土豆链节点快速配置手册

2018-08-01

本文档主要介绍土豆链节点编译安装和配置。 注意：因为命令和配置参数很多，这里只用实例介绍，详细的命令参数说明请查看nodepc和clpc命令文档。

1. 名词解释

1. 创世节点(BIOS) 用于创建初始区块，加载基础合约，供其它节点同步连接。
2. 矿工节点(BP) 主要用于生产同步区块，计算合约产生的费用归矿工账号所有。
3. 备用节点 不产生区块，同步区块信息，提供区块信息查询功能。
4. 私钥 用来进行签名操作，私钥可以生成唯一对应公钥。
5. 公钥 用来对私钥的签名进行验证。
6. 钱包 生成和保存私钥的地方，当需要进行签名操作时，会从钱包读取私钥列表，进行签名。
7. 带宽 带宽分为cpu带宽和net带宽两种，每次执行合约都会消耗一定的带宽（从账号中扣取）。
8. 合约 一段在链上可执行代码，绑定在账号上，每个账号只能绑定一份合约。
9. 账号 存储用户信息，包括余额、带宽、合约（如果有）等。
10. 总票数 总票数=总发行的货币数量。
11. 投票 投票者抵押货币投票给矿工，当矿工投票数大于总票数的15%时创世节点停止产生区块，转由矿工生产区块。

2. 节点代码编译、程序安装

注意：后面编译依赖GIT，所以务必保证代码根目录的.git完整，以及系统安装有git。 以下以ubuntu系统为例，其它系统步骤类似。

1. 先安装GIT，如果已经安装则忽略这一步

```
sudo apt update && sudo apt install git -y
```

2. 从GIT上克隆最新代码

```
git clone http://riseworlds.51vip.biz:8090/Potato-Token/potato.git --
recursive
```

3. 使用pcoin_build.sh自动编译

   脚本会检查系统依赖库，并自动安装，请注意管理员权限提示，如果下载依赖库时中断请检查网络状态，然后再次执行。

```
cd potato
./pcoin_build.sh
```

4. 安装程序到系统

   编译成功会出现'POTATO'的ASCII图样提示，请执行以下操作，把程序安装到系统，如果出现编译错误，请提交错误提示给相关人员查看解决。

```
sudo ./potato_install.sh
```

5. 重新编译 如果有修改代码，或者进行版本更新，请先关闭当前运行的钱包和节点程序，以及清除安装到系统的想着程序，再编译。

```
pkill -2 kpcd nodepc
sudo ./potaot_uninstall.sh
```

3. 默认配置文件所在目录

Linux: ~/.local/share/potato/nodepc/config Mac: ~/Library/Application Support/potato/nodepc/config

4. 创建默认钱包和公私钥

1. 启动钱包服务

```
kpcd --unlock-timeout 3600 # 3600秒后钱包重新锁定
```

2. 创建钱包

   请保存输出的钱包密码，下次打开钱包时用的到。

```
clpc wallet create -n default
```

打开钱包

```
clpc wallet open -n default
```

3. 创建公私钥

```
clpc create key
```

4. 把上步生成的私钥导入钱包

```
clpc wallet import -n default --private-key [private key]
```

5. 通用配置

参数名	示例	说明
agent-name	p2p网络中标识自己的节点的名字	"potato bios"
producer-name	矿工的账号名	"potato"
signature-provider	矿工的公钥私钥，用于签名	PC6MRyAjQq8ud7hVNYcfnVPJqcVpscN5So8BhtHuGYqET5GDW5CV=KEY:5KQwrPbwdL6PhXujxW37FSSQZ1JiwsST4cqQzDeyXtP79zkvFD3
p2p-server-address	p2p服务监听地址，默认监听0.0.0.0:9876	"0.0.0.0:9876"
http-server-address	http服务器监听地址，如果不□想提供http服务，可以把地址□留空，则不http服务不启动。注意不填写会使用默认地址127.0.0.1:8888	"127.0.0.1:8888"
enable-stale-production	启动后立即开始生产块。如果不是BIOS节点，这里填false。	true

参数名	示例	说明
p2p-peer-address	其它节点的p2p同步地址，用于同步数据，此参数可以有多个，即连接到多个节点。	"192.168.1.2:9876"

6. 创世节点配置

6.1. 启动参数配置

1. 创建pcoin系统账号的公私钥，参见4. 创建钱包和公私钥
2. 修改genesis.json的initial_key为上一步生成的公钥
3. 启动节点，注意填写第1步生成的公私钥

```
 nodepc --agent-name "potato bios" --producer-name "potato" \
     --signature-provider [public key]=KEY:[private key] \
     --p2p-server-address "0.0.0.0:9876" --http-server-address
"127.0.0.1:8888" --enable-stale-production
```

6.2. 部署系统合约

1. 创建系统账号

   OwnerKey与ActiveKey相同，是6.1生成的公钥

```
clpc create account potato pc.token [OwnerKey] [ActiveKey] -p potato
clpc create account potato pc.msig [OwnerKey] [ActiveKey] -p potato
clpc create account potato pc.bpay [OwnerKey] [ActiveKey] -p potato
clpc create account potato pc.names [OwnerKey] [ActiveKey] -p potato
clpc create account potato pc.ram [OwnerKey] [ActiveKey] -p potato
clpc create account potato pc.ramfee [OwnerKey] [ActiveKey] -p potato
clpc create account potato pc.saving [OwnerKey] [ActiveKey] -p potato
clpc create account potato pc.stake [OwnerKey] [ActiveKey] -p potato
clpc create account potato pc.vpay [OwnerKey] [ActiveKey] -p potato
```

2. 部署合约，请确认在代码根目录

```
clpc set contract pc.token build/contracts/pc.token -p pc.token
clpc set contract pc.msig build/contracts/pc.msig -p pc.msig
```

3. 创建货币

```
clpc push action pc.token create '["potato", "10000000000.0000 PC", 0, 0,
0]' -p pc.token
clpc push action pc.token issue '["potato", "10000000000.0000 PC", "issue"]'
-p potato
```

4. 部署系统合约，请确认在代码根目录

```
clpc set contract potato build/contracts/pc.system -p potato
clpc push action potato setpriv '["pc.msig", 1]' -p potato@active
```

7. 矿工节点配置

1. 注册矿工账号

1. 创建矿工账号的公私钥，参见4. 创建钱包和公私钥

2. 创建矿工账号

```
clpc system newaccount --stake-net [quantity] --stake-cpu [quantity] --buy-
ram-kbytes 8192 \
    [creater] [name] [OwnerKey] [ActiveKey] -p [creater]
# 参数说明
# creater 矿工账号的创建者账号。
# name 矿工账号的名字。
# quantity 购买带宽的资源，从创建者账号扣取。
# OwnerKey与ActiveKey相同，矿工账号的公钥，即上步生成的公钥。

# 示例
clpc system newaccount --stake-net "50.0000 PC" --stake-cpu "50.0000 PC" --
buy-ram-kbytes 8888888 potato pcbpa \
PC7n1U9Z2NQeVEvQZYjHCedNXRVWshmmuGH2j3r6bD4c8fH4U8QL
PC7n1U9Z2NQeVEvQZYjHCedNXRVWshmmuGH2j3r6bD4c8fH4U8QL -p potato
```

3. 转账给矿工账号

```
clpc transfer [from] [recipient] [amount] [momo]
# 参数说明
# from 转账账号
# recipient 接收转账账号
# amount 金额
# momo 注释

# 示例
clpc transfer potato pcbpa "25000000.0000 PC" "trans to pcbpa"
```

4. 注册成矿工

```
clpc system regproducer [name] [ActiveKey] [url]
# 参数说明
# name 矿工账号
# ActiveKey 矿工账号的公钥
# url 矿工的网站

#示例
clpc system regproducer pcbpa
PC7n1U9Z2NQeVEvQZYjHCedNXRVWshmmuGH2j3r6bD4c8fH4U8QL
https://202.46.36.57:8888
```

5. 抵押相应的资源

```
clpc system delegatebw [name] [name] [stake_net_quantity]
[stake_cpu_quantity] --transfer
# 参数说明
# name 矿工账号
# stake_net_quantity 抵押的网络带宽资源
# stake_cpu_quantity 抵押的CPU带宽资源

# 示例
clpc system delegatebw pcbpa pcbpa '12000000.0000 PC' '12000000.0000 PC' --
transfer
```

6. 投票给自己

```
clpc system voteproducer prods [voter] [producers] -p [voter]
# 参数说明
# voter 矿工账号
# producers 抵押的网络带宽资源

#示例
clpc system voteproducer prods pcbpa pcbpa
```

2. 启动参数配置

1. 创建potato系统账号的公私钥，参见创建钱包和公私钥

2. 修改genesis.json的initial_key为创世节点(BIOS)生成的公钥

3. 启动节点，注意填写第1步生成的公私钥

   假设已知bios节点202.46.36.56:9876，BP节点202.46.36.58:9876。

```
nodepc --agent-name "potato pb a" --producer-name "pba" \
    --signature-provider [public key]=KEY:[private key] \
    --p2p-server-address "0.0.0.0:9876" --http-server-address
"127.0.0.1:8888" \
    --p2p-peer-address "202.46.36.56:9876" --p2p-peer-address
"202.46.36.58:9876"
```

8. 备份节点配置

备份节点配置与矿工节点配置类似，只是不注册成矿工，备份节点常用来数据查询。建议开启bnet_plugin和mongo_db_plugin插件。 bnet_plugin是一个用来快速同步区块数据的插件，mongo_db_plugin是保存交易信息到mongodb的插件。

假设已知bios节点202.46.36.56:9876，BP节点202.46.36.57:9876、202.46.36.58:9876，并且这三个节点都开启了bnet_plugin插件。

```
nodepc --agent-name "potato pb a" --producer-name "pba" \
  --signature-provider [public key]=KEY:[private key] \
  --p2p-server-address "0.0.0.0:9876" --http-server-address "127.0.0.1:8888" \
  --p2p-peer-address "202.46.36.56:9876" --p2p-peer-address "202.46.36.57:9876" --
p2p-peer-address "202.46.36.58:9876" \
  --plugin potato::mongo_db_plugin --mongodb-uri mongodb://127.0.0.1:27017/POTATO
\
  --plugin potato::bnet_plugin --bnet-endpoint "0.0.0.0:4321" --bnet-threads 4 \
  --bnet-connect "202.46.36.56:4321" --bnet-connect "202.46.36.56:4321" --bnet-
connect "202.46.36.56:4321"
```

9. 建议

1. 从安全角度来说，不要私钥泄露给他人，知道账户私钥后，可以随意操纵账户。
2. 如果使用助记词生成的私钥，请使用复杂度较高的助记词。
3. 上面传入很多参数都是演示用的，如果正式部署，建议使用配置文件的方式。
4. 建议使用SIGINT信号来关闭节点，防止程序产生脏数据。

   ```
   pkill -2 kpcd nodepc
   ```

5. 如果有脏数据产生，请使用--replay-blockchain参数或者--delete-all-blocks参数启动节点，重新同步数据。