《区块链》期末 project 热身报告

陈铭涛

16340024

1. **以太坊的安装、私有链创世区块搭建、私有链节点的加入**

**安装**

本次使用 macOS 宿主机系统与在 Virtual Box 下运行的 Ubuntu 虚拟机系统作为两个节点进行。使用的以太坊客户端为 geth，使用从源码编译方式安装。

安装前的开发环境配置安装有 go 1.10.1，在目标目录下使用 git clone 获取 GitHub 上的 go-ethereum 仓库：

git clone https://github.com/ethereum/go-ethereum

在仓库目录下使用以下命令完成 geth 编译：

make geth

在 Ubuntu 下执行编译若出现错误，需从源码找到 vendor/github.com/karalabe/hid/hid\_enabled.go 文件并在其代码中的 cgo 编译参数找到并修改如下语句然后重新进行编译：

*#cgo linux,!android LDFLAGS: -lrt -liconv*

完成编译后将仓库目录下的 build/bin/geth 文件夹加入环境 PATH 中即可使用。

**私有链创世区块搭建**

使用类似如下的 genesis.json 文件作为创世区块的配置（注释为各值的作用解释，实际使用 JSON 时需删去）：

{

"config": {

*// 用于防止 Replay Attack*

"chainId": 666,

"homesteadBlock": 0,

"eip155Block": 0,

"eip158Block": 0

},

*//用于在创世区块直接向特定的地址加入指定数量的 Wei*

"alloc" : {},

*//成功挖出区块时奖励转入的地址*

"coinbase" : "0x0000000000000000000000000000000000000000",

*//挖矿过程中查找 nonce 值的难度*

"difficulty" : "0x00000001",

*//创世区块的extraData, 每个挖出的区块都可以有最长32字节的 extraData*

"extraData" : "",

*//最高 gas 花费*

"gasLimit" : "0xffffff",

*//nonce 为 PoW 挖矿时的数，与mixhash 结合以验证一个区块是否正确被挖出*

"nonce" : "0x0000000000000042",

"mixhash" : "0x0000000000000000000000000000000000000000000000000000000000000000",

*//上一区块头的哈希*

"parentHash" : "0x0000000000000000000000000000000000000000000000000000000000000000",

*//区块时间戳*

"timestamp" : "0x00"

}

使用如下命令完成创世区块创建：

geth --datadir data init genesis.json

其中 --datadir 后接区块链数据存放的目录，此处为当前目录下的 data 文件夹，也可改为其他文件夹。

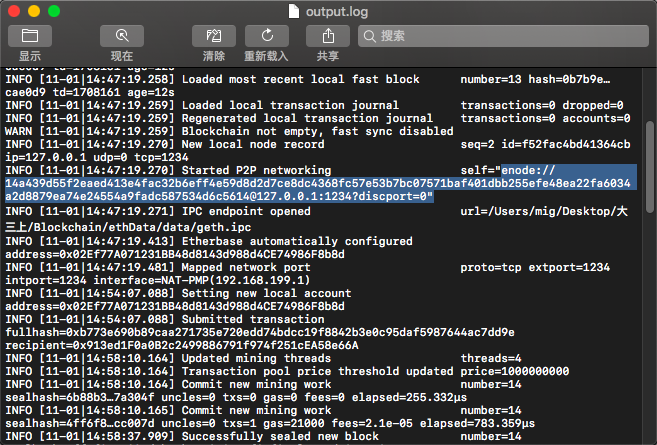
**私有链节点的加入**

在 macOS 和 Ubuntu 中都完成创世区块创建后使用如下命令启动 geth console：

geth --datadir data --nodiscover --port 1234 console 2>output.log

其中 --nodiscover 参数用于防止客户端自动连入其他 P2P 节点，--port 后接端口号，将程序的日志输出输出至 output.log 文件。

在 macOS 下的日志文件中找到节点 enode：

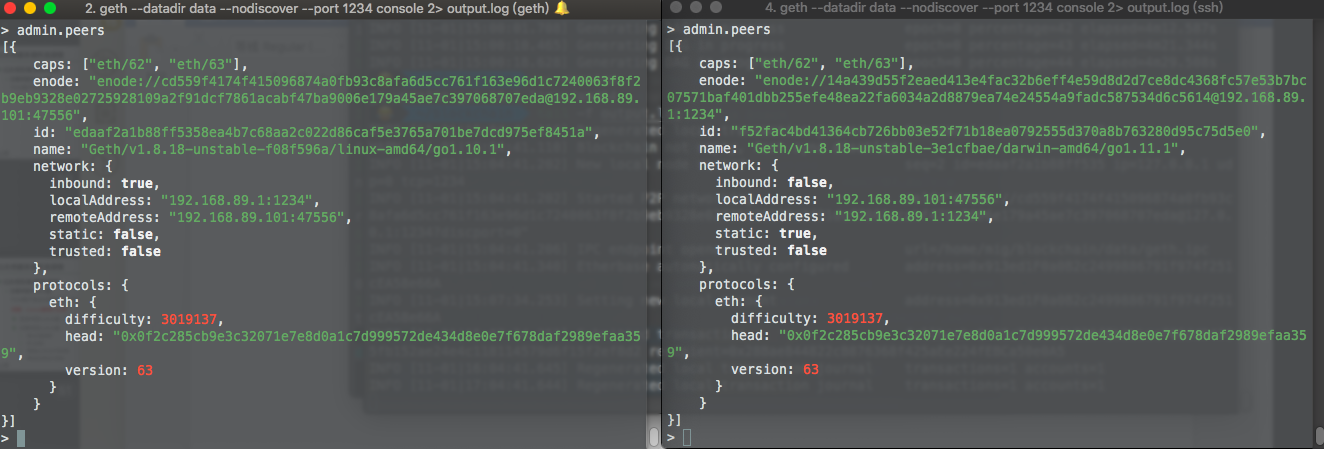


在虚拟机下的geth console 中使用如下代码添加节点：

admin.addPeer("enode://14a439d55f2eaed413e4fac32b6eff4e59d8d2d7ce8dc4368fc57e53b7bc07571baf401dbb255efe48ea22fa6034a2d8879ea74e24554a9fadc587534d6c5614@192.168.89.1:1234")

函数的参数为添加的节点的 enode，后面需加上添加节点的 ip 与端口。

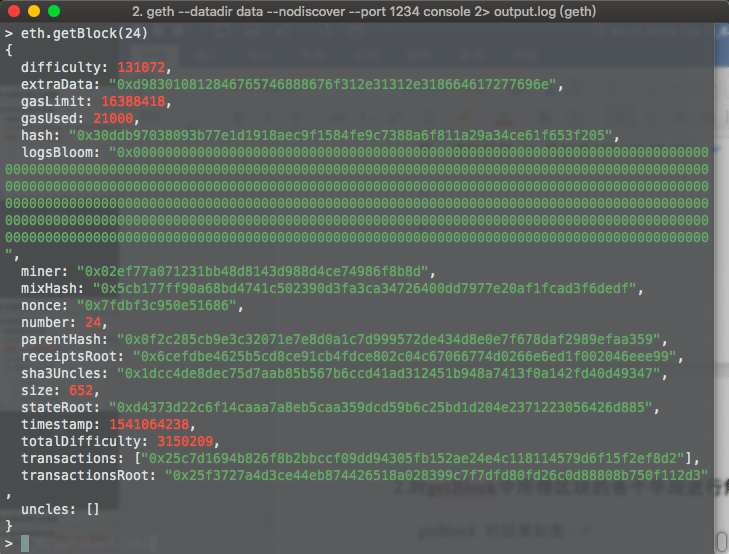
若成功查看 admin.peers 便可以看到添加的节点：



（图中左为 macOS 下的结果，右为使用 ssh 连接至虚拟机完成操作后的结果，虚拟机的 ip 地址为192.168.89.101）

1. **对getBlock中所得区块的各个字段进行解释**

getBlock 的结果如图：



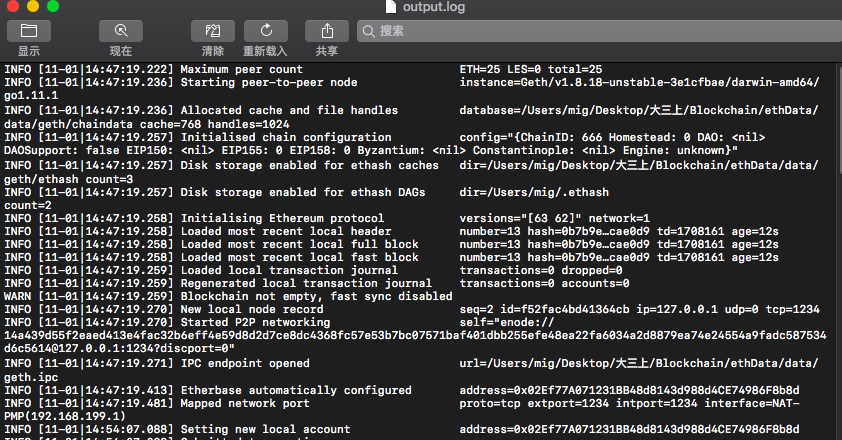
各字段的作用分别如下：

* difficulty: 当前区块的难度
* extraData: 当前区块的32字节额外信息
* gasLimit: 当前区块允许的最大 gas 花费
* gasUsed: 当前区块所有 transaction 所使用的总 gas
* hash: 区块的哈希值
* logsBloom: 区块日志的布隆过滤器
* miner: 挖掘出该区块的矿工地址
* mixHash: 用于 PoW 验证区块有效性。
* nonce: 与 mixHash 结合用于验证区块的有效性
* number: 区块的编号
* parentHash: 上一区块的哈希
* receiptsRoot: transaction 结果的 trie 根
* sha3Uncles: 区块 uncles 的 sha3结果
* size: 区块大小，单位为字节
* stateRoot: 区块最终状态的trie 根
* timestamp: 区块时间戳
* totalDifficulty: 区块链到此区块为止的总难度
* transactions: 当前区块记录的 transaction 的 hash 列表
* transactionsRoot: 当前区块的 transaction trie 的根
* uncles: uncle blocks 的列表，uncle blocks 是上一区块为当前区块祖先（前6个区块内）的其他区块。

1. **对日志输出进行解释**

日志主要的几种输出如下：

1. 客户端程序启动时日志如下：



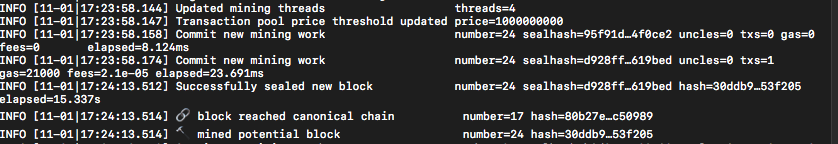
主要内容为对读取区块链配置，初始化以太坊协议，分配缓存与数据文件空间，读取最近的区块数据，启动 P2P 网络。

1. 创建一笔 transaction 时的日志如下：



包含了 transaction 的哈希和接收方地址，此时 transaction 将为 pending 状态，等待写入新区块。

1. 挖矿过程中日志如下：



其中updated mining threads 将挖矿线程设为4 表示使用4个进程开始挖矿；commit new mining word 表示 miner 已启动但是尚未完成区块；Successfully sealed new block 表示已完成一个区块的 PoW 求解。mined potential block 表示该区块已挖出；block reached canonical chain表示区块已加入该私有链的主链。

1. **编写简单的智能合约，在 remix 下进行调试，并部署在链上进行调用**

使用了如下的简单的智能合约，支持的功能为获取合约 balance，将合约 balance 取回 sender，以及将合约的一定量的 balance 发送至指定地址：

pragma solidity ^0.4.18;

**contract** buy {

**event** fallbackTrigged(bytes data);

function() **public** **payable**{emit fallbackTrigged(msg.data);}

**function** buyFrom(address seller, uint amount) **public** {

seller.transfer(amount);

}

**function** withdraw() **public** returns (uint) {

uint left = address(this).balance;

msg.sender.transfer(left);

return left;

}

**function** getCurrBalance() **public** **view** returns (uint) {

return address(this).balance;

}

}

部署前的 account balance:



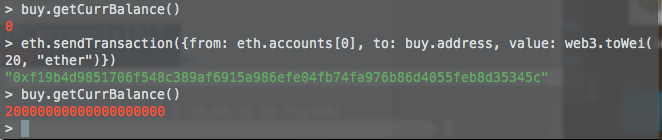
从 Remix 编译获取以下内容复制到 geth 中进行链上部署：



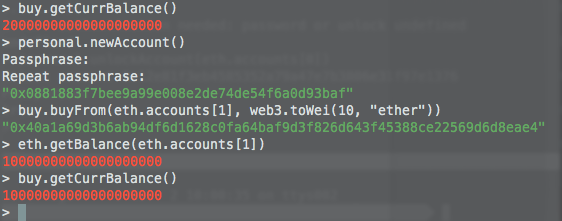
部署完后可查到 account balance 已被扣除对应的 gas：



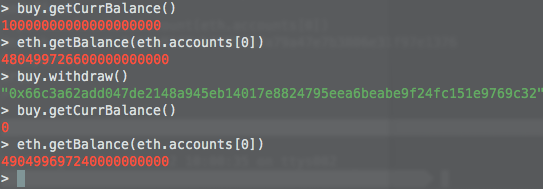
向合约转入 ether 前后调用 checkCurrBalance 查看当前合约的 balance:



调用buyFrom 向另一新创建的 account转入 balance:

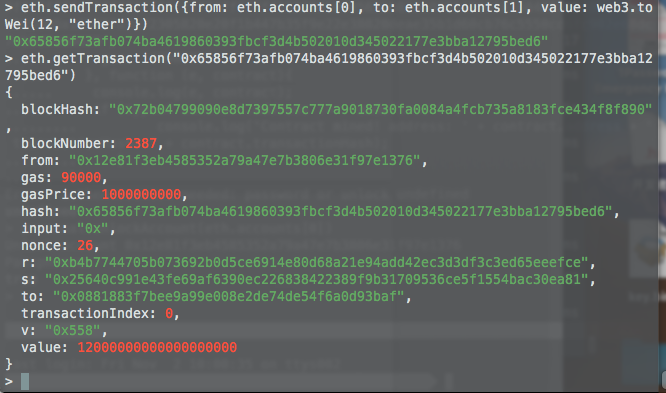


调用 withdraw 向原账户转回 balance:



1. **对交易的字段进行解释**

交易的各字段如下：



各字段解释如下：

* blockHash: 记录该条交易的区块哈希
* blockNumber: 记录该条交易的区块编号
* gas: 交易所需的 gas 数量
* gasPrice: gas 价格，gas \* gasPrice 为所支付的矿工费
* hash: 交易的哈希值
* input: 发往接收地址的消息的字节码
* nonce: 发送账号的发送交易序号
* r, s, v: 三者用于 ECDSA 签名
* from, to: 交易发送和接收方地址
* transactionIndex: 区块中该交易的索引值
* value: 交易数额，单位为 wei。