

区块链实验三实验报告

PB19071405 王昊元

2022 年 06 月 24 日

1 实验目的及要求

1.1 实验目的

- 了解 fabric 上的基本配置
- 在 fabric 网络中添加 peer，并加入网络
- 了解 fabric 上的基本证书和网络

1.2 实验要求

1. 使用 CA 服务器注册身份，并获得 CA 服务器颁发的身份证书
2. 在本地准备 Peer 节点启动所需要的文件，启动 Peer 节点
3. 将自己搭建的 Peer 节点加入通道 mychannel

2 实验平台

- OpenStack 平台
- Ubuntu 22.04 LTS (GNU/Linux 5.15.0-30-generic x86_64)

3 实验步骤

3.1 注册身份，并且获取证书

注册部分按照实验文档进行环境变量配置及命令的运行，脚本如下：

```

1 to_tls_certfiles="/home/ubuntu/fabric/tls-ca/crypto/tls-ca-cert.pem"
2 to_tls_ca_admin="/home/ubuntu/fabric/tls-ca/admin"
3 to_hold_tlsmsp="/home/ubuntu/fabric/org1/peer2/tls-msp"
4 PEERNAME="PB19071405"
5 PEERSECRET="12345678"
6 HOSTNAME=${PEERNAME}
7
8 export FABRIC_CA_CLIENT_TLS_CERTFILES=${to_tls_certfiles}
9
10 export FABRIC_CA_CLIENT_HOME=${to_tls_ca_admin}
11
12 # 注册身份
13 fabric-ca-client register --id.name ${PEERNAME} --id.secret ${PEERSECRET} --id.
    type peer -u https://172.16.4.35:7052
14
15 # 获得对应的tls_msp
16 # 存储tls_msp的位置
17 export FABRIC_CA_CLIENT_MSPDIR=${to_hold_tlsmsp}
18 fabric-ca-client enroll -u https://${PEERNAME}:${PEERSECRET}@172.16.4.35:7052
    --enrollment.profile tls --csr.hosts ${HOSTNAME}
19
20 # 注册成功后，可以把keystore下的文件存储为证书。当然，你也可以手动将其重新命名。
21 # mv tls-msp/keystore/*_sk tls-msp/keystore/key.pem
22 # mv msp/keystore/*_sk msp/keystore/key.pem
23 mv ${to_hold_tlsmsp}/keystore/*_sk ${to_hold_tlsmsp}/keystore/key.pem
24
25 unset FABRIC_CA_CLIENT_TLS_CERTFILES
26 unset FABRIC_CA_CLIENT_HOME
27 unset FABRIC_CA_CLIENT_MSPDIR
28
29
30 to_tls_certfiles="/home/ubuntu/fabric/org1/ca/crypto/ca-cert.pem"
31 to_org_ca_admin="/home/ubuntu/fabric/org1/ca/admin"
32 to_hold_msp="/home/ubuntu/fabric/org1/peer2/msp"
33
34 export FABRIC_CA_CLIENT_TLS_CERTFILES=${to_tls_certfiles}
35
36 export FABRIC_CA_CLIENT_HOME=${to_org_ca_admin}
37

```

```

38 fabric-ca-client register --id.name ${PEERNAME} --id.secret ${PEERSECRET} --id.
    type peer -u https://172.16.4.35:7054 --csr.hosts ${HOSTNAME}
39
40 # 存储msp的位置
41 export FABRIC_CA_CLIENT_MSPDIR=${to_hold_msp}
42 # 获得对应的msp
43 fabric-ca-client enroll -u https://${PEERNAME}:${PEERSECRET}@172.16.4.35:7054
44
45 mv ${to_hold_msp}/keystore/*_sk ${to_hold_msp}/keystore/key.pem
46
47 unset FABRIC_CA_CLIENT_TLS_CERTFILES
48 unset FABRIC_CA_CLIENT_HOME
49 unset FABRIC_CA_CLIENT_MSPDIR
50
51 unset PEERNAME
52 unset PEERSECRET
53 unset HOSTNAME

```

3.2 启动 Peer 节点

3.2.1 创建配置文件

先在相应的文件夹下创建 `config.yaml`，不同的文件所对应的证书不同，修改为对应的证书即可。例如在 `admin/msp/config.yaml` 中为

```

1 NodeOUs:
2   Enable: true
3   ClientOUIdentifier:
4     Certificate: cacerts/org1-ca-cert.pem
5     OrganizationalUnitIdentifier: client
6   PeerOUIdentifier:
7     Certificate: cacerts/org1-ca-cert.pem
8     OrganizationalUnitIdentifier: peer
9   AdminOUIdentifier:
10    Certificate: cacerts/org1-ca-cert.pem
11    OrganizationalUnitIdentifier: admin
12   OrdererOUIdentifier:
13     Certificate: cacerts/org1-ca-cert.pem
14     OrganizationalUnitIdentifier: orderer

```

3.2.2 配置 docker

在/home/ubuntu/fabric/org1/peer2 文件夹下创建 docker-compose.yaml 文件，修改对应的信息为自己的信息，如 HOSTNAME 等，PEER 端口为 7051，GOSSIP 端口为 7052。因命名格式与助教一致（主要反映在 peer 的命名上），所以其他信息不需要改动，核实一下签名证书等命名即可。

在/home/ubuntu/fabric/org1/cli 文件夹下创建 docker-compose.yaml 文件，并修改信息。

3.2.3 启动 docker container

分别在/home/ubuntu/fabric/org1/peer2 和/home/ubuntu/fabric/org1/cli 下通过命令 `docker-compose up -d` 启动 peer 和 cli 对应的容器（好像实验文档上没有说单独启动 cli 的容器，我配置完顺手就执行了，但好像也不影响，从 docker 容器的角度看，都可以通过 `docker exec -it cli-org1 /bin/bash` 命令在 container 中的运行 bash）。

3.3 添加通道

由于 docker mount 的路径中不包括 mychannel.block 的路径，所以我们需要将 mychannel.block 复制到/home/ubuntu/fabric/org1/peer2（被挂载的任何路径应该都可以）下

执行 `docker exec -it cli-org1 /bin/bash` 命令进入 cli 容器中的 bash，分别执行 `export CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/org1/admin/msp` 和 `peer channel join -b mychannel.block`（我理解的 mychannel.block 应该对应其路径，所以我先修改工作目录到相应位置才执行的），最后执行 `peer channel list` 查看是否添加成功。

4 实验结果

第一次启动 peer 之后的 log 截图如下：

```

[+] Running 1/1
  Container PB19071405 Started
ubuntu@blockchain-instance:~/fabric/org1/peer2$ docker logs PB19071405
2022-06-25 07:58:05.071 UTC [nodeCmd] serve -> INFO 001 Starting peer:
Version: 2.2.0
Commit SHA: 5ea85bc
Go version: go1.16.4
OS/Arch: linux/amd64
Chaincode:
Base Docker Label: org.hyperledger.fabric
Docker Namespace: hyperledger
2022-06-25 07:58:05.072 UTC [peer] getLocalAddress -> INFO 002 Auto-detected peer address: 172.18.0.2:7051
2022-06-25 07:58:05.072 UTC [peer] getLocalAddress -> INFO 003 Returning PB19071405:7051
2022-06-25 07:58:05.080 UTC [nodeCmd] initOrpSenaphores -> INFO 004 concurrency limit for endorser service is 2500
2022-06-25 07:58:05.080 UTC [nodeCmd] initOrpSenaphores -> INFO 005 concurrency limit for deliver service is 2500
2022-06-25 07:58:05.080 UTC [nodeCmd] serve -> INFO 006 Starting peer with TLS enabled
2022-06-25 07:58:05.377 UTC [ledgermgrnt] NewLedgerMgr -> INFO 007 Initializing LedgerMgr
2022-06-25 07:58:05.944 UTC [leveldbhelper] openDBAndCheckFormat -> INFO 008 DB is empty Setting db format as 2.0
2022-06-25 07:58:06.047 UTC [bktstorage] NewProvider -> INFO 009 Creating new file ledger directory at /var/hyperledger/production/ledgersData/chains/chains
2022-06-25 07:58:06.398 UTC [leveldbhelper] openDBAndCheckFormat -> INFO 009 DB is empty Setting db format as 2.0
2022-06-25 07:58:07.187 UTC [leveldbhelper] openDBAndCheckFormat -> INFO 008 DB is empty Setting db format as 2.0
2022-06-25 07:58:07.282 UTC [ledgermgrnt] NewLedgerMgr -> INFO 00c Initialized LedgerMgr
2022-06-25 07:58:07.293 UTC [gossip.service] New -> INFO 00d Initialize gossip with endpoint PB19071405:7051
2022-06-25 07:58:07.294 UTC [gossip.gossip] New -> INFO 00e Creating gossip service with self membership of Endpoint: PB19071405:7052, InternalEndpoint: PB19071405:7051
2022-06-25 07:58:07.295 UTC [lifecycle] InitializeLocalChaincodes -> INFO 00f Initialized lifecycle cache with 0 already installed chaincodes
2022-06-25 07:58:07.296 UTC [nodeCmd] computeChaincodeEndpoint -> INFO 010 Entering computeChaincodeEndpoint with peerHostname: PB19071405
2022-06-25 07:58:07.296 UTC [nodeCmd] computeChaincodeEndpoint -> INFO 011 Exit with ccEndpoint: PB19071405:7052
2022-06-25 07:58:07.296 UTC [nodeCmd] createChaincodeServer -> WARN 012 peer.chaincodeListenAddress is not set, using PB19071405:7052
2022-06-25 07:58:07.303 UTC [sccapi] DeploySysCC -> INFO 013 Deploying system chaincode 'lcc'
2022-06-25 07:58:07.304 UTC [sccapi] DeploySysCC -> INFO 014 Gossip instance PB19071405:7051 started
2022-06-25 07:58:07.305 UTC [sccapi] DeploySysCC -> INFO 015 Deploying system chaincode 'cacc'
2022-06-25 07:58:07.305 UTC [sccapi] DeploySysCC -> INFO 016 Deploying system chaincode 'qacc'
2022-06-25 07:58:07.305 UTC [sccapi] DeploySysCC -> INFO 017 Deploying system chaincode 'lifecycle'
2022-06-25 07:58:07.305 UTC [nodeCmd] serve -> INFO 018 Deployed system chaincodes
2022-06-25 07:58:07.306 UTC [discovery] NewService -> INFO 019 Created with config TLS: true, authCacheMaxSize: 1000, authCachePurgeRatio: 0.750000
2022-06-25 07:58:07.306 UTC [nodeCmd] registerDiscoveryService -> INFO 01a Discovery service activated
2022-06-25 07:58:07.306 UTC [nodeCmd] serve -> INFO 01b Starting peer with ID=[PB19071405], network ID=[dev], address=[PB19071405:7051]
2022-06-25 07:58:07.306 UTC [nodeCmd] serve -> INFO 01c Started peer with ID=[PB19071405], network ID=[dev], address=[PB19071405:7051]
2022-06-25 07:58:07.307 UTC [kvledger] LoadPresestHeight -> INFO 01d Loading preset height from path [/var/hyperledger/production/ledgersData/chains]
2022-06-25 07:58:07.307 UTC [bktstorage] preResetHTFiles -> INFO 01e No active channels passed
ubuntu@blockchain-instance:~/fabric/org1/peer2$
```

图 1: docker logs PB19071405 的结果（未添加 channel）

添加 channel 后在 cli 中执行 peer channel list 的结果如下：

```

[+] Running 1/1
  Container PB19071405 Started
ubuntu@blockchain-instance:~/fabric/org1/peer2$ docker logs PB19071405
2022-06-25 08:02:01.769 UTC [nodeCmd] serve -> INFO 001 Starting peer:
Version: 2.2.0
Commit SHA: 5ea85bc
Go version: go1.16.4
OS/Arch: linux/amd64
Chaincode:
Base Docker Label: org.hyperledger.fabric
Docker Namespace: hyperledger
2022-06-25 08:02:01.770 UTC [peer] getLocalAddress -> INFO 002 Auto-detected peer address: 172.18.0.2:7051
2022-06-25 08:02:01.770 UTC [peer] getLocalAddress -> INFO 003 Returning PB19071405:7051
2022-06-25 08:02:01.778 UTC [nodeCmd] initOrpSenaphores -> INFO 004 concurrency limit for endorser service is 2500
2022-06-25 08:02:01.778 UTC [nodeCmd] initOrpSenaphores -> INFO 005 concurrency limit for deliver service is 2500
2022-06-25 08:02:01.778 UTC [nodeCmd] serve -> INFO 006 Starting peer with TLS enabled
2022-06-25 08:02:01.778 UTC [ledgermgrnt] NewLedgerMgr -> INFO 007 Initializing LedgerMgr
2022-06-25 08:02:01.778 UTC [leveldbhelper] openDBAndCheckFormat -> INFO 008 DB is empty Setting db format as 2.0
2022-06-25 08:02:01.778 UTC [bktstorage] NewProvider -> INFO 009 Creating new file ledger directory at /var/hyperledger/production/ledgersData/chains/chains
2022-06-25 08:02:01.778 UTC [leveldbhelper] openDBAndCheckFormat -> INFO 009 DB is empty Setting db format as 2.0
2022-06-25 08:02:01.778 UTC [leveldbhelper] openDBAndCheckFormat -> INFO 008 DB is empty Setting db format as 2.0
2022-06-25 08:02:01.778 UTC [ledgermgrnt] NewLedgerMgr -> INFO 00c Initialized LedgerMgr
2022-06-25 08:02:01.778 UTC [gossip.service] New -> INFO 00d Initialize gossip with endpoint PB19071405:7051
2022-06-25 08:02:01.778 UTC [gossip.gossip] New -> INFO 00e Creating gossip service with self membership of Endpoint: PB19071405:7052, InternalEndpoint: PB19071405:7051
2022-06-25 08:02:01.778 UTC [lifecycle] InitializeLocalChaincodes -> INFO 00f Initialized lifecycle cache with 0 already installed chaincodes
2022-06-25 08:02:01.778 UTC [nodeCmd] computeChaincodeEndpoint -> INFO 010 Entering computeChaincodeEndpoint with peerHostname: PB19071405
2022-06-25 08:02:01.778 UTC [nodeCmd] computeChaincodeEndpoint -> INFO 011 Exit with ccEndpoint: PB19071405:7052
2022-06-25 08:02:01.778 UTC [nodeCmd] createChaincodeServer -> WARN 012 peer.chaincodeListenAddress is not set, using PB19071405:7052
2022-06-25 08:02:01.778 UTC [sccapi] DeploySysCC -> INFO 013 Deploying system chaincode 'lcc'
2022-06-25 08:02:01.778 UTC [sccapi] DeploySysCC -> INFO 014 Gossip instance PB19071405:7051 started
2022-06-25 08:02:01.778 UTC [sccapi] DeploySysCC -> INFO 015 Deploying system chaincode 'cacc'
2022-06-25 08:02:01.778 UTC [sccapi] DeploySysCC -> INFO 016 Deploying system chaincode 'qacc'
2022-06-25 08:02:01.778 UTC [sccapi] DeploySysCC -> INFO 017 Deploying system chaincode 'lifecycle'
2022-06-25 08:02:01.778 UTC [nodeCmd] serve -> INFO 018 Deployed system chaincodes
2022-06-25 08:02:01.778 UTC [discovery] NewService -> INFO 019 Created with config TLS: true, authCacheMaxSize: 1000, authCachePurgeRatio: 0.750000
2022-06-25 08:02:01.778 UTC [nodeCmd] registerDiscoveryService -> INFO 01a Discovery service activated
2022-06-25 08:02:01.778 UTC [nodeCmd] serve -> INFO 01b Starting peer with ID=[PB19071405], network ID=[dev], address=[PB19071405:7051]
2022-06-25 08:02:01.778 UTC [nodeCmd] serve -> INFO 01c Started peer with ID=[PB19071405], network ID=[dev], address=[PB19071405:7051]
2022-06-25 08:02:01.778 UTC [kvledger] LoadPresestHeight -> INFO 01d Loading preset height from path [/var/hyperledger/production/ledgersData/chains]
2022-06-25 08:02:01.778 UTC [bktstorage] preResetHTFiles -> INFO 01e No active channels passed
ubuntu@blockchain-instance:~/fabric/org1/peer2$
```

图 2: 添加通道后 peer channel list 的结果

再次查看 peer 容器的 log，如下：

```
report -- ubuntu@blockchain-instance: ~ -- ssh BCServer -- 159x42
2022-06-25 07:56:28.875 UTC [accapi] DeploySysGC -> INFO 017 deploying system chaincode '_lifecycle'
2022-06-25 07:56:28.875 UTC [nodeCmd] serve -> INFO 018 Deployed system chaincodes
2022-06-25 07:56:28.876 UTC [discovery] NewService -> INFO 019 Created with config TLS: true, authCacheMaxSize: 1000, authCachePurgeRatio: 0.750000
2022-06-25 07:56:28.876 UTC [nodeCmd] registerDiscoveryService -> INFO 01c Discovery service activated
2022-06-25 07:56:28.876 UTC [nodeCmd] serve -> INFO 01d Starting peer with ID=[org-peer], network ID=[dev], address=[PB19071405:7051]
2022-06-25 07:56:28.877 UTC [nodeCmd] serve -> INFO 01e Started peer with ID=[org-peer], network ID=[dev], address=[PB19071405:7051]
2022-06-25 07:56:28.877 UTC [validator] LoadPreseetHeight -> INFO 01d Loading preaset height from path [/var/hyperledger/production/ledgersData/chains]
2022-06-25 07:56:28.877 UTC [blkstorage] preResetHtFiles -> INFO 01e No active channels passed
2022-06-25 07:56:56.071 UTC [endorser] callChaincode -> INFO 01f finished chaincode: cscx duration: 12ms channel= txID=618d1733
2022-06-25 07:56:56.071 UTC [comm.grpc.server] 1 -> INFO 020 unary call completed grpc.service=protos.Endorser grpc.method=ProcessProposal grpc.peer_address=172.18.0.3:34696 grpc.code=OK grpc.call_duration=0.912587ms
2022-06-25 07:57:41.798 UTC [endorser] callChaincode -> INFO 021 finished chaincode: cscx duration: 0ms channel= txID=5525cfe5
2022-06-25 07:57:41.799 UTC [comm.grpc.server] 1 -> INFO 022 unary call completed grpc.service=protos.Endorser grpc.method=ProcessProposal grpc.peer_address=172.18.0.3:34700 grpc.code=OK grpc.call_duration=2.199722ms
2022-06-25 08:01:54.971 UTC [ledgermgr] CreateLedger -> INFO 023 Creating ledger [mychannel] with genesis block
2022-06-25 08:01:55.186 UTC [blkstorage] newBlockfileMgr -> INFO 024 Getting block information from block storage
2022-06-25 08:01:55.186 UTC [validator] CommitLegacy -> INFO 025 [mychannel] Committed block [0] with 1 transaction(s) in 1159ms (state_validation=0ms block_and_privdata_commit=763ms state_commit=233ms) commitHash=1
2022-06-25 08:01:54.964 UTC [ledgermgr] CreateLedger -> INFO 026 Created ledger [mychannel] with genesis block
2022-06-25 08:01:54.964 UTC [gossip.gossip] JoinChan -> INFO 027 Joining gossip network of channel mychannel with 1 organizations
2022-06-25 08:01:54.967 UTC [gossip.gossip] learnAnchorPeers -> INFO 028 No configured anchor peers -> WARN 029 For channel mychannel to learn about t = 0, next expected block is = 1
2022-06-25 08:01:57.061 UTC [gossip.state] NewGossipStateProvider -> INFO 029 Updating metadata information for channel mychannel, current ledger sequence is a t = 0, next expected block is = 1
2022-06-25 08:01:57.062 UTC [deliveryClient] StartDeliverForChannel -> INFO 02a This peer will retrieve blocks from ordering service and disseminate to other peers in the organization for channel mychannel
2022-06-25 08:01:57.109 UTC [endorser] callChaincode -> INFO 02b finished chaincode: cscx duration: 2224ms channel= txID=7d856297
2022-06-25 08:01:57.110 UTC [comm.grpc.server] 1 -> INFO 02c unary call completed grpc.service=protos.Endorser grpc.method=ProcessProposal grpc.peer_address=172.18.0.3:34702 grpc.code=OK grpc.call_duration=4.228548571s
2022-06-25 08:01:57.169 UTC [gossip.privdata] StoreBlock -> INFO 02d [mychannel] Received block [1] from buffer
2022-06-25 08:01:57.173 UTC [gossip.gossip] JoinChan -> INFO 02e Joining gossip network of channel mychannel with 1 organizations
2022-06-25 08:01:57.173 UTC [gossip.gossip] learnAnchorPeers -> INFO 02f Learning about the configured anchor peers of orgMSP for channel mychannel : [{peer1:org1:7051}]
2022-06-25 08:01:57.177 UTC [committer.txvalidator] Validate -> INFO 030 [mychannel] Validated block [1] in 8ms
2022-06-25 08:01:58.174 UTC [validator] CommitLegacy -> INFO 031 [mychannel] Committed block [1] with 1 transaction(s) in 1596ms (state_validation=0ms block_and_privdata_commit=642ms state_commit=567ms) commitHash=[47dc548c9aceb784a23079c1173e16bb0b0a87ae084de9117133560115f25a]
2022-06-25 08:01:58.174 UTC [gossip.privdata] StoreBlock -> INFO 032 [mychannel] Received block [2] from buffer
2022-06-25 08:01:58.777 UTC [committer.txvalidator] Validate -> INFO 033 [mychannel] Validated block [2] in 2ms
2022-06-25 08:02:01.819 UTC [endorser] callChaincode -> INFO 034 finished chaincode: cscx duration: 11ms channel= txID=b82f0aab
2022-06-25 08:02:01.819 UTC [comm.grpc.server] 1 -> INFO 035 unary call completed grpc.service=protos.Endorser grpc.method=ProcessProposal grpc.peer_address=172.18.0.3:34704 grpc.code=OK grpc.call_duration=12.152897ms
2022-06-25 08:02:02.785 UTC [gossip.privdata] fetchPrivateData -> WARN 036 Do not know any peer in the channel [mychannel] that matches the policies , abortin
g
```

图 3: 添加通道后 peer 容器的日志

图3为之后截图，可以看到下面已经产生了链上广播的一些信息（如果我没理解错的话），从另一个方面验证了结果的正确性。

5 实验总结及反馈

5.1 实验总结

- 自己实际操作了 Fabric 的注册 Peer、通过 Docker 启动再添加通道的过程，对 Fabric 的理解有了更深的理解。
- 由于实验是在考试之后完成的，对于“考完就忘”的现状来说，也起到了很大的巩固知识的作用。

5.2 实验反馈

可能由于新课程新实验的缘故，实验文档有点杂乱的感觉。但正因为是新课程新实验，我才决定反馈一些信息给助教和老师。

- 提及比较细小的流程。例如对于 **peer2** 目录的创建，是需要我们手动创建的，助教没有在文档中提及这一点，一直让我认为我注册的操作是有误的。
- 上一点提到的例子或许是因为跟助教的实际操作有些许不同，导致这个目录必须我手动创建，但这也意味着可能文档没有考虑全面（毕竟很多人做实验不可能一模一样，除非文档将每一步在哪里输什么看到什么像博客一样写出来，不过那样实验就失去了一定的意

义), 这也很正常, 只是希望助教可以从我们不同人的反馈中不断完善它。例如, 在注册所提供的命令重命名部分, 助教默认了我们所指定的存储位置是在当前文件夹下的 `msp` 和 `tls-msp` (另外可能写两个是包括了两个部分的重命名, 而非考虑两种命名情况, 算是一个小失误吧), 我在自己的脚本中引用了 `to_hold_tlsmsp` 变量来规避这一问题。

- 除此之外可能就是我认为的文档的一些小失误/小问题:
 - 两次注册命令里变量有修改, 但是说明部分没有修改
 - yaml 文件注释使用双斜杠
 - 对 docker 文件挂载路径对应部分的说明其实可以用文字简单说明一下, 我理解那个等号理解了很久 (也有可能是因为我很久没用 docker 而又不愿意看文档导致对 docker 不熟悉所以没反应过来), 对于不熟悉的同学可能理解上有困难。例如 “Docker 中 xx 路径” 会被挂载/对应 “磁盘中 xxx 路径” (主要是理解了很久那个等号)

此外, 如果实验报告中体现出来我哪里理解有误, 还希望助教指出。

最后还很感谢助教提供了详细的操作命令及配置文件模板, 帮助了我很多。