Hyperledger Fabric Samples 建立第一个网络

如果是linux用户，切换到 root 用户

初次使用Unbutu发现无法切换到Root权限状态，可以按如下到步骤做：

先设置root到密码：

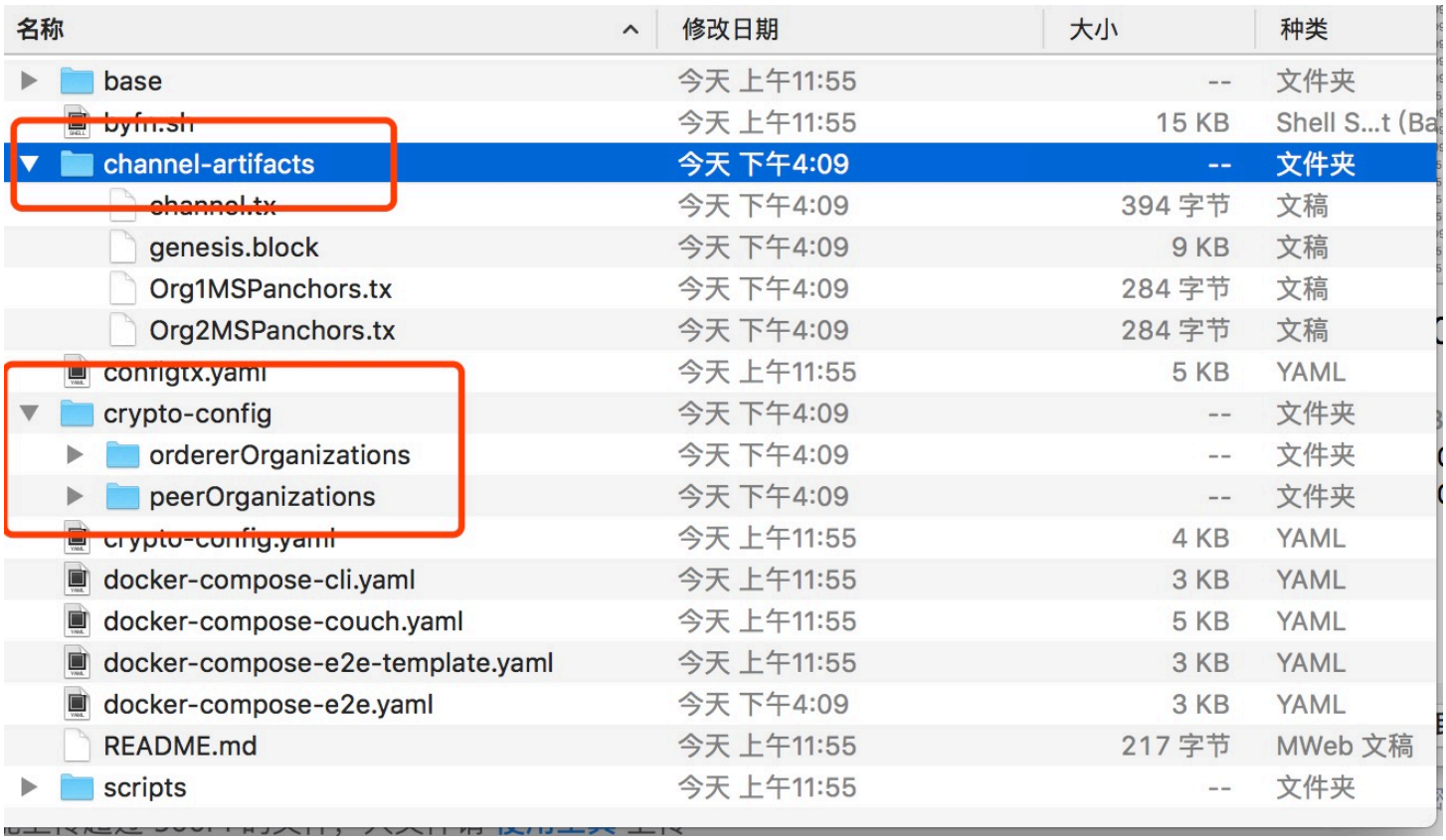
```
sudo passwd root
```

在控制台直接输入：`su root`，并输入密码
就可以获得root权限了。

生成网络配置

```
$ cd first-network
$ ./byfn.sh -m generate
```

运行结果：



生成初始区块

```
$ ../bin/cryptogen generate --config=./crypto-config.yaml
```

```
$ export FABRIC_CFG_PATH=$PWD
$ ../bin/configtxgen -profile TwoOrgsOrdererGenesis -outputBlock ./channel-a
rtifacts/genesis.block
```

生成应用通道的配置信息

```
$ export CHANNEL_NAME=mychannel

$ ../bin/configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./channe
l-artifacts/channel.tx -channelID $CHANNEL_NAME
```

生成锚节点配置更新文件

```
$ ../bin/configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./chan
nel-artifacts/Org1MSPanchors.tx -channelID $CHANNEL_NAME -asOrg Org1MSP
```

```
$ ../bin/configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./chan
nel-artifacts/Org2MSPanchors.tx -channelID $CHANNEL_NAME -asOrg Org2MSP
```

操作网络

编辑 `docker-compose-cli.yaml` ,注释到 command 命令

```
working_dir: /opt/gopath/src/github.com/hyperledger/fabric/peer
# command: /bin/bash -c './scripts/script.sh ${CHANNEL_NAME}; sleep $TIMEOUT
'
volumes
```

```
$ CHANNEL_NAME=$CHANNEL_NAME TIMEOUT=600 docker-compose -f docker-compose-cl
i.yaml up -d
```

```
[localhost:first-network fujinliang$ CHANNEL_NAME=$CHANNEL_NAME TIMEOUT=600 docke]
r-compose -f docker-compose-cli.yaml up -d
Starting peer1.org2.example.com ...
Starting peer0.org2.example.com ...
Starting peer0.org2.example.com
Starting peer1.org2.example.com
Starting orderer.example.com ...
Starting orderer.example.com
Starting peer0.org1.example.com ...
Starting peer1.org1.example.com ...
Starting orderer.example.com ... done
Starting peer1.org1.example.com ... done
Starting cli ...
Starting cli ... done
```

创建和加入通道

进入docker容器

```
$ docker exec -it cli bash
```

创建通道

```
$ export CHANNEL_NAME=mychannel
```

```
$ peer channel create -o orderer.example.com:7050 -c $CHANNEL_NAME -f ./chan
nel-artifacts/channel.tx --tls $CORE_PEER_TLS_ENABLED --cafile /opt/gopath/s
rc/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.co
m/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem
```

加入通道

```
$ peer channel join -b mychannel.block
```

链上代码 （链码）

安装链码

```
$ peer chaincode install -n mycc -v 1.0 -p github.com/hyperledger/fabric/exa
mples/chaincode/go/chaincode_example02
```

实例化链码

```
$ peer chaincode instantiate -o orderer.example.com:7050 --tls $CORE_PEER_TL
S_ENABLED --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto
/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacert
s/tlsca.example.com-cert.pem -C $CHANNEL_NAME -n mycc -v 1.0 -c '{"Args":["i
nit","a", "100", "b","200"]}' -P "OR ('Org1MSP.member','Org2MSP.member')"
```

查询

```
$ peer chaincode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","a"]}'
```

查询结果：

```
Query Result: 100
```

转账

```
$ peer chaincode invoke -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 -c '{"Args":["invoke","a","b","10"]}'
```

查询a账户的金额:

```
$ peer chaincode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","a"]}'

## 结果
Query Result: 90
```

问题记录

docker 运行时出错，可以查询docker正在运行的容器，删除运行的容器

```
# 查询正在运行的容器
docker ps -a

# 删除运行的容器
docker rm -f $(docker ps -aq)
```

清理网络

```
./byfn.sh -m down
```