生产环境部署

> 官方示例：https://hyperledger-fabric.readthedocs.io/en/release-2.5/test_network.html

1. 基础环境

角色	IP地址	依赖环境
orderer1	192.168.3.15	go1.21.4+Docker-Composev2.23.3+Docker20.10.10+jq-1.7+git1.8.3.1
orderer2	192.168.3.16	go1.21.4+Docker-Composev2.23.3+Docker20.10.10+jq-1.7+git1.8.3.1
orderer3	192.168.3.17	go1.21.4+Docker-Composev2.23.3+Docker20.10.10+jq-1.7+git1.8.3.1
peer1	192.168.3.18	go1.21.4+Docker-Composev2.23.3+Docker20.10.10+jq-1.7+git1.8.3.1
peer2	192.168.3.19	go1.21.4+Docker-Composev2.23.3+Docker20.10.10+jq-1.7+git1.8.3.1

1.1 安装git

```
yum -y install git
```

安装go

```
cd /usr/local/ && wget https://studygolang.com/dl/golang/go1.21.4.linux-amd64.ta
```

安装docker

```
sudo yum install -y yum-utils
sudo yum-config-manager \
    --add-repo \
    https://download.docker.com/linux/centos/docker-ce.repo
sudo yum install docker-ce docker-ce-cli containerd.io
```

```
6 sudo systemctl start docker
```

安装docker-compose

```
1  curl -L https://github.com/docker/compose/releases/download/v2.23.3/docker-comp
```

安装jq

```
1 wget https://github.com/jqlang/jq/releases/download/jq-1.7/jq-linux-amd64 && mkd
```

1.2 设置hosts

```
1 echo "192.168.3.15 orderer.example.com" >>/etc/hosts
2 echo "192.168.3.16 orderer2.example.com" >>/etc/hosts
3 echo "192.168.3.17 orderer3.example.com" >>/etc/hosts
4 echo "192.168.3.18 peer0.org1.example.com" >>/etc/hosts
5 echo "192.168.3.19 peer0.org2.example.com" >>/etc/hosts
6 scp -r /etc/hosts orderer2.example.com:/etc/hosts
7 scp -r /etc/hosts orderer3.example.com:/etc/hosts
8 scp -r /etc/hosts peer0.org1.example.com:/etc/hosts
9 scp -r /etc/hosts peer0.org1.example.com:/etc/hosts
```

2. 上传部署包

> 下载地址：https://deploystore.insightone.cn/middleware/Hyperledger-Fabric/
>
> 统一上传到/data/fabric

2.1 orderer1

https://deploystore.insightone.cn/middleware/Hyperledger-Fabric/fabric-orderer1.zip

2.2 orderer2

https://deploystore.insightone.cn/middleware/Hyperledger-Fabric/fabric-orderer2.zip

2.3 orderer3

https://deploystore.insightone.cn/middleware/Hyperledger-Fabric/fabric-orderer3.zip

2.4 peer1

https://deploystore.insightone.cn/middleware/Hyperledger-Fabric/fabric-peer1.zip

2.5 peer2

https://deploystore.insightone.cn/middleware/Hyperledger-Fabric/fabric-peer2.zip

3. 导入镜像

> 下载地址：https://deploystore.insightone.cn/middleware/Hyperledger-Fabric/images.zip
>
> 所有角色节点

```
unzip -d /data images.zip
cd /data/ && sh load.sh
```

4. 启动网络

4.1 orderer1

```
cd /data/fabric/fabric-orderer1/test-network
[root@orderer1 test-network]# ./network.sh up
```

将organizations传输给其余四台节点

```
scp -rq organizations orderer2.example.com:/data/fabric/fabric-orderer2/test-net
scp -rq organizations orderer3.example.com:/data/fabric/fabric-orderer3/test-net
scp -rq organizations peer0.org1.example.com:/data/fabric/fabric-peer1/test-netw
scp -rq organizations peer0.org2.example.com:/data/fabric/fabric-peer2/test-netw
```

4.2 orderer2

```
[root@orderer2 test-network]# ./network.sh up
```

4.3 orderer3

```
[root@orderer3 test-network]# ./network.sh up
```

4.4 peer1

```
[root@peer1 test-network]# ./network.sh up
```

4.5 peer2

```
[root@peer2 test-network]# ./network.sh up
```

5. Creating a channel

5.1 orderer1

```
./network.sh createChannel -c carbonchain
```

将channel-artifacts传输给peer节点

```
scp -r channel-artifacts/ peer0.org1.example.com:/data/fabric/fabric-peer1/test-
scp -r channel-artifacts/ peer0.org2.example.com:/data/fabric/fabric-peer2/test-
```

5.2 peer1

```
./network.sh createChannel -c carbonchain
```

5.3 peer2

```
./network.sh createChannel -c carbonchain
```

6. Starting a chaincode on the channel

> 在任意peer节点执行

6.1 peer1

```
./network.sh deployCC -ccn carbon_exchange -ccp ../asset-transfer-basic/chainco
```

7. Interacting with the network

7.1 peer1

```
export PATH=${PWD}/../bin:$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.e
export CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org1.examp
export CORE_PEER_ADDRESS=localhost:7051
```

运行以下命令用一些资产来初始化账本：

```
peer chaincode invoke -o orderer.example.com:7050 --ordererTLSHostnameOverride
orderer.example.com --tls --cafile
${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.
com/msp/tlscacerts/tlsca.example.com-cert.pem -C  carbonchain  -n
carbon_exchange --peerAddresses peer0.org1.example.com:7051 --tlsRootCertFiles
${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.exampl
e.com/tls/ca.crt --peerAddresses peer0.org2.example.com:9051 --
tlsRootCertFiles
${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.exampl
e.com/tls/ca.crt -c '{"function":"InitLedger","Args":[]}'
```

如果命令成功，您将观察到类似以下的输出：

```
-->INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. re
```

现在你可以用你的 CLI 工具来查询账本。运行以下指令来获取添加到通道账本的资产列表：

```
peer chaincode query -C carbonchain -n carbon_exchange -c '{"Args":["GetAllAsset
```

如果命令成功，您应该看到以下响应：

```
[{"AppraisedValue":300,"Color":"blue","ID":"asset1","Owner":"Tomoko","Size":5},
{"AppraisedValue":400,"Color":"red","ID":"asset2","Owner":"Brad","Size":5},
{"AppraisedValue":500,"Color":"green","ID":"asset3","Owner":"Jin
Soo","Size":10},
{"AppraisedValue":600,"Color":"yellow","ID":"asset4","Owner":"Max","Size":10},
{"AppraisedValue":700,"Color":"black","ID":"asset5","Owner":"Adriana","Size":15
},
{"AppraisedValue":800,"Color":"white","ID":"asset6","Owner":"Michel","Size":15}
]
```

当一个网络成员希望在账本上转一些或者改变一些资产，链码会被调用。使用以下的指令来通过调用 asset-transfer (basic) 链码改变账本上的资产所有者：

```
peer chaincode invoke -o orderer.example.com:7050 --ordererTLSHostnameOverride
orderer.example.com --tls --cafile
${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.
com/msp/tlscacerts/tlsca.example.com-cert.pem -C  carbonchain  -n
carbon_exchange  --peerAddresses peer0.org1.example.com:7051 --
tlsRootCertFiles
${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.exampl
e.com/tls/ca.crt --peerAddresses peer0.org2.example.com:9051 --
tlsRootCertFiles
${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.exampl
e.com/tls/ca.crt -c '{"function":"TransferAsset","Args":
["asset6","Christopher"]}'
```

如果命令成功，您应该看到以下响应：

```
INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. resul
```

7.2 peer2

调用链码之后，我们可以使用另一个查询来查看调用如何改变了区块链账本的资产。因为我们已经查询了 Org1 的 peer，我们可以把这个查询链码的机会通过 Org2 的 peer 来运行。设置以下的环境变量来操作 Org2：

```
export PATH=${PWD}/../bin:$PATH
export FABRIC_CFG_PATH=$PWD/../config/
export CORE_PEER_TLS_ENABLED=true
export CORE_PEER_LOCALMSPID="Org2MSP"
export CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations/org2.e
export CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org2.examp
export CORE_PEER_ADDRESS=localhost:9051
```

你可以查询运行在 `peer0.org2.example.com` asset-transfer (basic) 链码：

```
peer chaincode query -C  carbonchain  -n carbon_exchange  -c '{"Args":["ReadAsse
```

结果显示 `"asset6"` 转给了 Christopher:

```
{"AppraisedValue":800,"Color":"white","ID":"asset6","Owner":"Christopher","Size"
```

8. 关停网络

```
./network.sh down
```

该命令将停止并删除节点和链码容器，删除组织加密材料，并从Docker Registry移除链码镜像。 该命令还删除之前运行的通道项目和docker卷。如果您遇到任何问题，还允许您再次运行 `./network.sh up` 。

9. Hyperledger explorer

> 官网地址：https://github.com/hyperledger-labs/blockchain-explorer

```
[root@orderer1 test-network]# cd explorer/
[root@orderer1 explorer]# cp -r ../organizations/ .
[root@orderer1 explorer]# docker-compose up -d
```

用户：exploreradmin

密码：exploreradminpw

访问地址：http://IP:8080/