

以太坊区块链网络部署及验证实验

姓 名：范红乐

学 号：2025E8007382043

培养单位：国家空间科学中心

0. 实验目标

通过部署一个包含至少4个节点的以太坊私有区块链网络，掌握以太坊私链的搭建、节点间通信、挖矿机制以及智能合约的部署与调用方法，从而深入理解区块链网络的基本原理和操作流程

1. 环境搭建

- 基础系统：WSL2 搭载 Ubuntu 20.04.6 LTS
- 编译环境：Golang 1.19，为 Geth 客户端提供编译与运行支持
- 以太坊客户端：Geth 1.10.25，用于搭建和管理私有区块链网络

1.1. Go 1.19安装

1. 下载 Go (v1.19)

```
 wget https://dl.google.com/go/go1.19.linux-amd64.tar.gz
```

2. 解压文件

```
 sudo tar -C /usr/local -xzf go1.19.linux-amd64.tar.gz
```

3. 设置环境变量

```
 echo "export PATH=\$PATH:/usr/local/go/bin" >> ~/.profile
```

4. 使更改生效

```
 source ~/.profile
```

```
fanh@DESKTOP-AE1JHHA:~/mi < fanh@DESKTOP-AE1JHHA:~ > + -
```

```
fanh@DESKTOP-AE1JHHA:~$ uname -m  
x86_64  
fanh@DESKTOP-AE1JHHA:~$ ls /usr/local  
bin etc games include lib man sbin share src  
fanh@DESKTOP-AE1JHHA:~$ wget https://dl.google.com/go/g01.19.linux-amd64.tar.gz  
--2025-12-10 10:38:01-- https://dl.google.com/go/g01.19.linux-amd64.tar.gz  
Resolving dl.google.com (dl.google.com)... 120.253.253.97  
Connecting to dl.google.com (dl.google.com)|120.253.253.97|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 148796421 (142M) [application/x-gzip]  
Saving to: 'g01.19.linux-amd64.tar.gz'  
  
g01.19.linux-amd64.tar.gz 100%[=====] 141.90M 132KB/s in 16m 38s  
2025-12-10 10:54:39 (146 KB/s) - 'g01.19.linux-amd64.tar.gz' saved [148796421/148796421]  
  
fanh@DESKTOP-AE1JHHA:~$ sudo tar -C /usr/local -xzf g01.19.linux-amd64.tar.gz  
[sudo] password for fanh:  
fanh@DESKTOP-AE1JHHA:~$ ls /usr/local  
bin etc games go include lib man sbin share src  
fanh@DESKTOP-AE1JHHA:~$ ls /usr/local/go  
CONTRIBUTING.md PATENTS SECURITY.md api codereview.cfg lib pkg test  
LICENSE README.md VERSION bin doc misc src  
fanh@DESKTOP-AE1JHHA:~$ echo "export PATH=\$PATH:/usr/local/go/bin" >> ~/.profile  
fanh@DESKTOP-AE1JHHA:~$ source ~/.profile  
fanh@DESKTOP-AE1JHHA:~$ go version  
go version g01.19 linux/amd64  
fanh@DESKTOP-AE1JHHA:~$
```

1.2. Geth 1.10.25 安装

1. 下载 Geth (v1.10.25)

```
 wget https://gethstore.blob.core.windows.net/builds/geth-linux-amd64-  
1.10.25-69568c55.tar.gz
```

2. 解压文件

```
 tar -xvzf geth-linux-amd64-1.10.25-69568c55.tar.gz
```

3. 移动 Geth 到可执行目录

```
 sudo mv geth-linux-amd64-1.10.25-69568c55/geth /usr/local/bin/
```

```
fanh@DESKTOP-AE1JHHA:~/mi < fanh@DESKTOP-AE1JHHA:~ > + -
```

```
geth-linux-amd64-1.10.25-69568c55/geth  
fanh@DESKTOP-AE1JHHA:~$ sudo cp geth-linux-amd64-1.10.25-69568c55/geth/usr/local/bin  
[sudo] password for fanh:  
cp: cannot create regular file '/usr/local/bin': No such file or directory  
fanh@DESKTOP-AE1JHHA:~$ sudo cp geth-linux-amd64-1.10.25-69568c55/geth /usr/local/bin  
fanh@DESKTOP-AE1JHHA:~$ ls /usr/local  
bin etc games go include lib man sbin share src  
fanh@DESKTOP-AE1JHHA:~$ ls /usr/local/bin  
geth  
fanh@DESKTOP-AE1JHHA:~$ cd ~  
fanh@DESKTOP-AE1JHHA:~$ rm geth-linux-amd64-1.10.25-69568c55.tar.gz  
fanh@DESKTOP-AE1JHHA:~$ ls  
Desktop geth-linux-amd64-1.10.25-69568c55 g01.19.linux-amd64.tar.gz  
fanh@DESKTOP-AE1JHHA:~$ rm g01.19.linux-amd64.tar.gz  
fanh@DESKTOP-AE1JHHA:~$ ls  
Desktop geth-linux-amd64-1.10.25-69568c55  
fanh@DESKTOP-AE1JHHA:~$ geth version  
Geth  
Version: 1.10.25-stable  
Git Commit: 69568c554880b3567bace64f8848ff1be27d084d  
Git Commit Date: 20220915  
Architecture: amd64  
Go Version: go1.18.5  
Operating System: linux  
GOPATH=  
GOROOT=go  
fanh@DESKTOP-AE1JHHA:~$
```

2. 网络部署

2.1. 创世区块配置

- **自定义文件目录结构说明**

```
/home/fanh1/ethdata/      # 实验文件夹
├── genesis.json          # 创世区块信息文件
├── node1/                 # 节点1数据目录
├── node2/                 # 节点2数据目录
├── node3/                 # 节点3数据目录
├── node4/                 # 节点4数据目录
└── logs/                  # 节点运行日志
```

- **添加创世区块配置文件**

nano genesis.json

```
{
  "config": {
    "chainId": 1001,
    "homesteadBlock": 0,
    "eip150Block": 0,
    "eip155Block": 0,
    "eip158Block": 0
  },
  "coinbase": "0x000000000000000000000000000000000000000000000000000000000000000",
  "difficulty": "0x1111111",
  "extraData": "",
  "gasLimit": "0x2fefd8",
  "nonce": "0x0000000000000042",
  "mixhash": "0x000000000000000000000000000000000000000000000000000000000000000",
  "parentHash": "0x000000000000000000000000000000000000000000000000000000000000000",
  "timestamp": "0x00",
  "alloc": {}
}
```

创世区块配置文件字段说明

字段	取值范围	说明
<code>config</code>	对象类型	区块链的网络配置和硬分叉规则
<code>chainId</code>	整数: 1-65535	链ID, 用于区分不同以太坊网络 (1:主网, 3:Ropsten, 4:Rinkeby, 5:Goerli, 1337:常见开发链, 其他:私有链)
<code>homesteadBlock</code>	整数: 0-∞	从第0个区块启用该硬分叉 (0:立即启用, >0:在指定区块高度启用, null:禁用该分叉)
<code>eip150Block</code>	整数: 0-∞	0:立即启用, 与homesteadBlock类似
<code>eip155Block</code>	整数: 0-∞	0:立即启用, 通常与eip150Block设置相同
<code>eip158Block</code>	整数: 0-∞	0:立即启用, 必须≥eip155Block
<code>conibase</code>	20字节地址	创世区块的矿工地址 (以太坊中称 beneficiary), 此处为零地址, 因为创世区块无实际矿工
<code>difficulty</code>	十六进制字符串: 0x1- 0xFFFFFFFFFFFFFFFFF	初始挖矿难度。值较低便于私有链/测试网快速生成区块
<code>extraData</code>	十六进制字符串: 0-64字符(0-32字节)	附加信息, 可为任意数据 (最长32字节)。 此处为空, 常用于标识矿工或添加备注
<code>gasLimit</code>	十六进制字符串: 0x15F90- 0xFFFFFFFF	每个区块的Gas上限, 限制区块内交易的总计算量。此值约为以太坊主网初始值 (500万) 的60%
<code>nonce</code>	十六进制字符串: 0x0- 0xFFFFFFFFFFFFFFFFF	与mixhash配合用于工作量证明 (PoW) 的随机数。创世区块中该值通常固定
<code>mixhash</code>	32字节哈希值(64字符)	PoW算法中与nonce共同生成区块哈希的哈希值。创世区块固定为零哈希
<code>parentHash</code>	32字节哈希值(64字符)	父区块哈希。创世区块无父区块, 故为零哈希
<code>timestamp</code>	十六进制字符串: 0x0-0xFFFFFFFF	创世区块生成时间戳 (Unix时间戳)。0表示1970年1月1日, 实际运行时会更新
<code>alloc</code>	对象或空对象{}	预分配初始账户和余额。此处为空对象, 表示无预挖代币

```
fanhl@DESKTOP-AE1JHHA:~$ cd ~
fanhl@DESKTOP-AE1JHHA:~$ mkdir ethdata
fanhl@DESKTOP-AE1JHHA:~$ cd ethdata/
fanhl@DESKTOP-AE1JHHA:~/ethdata$ mkdir node1 node2 node3 node4 logs
fanhl@DESKTOP-AE1JHHA:~/ethdata$ ls
logs node1 node2 node3 node4
fanhl@DESKTOP-AE1JHHA:~/ethdata$ nano genesis.json
fanhl@DESKTOP-AE1JHHA:~/ethdata$ cat genesis.json
{
  "config": {
    "chainId": 1001,
    "homesteadBlock": 0,
    "eip150Block": 0,
    "eip155Block": 0,
    "eip158Block": 0
  },
  "coinbase" : "0x0000000000000000000000000000000000000000",
  "difficulty" : "0x1111111",
  "extraData" : "",
  "gasLimit" : "0x2fefd8",
  "nonce" : "0x0000000000000042",
  "mixhash" : "0x0000000000000000000000000000000000000000000000000000000000000000",
  "parentHash" : "0x0000000000000000000000000000000000000000000000000000000000000000",
  "timestamp" : "0x00",
  "alloc" : {}
}
fanhl@DESKTOP-AE1JHHA:~/ethdata$
```

2.2. 启动私链

- 初始化以太坊4个节点的Geth客户端

```
# 当前目录 /home/fanh1/ethdata

# node1
geth --datadir ./node1 init ./genesis.json

# node2
geth --datadir ./node2 init ./genesis.json

# node3
geth --datadir ./node3 init ./genesis.json

# node4
geth --datadir ./node4 init ./genesis.json
```

以太坊节点初始化命令说明

参数	说明
<code>geth</code>	可执行文件， Go Ethereum客户端的主程序
<code>--datadir ./node</code>	数据目录参数，指定节点数据的存储位置
<code>init</code>	初始化命名，创建新区块链的初始化操作
<code>./genesis.json</code>	创世文件路径，定义区块链初始状态的配置文件

```
fanhl@DESKTOP-AE1JHHA:~/ethdata$ ls -a
. . . genesis.json logs node1 node2 node3 node4
fanhl@DESKTOP-AE1JHHA:~/ethdata$ geth --datadir ./node1 init ./genesis.json
INFO [12-11|11:16:13.488] Maximum peer count                         ETH=50 LES=0 total=50
INFO [12-11|11:16:13.490] Smartcard socket not found, disabling      err="stat /run/pcscd/pcscd.comm: no such fi
le or directory"
INFO [12-11|11:16:13.496] Set global gas cap                           cap=50,000,000
INFO [12-11|11:16:13.497] Allocated cache and file handles           database=/home/fanhl/ethdata/node1/geth/cha
indata cache=16.00MiB handles=16
INFO [12-11|11:16:13.542] Opened ancient database                      database=/home/fanhl/ethdata/node1/geth/cha
indata/ancient/chain readonly=false
INFO [12-11|11:16:13.542] Writing custom genesis block                  nodes=0 size=0.00B time="5.073µs" gcnodes=0
INFO [12-11|11:16:13.542] Persisted trie from memory database          database=chaindata
gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [12-11|11:16:13.543] Successfully wrote genesis state            database=/home/fanhl/ethdata/node1/geth/lig
htchain
hash=cc97a1..9c57c9
INFO [12-11|11:16:13.543] Allocated cache and file handles           database=/home/fanhl/ethdata/node1/geth/lig
htchain
htchaindata cache=16.00MiB handles=16
INFO [12-11|11:16:13.586] Opened ancient database                      database=/home/fanhl/ethdata/node1/geth/lig
htchain
htchaindata/ancient/chain readonly=false
INFO [12-11|11:16:13.586] Writing custom genesis block                  nodes=0 size=0.00B time="4.204µs" gcnodes=0
INFO [12-11|11:16:13.588] Persisted trie from memory database          database=lightchain
gcsiz=0.00B gctime=0s livenodes=1 livesize=0.00B
INFO [12-11|11:16:13.588] Successfully wrote genesis state            database=lightchain
hash=cc97a1..9c57c9
fanhl@DESKTOP-AE1JHHA:~/ethdata$
```

- 启动私链

```
# 当前目录 /home/fanhl/ethdata

# node1
geth --datadir ./node1 --networkid 1001 --identity "node1" --port 30303 --
http --http.port 8545 --authrpc.port 8551 --nodiscover --verbosity 4 console
2>> node1.log

# node2
geth --datadir ./node2 --networkid 1001 --identity "node2" --port 30304 --
http --http.port 8546 --authrpc.port 8552 --nodiscover --verbosity 4
console 2>> node2.log

# node3
geth --datadir ./node3 --networkid 1001 --identity "node3" --port 30305 --
http --http.port 8547 --authrpc.port 8553 --nodiscover --verbosity 4
console 2>> node3.log

# node4
geth --datadir ./node4 --networkid 1001 --identity "node4" --port 30306 --
http --http.port 8548 --authrpc.port 8554 --nodiscover --verbosity 4
console 2>> node4.log
```

```
# 启动私链命令说明
```

参数	说明
<code>--datadir ./node</code>	指定存储节点数据的目录
<code>--network 1001</code>	设置私有网络的网络ID为1001，与配置文件中chainId一致，同一网络需相同
<code>--identity "node1"</code>	设置节点名称
<code>--port 30303</code>	设置节点间通信端口，默认为以太坊P2P端口
<code>--http</code>	开启HTTP-RPC接口，通过HTTP请求与节点交互
<code>--nodiscover</code>	禁用自动发现节点，使节点不主动发现其他节点，适用于私有网络
<code>--verbosity 4</code>	设置日志详细级别为4，提供较详细的日志输出
<code>console</code>	打开Geth的JavaScript控制台，允许节点交互
<code>2 > node.log</code>	将错误日志输出到文件

• 另开终端监听 log

```
# 实时查看 node1.log 文件内容
tail -f node1.log
```

```
fanh1@DESKTOP-AE1JHHA:~/ethdata$ tail -f node1.log
INFO [12-11|15:21:18.279] Transaction pool stopped
DEBUG[12-11|15:21:18.280] Journalled generator progress
DEBUG[12-11|15:21:18.280] Journalled disk layer
INFO [12-11|15:21:18.280] Writing snapshot state to disk
INFO [12-11|15:21:18.280] Persisted trie from memory database
1 livesize=0.00B
INFO [12-11|15:21:18.280] Writing clean trie cache to disk
INFO [12-11|15:21:18.281] Persisted the clean trie cache
INFO [12-11|15:21:18.281] Blockchain stopped
DEBUG[12-11|15:21:18.281] Read error
DEBUG[12-11|15:21:18.281] Deleting port mapping
tail: node1.log: file truncated
INFO [12-11|15:37:49.380] Maximum peer count
INFO [12-11|15:37:49.381] Smartcard socket not found, disabling
DEBUG[12-11|15:37:49.381] FS scan times
DEBUG[12-11|15:37:49.381] FS scan times
DEBUG[12-11|15:37:49.381] Sanitizing Go's GC trigger
INFO [12-11|15:37:49.383] Set global gas cap
INFO [12-11|15:37:49.385] Allocated trie memory caches
INFO [12-11|15:37:49.385] Allocated cache and file handles
8
DEBUG[12-11|15:37:49.485] Chain freezer table opened
items=0 size=0.00B
DEBUG[12-11|15:37:49.486] Chain freezer table opened
items=0 size=0.00B
DEBUG[12-11|15:37:49.486] Chain freezer table opened
items=0 size=0.00B
DEBUG[12-11|15:37:49.487] Chain freezer table opened
items=0 size=0.00B
DEBUG[12-11|15:37:49.488] Chain freezer table opened
items=0 size=0.00B
INFO [12-11|15:37:49.488] Opened ancient database
e
DEBUG[12-11|15:37:49.488] Current full block not old enough
INFO [12-11|15:37:49.488]
-----
```

2.3 多节点交互

1. 在节点1的控制台中获取enode信息

```
admin.nodeInfo.enode // 获取node1信息
```

```

fanhl@DESKTOP-AE1JHHA:~$ geth --datadir ./node1 --networkid 1001 --identity "node1" --port 30303 --http --http.port 8545 --nodiscover --verbosity 4 console 2> node1.log
Welcome to the Geth JavaScript console!

instance: Geth/node1/v1.10.25-stable-69568c55/linux-amd64/go1.18.5
at block: 0 (Thu Jan 01 1970 08:00:00 GMT+0800 (CST))
datadir: /home/fanhl/node1
modules: admin:1.0 debug:1.0 engine:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0
web3:1.0

To exit, press ctrl-d or type exit
> admin.nodeInfo.enode
"enode://df6531e1e056ae8cbaf827cf8fed38a9b8bd1a6bfe7e4f9b98529b3aa6abb8cd30103e08fe14f832a4e063ccc18b47fd8ffae0bb6fb86318ee81ce4c15cd123@127.0.0.1:30303?discport=0"

```

2. 另开新的终端，进入节点2的控制台，添加节点1信息并验证

```

admin.addPeer("从节点1获取的信息")
admin.peers // 验证节点信息
admin.nodeInfo.enode // 获取node2信息

```

```

fanhl@DESKTOP-AE1JHHA:~$ fanhl@DESKTOP-AE1JHHA:~$ fanhl@DESKTOP-AE1JHHA:~$ fanhl@DESKTOP-AE1JHHA:~$ fanhl@DESKTOP-AE1JHHA:~$ fanhl@DESKTOP-AE1JHHA:~$ fanhl@DESKTOP-AE1JHHA:~$ + - x
> admin.addPeer("enode://9cf19607ef873c7b8c4a50f2644e8b7ada2d52508c382f508197c3af3950ab7f2769a67415a9ea3792e1fb361f85fafc434a937af130bf5c875460efd0f3b31b@127.0.0.1:30303?discport=0")
true
> admin.peers

[{
  caps: ["eth/66", "eth/67", "snap/1"],
  enode: "enode://9cf19607ef873c7b8c4a50f2644e8b7ada2d52508c382f508197c3af3950ab7f2769a67415a9ea3792e1fb361f85fafc434a937af130bf5c875460efd0f3b31b@127.0.0.1:30303?discport=0",
  id: "d8ee3e5e1f5fb2e8b0cdfb406f1fdabeff80410ca54741eb2ffa873243caf77e",
  name: "Geth/node1/v1.10.25-stable-69568c55/linux-amd64/go1.18.5",
  network: {
    inbound: false,
    localAddress: "127.0.0.1:38380",
    remoteAddress: "127.0.0.1:30303",
    static: true,
    trusted: false
  },
  protocols: {
    eth: {
      difficulty: 17895697,
      head: "0xcc97a15707e1a3cf64433ce47735b831073c2995ad8b0618169abb8dac9c57c9",
      version: 67
    },
    snap: {
      version: 1
    }
  }
}
]
```

3. 另开新的终端，进入节点3的控制台，添加节点1、2信息并验证

```

admin.addPeer("从节点1获取的信息")
admin.addPeer("从节点2获取的信息")
admin.peers // 验证节点信息
admin.nodeInfo.enode // 获取node3信息

```

```

> admin.peers
[{
  caps: ["eth/66", "eth/67", "snap/1"],
  enode: "enode://3cccc628239790c6eabac4937e4b8c3cfab355824972035cad61244acdb0fc10dfa69c16a0a6b382613bf3e8fc869888460531396420e8e690b41d2336b7cce@127.0.0.1:30304?discport=0",
  id: "c9a15f10befa9a01d25cb0bc9983567a64da0bac3633934ba4da2fe8b3bbeac",
  name: "Geth/node2/v1.10.25-stable-69568c55/linux-amd64/go1.18.5",
  network: {
    inbound: false,
    localAddress: "127.0.0.1:36554",
    remoteAddress: "127.0.0.1:30304",
    static: true,
    trusted: false
  },
  protocols: {
    eth: {
      difficulty: 17895697,
      head: "0xcc97a15707e1a3cf64433ce47735b831073c2995ad8b0618169abb8dac9c57c9",
      version: 67
    },
    snap: {
      version: 1
    }
  }
}, {
  caps: ["eth/66", "eth/67", "snap/1"],
  enode: "enode://9cf19607ef873c7b8c4a50f2644e8b7ada2d52508c382f508197c3af3950ab7f2769a67415a9ea3792e1fb361f85fafc434a937af130bf5c875460efd0f3b31b@127.0.0.1:30303?discport=0",
  id: "d8ee3e5e1f5fb2e8b0cdfb406f1fdabeff80410ca54741eb2ffa873243caf77e",
  name: "Geth/node1/v1.10.25-stable-69568c55/linux-amd64/go1.18.5",
  network: {
    inbound: false,
    localAddress: "127.0.0.1:38418",
    remoteAddress: "127.0.0.1:30303",
    static: true,
    trusted: false
  },
  protocols: {
    eth: {
      difficulty: 17895697,
      head: "0xcc97a15707e1a3cf64433ce47735b831073c2995ad8b0618169abb8dac9c57c9",
      version: 67
    },
    snap: {
      version: 1
    }
  }
}]

```

4. 另开新的终端，进入节点4的控制台，添加节点1、2、3信息并验证

```
admin.addPeer("从节点1获取的信息")
admin.addPeer("从节点2获取的信息")
admin.addPeer("从节点3获取的信息")
admin.peers // 验证节点信息
```

2.4 节点信息

- node1

```
        difficulty: 17895697,
        genesis:
"0xcc97a15707e1a3cf64433ce47735b831073c2995ad8b0618169abb8dac9c57c9",
        head:
"0xcc97a15707e1a3cf64433ce47735b831073c2995ad8b0618169abb8dac9c57c9",
        network: 1001
    },
    snap: []
}
}
```

- node2

- node3

- node4

```
{  
    enode:  
        "enode://a360182b893a90f3f769d2116d239c0a354a020c7c6bb9463cd76822a82c060942d  
        3c92330ef7aaf59e416dc292011f5863a6c511623ec63567ece626e83e392@127.0.0.1:3030  
        6?discport=0",  
        enr: "enr:-  
        Jy4QHdRgoZqgJM5UKrisc12kHHc8RH0Awvhk0AtwH_JkpoA26cR8QomHcIRFn72JmfVfkw_JmEc  
        SYrEGYQoxx30y2GAZsMSzLPg2v0aMfGhJ0r3hCAGm1kgnY0gm1whH8AAAGJc2VjcdI1NmsxoQKjY  
        BgriTqq8_dp0hFtI5wKNUoCDHxruruY812giqCwGCYRzbmFwwIN0Y3CCdmI",  
        id: "5a153cbdee056a0b6a1485339705b71f8d9a179c0efe786b77edc132437f2eba",  
        ip: "127.0.0.1",  
        listenAddr: "[::]:30306",  
        name: "Geth/node4/v1.10.25-stable-69568c55/linux-amd64/go1.18.5",  
        ports: {  
            discovery: 0,  
            listener: 30306  
        },  
        protocols: {
```

3. Remix连接

3.1 创建账户与ETH获取

合约部署需要账户中ETH > 0，预先挖矿，获取一些ETH

1. 启动节点1，进入JavaScript控制台

```
geth --datadir ./node1 --networkid 1001 --identity "node1" --port 30303 --  
http --http.port 8545 --nodiscover --verbosity 4 console 2> node1.log
```

2. 创建新账户，将提示输入密码，并生成一个新的以太坊地址。

```
# 两种方式  
personal.newAccount() // 创建新账户，需要再设置密码  
personal.newAccount("lab1") // 创建一个密码为lab1的账户
```

```
> fanhl@DESKTOP-AE1JHHA:~/ethdata$ geth --datadir ./node1 --networkid 1001 --identity "node1" --port 30303 --http --http.port 8545 --nodiscover --verbosity 4 console 2> node1.log
Welcome to the Geth JavaScript console!

instance: Geth/node1/v1.10.25-stable-69568c55/linux-amd64/go1.18.5
at block: 0 (Thu Jan 01 1970 08:00:00 GMT+0800 (CST))
datadir: /home/fanhl/ethdata/node1
modules: admin:1.0 debug:1.0 engine:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0
web3:1.0

To exit, press ctrl-d or type exit
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x9ddc7add63e30d46bb8d18548614bcea3357b394"
>
```

3. 挖矿

启动挖矿后，虽然显示null但是实际上在后台运行，可以通过查看区块高度看到数字在增长

```
miner.start() // 启动挖矿  
eth.accounts // 显示在本地创建的所有账户地址  
eth.blockNumber // 获得当前区块高度  
miner.stop() // 关闭挖矿
```

```

> fanhl@DESKTOP-AE1JHHA:~/ethdata$ geth --datadir ./node1 --networkid 1001 --identity "node1" --port 30303 --http --http.port 8545 --nodiscover --verbosity 4 console 2> node1.log
Welcome to the Geth JavaScript console!

instance: Geth/node1/v1.10.25-stable-69568c55/linux-amd64/go1.18.5
at block: 0 (Thu Jan 01 1970 08:00:00 GMT+0800 (CST))
datadir: /home/fanh1/ethdata/node1
modules: admin:1.0 debug:1.0 eth:1.0 engine:1.0 ethash:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0
web3:1.0

To exit, press ctrl-d or type exit
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x9ddc7add63e30d46bb8d18548614bcea3357b394"
> miner.start()
null
> miner.stop()
null
> eth.accounts
["0x9ddc7add63e30d46bb8d18548614bcea3357b394"]
> eth.blockNumber
0
> ■

> miner.start()
null
> eth.blockNumber
13
> eth.accounts
["0x9ddc7add63e30d46bb8d18548614bcea3357b394", "0xf595c507fdbb3d00ab3deca5693a1053f6f6e3e8"]
> ■

```

3.2 创建并编译合约

- 进入 <https://remix.ethereum.org/>，在 Remix 顶部选择默认工作台 default_workspace
- 在 File Explorer 页面新建 contract.sol 文件，用于存储与查询键值对

```

// SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.4.25;

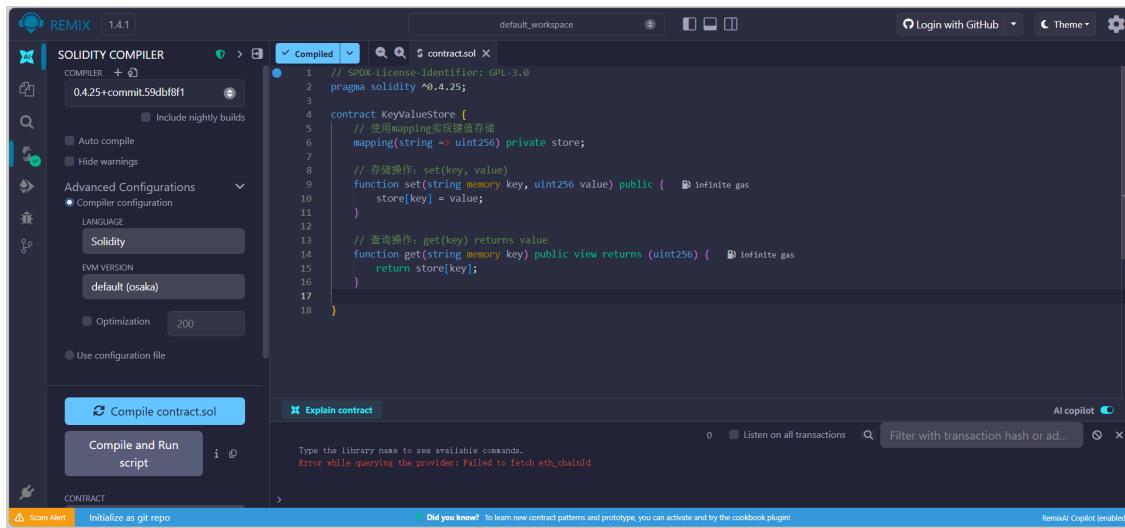
contract Keyvaluestore {
    // 使用mapping实现键值存储
    mapping(string => uint256) private store;

    // 存储操作: set(key, value)
    function set(string memory key, uint256 value) public {
        store[key] = value;
    }

    // 查询操作: get(key) returns value
    function get(string memory key) public view returns (uint256) {
        return store[key];
    }
}

```

- 在 Solidity Compiler 页面点击 Compiled 编译该文件，出现 ✓ 即编译成功



3.3 连接Remix并部署合约

1. 添加http参数后，重新启动节点1，否则无法连接Remix

```
# node1
geth --datadir ./node1 --networkid 1001 --identity "node1" --port 30303 --
http --http.port 8545 --http.corsdomain "https://remix.ethereum.org" --
http.api "web3,eth,debug,personal,net" --authrpc.port 8551 --allow-insecure-
unlock --nodiscover --verbosity 4 console 2>> node1.log
```

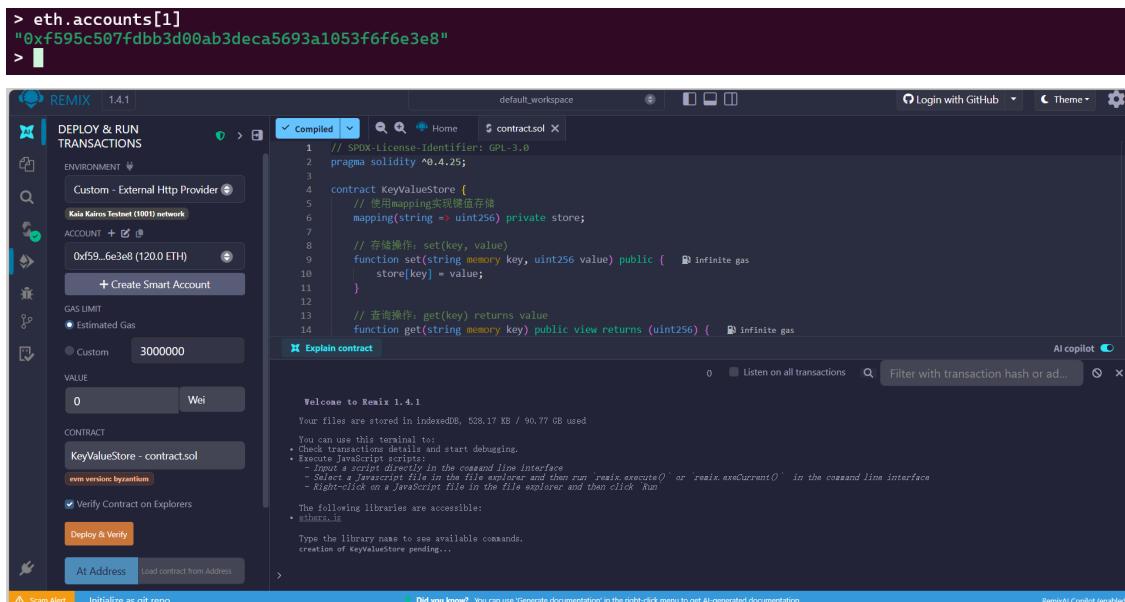
2. 在 Deploy & run transactions 页面，配置相关信息，点击 Deploy & verify

可能出现报错的几种情况与解决方法

1. 环境配置下面出现can't detect network提示

- 可能是选择 Custom - External Http Provider 时需要填写的http与实际的节点1启动时的http.port参数值不相符
 - 可能说明节点1的http配置有问题需要检查参数并重新启动
 - 也可能是--allow-insecure-unlock未生效，启动节点1后并未解锁账户，在启动节点1后使用命令personal.unlockAccount(eth.accounts[0], "你的密码", 0)单独解锁账户，结果返回true则解锁成功
 - 以上操作完成后，需要刷新remix页面

2. ETH为0导致部署失败，需要在节点1中预先挖矿，获取一定ETH，Remix中选择ACCOUNT时可以看见账户余额



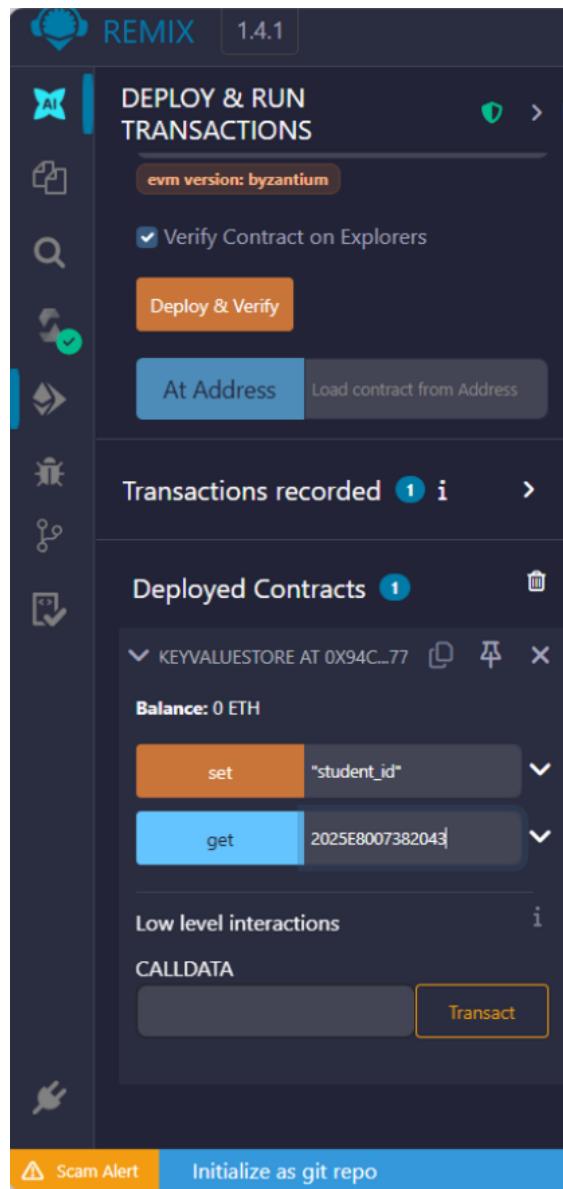
3. 部署成功可以看到 `creation of Keyvaluestore pending...` 表示交易正在提交，这时在节点1中输入miner.start(), 等待一会儿，当deployed contracts有内容可展开时，再miner.stop(), 此时 `Deployed Contracts` 可以展开

合约地址: 0x94C28129b02D861a002367d3d9820F3bff377C3f

填写set内容，点击 `transact`

key: "user_fanh_l_balance"

value: "123456"



5. 在节点1中输入miner.start(), 短暂挖矿确认交易，等待一会儿，当终端有内容可展开时，再 miner.stop()

6. 在Remix的 Deployed Contracts 中 get 输入相应的key后，点击call，可以获取对应value