# 搭建Fabric环境

参照《搭建第一个个fabric网络》,能在一台服务器上，运行一个Fabric网络。然后在通过cli容器进行创建channel、节点加入channel、创建chaincode、安装chaincode、实例化chaincode、执行查询修改chaincode。能实现以上操作，视为Fabric网络环境搭建完成。

因为作者使用java连接网络时的是关闭TLS验证的，所以在cli容器进行创建channel和实例化chaincode时需要将tls参数去除，否则java连接网络时会出现报TLS错误。这里不做java连接网络并使用TLS验证的做法。

关闭网络TLS验证

1修改e2e\_cli/base 目录下的两个配置文件中CORE\_PEER\_TLS\_ENABLED为false

2.修改e2e\_cli目录下docker-compose-cli.yaml文件中CORE\_PEER\_TLS\_ENABLED为false

3.最后重启网络，同时fabric-sdk-java的调用demo中，peer节点及orderer节点通讯链接不要使用grpcs，使用grpc进行通讯。

以下是创建和实例化chaincode有无TLS参数命令的对比。

创建chaincode

peer channel create -o orderer.example.com:7050 -c $CHANNEL\_NAME -f ./channel-artifacts/channel.tx --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem

peer channel create -o orderer.example.com:7050 -c $CHANNEL\_NAME -f ./channel-artifacts/channel.tx --tls $CORE\_PEER\_TLS\_ENABLED --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem

实例化chaincode

peer chaincode instantiate -o orderer.example.com:7050 --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C $CHANNEL\_NAME -n mycc -v 1.0 -c '{"Args":["init","a", "100", "b","200"]}' -P "OR ('Org1MSP.member','Org2MSP.member')"

peer chaincode instantiate -o orderer.example.com:7050 --tls $CORE\_PEER\_TLS\_ENABLED --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C $CHANNEL\_NAME -n mycc -v 1.0 -c '{"Args":["init","a", "100", "b","200"]}' -P "OR ('Org1MSP.member','Org2MSP.member')"

# 复制证书、密钥、配置文件

在构建好的Fabric网络环境中将存储证书、密钥、配置文件夹crypto-config和channel-artifacts复制至运行java代码的机器上。等待java代码运行时调用。

# 搭建java运行环境

因为要实现fabric-java-sdk的例子需要有依赖包，这里通过maven的方式添加依赖。主要通过一个pom.xml实现。

# 编写java代码

测试的java代码功能主要为连接Fabric网络、执行已安装的chaincode查询、执行已安装的chaincode修改。主要的代码hyperledger-fabric由官网提供如：HFClient、Channel、ChaincodeID等对象，都是直接引用，在此不多做解释。以下主要讲解使用官网提供的代码对网络进行操作的步骤和具体方式。

## 初始化连接网络的配置

连接配置类中有以下属性，其中属性都得根据网络服务器正确配置

/\*\* 节点服务器对象 \*/

**private** Peers peers;

/\*\* 排序服务器对象 \*/

**private** Orderers orderers;

/\*\* 智能合约对象 \*/

**private** Chaincode chaincode;

/\*\* channel-artifacts所在路径

**private** String channelArtifactsPath;

/\*\* crypto-config所在路径

**private** String cryptoConfigPath;

**private** **boolean** registerEvent = **false**;

### channel-artifacts所在路径

channelArtifactsPath = "E:\\chsoft\\Git\\fabric-samples\\first-network\\channel-artifacts\\"

（此路径为复制过来channel-artifacts的真实路径）

### crypto-config所在路径

cryptoConfigPath = "E:\\chsoft\\Git\\fabric-samples\\first-network\\crypto-config\\";

（此路径为复制过来crypto-config的真实路径）

### 节点服务器对象

FabricConfig fabricConfig = **new** FabricConfig();

Peers peers = **new** Peers();//新建节点对象数组

peers.setOrgName("Org1");//组织名称

peers.setOrgMSPID("Org1MSP");//组织MSP的ID

peers.setOrgDomainName("org1.example.com");//组织所在根域名

peers.addPeer("peer0.org1.example.com", //节点域名

"peer0.org1.example.com", //节点事件域名

"grpc://192.168.15.133:7051",//节点访问地址

"grpc://192.168.15.133:7053", //节点事件监听访问地址

"grpc://192.168.15.133:7054");//节点ca访问地址

fabricConfig.setPeers(peers);//连接配置文件添加连接节点对象

### 排序服务器对象

Orderers orderers = **new** Orderers();//新建排序节点对象

orderers.setOrdererDomainName("example.com");//组织所在根域名

orderers.addOrderer("orderer.example.com",//排序服务器的域名

"grpc://192.168.15.133:7050");//排序服务器的访问地址

fabricConfig.setOrderers(orderers);//连接配置文件添加排序节点对象

### 智能合约对象

chaincode.setChaincodePath("E:\\chsoft\\Git\\fabric-samples\\chaincode\\chaincode\_example02\\go");

chaincode.setChannelName("mychannel");//智能合约所在通道名称

chaincode.setChaincodeName("mycc");//智能合约名称

chaincode.setChaincodeVersion("1.0");//智能合约版本

fabricConfig.setChaincode(chaincode);//连接配置文件添加智能合约对象

## 根据连接配置生成链码管理器

ChaincodeManager c = **new** ChaincodeManager(fabricConfig);

构造方法如下：

**public** ChaincodeManager(FabricConfig fabricConfig){

**this**.config = fabricConfig;

//根据连接配置文件对属性进行赋值

orderers = **this**.config.getOrderers();

peers = **this**.config.getPeers();

chaincode = **this**.config.getChaincode();

//生成默认连接器

client = HFClient.*createNewInstance*();

*log*.debug("Create instance of HFClient");

client.setCryptoSuite(CryptoSuite.Factory.*getCryptoSuite*());

*log*.debug("Set Crypto Suite of HFClient");

/\*

生成联盟、通道、智能合约等信息，在讲解java-sdk连接网络细节时再做详细讲解\*/

fabricOrg = getFabricOrg();

channel = getChannel();

chaincodeID = getChaincodeID();

// 也许是1.0.0测试版的bug，只有节点管理员可以调用链码

client.setUserContext(fabricOrg.getPeerAdmin());

}

## 使用链码管理器进行查询和修改账本

//定义智能合约查询方法中的参数（查询a的余额）

String[] queryArg = **new** String[] { "a" };

//根据chaincode方法名和参数，执行查询返回结果

Map queryResult = c.query("query",queryArg);

//定义智能合约查询方法中对的参数（将a的余额转账10给b）

String[] invokeArg = **new** String[] { "a","b","10" };

//根据chaincode方法名和参数，执行修改返回结果

Map invokeResult = c.invoke("invoke",invokeArg);

# 问题汇总

Caused by: org.hyperledger.fabric.sdk.exception.ProposalException: getConfigBlock for channel mychannel failed with peer peer0.org1.example.com. Status FAILURE, details: Sending proposal to peer0.org1.example.com failed because of: gRPC failure=Status{code=UNKNOWN, description=Failed to deserialize creator identity, err The supplied identity is not valid, Verify() returned x509: certificate signed by unknown authority (possibly because of "x509: ECDSA verification failure" while trying to verify candidate authority certificate "ca.org1.example.com"), cause=null}

这个问题是因为证书不正确导致的，我们要保证在搭建Fabric网络时生成文件夹crypto-config和channel-artifacts里的文件，和我们代码里使用的文件内容一模一样。就是第二步操作得正确。将Fabric网络这两个文件夹复制到java代码里指定的文件夹即可解决问题。

报tls错误时：

1修改e2e\_cli/base 目录下的两个配置文件中CORE\_PEER\_TLS\_ENABLED为false

2.修改e2e\_cli目录下docker-compose-cli.yaml文件中CORE\_PEER\_TLS\_ENABLED为false

3.最后重启网络，同时fabric-sdk-java的调用demo中，peer节点及orderer节点通讯链接不要使用grpcs，使用grpc进行通讯。

# 附件代码文件

hyperledger-fabric提供的代码，git地址如下：

<https://github.com/hyperledger/fabric-sdk-java.git>

作者自己测试通过的例子，文件链接如下：

链接：https://pan.baidu.com/s/1DMLyF\_atdzYyx9GWKJmeQw 密码：f11f