# Fabric多机环境部署

目录

[Fabric多机环境部署 1](#_Toc30000)

[1. 多机环境部署 2](#_Toc26837)

[1.1. 关闭selinux 3](#_Toc10253)

[1.2. 关闭防火墙 3](#_Toc21342)

[1.3. 配置域名解析 4](#_Toc12333)

[1.4. 证书配置 4](#_Toc28784)

[1.5. 通道配置 4](#_Toc4806)

[1.5.1. 生成创世区块 5](#_Toc3284)

[1.5.2. 生成通道配置区块 5](#_Toc7210)

[1.5.3. 生成锚节点信息 5](#_Toc21403)

[1.6. Docker配置 5](#_Toc8989)

[1.7. 启动 5](#_Toc8329)

[1.8. 清空docker环境 5](#_Toc15391)

[1.9. 附件 6](#_Toc26304)

[1.10. 参考资料 6](#_Toc993)

# 多机环境部署

本文档基于fabric1.4.4，在centos上搭建多机环境，fabric镜像和docker环境的安装请参考官方文档。

使用raft共识，3个order，2个组织，每个组织2个peer，raft共识需开启tls。

有3台服务器，docker布局和端口分配方案如下表：

|  |  |  |
| --- | --- | --- |
| 172.16.10.112 | 172.16.10.164 | 172.16.10.187 |
| Order1:7050 | Order2:7050 | Order3:7050 |
| Peer0.org1:7051->couchdb1:5984 | Peer1.org1:7051->couchdb1:5984  Peer1.org2:8051->couchdb2:6984 | Peer0.org2:7051->couchdb1:5984 |
| CLI | CLI | CLI |

每个服务器的目录结构如下：

.

└── test

├── chaincode

│   ├── abac

│   │   └── go

│   ├── chaincode\_example02

│   │   ├── go

│   │   ├── java

│   │   └── node

│   ├── fabcar

│   │   ├── go

│   │   ├── java

│   │   ├── javascript

│   │   ├── javascript-low-level

│   │   └── typescript

│   ├── marbles02

│   │   ├── go

│   │   └── node

│   ├── marbles02\_private

│   │   ├── collections\_config.json

│   │   └── go

│   └── sacc

│   └── sacc.go

└── multi\_node

├── base

│   └── peer-base.yaml

├── channel-artifacts

│   ├── channel.tx

│   ├── genesis.block

│   ├── mychannel.block

│   ├── Org1MSPanchors.tx

│   └── Org2MSPanchors.tx

├── configtx.yaml

├── crypto-config

│   ├── ordererOrganizations

│   └── peerOrganizations

├── crypto-config.yaml

├── docker-compose-host.yaml

├── scripts

│   ├── installcc.sh

│   └── utils.sh

└── start.sh

## 关闭selinux

查看selinux状态命令：

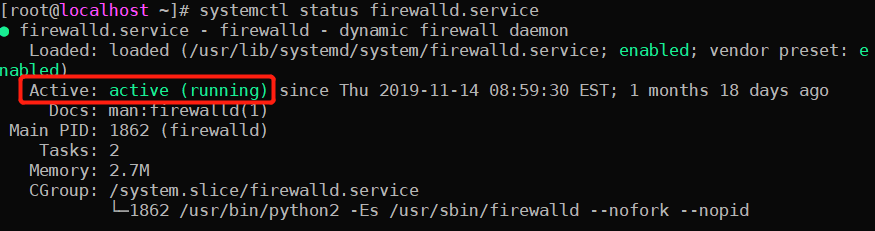
/usr/sbin/sestatus -v

临时关闭selinux命令，不需重启：

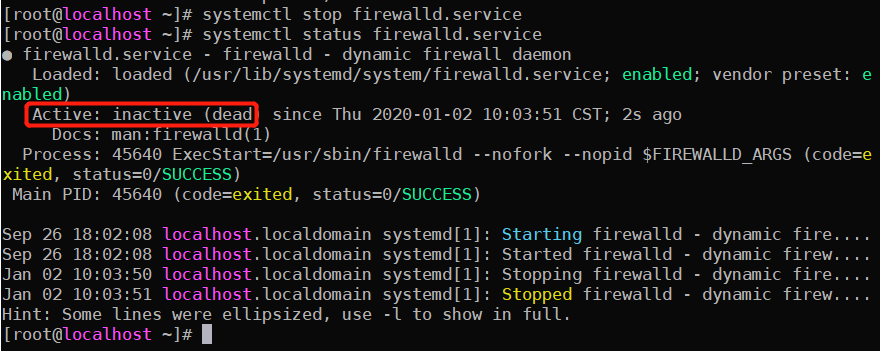
setenforce 0

## 关闭防火墙

systemctl status firewalld.service查看防火墙状态，执行后可以看到绿色字样标注的“active（running）”，说明防火墙是开启状态



systemctl stop firewalld.service 关闭运行的防火墙



## 配置域名解析

在每台的服务器上都需要配置order、peer、ca的域名，需要修改本机的域名和docker容器的域名。本机域名修改/etc/hosts。Docker容器的域名在docker-composer的extra\_hosts下面进行配置。

112的/etc/hosts文件如下：

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4 orderer1.example.com peer0.org1.example.com

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

172.16.10.164 orderer2.example.com peer1.org1.example.com peer1.org2.example.com

172.16.10.187 orderer3.example.com peer0.org2.example.com

164的/etc/hosts文件如下：

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4 orderer2.example.com peer1.org1.example.com peer1.org2.example.com

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

172.16.10.112 orderer1.example.com peer0.org1.example.com

172.16.10.187 orderer3.example.com peer0.org2.example.com

187的/etc/hosts文件如下：

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4 orderer3.example.com peer0.org2.example.com

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

172.16.10.112 orderer1.example.com peer0.org1.example.com

172.16.10.164 orderer2.example.com peer1.org1.example.com peer1.org2.example.com

## 证书配置

crypto-config.yaml配置参考附件《fabric多机环境》

只需要在一台服务器生成证书，并复制到其他服务器的证书目录中。

注意：使用同样的crypto-config.yaml配置，每次生成的证书也是不一样的，所以证书只能生成一次，然后复制到其他服务器上。

在112服务器生成证书：

cryptogen generate --config=./crypto-config.yaml

证书复制到另外两台服务器：

scp -r ./crypto-config/\* [root@172.16.10.164:/root/hyperledger-fabric-1.4.4/test/multi\_node/crypto-config/](mailto:root@172.16.10.164:/root/hyperledger-fabric-1.4.4/test/multi_node/crypto-config/)

scp -r ./crypto-config/\* [root@172.16.10.187:/root/hyperledger-fabric-1.4.4/test/multi\_node/crypto-config/](mailto:root@172.16.10.164:/root/hyperledger-fabric-1.4.4/test/multi_node/crypto-config/)

## 通道配置

configtx.yaml 配置参加附件《fabric多机环境》

只需在一台服务器生成创世块、通道信息即可。下面命令皆在112服务器执行

### 生成创世区块

configtxgen -profile SampleMultiNodeEtcdRaft -channelID $SYS\_CHANNEL -outputBlock ./channel-artifacts/genesis.block

### 生成通道配置区块

configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./channel-artifacts/mychannel.tx -channelID mychannel

### 生成锚节点信息

configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./channel-artifacts/Org1MSPanchors.tx -channelID mychannel -asOrg Org1MSP

configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./channel-artifacts/Org2MSPanchors.tx -channelID mychannel -asOrg Org2MSP

## Docker配置

为了使用统一的启动脚本，所有服务器的docker配置均写到docker-compose-host.yaml中，所有容器都使用容器名称作为数据持久化的目录。

每台服务器的docker-compose配置参考附件《fabric多机环境》。

注意docker容器内的couchdb端口必须是5984，不能修改，一台服务器启动多个couchdb时，只需要修改本机对外映射的端口号。

## 启动

启动脚本为start.sh，三台服务器同时执行start.sh，重点关注112的输出日志。

三台服务器启动各自的docker容器。112比其他两个服务器多了锚节点更新、channel创建、链码部署、链码交易和查询。链码使用的fabric-samples中的chaincode

112的start.sh脚本比其他两个服务器的脚本多了锚节点更新、channel创建、链码部署、链码交易和查询。

三台服务器的启动脚本参考附件《fabric多机环境》。

## 清空docker环境

彻底清空不使用的docker环境，需要执行以下命令：

stop停止所有容器

docker stop $(docker ps -a -q)

remove删除所有容器

docker rm $(docker ps -a -q)

清空所有未使用的docker挂载信息（慎用）

docker volume prune

清理没有再被任何容器引用的networks

docker network prune

## 附件



## 参考资料

<https://www.cnblogs.com/cbkj-xd/p/11067814.html>

<https://www.cnblogs.com/llongst/p/9571363.html>

<https://www.cnblogs.com/si812cn/p/9860391.html>

<https://blog.csdn.net/shebao3333/article/details/103687983>

<https://www.jianshu.com/p/2ddb65a1f27c>

<https://blog.csdn.net/meltsnow/article/details/94548066>

<https://blog.csdn.net/beijihukk/article/details/103500970>