自动化部署脚本说明

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 变更日期 | 变更人 | 变更内容 |
| 20191011/初版 | 20191011 | 邹丽丽 | 2.1底层链自动化部署 |
| 20191012 | 20191012 | 邹丽丽 | 1.增加共识类型可配置 setting.ini type  2.增加更新配置文件脚本updateconfig.sh  3.增加更新节点sdk、tool可执行文件版本 |
| 20191209 | 20191209 | 邹丽丽 | 1.部署脚本增加远程主机账密可配置setting.ini IPUserPwd  (暂未调试仅更新配置文件和版本部分)  2.根据新版本节点配置文件内容更新deploy.sh配置信息 |

## 准备工作

1.部署机器linux系统，预先安装expect

2.确认各个集群之间相互可以连通 所有涉及端口均开放

3.执行前确认所有的shell脚本具有可执行权限

## 脚本作用

实现同网段底层链节点及SDK自动化部署，包括配置文件信息适配、节点及sdk部署；节点及sdk进程启动，sdk权限赋值

## 20191012版本限制说明

1. 实现不支持同一台机器部署多个节点或者部署多个sdk（节点和sdk可以使用同一台主机）
2. 已测试系统 ubuntu 、centos
3. 仅支持同网络部署，即部署主机与待部署服务器需要网络连通
4. 目前适配版本2.1 测试构建版本时间 2019年10月 9日 17:39:25
5. 对配置文件内容有依赖性，因此若有改动需要再进行适配调整
6. 当前脚本仅支持全新部署，不支持仅更新节点、sdk执行文件（有需求后续考虑）

## 部署说明

### 部署执行

#### 获取完整部署文件

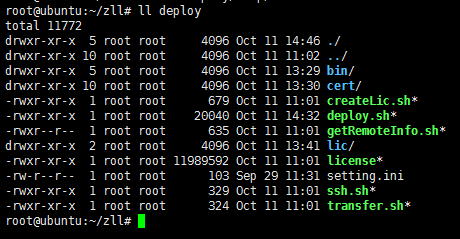
1. 获取deploy文件 参见[附录](#_附录)deploy.7z

有预放一些文件，部署前先删除cert及lic目录下的文件

2.拷贝使用build.sh成功构建后的版本文件bin文件夹至deploy目录

3获取cert相关证书：tls通讯证书 ca证书 说明参见[cert目录](#_Cert目录)

4.获取各个peer主机的peer.lic 根据主机IP列表（setting.ini中[peer]项下的主机列表）依次命令为peer1.lic peer2.lic ..... 拷贝到deploy的lic目录下（如可以自动生成，则可忽略此步骤）

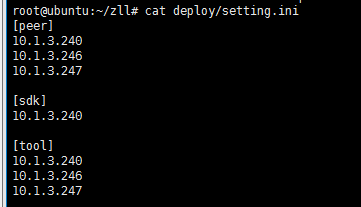


#### setting.ini文件配置

配置主机信息、端口信息、配置文件指定项信息

涉及文件：setting.ini deploy.sh

Setting.ini:配置peer、sdk、tool信息，详细参见部署文件说明中的[setting.ini](#_setting.ini)



20191209新增：为远程部署所有主机及对应的账户密码 以单个空格隔开

[IPUserPwd]

10.1.3.162 root root

#### deploy.sh关键信息配置

#节点部署目录 仅支持创建最后一级目录，即主机当前必须存在deployRootdir目录

deployRootdir=/root $deploydir的上一层目录，所有主机上必须是已存在的文件夹

deploydir=$deployRootdir/ate 待部署目录，此目录会做部署前删除再创建操作，因此此文件建议是系统中$deployRootdir（即/root）目录下不存在的目录，之后节点、sdk、管理工具均会传送至该目录下

remotepeerName=wtpeer 部署完成后节点进程名称，可以根据实际需要修改

remotesdkName=wtcli 部署完成后sdk进程名称，可以根据实际需要修改

remotetoolName=wtkit 部署完成后管理工具进程名称，可以根据实际需要修改

其他关键信息即为节点base.toml和sdk config.toml文件中的一些项目配置

所有节点及sdk配置项信息相同，即所有节点使用的rpc端口都是9800（因此目前不支持同一台主机上部署多个节点）

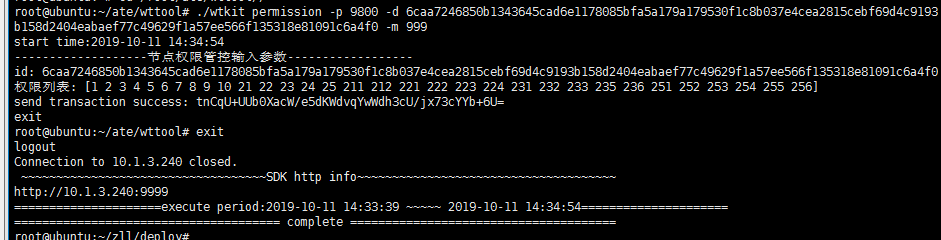


#### 执行部署文件deploy.sh

需要确认deploy目录下所有的sh脚本存在执行权限(chmod +x \*.sh可以全部添加执行权限)

在deploy目录下执行./deploy.sh即可

执行完成，可以看到如下信息

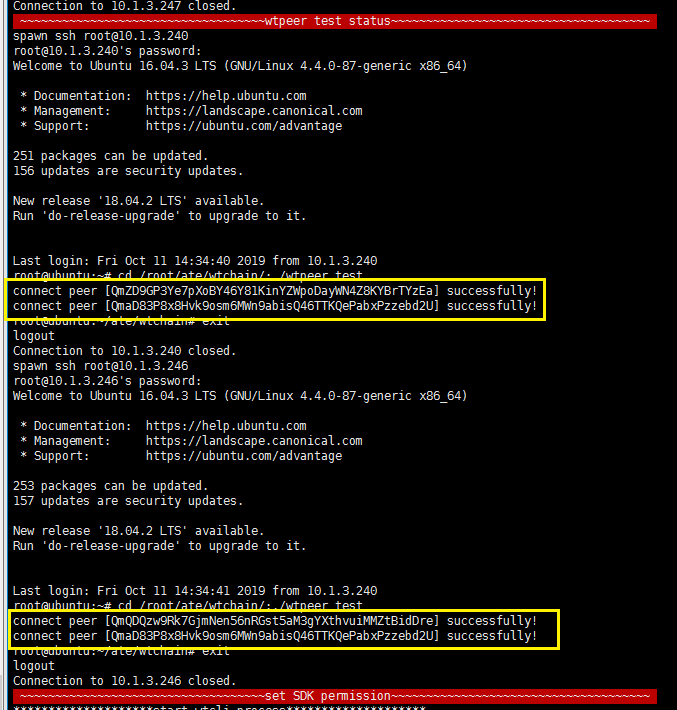


#### 执行后检查

##### 节点状态检查

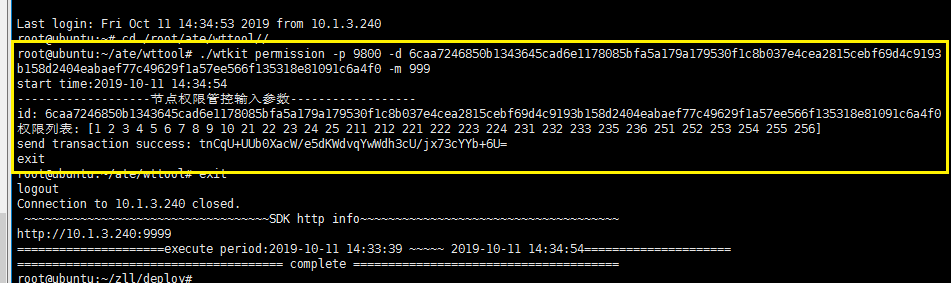
执行时控制台会出现如下高亮执行段，会显示节点状态（./peer test）

目前在两个节点上执行test，全部显示successfully则可认为是正常开启的状态



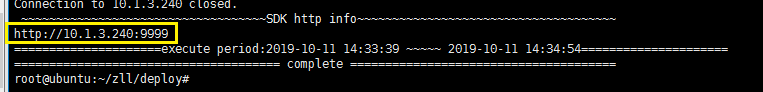
##### sdk权限检查

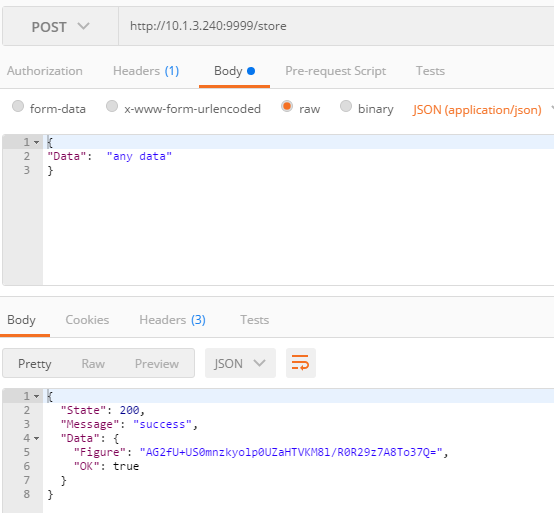
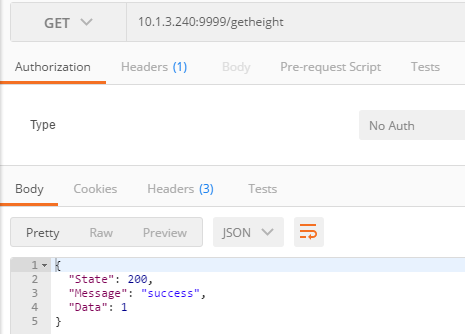
在set SDK permission高亮显示下方有如下sdk权限赋值操作



##### sdk运行检查

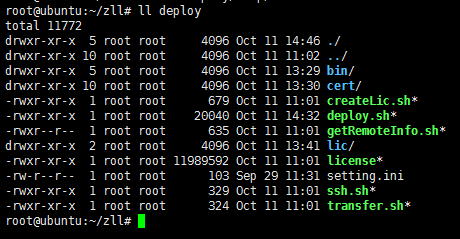
根据执行结束后打印出sdk及端口信息 执行获取高度及存在操作（借助postman或其他工具）





### 部署文件说明

部署目录及文件列表如下

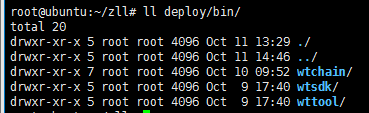


目录及sh脚本简要说明：

#### Bin目录

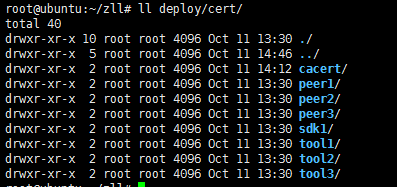
该目录可以在使用build.sh成功构建版本后直接拷贝bin文件夹过来

目前有如下三个目录：节点、sdk和管理工具



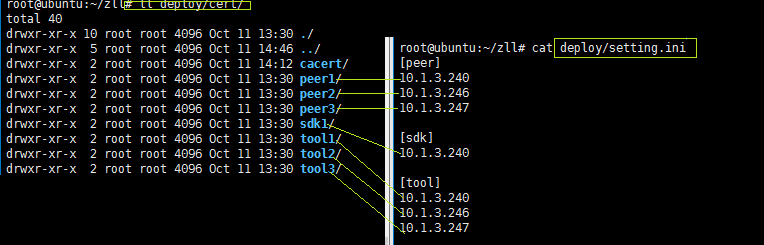
#### Cert目录

为节点、sdk、管理工具在正常运行（tls打开）时必须的证书



cacert目录下证书 为节点CA配置项下ca/crypt目录下的证书，目前不区分节点

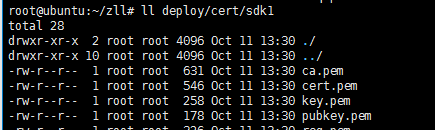
peer\*目录下证书 为节点cert及tls目录下使用证书，peer1下证书会传至setting.ini [peer]下的第一个服务器节点对应的cert/tls目录中



sdk\*为sdk tls auth下使用证书

tool\*为管理工具tls auth下使用证书（目前使用可以是同一个，否则需要赋权限）

目前每个证书目录下须含有如下四个文件，后期根据需求调整（没有的话传输文件报错）



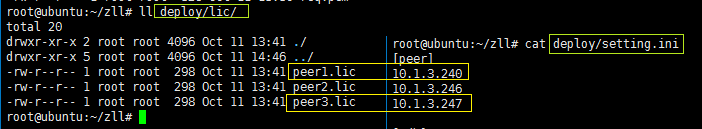
#### lic目录

放置节点启动使用的peer.lic

该目录不是必须，支持如下两种场景：外出部署场景、内部测试部署场景

外出部署场景：为必须，同时deploy.sh中设置配置项licAuto=false，需要预先准备好节点的peer.lic放置在该目录中

peer\*.lic文件需要与setting.ini文件中节点一一对应，即peer1.lic与setting [peer]项下的第一节点地址对应，因peer.lic与目前与mac地址绑定，不对应的话节点启动失败



内部测试部署场景：

lic目录可无，但是必须有license和createLic.sh文件，用来自动生成peer.lic文件，同时deploy.sh中设置配置项licAuto=true

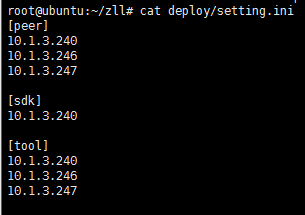
#### setting.ini

节点、sdk、管理工具所需配置的IP信息，目前不支持重复IP，即同一台主机上部署多个peer节点或者部署多个sdk节点（后续有需求再重新做调整）

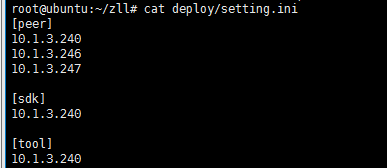
目前要求节点地址顺序必须要与管理工具地址顺序完全一致，即peer第一个是10.1.3.240则tool的第一个IP也必须是10.1.3.240

如果peer.lic需要自动生成（licAuto=true的场景）tool和peer下的主机IP必须完全一致；licAuto=false时tool下的IP的第一个地址必须是peer第一个地址

即如下，

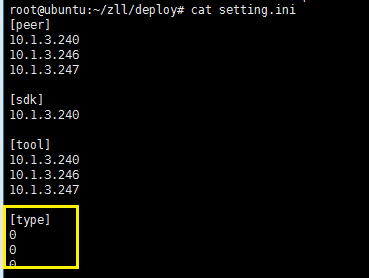


或



20191012：增加共识类型设置

若不设置该项目，则默认为type = 0，即共识节点



#### license

生成节点peer.lic的工具，不对外呈现；licAuto=true时必须

#### sh脚本

所有脚本必须要先有执行权限

目前包含如下几个shell脚本

deploy.sh 部署脚本

createLic.sh 生成节点peer.lic使用脚本

getRemoteInfo.sh 在远程主机上执行命令后获取执行结果指定信息脚本，例如远程获取节点ID、sdk ID、管理工具ID

ssh.sh 远程主机执行命令，不检查响应信息，例如启动、关闭节点进程

transfer.sh scp文件到远程主机，允许expect进行交互

20191012增加如下脚本

updateconfig.sh 更新节点及sdk配置文件信息脚本

upgradeexe.sh 更新节点/sdk/管理工具可执行文件脚本

以上所有内容供参考使用

## 附录

此部分附带所有涉及的脚本及配置文件 bin、cert、lic目录下内容需更新

