LAB3实验报告

【实验名称】Fabric搭建peer并加入通道

姓名 李泓民 学号 PB18071495

【实验过程】

step1

> 使用CA服务器注册身份，并获得CA服务器颁发的身份证书

在助教给出的kecheng账号scp相应的文件夹，对于给出的命令修改相应的参数，执行`fabric-ca-client register --id.name leehm --id.secret 123456 --id.type peer -u http://222.195.70.186:7054 --mspdir ca-msp`,注册节点:

```
UserPB18071495@block:~$ fabric-ca-client register --id.name leehm --id.secret 123456 --id.type peer
 -u http://222.195.70.186:7054 --mspdir ca-msp
2021/06/10 08:20:11 [INFO] Configuration file location: /home/UserPB18071495/fabric-ca-client-confi
g.yaml
Password: 123456
```

然后再把注册成功之后的peer的mspenroll到本地:

```
UserPB18071495@block:~$ fabric-ca-client enroll -u http://leehm:123456@222.195.70.186:7054 --mspdir ./peer
2021/06/17 04:04:29 [INFO] generating key: &{A:ecdsa S:256}
2021/06/17 04:04:29 [INFO] encoded CSR
2021/06/17 04:04:29 [INFO] Stored client certificate at /home/UserPB18071495/peer/signcerts/cert.pem
2021/06/17 04:04:29 [INFO] Stored root CA certificate at /home/UserPB18071495/peer/cacerts/222-195-70-186-7054.pem
2021/06/17 04:04:29 [INFO] Stored Issuer public key at /home/UserPB18071495/peer/IssuerPublicKey
2021/06/17 04:04:29 [INFO] Stored Issuer revocation public key at /home/UserPB18071495/peer/IssuerRevocationPublicKey
```

查看自己的证书:

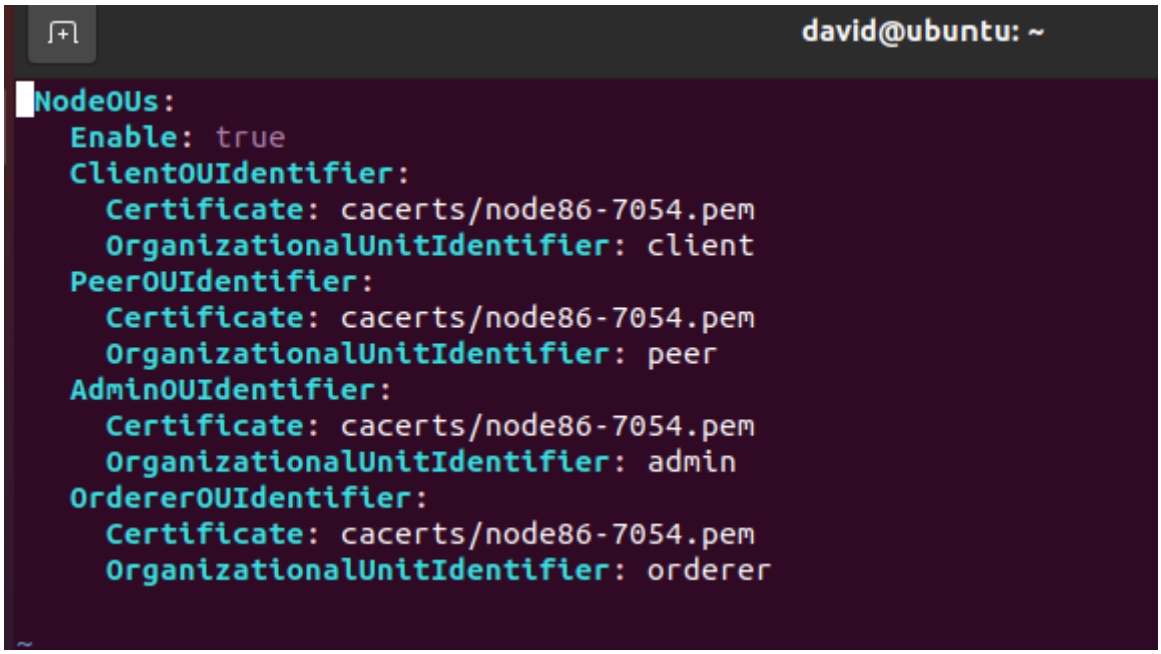
```
UserPB18071495@block:~$ openssl x509 -in ./peer/signcerts/cert.pem -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            48:4f:10:03:3a:cd:a0:82:95:65:5c:4e:60:27:84:b4:f1:10:76:6b
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: C = Ch, ST = Anhui, L = Hefei, O = blockchain-class, OU = Fabric, CN = fabric-ca-server
        Validity
            Not Before: Jun 17 03:59:00 2021 GMT
            Not After : Jun 17 04:04:00 2022 GMT
        Subject: C = US, ST = North Carolina, O = Hyperledger, OU = peer, CN = leehm
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:e0:0a:14:c5:3c:8c:95:3a:dd:31:7d:43:22:e1:
                    da:d7:6d:a3:80:e9:51:ad:84:c5:5a:1e:e0:bd:c0:
                    93:b8:9d:f7:27:50:f1:8f:05:72:13:00:0f:eb:3a:
                    af:e0:2d:6d:d7:c2:b2:5b:0f:3a:9f:38:c0:4c:7a:
                    6d:d9:c8:ef:58
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature
            X509v3 Basic Constraints: critical
                CA:FALSE
            X509v3 Subject Key Identifier:
                D1:7A:0C:8E:8D:A0:94:6B:7C:54:97:34:92:EB:57:CD:72:1F:E0:09
            X509v3 Authority Key Identifier:
                keyid:AA:3F:D0:09:16:9B:5F:98:87:14:3B:99:E4:E3:FD:5E:F9:C2:40:A6

            X509v3 Subject Alternative Name:
                DNS:block
            1.2.3.4.5.6.7.8.1:
                {"attrs":{"hf.Affiliation":"","hf.EnrollmentID":"leehm","hf.Type":"peer"}}
    Signature Algorithm: ecdsa-with-SHA256
         30:45:02:21:00:f6:36:06:7e:1f:7f:6f:67:8d:19:86:bf:ff:
         90:fb:48:5b:ee:08:43:55:2e:48:43:3a:2f:5d:7a:18:93:81:
         23:02:20:72:9c:3b:93:aa:00:aa:82:12:a5:df:1a:9b:a4:b2:
         38:1f:cf:cd:9f:39:a2:30:ce:d9:69:12:ce:29:80:8f:b6
-----BEGIN CERTIFICATE-----
MIICnTCCAkOgAwIBAgIUSE8QAzrNoIKVZVxOYCeEtPEQdmswCgYIKoZIzj0EAwIw
dDELMAkGA1UEBhMCQ2gxDjAMBgNVBAgTBUFuaHVpMQ4wDAYDVQQHEwVIZWZlaTEZ
```

step2

> 在本地准备Peer节点启动所需要的文件，启动Peer节点

查看config.yaml文件:



使用

```
export FABRIC_CFG_PATH=/home/UserPB18071495/peer
```

指定上传的core.yaml的路径,然后修改 peer 节点的 config.yaml 配置文件,修改参数:

```
NodeOUs:
  Enable: true
  # For each identity classification that you would like to utilize,
  # an OU identifier.
  # You can optionally configure that the OU identifier must be issue
  # or intermediate certificate from your organization. However, it i
  # configure a specific Certificate. By not configuring a specific C
  # able to add other CA or intermediate certs later, without having
  # For this reason, the sample below comments out the Certificate fi
  ClientOUIdentifier:
    Certificate: "cacerts/222-195-70-186-7054.pem"
    OrganizationalUnitIdentifier: "client"
  PeerOUIdentifier:
    Certificate: "cacerts/222-195-70-186-7054.pem"
    OrganizationalUnitIdentifier: "peer"
  AdminOUIdentifier:
    Certificate: "cacerts/222-195-70-186-7054.pem"
    OrganizationalUnitIdentifier: "admin"
  OrdererOUIdentifier:
    Certificate: "cacerts/222-195-70-186-7054.pem"
    OrganizationalUnitIdentifier: "orderer"
```

根据cacerts下面的文件名修改为222-195-70-186-7054.pem.

然后是修改core.yaml文件:

- 查询到9999端口是可以使用的,指派9999端口给listenAddress.node88代表的含义和222.195.70.188是一样的.

```
    # The peer id provides a name for this peer instance and is used whe
    # naming docker resources.
    id: leehm

    # The networkId allows for logical separation of networks and is use
    # naming docker resources.
    networkId: dong

    # The Address at local network interface this Peer will listen on.
    # By default, it will listen on all network interfaces      dongjl1
    listenAddress: node88:9999
```

```
38      # When used as peer config, this represents the endpoint to other pe
39      # in the same organization. For peers in other organization, see
40      # gossip.externalEndpoint for more info.
41      # When used as CLI config, this means the peer's endpoint to interac
42      address: node88:9999
```

修改存储数据的路径:

```
    # modification that might corrupt the peer operations.
    fileSystemPath: /home/UserPB18071495/peer/data

    # BCCSP (Blockchain crypto provider): Select which crypto implem
```

将mspConfigPath设置为拷贝的助教的具有admin权限的msp:

```
    # Path on the file system where peer will find MSP local configur
    mspConfigPath: /home/UserPB18071495/peer-msp

    # Identifier of the local MSP
    # ----!!!!IMPORTANT!!!-!!!IMPORTANT!!!-!!!IMPORTANT!!!!----
    # Deployers need to change the value of the localMspId string.
    # In particular, the name of the local MSP ID of a peer needs
    # to match the name of one of the MSPs in each of the channel
    # that this peer is a member of. Otherwise this peer's messages
    # will not be identified as valid by other nodes.
    localMspId: Peer
```

将chaincodeListenAddress设置为:

```
            endorserService: 2500
            # deliverService limits concurrent event listeners registere
            deliverService: 2500
    chaincodeListenAddress: node88:9990
##############################################################################
#
#    VM section
#
##############################################################################
```

修改snapshots储存路径:

```
    snapshots:
        # Path on the file system where peer will store ledger snapshots
        rootDir: /home/UserPB18071495/peer/snapshots

##############################################################################
```

设置完毕后启动节点:

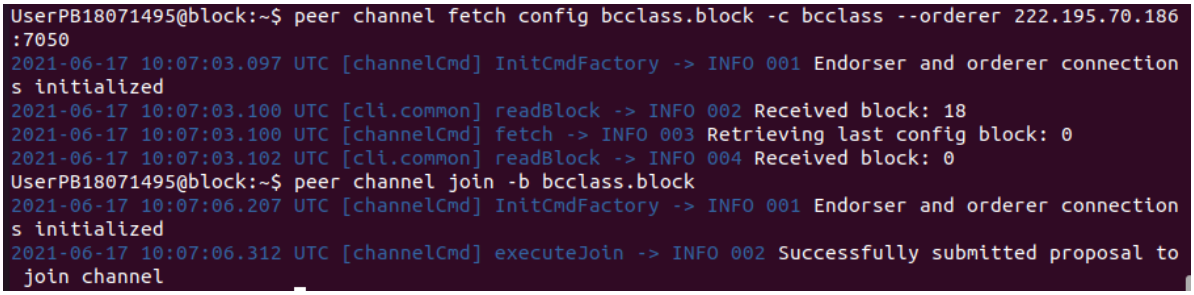
```
UserPB18071495@block:~/peer$ peer node start
2021-06-17 04:43:12.780 UTC [nodeCmd] serve -> INFO 001 Starting peer:
 Version: 2.2.1
 Commit SHA: 344fda602
 Go version: go1.14.4
 OS/Arch: linux/amd64
 Chaincode:
  Base Docker Label: org.hyperledger.fabric
  Docker Namespace: hyperledger
2021-06-17 04:43:12.781 UTC [peer] getLocalAddress -> INFO 002 Auto-detected peer address: 222.195.70.188:9999
2021-06-17 04:43:12.782 UTC [peer] getLocalAddress -> INFO 003 Returning node88:9999
2021-06-17 04:43:12.792 UTC [nodeCmd] initGrpcSemaphores -> INFO 004 concurrency limit for endorser service is 2500
2021-06-17 04:43:12.792 UTC [nodeCmd] initGrpcSemaphores -> INFO 005 concurrency limit for deliver service is 2500
2021-06-17 04:43:12.811 UTC [certmonitor] trackCertExpiration -> INFO 006 The enrollment certificate will expire on 2022-06-17 04:04:00 +0000 UTC
2021-06-17 04:43:12.817 UTC [ledgermgmt] NewLedgerMgr -> INFO 007 Initializing LedgerMgr
2021-06-17 04:43:13.192 UTC [ledgermgmt] NewLedgerMgr -> INFO 008 Initialized LedgerMgr
2021-06-17 04:43:13.201 UTC [gossip.service] New -> INFO 009 Initialize gossip with endpoint node89:7061
2021-06-17 04:43:13.202 UTC [gossip.gossip] New -> INFO 00a Creating gossip service with self membership of Endpoint: node89:7061, InternalEndpoint: node89:7061, PKI-ID: 1352484766c47167499ee245c315c26cf411
cc377da38da7a4239ce52bb9d754, Metadata:
2021-06-17 04:43:13.202 UTC [lifecycle] InitializeLocalChaincodes -> INFO 00b Initialized lifecycle cache with 0 already installed chaincodes
2021-06-17 04:43:13.202 UTC [nodeCmd] computeChaincodeEndpoint -> INFO 00c Entering computeChaincodeEndpoint with peerHostname: node88
2021-06-17 04:43:13.202 UTC [nodeCmd] computeChaincodeEndpoint -> INFO 00d Exit with ccEndpoint: node88:9990
2021-06-17 04:43:13.202 UTC [gossip.gossip] start -> INFO 00e Gossip instance node89:7061 started
2021-06-17 04:43:13.206 UTC [sccapi] DeploySysCC -> INFO 00f deploying system chaincode 'lscc'
2021-06-17 04:43:13.207 UTC [sccapi] DeploySysCC -> INFO 010 deploying system chaincode 'cscc'
2021-06-17 04:43:13.207 UTC [sccapi] DeploySysCC -> INFO 011 deploying system chaincode 'qscc'
2021-06-17 04:43:13.207 UTC [sccapi] DeploySysCC -> INFO 012 deploying system chaincode '_lifecycle'
2021-06-17 04:43:13.207 UTC [nodeCmd] serve -> INFO 013 Deployed system chaincodes
2021-06-17 04:43:13.207 UTC [discovery] NewService -> INFO 014 Created with config TLS: false, authCacheMaxSize: 1000, authCachePurgeRatio: 0.750000
2021-06-17 04:43:13.207 UTC [nodeCmd] registerDiscoveryService -> INFO 015 Discovery service activated
2021-06-17 04:43:13.207 UTC [nodeCmd] serve -> INFO 016 Starting peer with ID=[leehm], network ID=[dong], address=[node88:9999]
2021-06-17 04:43:13.208 UTC [nodeCmd] func6 -> INFO 017 Starting profiling server with listenAddress = 222.195.70.188:6060
2021-06-17 04:43:13.208 UTC [nodeCmd] serve -> INFO 018 Started peer with ID=[leehm], network ID=[dong], address=[node88:9999]
2021-06-17 04:43:13.208 UTC [kvledger] LoadPreResetHeight -> INFO 019 Loading prereset height from path [/home/UserPB18071495/peer/data/ledgersData/chains]
2021-06-17 04:43:13.208 UTC [blkstorage] preResetHtFiles -> INFO 01a No active channels passed
```

获取配置区块

```
peer channel fetch config bcclass.block -c bcclass --orderer
222.195.70.186:7050
```

peer加入通道（加入通道的创世区块可以使用peer channel fetch获得）

```
peer channel join -b bcclass.block
```



最后调用peer channel list查看已加入的节点: