区块链期末项目热身

内容

1. 使用已有的开源区块链系统FISCO-BCOS，完成私有链的搭建以及新节点的加入。（截图说明搭建流程）
2. 自行编写一个智能合约并部署到私有链上，同时完成合约调用。（截图说明部署流程）
3. 使用命令查看一个区块，并对各个字段进行解释。

环境和参考

> 环境说明：使用的是 win10 环境下的 wsl ， linux 版本为 ubuntu 18.04

```
root@LAPTOP-QTCGESHO:/mnt/d/blog# uname -a
Linux LAPTOP-QTCGESHO 4.4.0-19041-Microsoft #488-Microsoft Mon Sep 01 13:43:00 PST 2020 x86_64 x86_64 x86_64 GNU/Linux
```

> 教程参考为 **fisco 官方文档**

单群组 FISCO BCOS 联盟链的搭建

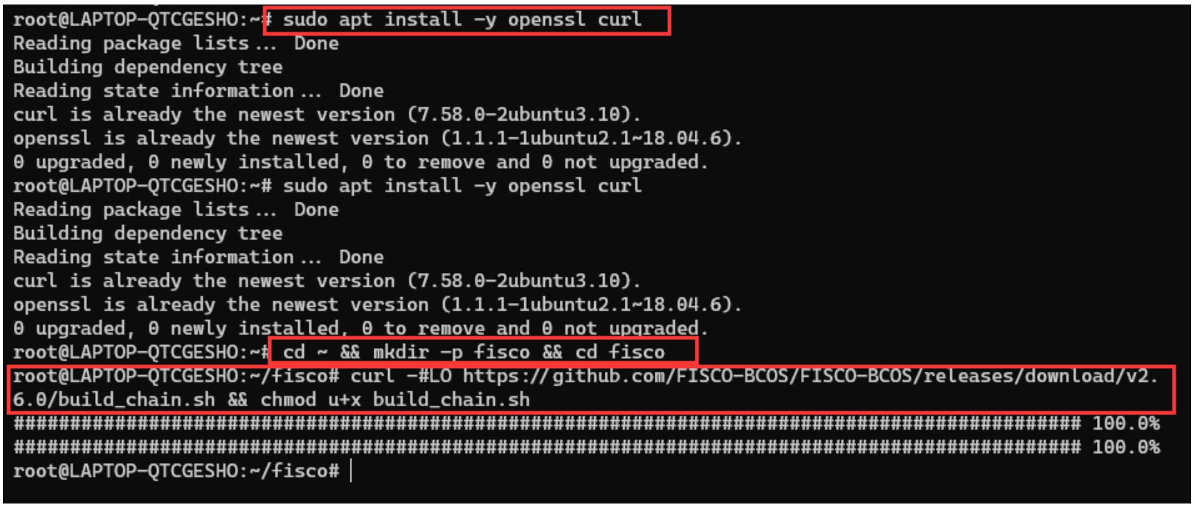
准备环境

根据教程步骤

- 安装 `curl` 依赖
- 创建操作目录
- 下载对应脚本 `build_chain.sh`

```
sudo apt install -y openssl curl
cd ~ && mkdir -p fisco && cd fisco
curl -#LO https://github.com/FISCO-BCOS/FISCO-BCOS/releases/download/v2.6.0/build_chain.sh && chmod u+x build_chain.sh
```

结果如下



```
root@LAPTOP-QTCGESHO:~# sudo apt install -y openssl curl
Reading package lists... Done
Building dependency tree
Reading state information... Done
curl is already the newest version (7.58.0-2ubuntu3.10).
openssl is already the newest version (1.1.1-1ubuntu2.1~18.04.6).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@LAPTOP-QTCGESHO:~# sudo apt install -y openssl curl
Reading package lists... Done
Building dependency tree
Reading state information... Done
curl is already the newest version (7.58.0-2ubuntu3.10).
openssl is already the newest version (1.1.1-1ubuntu2.1~18.04.6).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@LAPTOP-QTCGESHO:~# cd ~ && mkdir -p fisco && cd fisco
root@LAPTOP-QTCGESHO:~/fisco# curl -#LO https://github.com/FISCO-BCOS/FISCO-BCOS/releases/download/v2.
6.0/build_chain.sh && chmod u+x build_chain.sh
############################################################################################### 100.0%
############################################################################################### 100.0%
root@LAPTOP-QTCGESHO:~/fisco#
```

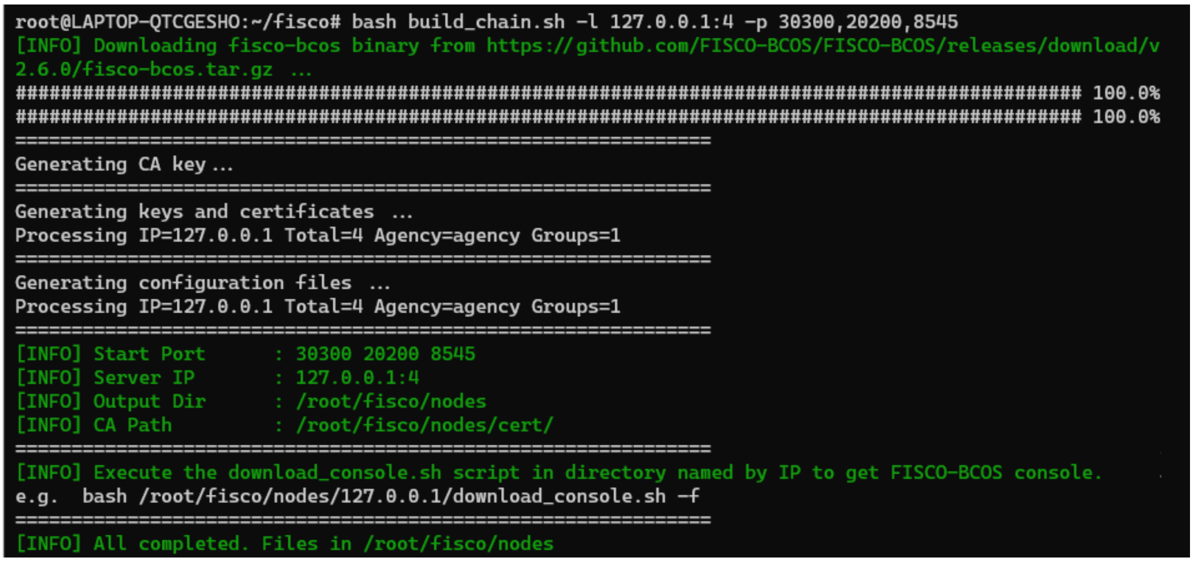
搭建单群组4节点联盟链

命令为

```
bash build_chain.sh -l 127.0.0.1:4 -p 30300,20200,8545
```

> - -p 指定起始端口，分别是 p2p_port, channel_port, jsonrpc_port
> - -l 指定对应 ip 和端口

结果如图：



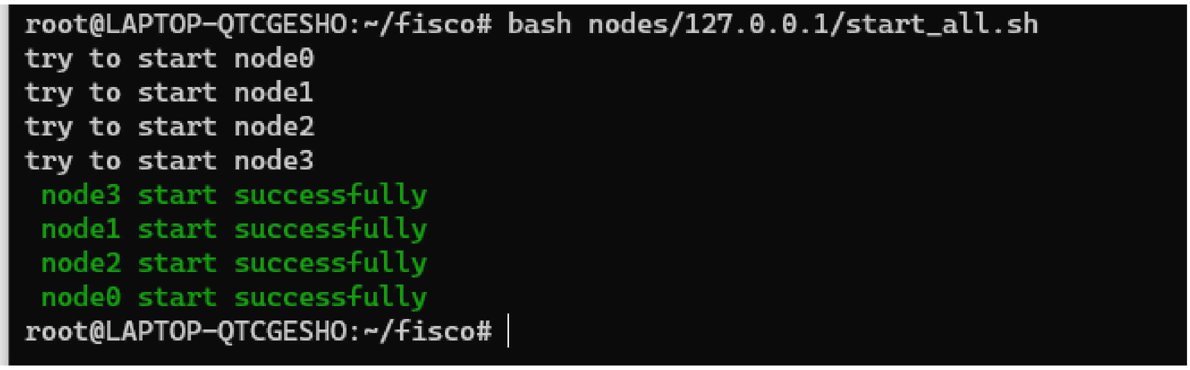
```
root@LAPTOP-QTCGESHO:~/fisco# bash build_chain.sh -l 127.0.0.1:4 -p 30300,20200,8545
[INFO] Downloading fisco-bcos binary from https://github.com/FISCO-BCOS/FISCO-BCOS/releases/download/v
2.6.0/fisco-bcos.tar.gz ...
############################################################################################# 100.0%
############################################################################################# 100.0%
==============================================================
Generating CA key...
==============================================================
Generating keys and certificates ...
Processing IP=127.0.0.1 Total=4 Agency=agency Groups=1
==============================================================
Generating configuration files ...
Processing IP=127.0.0.1 Total=4 Agency=agency Groups=1
==============================================================
[INFO] Start Port      : 30300 20200 8545
[INFO] Server IP       : 127.0.0.1:4
[INFO] Output Dir      : /root/fisco/nodes
[INFO] CA Path         : /root/fisco/nodes/cert/
==============================================================
[INFO] Execute the download_console.sh script in directory named by IP to get FISCO-BCOS console.
e.g.  bash /root/fisco/nodes/127.0.0.1/download_console.sh -f
==============================================================
[INFO] All completed. Files in /root/fisco/nodes
```

启动联盟链

执行如下命令：

```
bash nodes/127.0.0.1/start_all.sh
```

结果如图：

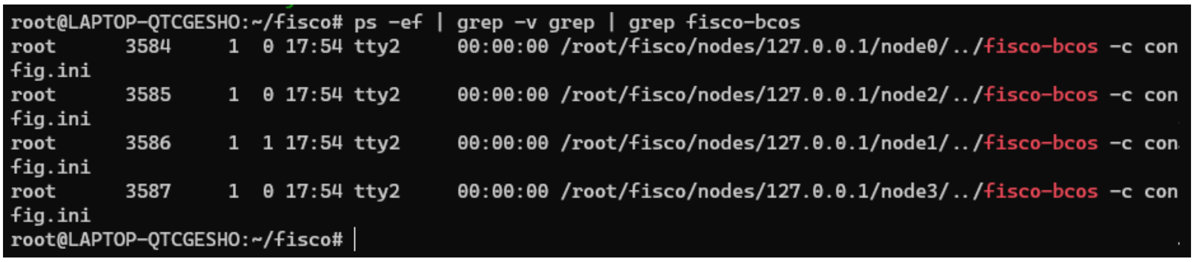


```
root@LAPTOP-QTCGESHO:~/fisco# bash nodes/127.0.0.1/start_all.sh
try to start node0
try to start node1
try to start node2
try to start node3
 node3 start successfully
 node1 start successfully
 node2 start successfully
 node0 start successfully
root@LAPTOP-QTCGESHO:~/fisco# 
```

检查进程和日志输出

检查进程：

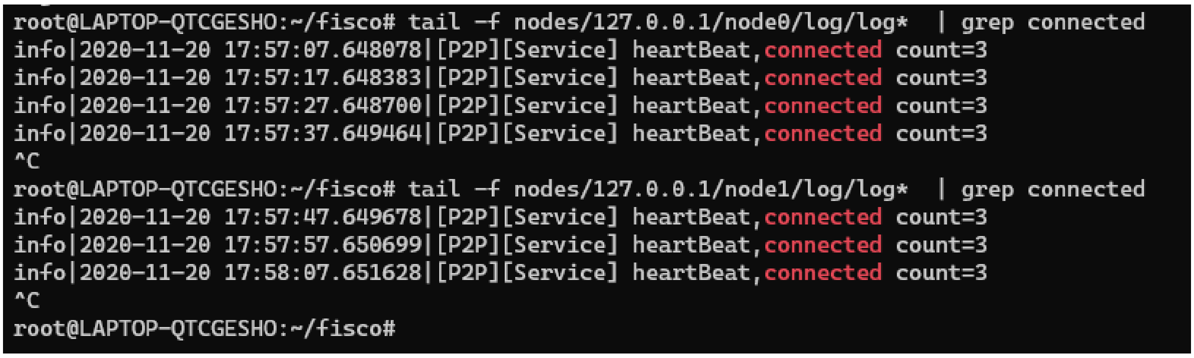
```
ps -ef | grep -v grep | grep fisco-bcos
```



```
root@LAPTOP-QTCGESHO:~/fisco# ps -ef | grep -v grep | grep fisco-bcos
root      3584     1  0 17:54 tty2     00:00:00 /root/fisco/nodes/127.0.0.1/node0/../fisco-bcos -c con
fig.ini
root      3585     1  0 17:54 tty2     00:00:00 /root/fisco/nodes/127.0.0.1/node2/../fisco-bcos -c con
fig.ini
root      3586     1  1 17:54 tty2     00:00:00 /root/fisco/nodes/127.0.0.1/node1/../fisco-bcos -c con
fig.ini
root      3587     1  0 17:54 tty2     00:00:00 /root/fisco/nodes/127.0.0.1/node3/../fisco-bcos -c con
fig.ini
root@LAPTOP-QTCGESHO:~/fisco# 
```

检查日志输出：

```
tail -f nodes/127.0.0.1/node0/log/log*  | grep connected
```



```
root@LAPTOP-QTCGESHO:~/fisco# tail -f nodes/127.0.0.1/node0/log/log*  | grep connected
info|2020-11-20 17:57:07.648078|[P2P][Service] heartBeat,connected count=3
info|2020-11-20 17:57:17.648383|[P2P][Service] heartBeat,connected count=3
info|2020-11-20 17:57:27.648700|[P2P][Service] heartBeat,connected count=3
info|2020-11-20 17:57:37.649464|[P2P][Service] heartBeat,connected count=3
^C
root@LAPTOP-QTCGESHO:~/fisco# tail -f nodes/127.0.0.1/node1/log/log*  | grep connected
info|2020-11-20 17:57:47.649678|[P2P][Service] heartBeat,connected count=3
info|2020-11-20 17:57:57.650699|[P2P][Service] heartBeat,connected count=3
info|2020-11-20 17:58:07.651628|[P2P][Service] heartBeat,connected count=3
^C
root@LAPTOP-QTCGESHO:~/fisco#
```

> 图中分别查看了两个节点 node0 和 node1 的日志

配置并且使用控制台

> 选择的基于 `Java JDK` 实现的控制台2.6

准备依赖

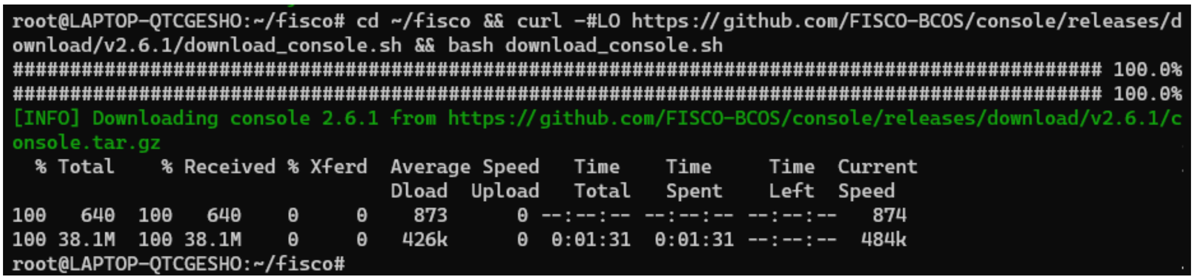
> 需要安装 JDK，命令如下

```
sudo apt install -y default-jdk # ubuntu系统安装java
```

- 获取控制台：

```
cd ~/fisco && curl -#LO https://github.com/FISCO-BCOS/console/releases/download/v2.6.1/download_console.sh && bash download_console.sh
```

结果如下：



```
root@LAPTOP-QTCGESHO:~/fisco# cd ~/fisco && curl -#LO https://github.com/FISCO-BCOS/console/releases/d
ownload/v2.6.1/download_console.sh && bash download_console.sh
############################################################################################### 100.0%
############################################################################################### 100.0%
[INFO] Downloading console 2.6.1 from https://github.com/FISCO-BCOS/console/releases/download/v2.6.1/c
onsole.tar.gz
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   640  100   640    0     0    873      0 --:--:-- --:--:-- --:--:--   874
100 38.1M  100 38.1M    0     0   426k      0  0:01:31  0:01:31 --:--:--  484k
root@LAPTOP-QTCGESHO:~/fisco#
```

- 拷贝控制台配置文件：

```
cp -n console/conf/config-example.toml console/conf/config.toml # 最新版本控制台使用如下命令拷贝配置文件
```

> 若节点未采用默认端口，请将文件中的20200替换成节点对应的channel端口。

- 配置控制台证书：

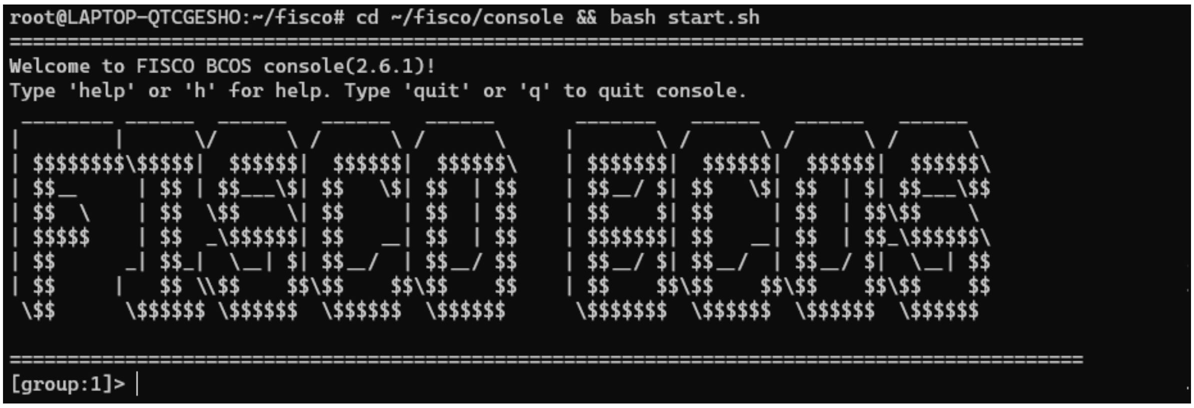
```
cp -r nodes/127.0.0.1/sdk/* console/conf/
```

启动控制台

- 启动控制台：

```
cd ~/fisco/console && bash start.sh
```

输出如下，说明控制台配置成功：

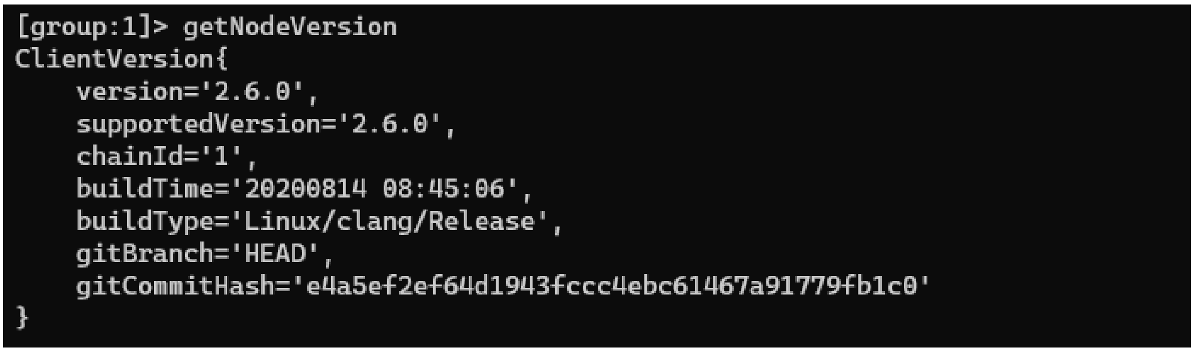


```
root@LAPTOP-QTCGESHO:~/fisco# cd ~/fisco/console && bash start.sh
=============================================================================================
Welcome to FISCO BCOS console(2.6.1)!
Type 'help' or 'h' for help. Type 'quit' or 'q' to quit console.
 ________ ______  ______   ______   ______       _______   ______   ______   ______
|        |      \/      \ /      \ /      \     |       \ /      \ /      \ /      \
| $$$$$$$$\$$$$$|  $$$$$$|  $$$$$$|  $$$$$$\    | $$$$$$$|  $$$$$$|  $$$$$$|  $$$$$$\
| $$__     | $$ | $$___\$| $$   \$| $$  | $$    | $$__/ $| $$   \$| $$  | $| $$___\$$
| $$  \    | $$  \$$    \| $$     | $$  | $$    | $$    $| $$     | $$  | $$\$$    \
| $$$$$    | $$  _\$$$$$$| $$   __| $$  | $$    | $$$$$$$| $$   __| $$  | $$_\$$$$$$\
| $$      _| $$_|  \__| $| $$__/  | $$__/ $$    | $$__/ $| $$__/  | $$__/ $|  \__| $$
| $$     |   $$ \\$$    $$\$$    $$\$$    $$    | $$    $$\$$    $$\$$    $$\$$    $$
 \$$      \$$$$$$ \$$$$$$  \$$$$$$  \$$$$$$      \$$$$$$$  \$$$$$$  \$$$$$$  \$$$$$$

=============================================================================================
[group:1]> 
```

通过控制台获取信息

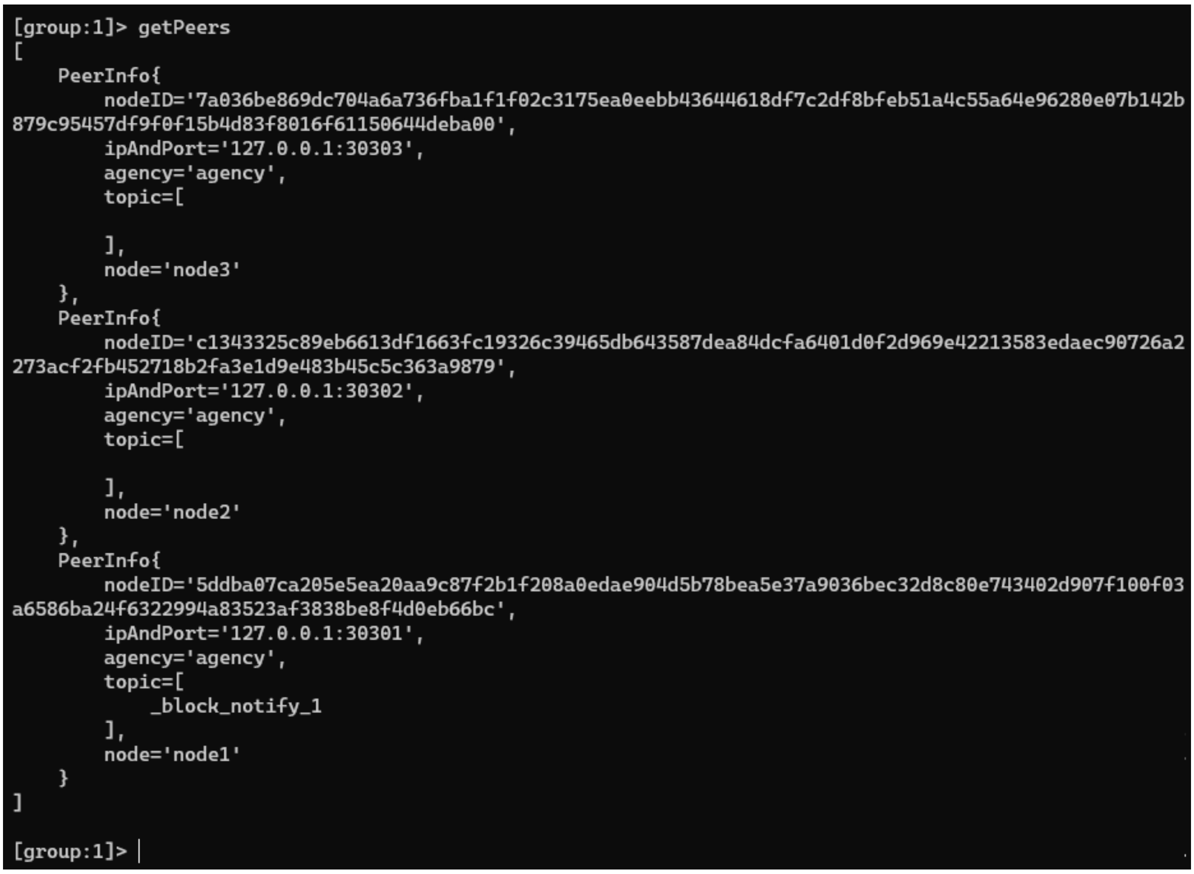
在控制台执行 `getNodeVersion` 和 `getPeers` 获取客户端版本和节点信息

获取客户端版本：



```
[group:1]> getNodeVersion
ClientVersion{
    version='2.6.0',
    supportedVersion='2.6.0',
    chainId='1',
    buildTime='20200814 08:45:06',
    buildType='Linux/clang/Release',
    gitBranch='HEAD',
    gitCommitHash='e4a5ef2ef64d1943fccc4ebc61467a91779fb1c0'
}
```

获取节点链接信息：

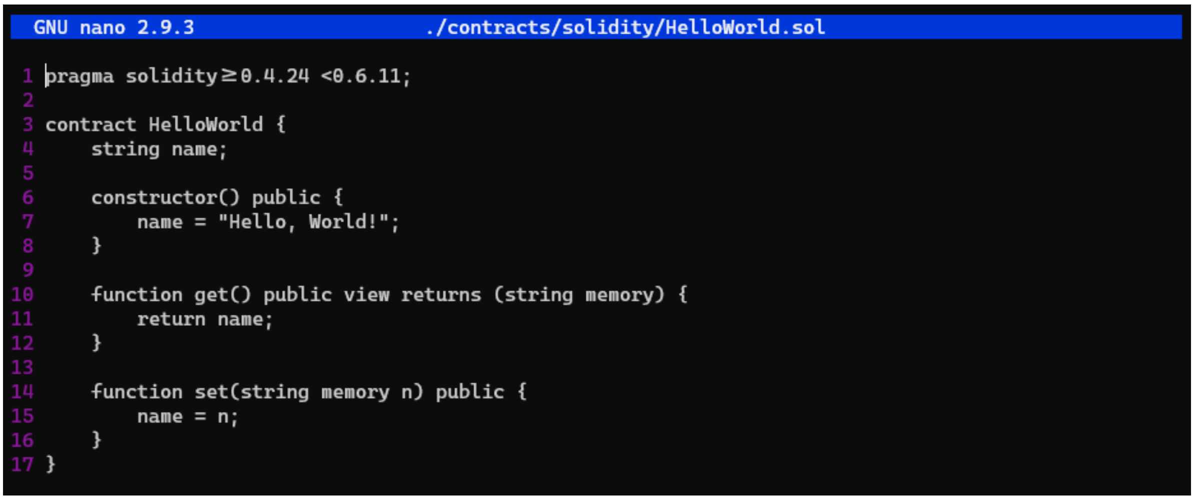


```
[group:1]> getPeers
[
    PeerInfo{
        nodeID='7a036be869dc704a6a736fba1f1f02c3175ea0eebb43644618df7c2df8bfeb51a4c55a64e96280e07b142b
879c95457df9f0f15b4d83f8016f61150644deba00',
        ipAndPort='127.0.0.1:30303',
        agency='agency',
        topic=[

        ],
        node='node3'
    },
    PeerInfo{
        nodeID='c1343325c89eb6613df1663fc19326c39465db643587dea84dcfa6401d0f2d969e42213583edaec90726a2
273acf2fb452718b2fa3e1d9e483b45c5c363a9879',
        ipAndPort='127.0.0.1:30302',
        agency='agency',
        topic=[

        ],
        node='node2'
    },
    PeerInfo{
        nodeID='5ddba07ca205e5ea20aa9c87f2b1f208a0edae904d5b78bea5e37a9036bec32d8c80e743402d907f100f03
a6586ba24f6322994a83523af3838be8f4d0eb66bc',
        ipAndPort='127.0.0.1:30301',
        agency='agency',
        topic=[
            _block_notify_1
        ],
        node='node1'
    }
]

[group:1]> |
```

部署和调用智能合约

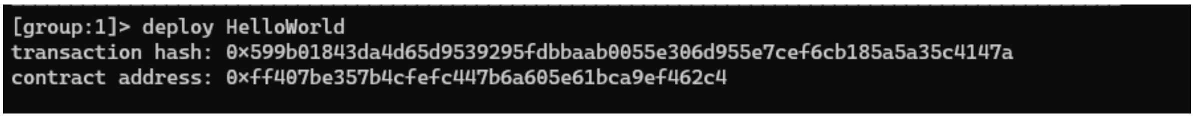
部署 HelloWorld 合约

在控制台目录下 `/contracts/solidity/` 已经已经有 `HelloWorld.sol` ，查看代码如下：



```
 GNU nano 2.9.3                    ./contracts/solidity/HelloWorld.sol

pragma solidity>=0.4.24 <0.6.11;

contract HelloWorld {
    string name;

    constructor() public {
        name = "Hello, World!";
    }

    function get() public view returns (string memory) {
        return name;
    }

    function set(string memory n) public {
        name = n;
    }
}
```

部署该合约，得到如下输出



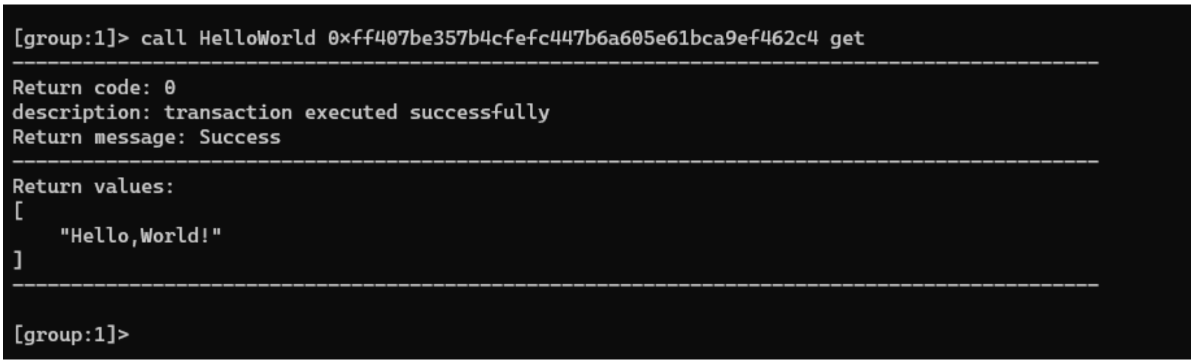
```
[group:1]> deploy HelloWorld
transaction hash: 0x599b01843da4d65d9539295fdbbaab0055e306d955e7cef6cb185a5a35c4147a
contract address: 0xff407be357b4cfefc447b6a605e61bca9ef462c4
```

> 返回的合约地址 `0xff407be357b4cfefc447b6a605e61bca9ef462c4` 比较重要，因为后续调用合约需要用到

调用 HelloWord 合约

查看变量

调用 get 接口，获取 `name` 变量，输出如下：



```
[group:1]> call HelloWorld 0xff407be357b4cfefc447b6a605e61bca9ef462c4 get
---------------------------------------------------------------------------------------------
Return code: 0
description: transaction executed successfully
Return message: Success
---------------------------------------------------------------------------------------------
Return values:
[
    "Hello,World!"
]
---------------------------------------------------------------------------------------------

[group:1]>
```

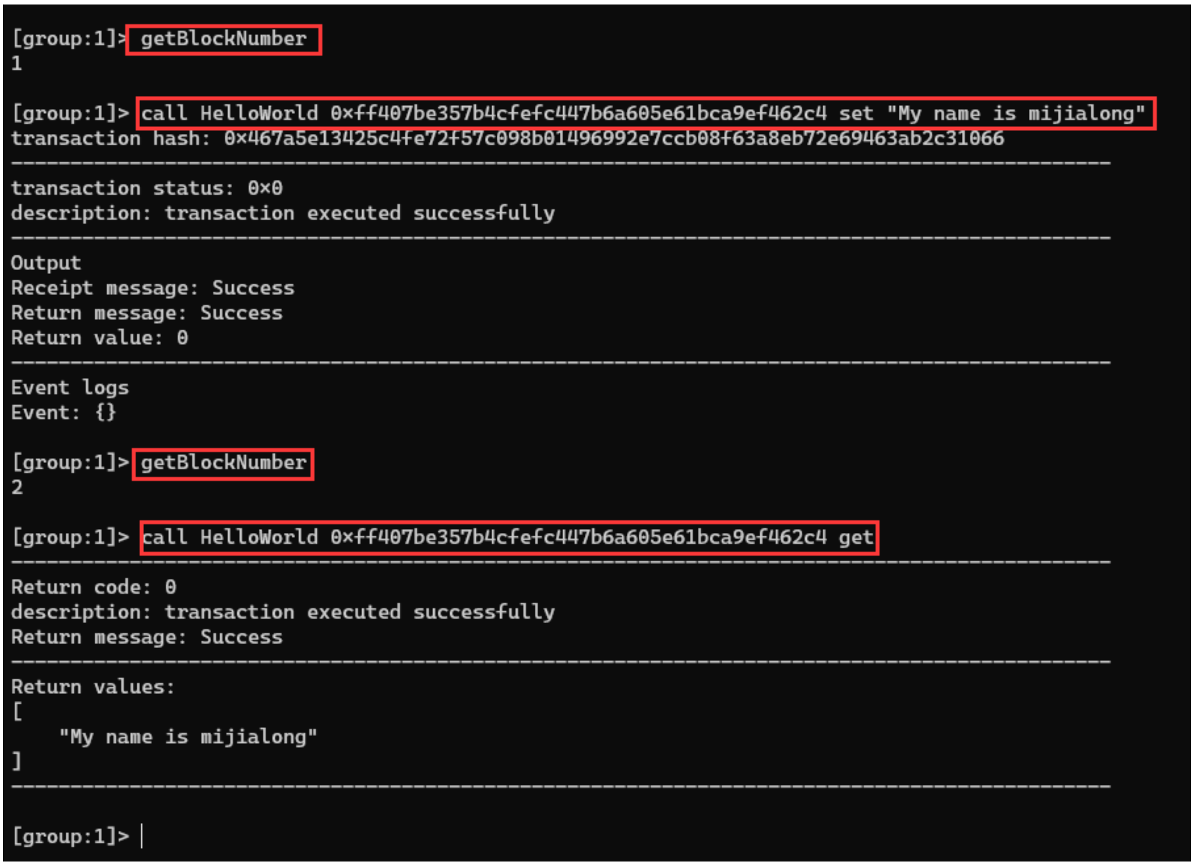
修改变量

> 修改变量会导致块增加

进行如下操作

- 先查看一次当前区块数量
- 设置 `name` 变量值为 `My name is mijialong.`
- 再次查看当前区块数量
- 再次调用 get 接口查看 `name` 变量的值

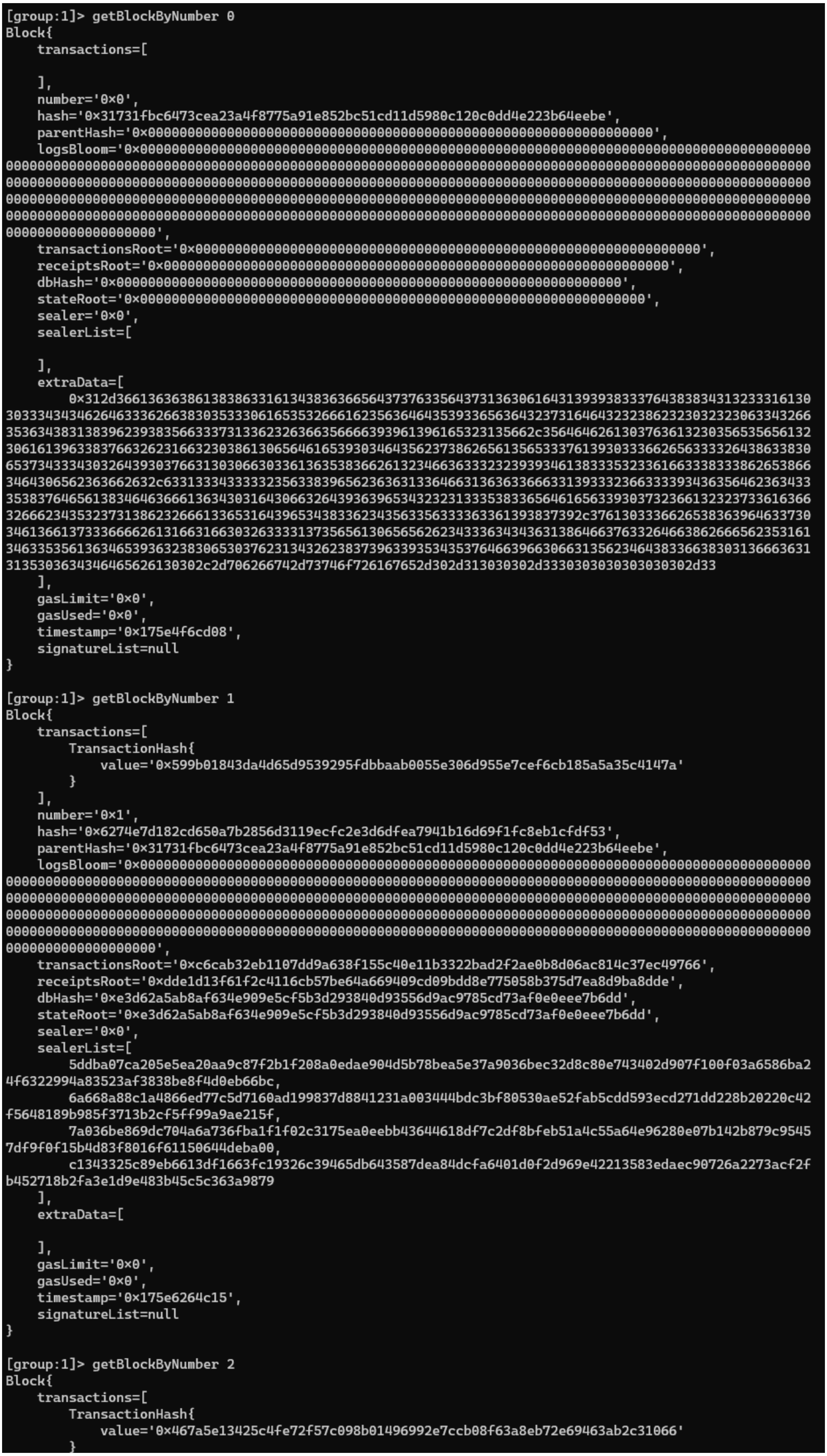
操作结果如下：



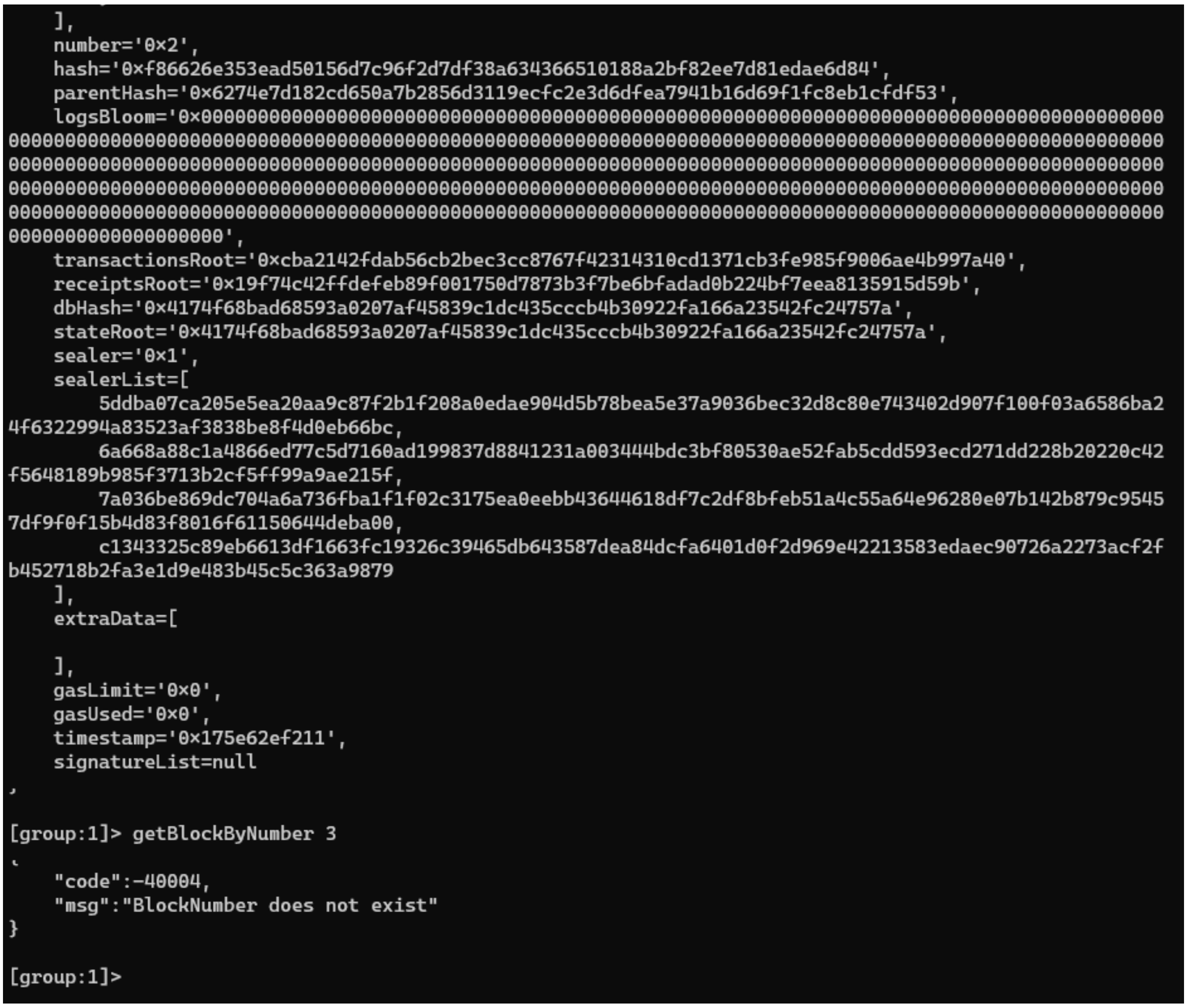
```
[group:1]> getBlockNumber
1

[group:1]> call HelloWorld 0xff407be357b4cfefc447b6a605e61bca9ef462c4 set "My name is mijialong"
transaction hash: 0x467a5e13425c4fe72f57c098b01496992e7ccb08f63a8eb72e69463ab2c31066
---------------------------------------------------------------------------------------------
transaction status: 0x0
description: transaction executed successfully
---------------------------------------------------------------------------------------------
Output
Receipt message: Success
Return message: Success
Return value: 0
---------------------------------------------------------------------------------------------
Event logs
Event: {}

[group:1]> getBlockNumber
2

[group:1]> call HelloWorld 0xff407be357b4cfefc447b6a605e61bca9ef462c4 get
---------------------------------------------------------------------------------------------
Return code: 0
description: transaction executed successfully
Return message: Success
---------------------------------------------------------------------------------------------
Return values:
[
    "My name is mijialong"
]
---------------------------------------------------------------------------------------------

[group:1]> |
```

查看区块

可以通过 `getBlockByNumber` 方法查看每个区块：



```
[group:1]> getBlockByNumber 0
Block{
    transactions=[

    ],
    number='0×0',
    hash='0×31731fbc6473cea23a4f8775a91e852bc51cd11d5980c120c0dd4e223b64eebe',
    parentHash='0×0000000000000000000000000000000000000000000000000000000000000000',
    logsBloom='0×0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000',
    transactionsRoot='0×0000000000000000000000000000000000000000000000000000000000000000',
    receiptsRoot='0×0000000000000000000000000000000000000000000000000000000000000000',
    dbHash='0×0000000000000000000000000000000000000000000000000000000000000000',
    stateRoot='0×0000000000000000000000000000000000000000000000000000000000000000',
    sealer='0×0',
    sealerList=[

    ],
    extraData=[
        0×312d3661363638613838633161343836366564373763356437313630616431393938333764383834313233316130
303334343462646333626638303533306165353266616235636464353933656364323731646432323862323032323063343266
353634383138396239383566333731336232636635666639396139616532313566 2c3564646261303763613230356535656132
306161396338376632623166323038613065646165393034643562373862656135653337613930333662656333326438633830
653734333430326439303766313030663033613635383662613234663633323239393461383335323361663338333862653866
3464306562363662632c63313334333332356338396562363631336466313636336663313933323663333934363564623634 33
353837646561383464636661363430316430663264393639653432323133353833656461656339303732366132323733616366
32666234353237313862326661336531643965343833623435633563333633613938373932c37613033366265383639646337 30
346136613733366662613166316630326333313735656130656562623433363434363138646637633264663862666562353161
346335356136346539363238306530376231343262383739633935343537646639663066313562346438336638303136663631
3135303634346465626130302c2d706266742d73746f726167652d302d313030302d3330303030303030302d33
    ],
    gasLimit='0×0',
    gasUsed='0×0',
    timestamp='0×175e4f6cd08',
    signatureList=null
}

[group:1]> getBlockByNumber 1
Block{
    transactions=[
        TransactionHash{
            value='0×599b01843da4d65d9539295fdbbaab0055e306d955e7cef6cb185a5a35c4147a'
        }
    ],
    number='0×1',
    hash='0×6274e7d182cd650a7b2856d3119ecfc2e3d6dfea7941b16d69f1fc8eb1cfdf53',
    parentHash='0×31731fbc6473cea23a4f8775a91e852bc51cd11d5980c120c0dd4e223b64eebe',
    logsBloom='0×0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000',
    transactionsRoot='0×c6cab32eb1107dd9a638f155c40e11b3322bad2f2ae0b8d06ac814c37ec49766',
    receiptsRoot='0×dde1d13f61f2c4116cb57be64a669409cd09bdd8e775058b375d7ea8d9ba8dde',
    dbHash='0×e3d62a5ab8af634e909e5cf5b3d293840d93556d9ac9785cd73af0e0eee7b6dd',
    stateRoot='0×e3d62a5ab8af634e909e5cf5b3d293840d93556d9ac9785cd73af0e0eee7b6dd',
    sealer='0×0',
    sealerList=[
        5ddba07ca205e5ea20aa9c87f2b1f208a0edae904d5b78bea5e37a9036bec32d8c80e743402d907f100f03a6586ba2
4f6322994a83523af3838be8f4d0eb66bc,
        6a668a88c1a4866ed77c5d7160ad199837d8841231a003444bdc3bf80530ae52fab5cdd593ecd271dd228b20220c42
f5648189b985f3713b2cf5ff99a9ae215f,
        7a036be869dc704a6a736fba1f1f02c3175ea0eebb43644618df7c2df8bfeb51a4c55a64e96280e07b142b879c9545
7df9f0f15b4d83f8016f61150644deba00,
        c1343325c89eb6613df1663fc19326c39465db643587dea84dcfa6401d0f2d969e42213583edaec90726a2273acf2f
b452718b2fa3e1d9e483b45c5c363a9879
    ],
    extraData=[

    ],
    gasLimit='0×0',
    gasUsed='0×0',
    timestamp='0×175e6264c15',
    signatureList=null
}

[group:1]> getBlockByNumber 2
Block{
    transactions=[
        TransactionHash{
            value='0×467a5e13425c4fe72f57c098b01496992e7ccb08f63a8eb72e69463ab2c31066'
        }
```



```
    ],
    number='0x2',
    hash='0xf86626e353ead50156d7c96f2d7df38a634366510188a2bf82ee7d81edae6d84',
    parentHash='0x6274e7d182cd650a7b2856d3119ecfc2e3d6dfea7941b16d69f1fc8eb1cfdf53',
    logsBloom='0x0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000',
    transactionsRoot='0xcba2142fdab56cb2bec3cc8767f42314310cd1371cb3fe985f9006ae4b997a40',
    receiptsRoot='0x19f74c42ffdefeb89f001750d7873b3f7be6bfadad0b224bf7eea8135915d59b',
    dbHash='0x4174f68bad68593a0207af45839c1dc435cccb4b30922fa166a23542fc24757a',
    stateRoot='0x4174f68bad68593a0207af45839c1dc435cccb4b30922fa166a23542fc24757a',
    sealer='0x1',
    sealerList=[
        5ddba07ca205e5ea20aa9c87f2b1f208a0edae904d5b78bea5e37a9036bec32d8c80e743402d907f100f03a6586ba2
4f6322994a83523af3838be8f4d0eb66bc,
        6a668a88c1a4866ed77c5d7160ad199837d8841231a003444bdc3bf80530ae52fab5cdd593ecd271dd228b20220c42
f5648189b985f3713b2cf5ff99a9ae215f,
        7a036be869dc704a6a736fba1f1f02c3175ea0eebb43644618df7c2df8bfeb51a4c55a64e96280e07b142b879c9545
7df9f0f15b4d83f8016f61150644deba00,
        c1343325c89eb6613df1663fc19326c39465db643587dea84dcfa6401d0f2d969e42213583edaec90726a2273acf2f
b452718b2fa3e1d9e483b45c5c363a9879
    ],
    extraData=[

    ],
    gasLimit='0x0',
    gasUsed='0x0',
    timestamp='0x175e62ef211',
    signatureList=null
,

[group:1]> getBlockByNumber 3
{
    "code":-40004,
    "msg":"BlockNumber does not exist"
}

[group:1]>
```

参考文档中的 getBlockByNumber 函数接口，具体如下：

- 参数
  - `groupID` : `unsigned int` - 群组ID
  - `blockNumber` : `string` - 区块高度(十进制字符串或0x开头的十六进制字符串)
  - `includeTransactions` : `bool` - 包含交易标志(true 显示交易详细信息，false 仅显示交易的hash)
- 返回值
  - object - 区块信息，字段如下：
    - `extraData` : `array` - 附加数据
    - `gasLimit` : `string` - 区块中允许的gas最大值
    - `gasUsed` : `string` - 区块中所有交易消耗的gas
    - `hash` : `string` - 区块哈希
    - `logsBloom` : `string` - log 的布隆过滤器值
    - `number` : `string` - 区块高度
    - `parentHash` : `string` - 父区块哈希
    - `sealer` : `string` - 共识节点序号
    - `sealerList` : `array` - 共识节点列表
    - `stateRoot` : `string` - 状态根哈希
    - `timestamp` : `string` - 时间戳
    - `transactions` : `array` - 交易列表，当 `includeTransactions` 为 false 时，显示交易的哈希。当 `includeTransactions` 为 true 时，显示交易详细信息

以 `getBlockByNumber 2` 的结果为例：

```
Block{
    transactions=[
        TransactionHash{
            value='0x467a5e13425c4fe72f57c098b01496992e7ccb08f63a8eb72e69463ab2c31066' // 转移的值
        }
    ], // 交易列表
    number='0x2', // 区块高度
    hash='0xf86626e353ead50156d7c96f2d7df38a634366510188a2bf82ee7d81edae6d84', // 区块哈希
    parentHash='0x6274e7d182cd650a7b2856d3119ecfc2e3d6dfea7941b16d69f1fc8eb1cfdf53', // 父区块哈希
    logsBloom='0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000', // log 的布隆过滤器值
    transactionsRoot='0xcba2142fdab56cb2bec3cc8767f42314310cd1371cb3fe985f9006ae4b997a40', // 交易根哈希
    receiptsRoot='0x19f74c42ffdefeb89f001750d7873b3f7be6bfadad0b224bf7eea8135915d59b', // 接受根哈希
    dbHash='0x4174f68bad68593a0207af45839c1dc435cccb4b30922fa166a23542fc24757a', // 数据库哈希
    stateRoot='0x4174f68bad68593a0207af45839c1dc435cccb4b30922fa166a23542fc24757a', // 状态根哈希
    sealer='0x1', // 共识节点序号
    sealerList=[
        5ddba07ca205e5ea20aa9c87f2b1f208a0edae904d5b78bea5e37a9036bec32d8c80e743402d907f100f03a6586ba24f6322994a83523af3838be8f4d0eb66bc,
        6a668a88c1a4866ed77c5d7160ad199837d8841231a003444bdc3bf80530ae52fab5cdd593ecd271dd228b20220c42f5648189b985f3713b2cf5ff99a9ae215f,
        7a036be869dc704a6a736fba1f1f02c3175ea0eebb43644618df7c2df8bfeb51a4c55a64e96280e07b142b879c95457df9f0f15b4d83f8016f61150644deba00,
        c1343325c89eb6613df1663fc19326c39465db643587dea84dcfa6401d0f2d969e42213583edaec90726a2273acf2fb452718b2fa3e1d9e483b45c5c363a9879
    ], // 共识节点列表
    extraData=[

    ], // 额外数据
    gasLimit='0x0', // 区块中允许的 gas 最大值
    gasUsed='0x0', // 区块中所有交易消耗的 gas
    timestamp='0x175e62ef211', // 时间戳
    signatureList=null // 签名列表
}
```

新增节点

参考 **群组新增节点** 文档进行操作。

为新节点生成私钥证书

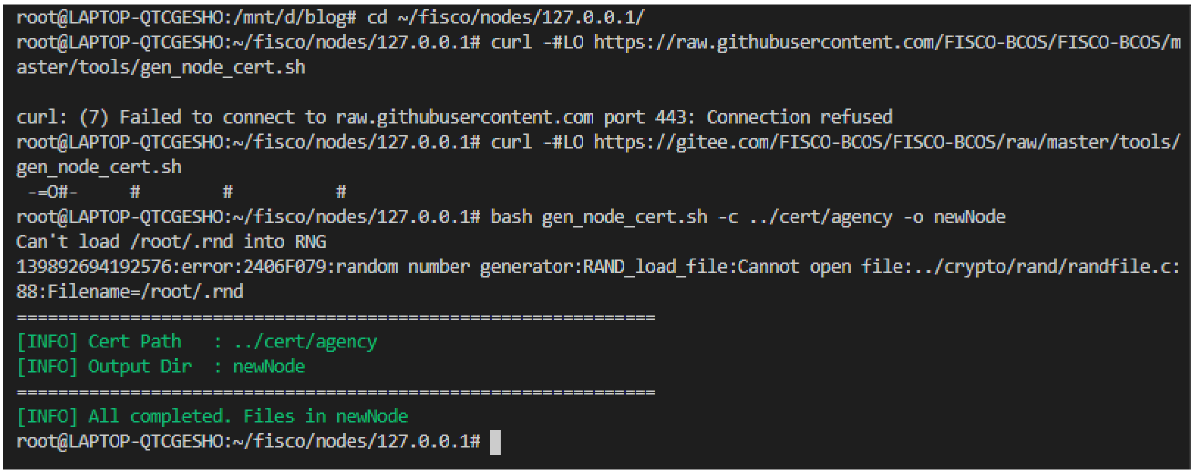
1. 获取证书生成脚本。
2. 生成新节点私钥证书。

> 命令都在 `nodes/127.0.0.1` 目录下进行操作

命令分别如下：

```
# 1. 获取证书生成脚本
curl -#LO https://raw.githubusercontent.com/FISCO-BCOS/FISCO-BCOS/master/tools/gen_node_cert.sh
# 如果因为网络问题导致长时间无法下载，请尝试
# curl -#LO https://gitee.com/FISCO-BCOS/FISCO-BCOS/raw/master/tools/gen_node_cert.sh

# 2. 生成新节点私钥证书
# -c指定机构证书及私钥所在路径
# -o输出到指定文件夹，其中newNode/conf中会存在机构agency新签发的证书和私钥
bash gen_node_cert.sh -c ../cert/agency -o newNode
```

执行命令结果如下：



```
root@LAPTOP-QTCGESHO:/mnt/d/blog# cd ~/fisco/nodes/127.0.0.1/
root@LAPTOP-QTCGESHO:~/fisco/nodes/127.0.0.1# curl -#LO https://raw.githubusercontent.com/FISCO-BCOS/FISCO-BCOS/master/tools/gen_node_cert.sh

curl: (7) Failed to connect to raw.githubusercontent.com port 443: Connection refused
root@LAPTOP-QTCGESHO:~/fisco/nodes/127.0.0.1# curl -#LO https://gitee.com/FISCO-BCOS/FISCO-BCOS/raw/master/tools/gen_node_cert.sh
 -=O#-     #        #          #
root@LAPTOP-QTCGESHO:~/fisco/nodes/127.0.0.1# bash gen_node_cert.sh -c ../cert/agency -o newNode
Can't load /root/.rnd into RNG
139892694192576:error:2406F079:random number generator:RAND_load_file:Cannot open file:../crypto/rand/randfile.c:88:Filename=/root/.rnd
==============================================================
[INFO] Cert Path   : ../cert/agency
[INFO] Output Dir  : newNode
==============================================================
[INFO] All completed. Files in newNode
root@LAPTOP-QTCGESHO:~/fisco/nodes/127.0.0.1#
```

> 在图中可以看到因为网络问题引起的下载失败，还好文档给出了另一个下载方式。

准备配置文件

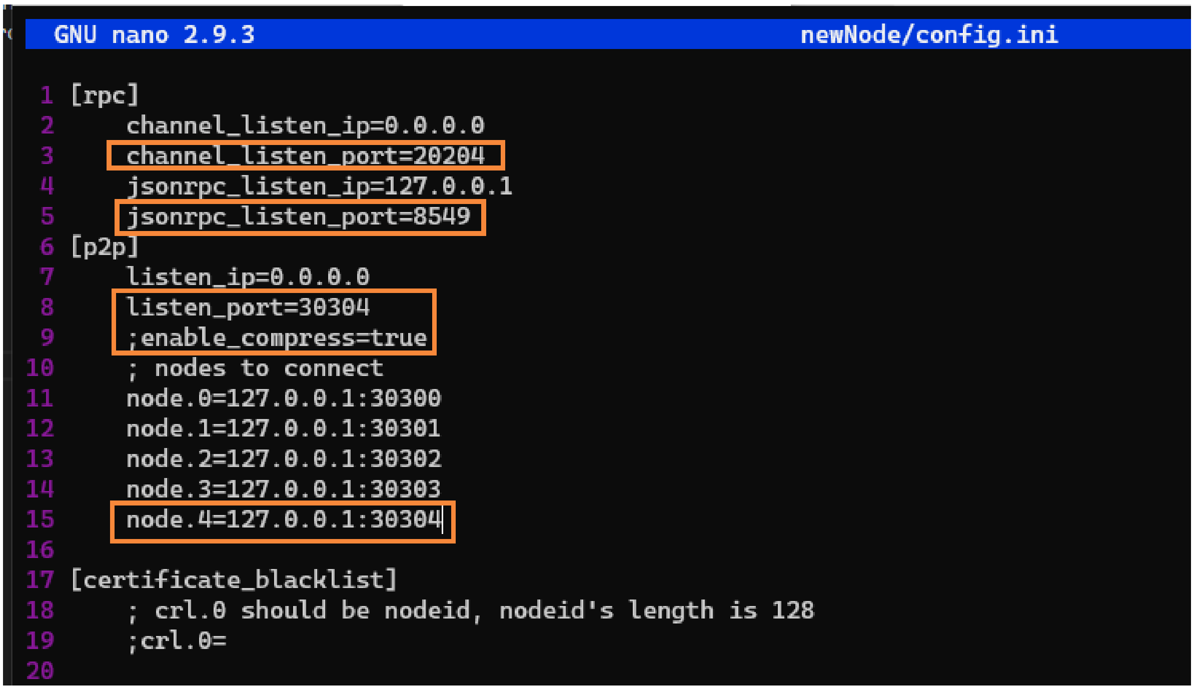
1. 从群组1中的节点 `node0` 拷贝配置文件和工具脚本
2. 更新 `newNode/config.ini` 中监听的 IP 和端口，将新节点的 P2P 配置中的 IP 和 Port 加入原有节点的 `config.ini` 中的 `[p2p]` 字段。：
   - 对于 `[rpc]` 模块，修改 `listen_ip` 、 `channel_listen_port` 和 `jsonrpc_listen_port` ；
   - 对于 `[p2p]` 模块，修改 `listen_port`

具体命令如下：

```
# 1. 拷贝配置文件和脚本
cp node0/config.ini newNode/config.ini
cp node0/conf/group.1.genesis newNode/conf/group.1.genesis
cp node0/conf/group.1.ini newNode/conf/group.1.ini
cp node0/*.sh newNode/
cp -r node0/scripts newNode/
```

```
# 2. 修改对应的 .ini 文件，参考如下
[rpc]
    channer_listen_ip=0.0.0.0
    channel_listen_port=20204
    jsonrpc_listen_ip=127.0.0.1
    jsonrpc_listen_port=8549
[p2p]
    listen_ip=0.0.0.0
    listen_port=30304
    ;enable_compress=true
    ; nodes to connect
    node.0=127.0.0.1:30300
    node.1=127.0.0.1:30301
    node.2=127.0.0.1:30302
    node.3=127.0.0.1:30303
    node.4=127.0.0.1:30304
```

具体修改如下图，框出来的为修改过的部分：

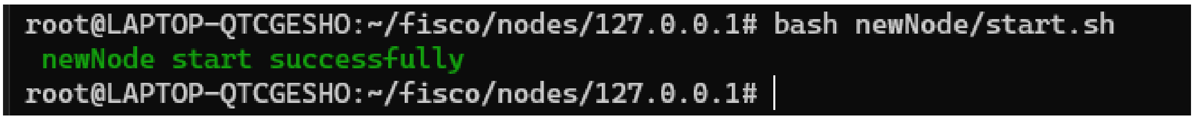


```
GNU nano 2.9.3                                    newNode/config.ini

[rpc]
    channel_listen_ip=0.0.0.0
    channel_listen_port=20204
    jsonrpc_listen_ip=127.0.0.1
    jsonrpc_listen_port=8549
[p2p]
    listen_ip=0.0.0.0
    listen_port=30304
    ;enable_compress=true
    ; nodes to connect
    node.0=127.0.0.1:30300
    node.1=127.0.0.1:30301
    node.2=127.0.0.1:30302
    node.3=127.0.0.1:30303
    node.4=127.0.0.1:30304

[certificate_blacklist]
    ; crl.0 should be nodeid, nodeid's length is 128
    ;crl.0=
```

启动新节点

```
# 3. 启动新节点
bash newNode_start.sh
```

节点启动成功，输出如下：



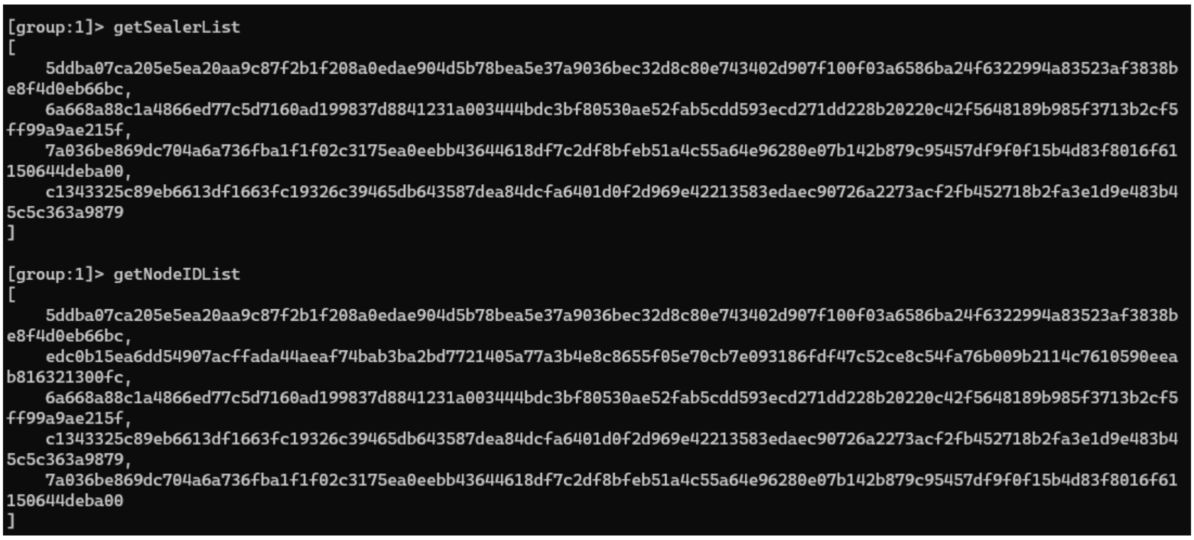
```
root@LAPTOP-QTCGESHO:~/fisco/nodes/127.0.0.1# bash newNode/start.sh
 newNode start successfully
root@LAPTOP-QTCGESHO:~/fisco/nodes/127.0.0.1#
```

加入群组1

通过：

- `getSealerList` 查看共识节点列表。
- `getNodeIDList` 查看节点及连接p2p节点的nodeId列表。

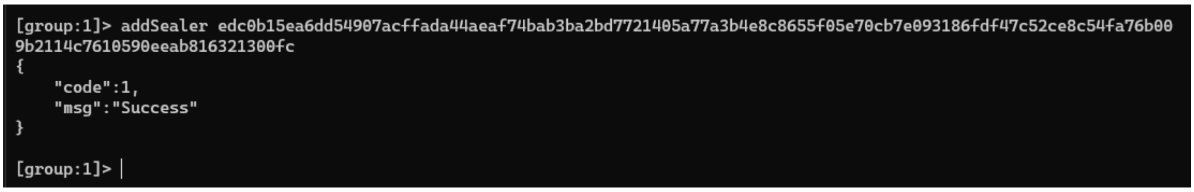
输出结果如下：



```
[group:1]> getSealerList
[
    5ddba07ca205e5ea20aa9c87f2b1f208a0edae904d5b78bea5e37a9036bec32d8c80e743402d907f100f03a6586ba24f6322994a83523af3838b
e8f4d0eb66bc,
    6a668a88c1a4866ed77c5d7160ad199837d8841231a003444bdc3bf80530ae52fab5cdd593ecd271dd228b20220c42f5648189b985f3713b2cf5
ff99a9ae215f,
    7a036be869dc704a6a736fba1f1f02c3175ea0eebb43644618df7c2df8bfeb51a4c55a64e96280e07b142b879c95457df9f0f15b4d83f8016f61
150644deba00,
    c1343325c89eb6613df1663fc19326c39465db643587dea84dcfa6401d0f2d969e42213583edaec90726a2273acf2fb452718b2fa3e1d9e483b4
5c5c363a9879
]

[group:1]> getNodeIDList
[
    5ddba07ca205e5ea20aa9c87f2b1f208a0edae904d5b78bea5e37a9036bec32d8c80e743402d907f100f03a6586ba24f6322994a83523af3838b
e8f4d0eb66bc,
    edc0b15ea6dd54907acffada44aeaf74bab3ba2bd7721405a77a3b4e8c8655f05e70cb7e093186fdf47c52ce8c54fa76b009b2114c7610590eea
b816321300fc,
    6a668a88c1a4866ed77c5d7160ad199837d8841231a003444bdc3bf80530ae52fab5cdd593ecd271dd228b20220c42f5648189b985f3713b2cf5
ff99a9ae215f,
    c1343325c89eb6613df1663fc19326c39465db643587dea84dcfa6401d0f2d969e42213583edaec90726a2273acf2fb452718b2fa3e1d9e483b4
5c5c363a9879,
    7a036be869dc704a6a736fba1f1f02c3175ea0eebb43644618df7c2df8bfeb51a4c55a64e96280e07b142b879c95457df9f0f15b4d83f8016f61
150644deba00
]
```

可以看到存在一个 hash 值为 `edc0b15ea6dd54907acffada44aeaf74bab3ba2bd7721405a77a3b4e8c8655f05e70cb7e093186fdf47c52ce8c54fa76b009b2114c7610590eeab816321300fc` 的节点未加入共识，该节点就是新增节点

使用命令 `addSealer edc0b15ea6dd54907acffada44aeaf74bab3ba2bd7721405a77a3b4e8c8655f05e70cb7e093186fdf47c52ce8c54fa76b009b2114c7610590eeab816321300fc` 将其加入共识节点，得到输出如下：

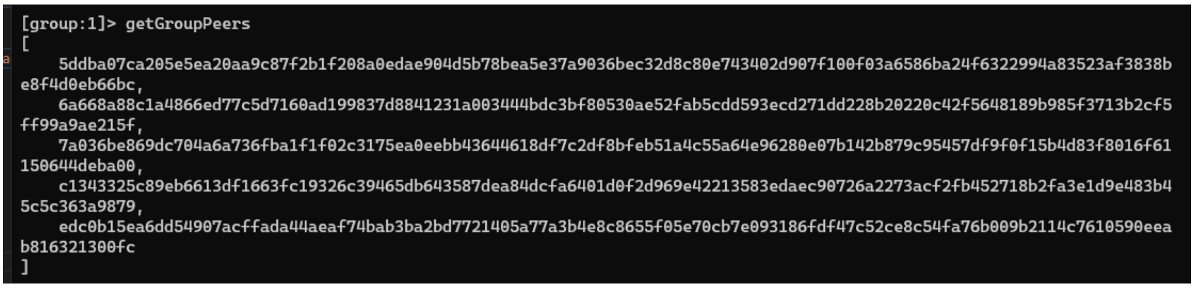


```
[group:1]> addSealer edc0b15ea6dd54907acffada44aeaf74bab3ba2bd7721405a77a3b4e8c8655f05e70cb7e093186fdf47c52ce8c54fa76b00
9b2114c7610590eeab816321300fc
{
    "code":1,
    "msg":"Success"
}

[group:1]> |
```

此时通过：

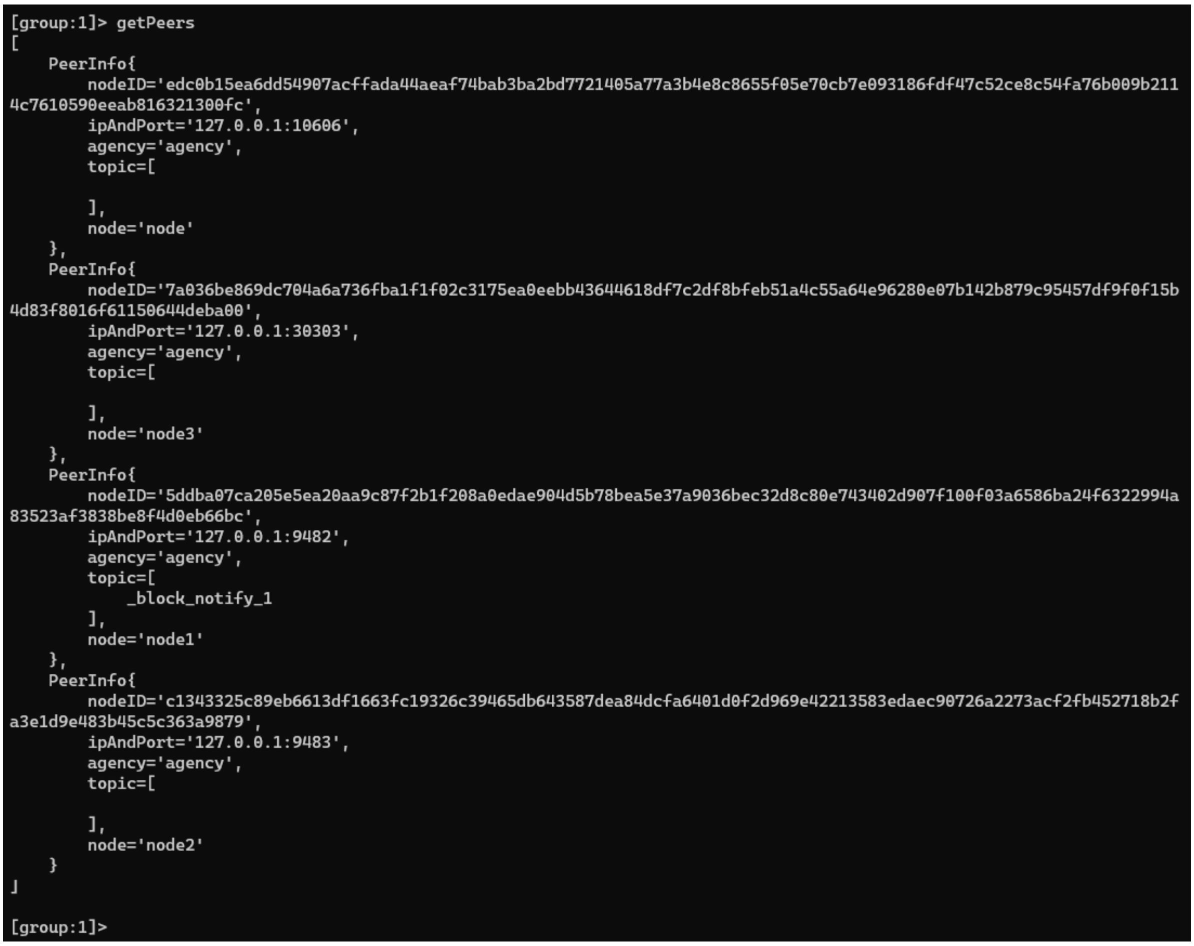
- `getGroupPeers` 查看当前群组的节点列表
- `getPeers` 查看当所有前节点状态

节点列表如下：



```
[group:1]> getGroupPeers
[
    5ddba07ca205e5ea20aa9c87f2b1f208a0edae904d5b78bea5e37a9036bec32d8c80e743402d907f100f03a6586ba24f6322994a83523af3838b
e8f4d0eb66bc,
    6a668a88c1a4866ed77c5d7160ad199837d8841231a003444bdc3bf80530ae52fab5cdd593ecd271dd228b20220c42f5648189b985f3713b2cf5
ff99a9ae215f,
    7a036be869dc704a6a736fba1f1f02c3175ea0eebb43644618df7c2df8bfeb51a4c55a64e96280e07b142b879c95457df9f0f15b4d83f8016f61
150644deba00,
    c1343325c89eb6613df1663fc19326c39465db643587dea84dcfa6401d0f2d969e42213583edaec90726a2273acf2fb452718b2fa3e1d9e483b4
5c5c363a9879,
    edc0b15ea6dd54907acffada44aeaf74bab3ba2bd7721405a77a3b4e8c8655f05e70cb7e093186fdf47c52ce8c54fa76b009b2114c7610590eea
b816321300fc
]
```

所有节点状态：



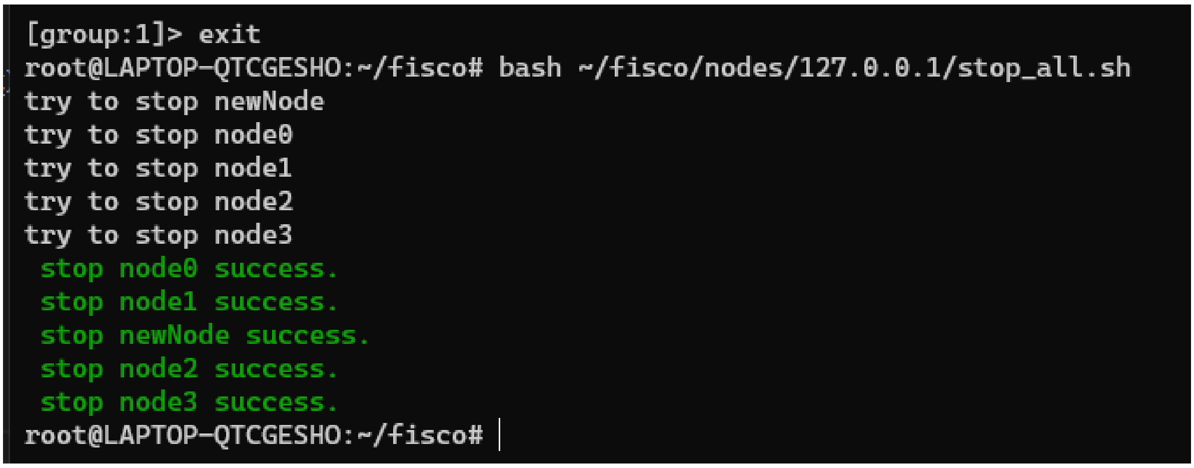
```
[group:1]> getPeers
[
    PeerInfo{
        nodeID='edc0b15ea6dd54907acffada44aeaf74bab3ba2bd7721405a77a3b4e8c8655f05e70cb7e093186fdf47c52ce8c54fa76b009b211
4c7610590eeab816321300fc',
        ipAndPort='127.0.0.1:10606',
        agency='agency',
        topic=[

        ],
        node='node'
    },
    PeerInfo{
        nodeID='7a036be869dc704a6a736fba1f1f02c3175ea0eebb43644618df7c2df8bfeb51a4c55a64e96280e07b142b879c95457df9f0f15b
4d83f8016f61150644deba00',
        ipAndPort='127.0.0.1:30303',
        agency='agency',
        topic=[

        ],
        node='node3'
    },
    PeerInfo{
        nodeID='5ddba07ca205e5ea20aa9c87f2b1f208a0edae904d5b78bea5e37a9036bec32d8c80e743402d907f100f03a6586ba24f6322994a
83523af3838be8f4d0eb66bc',
        ipAndPort='127.0.0.1:9482',
        agency='agency',
        topic=[
            _block_notify_1
        ],
        node='node1'
    },
    PeerInfo{
        nodeID='c1343325c89eb6613df1663fc19326c39465db643587dea84dcfa6401d0f2d969e42213583edaec90726a2273acf2fb452718b2f
a3e1d9e483b45c5c363a9879',
        ipAndPort='127.0.0.1:9483',
        agency='agency',
        topic=[

        ],
        node='node2'
    }
]

[group:1]>
```

停止所有节点

使用如下命令停止所有节点：

```
bash ~/fisco/nodes/127.0.0.1/stop_all.sh
```

输出结果如下：



```
[group:1]> exit
root@LAPTOP-QTCGESHO:~/fisco# bash ~/fisco/nodes/127.0.0.1/stop_all.sh
try to stop newNode
try to stop node0
try to stop node1
try to stop node2
try to stop node3
 stop node0 success.
 stop node1 success.
 stop newNode success.
 stop node2 success.
 stop node3 success.
root@LAPTOP-QTCGESHO:~/fisco#
```

说明所有节点已经正常退出