresses peer0.vao.com:7154 --tlsRootCertFiles ${PWD}/../organizations/peerOrganizations/vao.com/peers/peer0.vao.com/tls/ca.crt -c '{"function":"deleteCar","Args":["car7"]}'

# 编写APP

*/home/rain/exp/organizations/peerOrganizations/police.com/peers/peer0.police.com/tls/ca.crt与~/exp/organizations/peerOrganizations/rsu.com/tlsca/tlsca.rsu.com-cert.pem是一回事，在之前搭博文内容的区块链时使用的是后者，这次为了达到最小化证书的目的，决定使用前者*

*关于police与vao的需要按上述变更，但是rsu不需要*