**搭建私有链**

以太坊1.8版本目前支持2种共识算法：工作量证明机制（PoW）和权威证明机制（PoA）。其中PoW适用于公有链，而PoA适用于私有链和联盟链，这两种机制的具体区别将在后续章节中详细介绍。

# 1 安装环境

## 1.1 安装GO

本书安装的以太坊为1.8版本，其要求Go版本不能低于1.10，否则编译以太坊源码时将会出现异常。

**1.下载和解压**

读者可到https://golang.org/dl/下载对应版本Go源码，并将源码解压至安装目录。下载操作如下：

|  |
| --- |
| // 下载源码  $ wget <https://dl.google.com/go/go1.10.2.linux-amd64.tar.gz> |
| // 解压源码  $ sudo tar -zxvf go1.10.2.linux-amd64.tar.gz /usr/local/src/go |

**2.修改环境变量**

在目录/etc/中找到profile文件，并在其中修改或添加环境变量$GOPATH、$GOROOT和$PATH的路径。

|  |
| --- |
| export GOPATH=/work/Go  export GOROOT=/usr/src/go  export PATH=$PATH:$GOROOT/bin:$GOPATH/bin |

完成以上全部操作后，再在命令行输入`source profile`便可使用go工具，此时也就完成了Go程序的安装。使用Go不是本书的重点，读者可查阅其他相关资料。

|  |
| --- |
| *备注：*  *1.$GOROOT表示的是Go的安装路径。*  *2.$GOPATH下的src是Golang开发程源码的目录，其可配置多个路径，各路径之间使用分号’;’进行分隔。本书中为了方便操作，只为其指定一个目录。* |

## 1.2 安装GETH

Geth是使用Golang实现的以太坊工具。其可作为以太坊网络中的一个结点，我们可通过其进行挖矿、转账、查询区块信息等操作。本书以安装Geth 1.8版本为例，以下是使用Geth源码进行安装的过程。

|  |
| --- |
| *注意：在Ubuntu系统中使用`sudo apt-get install geth`虽然可以完成geth的安装，但版本问题，使用过程中可能会出现各种莫名其妙的问题．因此，强烈建议使用源码安装。* |

**1.下载源码**

读者可到https://github.com/ethereum/go-ethereum.git下载Geth源码。下载操作如下：

|  |
| --- |
| $ go get github.com/ethereum/go-ethereum |

完成以上操作后，源码将会被自动下载到$GOPATH/github.com/ethereum/go-ethereum目录。

|  |
| --- |
| *注意：在这我们之所以不使用`git clone* github.com/ethereum/go-ethereum*.git`下载源码，是为了防止我们后续编译的geth版本与我们开发中所引用的geth库版本不一致的问题。如果出现版本不一致的问题，很可能导致后续开发的程序在编译过程中出现各种异常提示。* |

**2.切换版本**

完成源码下载后，默认为master分支。本文使用Geth 1.8版本，因此，需要手动切换分支。

|  |
| --- |
| // 源码目录  $ cd $GOPATH/github.com/ethereum/go-ethereum  // 切换版本  $ git chekcout release/1.8 |

**3.编译源码**

源码中存在Makefile文件，我们只需使用命令`make all`便可完成所有模块的编译。

|  |
| --- |
| $ make all |

编译完成后，可在源码目录下的`./build/bin`中查看所有编译后生成的所有工具。

|  |  |  |
| --- | --- | --- |
| **序号** | **命令** | **描述** |
| 01 | geth | 使用Golang编写的以太坊客户端工具。 |
| 02 | abigen | 可根据智能合约的sol或者abi文件生成指定语言的封装接口，当前支持三种语言：Golang，ObjectC和Java。 |
| 03 | puppeth | 用于生成以太坊Geth配置文件。 |

**4.修改环境变量**

找到/etc目录中的profile文件，其中添加环境变量$GETH，并修改环境变量$PATH。

|  |
| --- |
| // 安装目录  export GETH=$GOPATH/github.com/ethereum/go-ethereum/  // 命令目录  export PATH=$PATH:$GETH/build/bin |

完成以上操作，并执行`source profile`后，便可开始搭建自己的以太坊私有链平台。

# 2 搭建PoA私有链

权威证明机制简称PoA，其性能很高，功耗低，但却牺牲了信任。PoA适用于私有链和联盟链。本节将详细讲解基于PoA共识引擎的私有链搭建流程。

## 2.1搭建私有链单点

为了便于大家能够快速的掌握PoA私有链的搭建流程，我们先尝试搭建一个私有链单点，下一章节再介绍私有链集群的搭建。

### 2.1.1 指定目录

首先，我们需要指定私有链的搭建目录。后续搭建和运行私有链过程中产生的所有数据均会存储在该目录下。在/etc/profile中添加如下内容：

|  |
| --- |
| // 私有链目录  export CHAIN=/Work/Chain/ |

### 2.1.2 新建账号

生成配置前，需要新建一个或多个以太坊账号，这些账号将会在后续的配置过程中被指定为挖矿账户或预分配资金账户。为方便后续操作，本书中所有账号的密码均为”111111”。新建账号的具体操作如下：

|  |
| --- |
| $ cd $CHAIN  $ geth --datadir ./data account new  Your new account is locked with a password. Please give a password. Do not forget this password.  Passphrase: // 输入密码: 111111  Repeat passphrase: // 输入密码: 111111  Address: {b9bc3a727a557ea0489b34cf9b16b35967e9c42b} |

|  |
| --- |
| *注意：一定要牢记自己账号的密码。密码一旦丢失，账户中的代币将永远无法被尘封。* |

### 2.1.3 生成配置

在目录$GETH/build/bin中可以找到以太坊官方提供的geth配置生成工具：puppeth。首先，进入$CHAIN目录，并开始配置处理：

|  |
| --- |
| $ cd $CHAIN  $ puppet |

**1.网络名称**

|  |
| --- |
| Please specify a network name to administer (no spaces or hyphens, please)  > **YiChain** |

**2.配置创世块**

由于当前需要搭建一个全新的私有链，因此，我们选择`2. Configure new genesis`开始创世块的配置。

|  |
| --- |
| What would you like to do? (default = stats)  1. Show network stats  **2. Configure new genesis**  3. Track new remote server  4. Deploy network components  > **2** |

**3.共识引擎**

当前以太坊支持Pow和PoA两种共识引擎。由于PoW适用于公有链，而PoA适用于私有链。因此，我们输入2来选择PoA。

|  |
| --- |
| Which consensus engine to use? (default = clique)  1. Ethash - proof-of-work  **2. Clique - proof-of-authority**  > **2** |

**4.出块间隔**

以太坊PoW共识机制通过调整难度来控制链的出块时间，而PoA共识机制则通过睡眠指定时间来控制出块时间。为了提高私有链的交易处理性能，在此我们设置出块时间间隔为0秒。

|  |
| --- |
| How many seconds should blocks take? (default = 15)  > **0** |

|  |
| --- |
| *备注：控制出块的时间间隔。*  *1.当该值为0时，只要链平台收到交易就会产生区块，没有收到交易则不产生区块。即：不会出现空块的情况，且每个区块中至少包含了1条交易数据。*  *2.当该值不为0时，只要达到指定间隔时间，即使链平台未收到交易，平台依然会产生一个区块。即：可能出现空块的情况，也就是有些区块中不包含任何交易数据。* |

**5.挖矿账号**

如果使用PoW共识引擎，则其允许任何账号进行挖矿操作。但是如果使用PoA共识引擎，则其对挖矿权利有严格的限制，只有得到1/2以上结点认可的结点才能有挖矿的权利。PoA共识引擎之所以对挖矿权利进行限制，是为了保证私有链数据的安全。

|  |
| --- |
| Which accounts are allowed to seal? (mandatory at least one)  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b**  > 0x |

**6.预分配账号**

为什么搭建私有链的过程中需要给某些账号预分配以太币？这些以太币将会是驱动私有链平台正常运行的基石，具有非常重要的作用。且可以设置多个预分配账号。

|  |
| --- |
| Which accounts should be pre-funded? (advisable at least one)  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b** |

|  |
| --- |
| *设想：假如我们使用PoA共识引擎，设置的出块时间为0秒，且没有给任何账号预分配资金。这将出现什么情况？*  *1.使用PoA共识引擎，设置的出块时间为0秒时，则不会出现空块的情况。在没有交易的情况下，不出块也就无任何账号可以获得以太币的奖励。*  *2.那我们就发起交易，让挖矿账号出块并获得奖励。但是发起交易是需要消耗GAS的，而GAS的获取却需要消耗以太币。因此，无任何账号拥有以太币，也就无法发起交易。*  *总结：综合以上2种情况，可得出结论：不进行以太币的预分配，则会导致私有链平台无法正常使用。* |

**7.网络ID**

某个以太坊结点像加入到私有链集群中，那么其必须与私有链集群拥有相同的网络(链)ID。如果此时不指定，则puppeth将会为你随机生成网络(链)ID。

|  |
| --- |
| Specify your chain/network ID if you want an explicit one (default = random)  > **10000** |

**8.导出配置**

至此我们已经完成私有链所有属性的配置，只需继续按照如下提示便可将配置信息导出到指定文件中。具体操作如下：

|  |
| --- |
| What would you like to do? (default = stats)  1. Show network stats  **2. Manage existing genesis**  3. Track new remote server  4. Deploy network components  > **2**  1. Modify existing fork rules  **2. Export genesis configuration**  3. Remove genesis configuration  > **2**  Which file to save the genesis into? (default = YiChain.json)  > **YiChain.json** |

完成以上操作后，便可使用`Ctrl-C`退出puppeth的执行。此时，我们可在puppeth执行目录$CHAIN中看到生成的配置文件YiChain.json。

### 2.1.4 配置详解

以下是使用PoA共识引擎的配置信息，具体格式如下所示．

|  |
| --- |
| {  "config": {  "chainId": 10000,  "homesteadBlock": 1,  "eip150Block": 2,  "eip150Hash": "0x0000000000000000000000000000000000000000000000000000000000000000",  "eip155Block": 3,  "eip158Block": 3,  "byzantiumBlock": 4,  "clique": {  "period": 0,  "epoch": 30000  }  },  "nonce": "0x0",  "timestamp": "0x5af78862",  "extraData": "0x0000000000000000000000000000000000000000000000000000000000000000c162183c6a58542baa001cbc2d25cd62a0b58f2b0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",  "gasLimit": "0x47b760",  "difficulty": "0x1",  "mixHash": "0x0000000000000000000000000000000000000000000000000000000000000000",  "coinbase": "0x0000000000000000000000000000000000000000",  "alloc": {  "0000000000000000000000000000000000000000": {  "balance": "0x1"  },  "0000000000000000000000000000000000000001": {  "balance": "0x1"  },  "0000000000000000000000000000000000000002": {  "balance": "0x1"  },  **.... // 03 ~ fe部分省略**  "00000000000000000000000000000000000000ff": {  "balance": "0x1"  },  "c162183c6a58542baa001cbc2d25cd62a0b58f2b": {  "balance": "0x200000000000000000000000000000000000000000000000000000000000000"  }  },  "number": "0x0",  "gasUsed": "0x0",  "parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000"  } |

以上配置中各字段含义如下．

|  |  |  |
| --- | --- | --- |
| **序号** | **命令** | **描述** |
| **01** | config.chainId | 私有链平台的网络(链)ID |
| **02** | config.clique | PoA共识引擎的相关配置 |
| **03** | config.clique.period | 用于控制PoA出块的时间间隔。  1.当该值为0时，只要私有链平台收到交易就会产生区块，没有收到交易则不产生区块。即：不会出现空块的情况，也就是每个区块中至少1条交易数据。  2.当该值不为0时，只要达到指定间隔时间，即使私有链平台未收到交易，平台依然会产生一个区块。即：可能出现空块的情况，也就是有些区块中没人任何交易数据。 |
| **04** | config.clique.epoch | 每epoch个块算一个世纪。无需修改该值。 |
| **05** | nonce | 创世块nonce值。无需修改该值。 |
| **06** | timestamp | 生成配置文件的时间戳 |
| **07** | extraData | 扩展数据 |
| **08** | gasLimit | 交易Gas限制 |
| **09** | difficulty | 设置挖矿难度。使用PoA时，无需修改该值。 |
| **10** | mixHash |  |
| **11** | coinbase | 结点挖矿钱包地址 |
| **12** | alloc | 各钱包地址预分配以太币数量 |
| **13** | number | 创世块ID |
| **14** | gasUsed | 花费的Gas数量 |
| **15** | parentHash | 设置父区块哈希值。创世块的值为父区块哈希值为0x0000000000000000000000000000000000000000000000000000000000000000。 |

### 2.1.5 创世区块

配置信息虽中有创世块的相关配置，但是我们还未创建创世块。要正常启动私有链结点，需在私有链目录$CHAIN中通过如下命令新建创世区块。其具体操作如下：

|  |
| --- |
| $ geth --datadir ./data init YiChain.json |

### 2.1.6 启动结点

完成创世区块的创建后，便可正常启动一个私有链结点。启动可交互的私有链结点的具体操作如下：

|  |
| --- |
| $ geth --datadir ./data --networkid 10000 --port 61910 --rpc --rpcaddr 10.168.0.2 --rpcport 8545 --rpcapi 'db,net,eth,web3' --rpccorsdomain '\*' --nodiscover --mine console |

私有链启动命令中的相关命令和选项的描述如下：

|  |  |  |
| --- | --- | --- |
| **序号** | **命令或选项** | **功能描述** |
| **01** | --datadir | 指定数据库和密钥的数据目录 |
| **02** | --networkid | 网络标识符(整型)(默认:1)   1. Frontier 2. Morden(弃用) 3. Ropsten 4. Rinkeby |
| **03** | --port | 指定结点间通信的端口 |
| **04** | --rpc | 开启HTTP-RPC服务 |
| **05** | --rpcaddr | 指定HTTP-RPC服务绑定的网卡IP(默认值:"localhost"). |
| **06** | --rpcport | 指定HTTP-RPC服务监听端口(默认值:8545) |
| **07** | --rpcapi | 指定HTTP-RPC服务提供的API类型 |
| **08** | --rpccorsdomain | 允许跨域请求的域名列表(逗号分隔)(浏览器强制) |
| **09** | --nodiscover | 禁用结点发现机制(手动添加结点) |
| **10** | --mine | 开启挖矿功能 |
| **11** | console | 开启交互模式 |

## 2.2搭建私有链集群

在2.1节中描述了配置和启动单个私有链结点的详细过程，而只有一个结点的私有链显然并不能满足生产环境的实际要求。

假设我们当前有3台机器(10.168.0.1~3)，本节将描述在这3台机器上搭建私有链集群的过程。

### 2.2.1 准备环境

在提供的3台机器上按照1.1和1.2节的描述完成GO和GETH的安装，并按照2.1.1节中的描述新建私有链目录。

### 2.2.2 新建账号

在进行配置前，我们需要分别登录3台机器，并执行类似2.1.2节的操作新建私有链账号。

|  |
| --- |
| $ cd $CHAIN  // 新建账号  $ geth --datadir ./data account new  Your new account is locked with a password. Please give a password. Do not forget this password.  Passphrase: // 输入密码: 111111  Repeat passphrase: // 输入密码: 111111  Address: {b9bc3a727a557ea0489b34cf9b16b35967e9c42b} |

完成以上操作后，我们可以得到3个私有链账户，分别为：

|  |  |  |
| --- | --- | --- |
| **序号** | **钱包地址** | **密码** |
| **01** | b9bc3a727a557ea0489b34cf9b16b35967e9c42b | 111111 |
| **02** | 30ab3bb668038da15d87c51ddfd09d7cf7e4b155 | 111111 |
| **03** | b9bc3a727a557ea0489b34cf9b16b35967e9c42b | 111111 |

### 2.2.3 生成配置

搭建私有链集群和私有链的配置流程是完全一样的，且只需在其中一台机器上生成便可。生成配置的流程可以参考2.1.3节，只需适当调整挖矿账号和预分配账号的配置。

**1.挖矿账号**

|  |
| --- |
| Which accounts are allowed to seal? (mandatory at least one)  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b**  > **0x30ab3bb668038da15d87c51ddfd09d7cf7e4b155**  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b** |

**2.预分配账号**

|  |
| --- |
| Which accounts should be pre-funded? (advisable at least one)  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b**  > **0x30ab3bb668038da15d87c51ddfd09d7cf7e4b155**  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b** |

完成配置文件的生成后，需要将该配置原封不动的同步到其他2台机器的指定目录$CHAIN中。

|  |
| --- |
| *注意:将生成的配置文件同步到其他2台机器上，然后再在3台机器执行创世块初始化操作。切记3台机器的配置文件必须一致，否则搭建集群时将会提示配置文件MD5值不一致的异常。* |

### 2.2.4 创世区块

该流程与2.1.5节的描述一致，且在3台机器均需要执行创世区块的初始化操作：

|  |
| --- |
| $ geth --datadir ./data init YiChain.json |

### 2.2.5 启动结点

完成创世区块的创建后，需要在3台机器上均启动私有链结点。在10.168.0.1上启动可交互的私有链结点指令如下，其他2台机器上只需适当调整RPC地址。

|  |
| --- |
| // 10.168.0.1  $ geth --datadir ./data --networkid 10000 --port 61910 --rpc --rpcaddr 10.168.0.1 --rpcport 8545 --rpcapi 'db,net,eth,web3' --rpccorsdomain '\*' --nodiscover --mine console |

### 2.2.6 构建集群

虽然我们在3台机器上分别启动了私有链结点，但是他们相互之间并不知道其他结点的存在。因此，目前还不能算是一个私有链集群。要想将相互隔离的私有链结点构建成一个私有链集群，我们可按如下流程进行操作：

**1.查看结点信息**

每个以太坊结点的enode是唯一的。新建结点要想加入到私有链集群中，需要将其enode添加到私有链其他结点的远程结点列表中。可通过在交互模式中输入如下指令查看enode信息：

|  |
| --- |
| // 10.168.0.1  > admin.nodeinfo.enode  “enode://3a5d4f2c15ae01e16bbf1e7c819558373e0b6ddfcb72a931c3e59ed72f01fc217c97c3556ebb3f0c09afab94fda136f4cb80d757ba75c9f3be7742b3c8834169@[::]:28000?discport=0” |

|  |
| --- |
| // 10.168.0.2  > admin.nodeinfo.enode  “enode://4b6c5a3d26bf12f27cca2f8d920669484f1c7eeadc83b042d4f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3c9945270@[::]:28000?discport=0” |

|  |
| --- |
| // 10.168.0.3  > admin.nodeinfo.enode  “enode://5c7d6b4c37ca23a27cca2f8d920669484f1c8ffbed94c153e5f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3d0056381 @[::]:28000?discport=0” |

**2.添加PEER结点**

要想让私有链集群中的结点相互感知到其他结点的存在，则需在某结点上将其他结点的enode添加到该结点peers列表中。

机器10.168.0.1上的私有链结点将其他2台机器上的私有链结点信息到自身的peers列表中的操作如下：

|  |
| --- |
| > admin.addPeer(“enode://4b6c5a3d26bf12f27cca2f8d920669484f1c7eeadc83  b042d4f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3c9945270@**10.168.0.2**:28000?discport=0”)  > admin.addPeer(“enode://5c7d6b4c37ca23a27cca2f8d920669484f1c8ffbed94  c153e5f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3d0056381@**10.168.0.3**:28000?discport=0”) |

机器10.168.0.2可参照以上格式将10.168.0.1和10.168.0.3上的私有链结点加入到自身的peers列表；机器10.168.0.3同样参照如上方式。

**3.查看PEER结点**

为了确定刚才添加PEER结点的操作是否有误，我们可在交互模式中输入如下指令查看PEER节点的状态。

|  |
| --- |
| > admin.peers  “enode://4b6c5a3d26bf12f27cca2f8d920669484f1c7eeadc83b042d4f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3c9945270@**10.168.0.2**:28000?discport=0”  “enode://5c7d6b4c37ca23a27cca2f8d920669484f1c8ffbed94  c153e5f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3d0056381@**10.168.0.3**:28000?discport=0” |

|  |
| --- |
| *注意:如果发现PEER列表中的结点存在异常，我们该如何解决呢？*  *1.结点信息是否正确。由于结点enode信息很长，很可能输入时出现失误。*  *2.结点IP地址是否正确。由于添加PEER时，需手动填入结点IP地址，很可能输入时出现失误。*  *3.结点是否正常运行。远程结点目前均是交互模式启动，很容易误操作导致结点结束运行。* |

### 2.2.7 授权挖矿

我们搭建的私有链选择的是PoA共识引擎，为保证私有链的安全性，其对挖矿权进行了严格的限制。只有集群中1/2以上的结点认可了某结点的挖矿权，某结点才拥有挖矿的权利。

我们搭建的私有链集群中的每个结点可通过如下命令对其他2个结点的挖矿权进行认可。

|  |
| --- |
| > admin.peers |

完成以上所有操作后，我们的私有链集群才算真正搭建完成。此后，我们可以使用该私有链进行基于区块链项目的开发了。

# 3 搭建PoW私有链

工作量证明简称PoW，其将会消耗较高的CPU资源和内存资源，存在51%的算力攻击风险。PoW适合于公有链。本节将详细讲解基于PoW共识引擎的私有链搭建流程。

## 3.1搭建私有链单点

为了便于大家能够快速的掌握PoW私有链的搭建流程，我们先尝试搭建一个私有链单点，下一章节再介绍私有链集群的搭建。

### 3.1.1 新建目录

首先，我们需要指定私有链的搭建目录。后续搭建和运行私有链过程中产生的所有数据均会存储在该目录下。在/etc/profile中添加如下内容：

|  |
| --- |
| // 私有链目录  export CHAIN=/Work/Chain/ |

### 3.1.2 新建账号

生成配置前，需要新建一个或多个以太坊账号，这些账号将会在后续的配置过程中被指定为挖矿账户或预分配资金账户。为方便后续操作，本书中所有账号的密码均为”111111”。新建账号的具体操作如下：

|  |
| --- |
| $ cd $CHAIN  $ geth --datadir ./data account new  Your new account is locked with a password. Please give a password. Do not forget this password.  Passphrase: // 输入密码: 111111  Repeat passphrase: // 输入密码: 111111  Address: {b9bc3a727a557ea0489b34cf9b16b35967e9c42b} |

|  |
| --- |
| *注意：一定要牢记自己账号的密码。密码一旦丢失，账户中的代币将永远无法被尘封。* |

### 3.1.3 生成配置

在目录$GETH/build/bin中可以找到以太坊官方提供的geth配置生成工具：puppeth。首先，进入$CHAIN目录，并开始配置处理：

|  |
| --- |
| $ cd $CHAIN  $ puppet |

**1.网络名称**

|  |
| --- |
| Please specify a network name to administer (no spaces or hyphens, please)  > **YiChain** |

**2.配置创世块**

由于当前需要搭建一个全新的私有链，因此，我们选择`2. Configure new genesis`开始创世块的配置。

|  |
| --- |
| What would you like to do? (default = stats)  1. Show network stats  **2. Configure new genesis**  3. Track new remote server  4. Deploy network components  > **2** |

**3.共识引擎**

|  |
| --- |
| Which consensus engine to use? (default = clique)  **1. Ethash - proof-of-work**  2. Clique - proof-of-authority  > **1** |

**4.预分配账号**

之前我们提到使用PoA共识引擎，且设置的出块时间为0秒，必须指定预分配账号，否则PoA私有链平台将无法正常运行。但是如果使用PoW私有链，情况则有所不同。

使用PoW私有链时，其达到指定时间必然会出块，出块也就意味着存在奖励。因此，即使未给指定账号预分配以太币，PoW私有链平台也能够正常运行。但依然强烈建议为搭建的PoW私有链指定预分配账号，这样才能更好的保证基于此链的业务运行。

|  |
| --- |
| Which accounts should be pre-funded? (advisable at least one)  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b** |

**5.网络ID**

某个以太坊结点像加入到私有链集群中，那么其必须与私有链集群拥有相同的网络(链)ID。如果此时不指定，则puppeth将会为你随机生成网络(链)ID。

|  |
| --- |
| Specify your chain/network ID if you want an explicit one (default = random)  > **10000** |

**6.导出配置**

至此我们已经完成私有链所有属性的配置，只需继续按照如下提示便可将配置信息导出到指定文件中。具体操作如下：

|  |
| --- |
| What would you like to do? (default = stats)  1. Show network stats  **2. Manage existing genesis**  3. Track new remote server  4. Deploy network components  > **2**  1. Modify existing fork rules  **2. Export genesis configuration**  3. Remove genesis configuration  > **2**  Which file to save the genesis into? (default = YiChain.json)  > **YiChain.json** |

完成以上操作后，便可使用`Ctrl-C`退出puppeth的执行。此时，我们可在puppeth执行目录$CHAIN中看到生成的配置文件YiChain.json。

### 3.1.4 配置详解

以下是使用PoW共识引擎的配置信息，具体格式如下所示．

|  |
| --- |
| {  "config": {  "chainId": 10000,  "homesteadBlock": 1,  "eip150Block": 2,  "eip150Hash": "0x0000000000000000000000000000000000000000000000000000000000000000",  "eip155Block": 3,  "eip158Block": 3,  "byzantiumBlock": 4,  "ethash": {}  },  "nonce": "0x0",  "timestamp": "0x5af78862",  "extraData": "0x0000000000000000000000000000000000000000000000000000000000000000c162183c6a58542baa001cbc2d25cd62a0b58f2b0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",  "gasLimit": "0x47b760",  "difficulty": "0x1",  "mixHash": "0x0000000000000000000000000000000000000000000000000000000000000000",  "coinbase": "0x0000000000000000000000000000000000000000",  "alloc": {  "0000000000000000000000000000000000000000": {  "balance": "0x1"  },  "0000000000000000000000000000000000000001": {  "balance": "0x1"  },  "0000000000000000000000000000000000000002": {  "balance": "0x1"  },  **.... // 03 ~ fe部分省略**  "00000000000000000000000000000000000000ff": {  "balance": "0x1"  },  "c162183c6a58542baa001cbc2d25cd62a0b58f2b": {  "balance": "0x200000000000000000000000000000000000000000000000000000000000000"  }  },  "number": "0x0",  "gasUsed": "0x0",  "parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000"  } |

以上配置中各字段含义如下．

|  |  |  |
| --- | --- | --- |
| **序号** | **命令** | **描述** |
| 01 | config.chainId | 私有链平台的网络(链)ID |
| 02 | config.ethash | PoW共识引擎的相关配置 |
| 05 | nonce | 创世块nonce值。无需修改该值。 |
| 06 | timestamp | 生成配置文件的时间戳 |
| 07 | extraData | 扩展数据 |
| 08 | gasLimit | 交易Gas限制 |
| 09 | difficulty | 设置挖矿难度。使用PoA时，无需修改该值。 |
| 10 | mixHash |  |
| 11 | coinbase | 结点挖矿钱包地址 |
| 12 | alloc | 各钱包地址预分配以太币数量 |
| 13 | number | 创世块ID |
| 14 | gasUsed | 花费的Gas数量 |
| 15 | parentHash | 设置父区块哈希值。创世块的值为父区块哈希值为0x0000000000000000000000000000000000000000000000000000000000000000。 |

### 3.1.5 创世区块

配置信息虽中有创世块的相关配置，但是我们还未创建创世块。要正常启动私有链结点，需在私有链目录$CHAIN中通过如下命令新建创世区块。其具体操作如下：

|  |
| --- |
| $ geth --datadir ./data init YiChain.json |

### 3.1.6 启动结点

完成创世区块的创建后，便可正常启动一个私有链结点。启动可交互的私有链结点的具体操作如下：

|  |
| --- |
| $ geth --datadir ./data --networkid 10000 --port 61910 --rpc --rpcaddr 10.168.0.2 --rpcport 8545 --rpcapi 'db,net,eth,web3' --rpccorsdomain '\*' --nodiscover --mine console |

私有链启动命令中的相关命令和选项的描述如下：

|  |  |  |
| --- | --- | --- |
| **序号** | **命令或选项** | **功能描述** |
| 01 | --datadir | 指定数据库和密钥的数据目录 |
| 02 | --networkid | 网络标识符(整型)(默认:1)   1. Frontier 2. Morden(弃用) 3. Ropsten 4. Rinkeby |
| 03 | --port | 指定结点间通信的端口 |
| 04 | --rpc | 开启HTTP-RPC服务 |
| 05 | --rpcaddr | 指定HTTP-RPC服务绑定的网卡IP(默认值:"localhost"). |
| 06 | --rpcport | 指定HTTP-RPC服务监听端口(默认值:8545) |
| 07 | --rpcapi | 指定HTTP-RPC服务提供的API类型 |
| 08 | --rpccorsdomain | 允许跨域请求的域名列表(逗号分隔)(浏览器强制) |
| 09 | --nodiscover | 禁用结点发现机制(手动添加结点) |
| 10 | --mine | 开启挖矿功能 |
| 11 | console | 开启交互模式 |

## 3.2搭建私有链集群

在3.1节中描述了配置和启动单个私有链结点的详细过程，而只有一个结点的私有链显然并不能满足生产环境的实际要求。

假设我们当前有3台机器(10.168.0.1~3)，本节将描述在这3台机器上搭建私有链集群的过程。

### 3.2.1 准备环境

在提供的3台机器上按照1.1和1.2节的描述完成GO和GETH的安装，并按照3.1.1节中的描述新建私有链目录。

### 3.2.2 新建账号

在进行配置前，我们需要分别登录3台机器，并执行类似3.1.2节的操作新建私有链账号。

|  |
| --- |
| $ cd $CHAIN  // 新建账号  $ geth --datadir ./data account new  Your new account is locked with a password. Please give a password. Do not forget this password.  Passphrase: // 输入密码: 111111  Repeat passphrase: // 输入密码: 111111  Address: {b9bc3a727a557ea0489b34cf9b16b35967e9c42b} |

完成以上操作后，我们可以得到3个私有链账户，分别为：

|  |  |  |
| --- | --- | --- |
| **序号** | **钱包地址** | **密码** |
| 01 | b9bc3a727a557ea0489b34cf9b16b35967e9c42b | 111111 |
| 02 | 30ab3bb668038da15d87c51ddfd09d7cf7e4b155 | 111111 |
| 03 | b9bc3a727a557ea0489b34cf9b16b35967e9c42b | 111111 |

### 3.2.3 生成配置

搭建私有链集群和私有链的配置流程是完全一样的，且只需在其中一台机器上生成便可。生成配置的流程可以参考3.1.3节，只需适当调整挖矿账号和预分配账号的配置。

**1.挖矿账号**

|  |
| --- |
| Which accounts are allowed to seal? (mandatory at least one)  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b**  > **0x30ab3bb668038da15d87c51ddfd09d7cf7e4b155**  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b** |

**2.预分配账号**

|  |
| --- |
| Which accounts should be pre-funded? (advisable at least one)  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b**  > **0x30ab3bb668038da15d87c51ddfd09d7cf7e4b155**  > **0xc162183c6a58542baa001cbc2d25cd62a0b58f2b** |

完成配置文件的生成后，需要将该配置原封不动的同步到其他2台机器的指定目录$CHAIN中。

|  |
| --- |
| *注意:将生成的配置文件同步到其他2台机器上，然后再在3台机器执行创世块初始化操作。切记3台机器的配置文件必须一致，否则搭建集群时将会提示配置文件MD5值不一致的异常。* |

### 3.2.4 创世区块

该流程与3.1.5节的描述一致，且在3台机器均需要执行创世区块的初始化操作：

|  |
| --- |
| $ geth --datadir ./data init YiChain.json |

### 3.2.5 启动结点

完成创世区块的创建后，需要在3台机器上均启动私有链结点。在10.168.0.1上启动可交互的私有链结点指令如下，其他2台机器上只需适当调整RPC地址。

|  |
| --- |
| // 10.168.0.1  $ geth --datadir ./data --networkid 10000 --port 61910 --rpc --rpcaddr 10.168.0.1 --rpcport 8545 --rpcapi 'db,net,eth,web3' --rpccorsdomain '\*' --nodiscover --mine console |

### 3.2.6 构建集群

虽然我们在3台机器上分别启动了私有链结点，但是他们相互之间并不知道其他结点的存在。因此，目前还不能算是一个私有链集群。要想将相互隔离的私有链结点构建成一个私有链集群，我们可按如下流程进行操作：

**1.查看结点信息**

每个以太坊结点的enode是唯一的。新建结点要想加入到私有链集群中，需要将其enode添加到私有链其他结点的远程结点列表中。可通过在交互模式中输入如下指令查看enode信息：

|  |
| --- |
| // 10.168.0.1  > admin.nodeinfo.enode  “enode://3a5d4f2c15ae01e16bbf1e7c819558373e0b6ddfcb72a931c3e59ed72f01fc217c97c3556ebb3f0c09afab94fda136f4cb80d757ba75c9f3be7742b3c8834169@[::]:28000?discport=0” |

|  |
| --- |
| // 10.168.0.2  > admin.nodeinfo.enode  “enode://4b6c5a3d26bf12f27cca2f8d920669484f1c7eeadc83b042d4f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3c9945270@[::]:28000?discport=0” |

|  |
| --- |
| // 10.168.0.3  > admin.nodeinfo.enode  “enode://5c7d6b4c37ca23a27cca2f8d920669484f1c8ffbed94c153e5f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3d0056381 @[::]:28000?discport=0” |

**2.添加PEER结点**

要想让私有链集群中的结点相互感知到其他结点的存在，则需在某结点上将其他结点的enode添加到该结点peers列表中。

机器10.168.0.1上的私有链结点将其他2台机器上的私有链结点信息到自身的peers列表中的操作如下：

|  |
| --- |
| > admin.addPeer(“enode://4b6c5a3d26bf12f27cca2f8d920669484f1c7eeadc83  b042d4f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3c9945270@**10.168.0.2**:28000?discport=0”)  > admin.addPeer(“enode://5c7d6b4c37ca23a27cca2f8d920669484f1c8ffbed94  c153e5f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3d0056381@**10.168.0.3**:28000?discport=0”) |

机器10.168.0.2可参照以上格式将10.168.0.1和10.168.0.3上的私有链结点加入到自身的peers列表；机器10.168.0.3同样参照如上方式。

**3.查看PEER结点**

为了确定刚才添加PEER结点的操作是否有误，我们可在交互模式中输入如下指令查看PEER节点的状态。

|  |
| --- |
| > admin.peers  “enode://4b6c5a3d26bf12f27cca2f8d920669484f1c7eeadc83b042d4f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3c9945270@**10.168.0.2**:28000?discport=0”  “enode://5c7d6b4c37ca23a27cca2f8d920669484f1c8ffbed94  c153e5f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3d0056381@**10.168.0.3**:28000?discport=0” |

|  |
| --- |
| *注意:如果发现PEER列表中的结点存在异常，我们该如何解决呢？*  *1.结点信息是否正确。由于结点enode信息很长，很可能输入时出现失误。*  *2.结点IP地址是否正确。由于添加PEER时，需手动填入结点IP地址，很可能输入时出现失误。*  *3.结点是否正常运行。远程结点目前均是交互模式启动，很容易误操作导致结点结束运行。* |

# 4 配置进阶

## 4.1 静态结点

搭建私有链的过程中已描述了通过交互模式将私有链中其他结点添加至本结点peers列表的过程。但当私有链结点过多时，使用交互模式进行操作显然过于繁琐。以太坊Geth支持通过配置statistic.json文件的方式来应对这种情况。

将10.168.0.2和10.168.0.3结点信息加入10.168.0.1的statistic.json中的操作如下：

|  |
| --- |
| enode://4b6c5a3d26bf12f27cca2f8d920669484f1c7eeadc83b042d4f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3c9945270@**10.168.0.2**:28000?discport=0  enode://5c7d6b4c37ca23a27cca2f8d920669484f1c8ffbed94c153e5f69fe83a12ad328d08b4667fcc4a1b10babc05acb247a5dc91d757ba75c9f3be7742b3d0056381@**10.168.0.3**:28000?discport=0 |

# 5 总结

以太坊1.8版本目前支持2种共识算法：工作量证明机制（PoW）和权威证明机制（PoA）。其中PoW消耗的内存多，功耗较高，适用于公网，但存在51%算力攻击的风险；而PoA消耗的功率低，性能高，适用于私有链和联盟链，但却牺牲了信任度。

在选择使用PoW还是PoA搭建私有链时，我们需要对两种共识引擎进行全面的了解，再根据我们实际的业务要求选择更合理的私有链模型。