**《课程综合实践Ⅱ》实验报告四**

**以太坊客户端geth安装及metamask使用**

**姓名\_\_\_ 沈云昊 专业 电子信息工程**

**学号 3220104126 联系方式 2766636730@qq.com**

## 一、实验目的

1. 在Ubuntu系统上配置必需的Golang环境。

2. 在Ubuntu系统上安装以太坊客户端geth。

3. 借助metamask钱包工具，对以太坊账户进行管理。

## 二、实验内容及要求

**准备工作：为了方便后续的学习和实践，建议同学们自带电脑，在自己电脑上进行实验内容，并完成实验报告。**

**实验步骤（仔细阅读，按照步骤完成实验）**

**（本次实验全程在ubuntu上进行操作，学在浙大上有规定版本的geth客户端文件）**

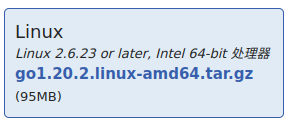
### Golang环境安装

Golang是一种静态强类型的编译型语言，语言语法与C相近，比较容易上手，实验用到的以太坊客户端geth依赖于Golang， 因此我们需要在Ubuntu中配置Golang环境，具体配置方法如下：

1. 在自带的火狐浏览器中，打开网址

https://studygolang.com/dl

选择linux-amd64版本的Go安装包下载。



1. 下载完成后，进入到Go安装包所在的目录，打开终端，使用tar命令解压到/usr/local目录中

sudo tar -C /usr/local/ -xvf go1.20.2.linux-amd64.tar.gz

1. 下面我们需要配置环境变量，使Go环境在虚拟机内生效，进入到profile文件目录，输入命令如下：

cd /etc

sudo gedit profile

在该文件中追加内容如下：

#golang

export GOROOT=/usr/local/go

export PATH=$PATH:$GOROOT/bin

export GOARCH=amd64

输入命令，使配置文件生效

source /etc/profile

至此，go环境配置完成，查看go版本验证是否配置成功。

go version

如果遇到每次重启控制台都需要运行一次 source命令才能让go生效的问题，可以查看2.1末尾尝试解决。

1. 下面我们写一个简单的文件测试

新建一个hello.go文件，在其中输入代码

package main

import "fmt"

func main() {

fmt.Println("Hello World!")

}

保存后，运行该文件，

go run hello.go

### Geth客户端安装及操作

### 2.1 Geth客户端编译配置

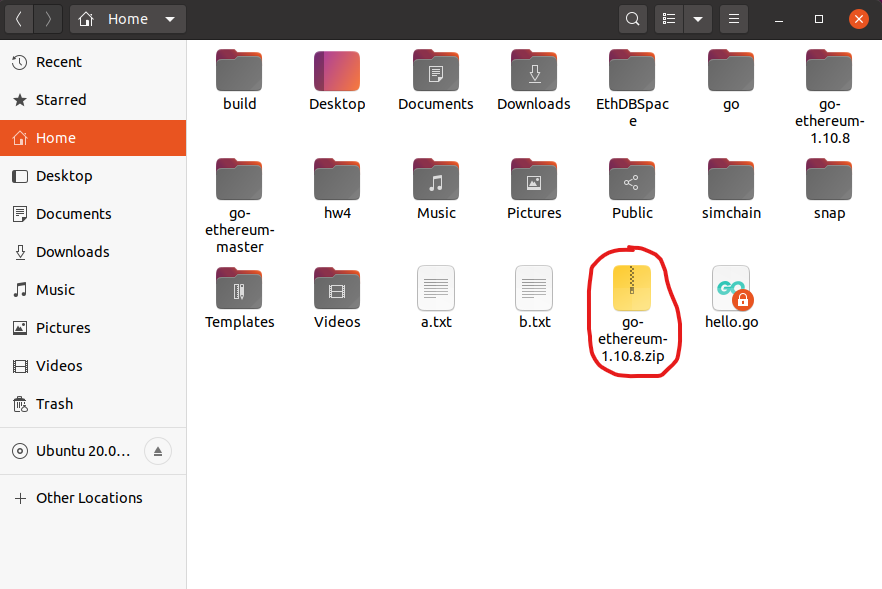
Geth是一种以后台命令行形式运行的以太坊客户端，用户可借助Geth方

便的构建自己的以太坊私有链并与以太坊网络交互

我们用源码编译的方式下载安装Geth，具体安装方法如下：

1. 在学在浙大上下载指定版本的Geth。

将geth安装包拖动到ubuntu虚拟机工作目录内，



1. 进入该文件所在目录，打开控制台输入命令解压：

unzip go-ethereum-1.10.8.zip

为方便后一步的编译，我们需要修改Go的下载源，在控制台中输入命令：

go env -w GOPROXY=https://goproxy.cn

修改完成后，进入刚刚解压的目录，

cd go-ethereum-1.10.8

输入命令开始编译，

make all

1. 编译完成后，会在go-ethereum-1.10.8目录下生成build目录，在该目录下执行命令，查看是否安装成功。

./bin/geth help

1. 进入/usr/local目录下，创建文件夹ETH；进入ETH文件夹，创建文件夹geth。

创建完成后，将刚刚编译出来的build目录移动到geth目录下，输入命令

sudo mv ./build /usr/local/ETH/geth

为了让geth相关的命令全局生效，我们需要在/etc/profile文件中追加环境变量，输入命令如下：

sudo gedit /etc/profile

在其中追加内容如下，

#geth

export GETH\_HOME=/usr/local/ETH/geth/build/bin

export PATH=$PATH:$GETH\_HOME

保存后，令环境变量生效，

source /etc/profile

输入命令测试，

geth help

至此geth客户端安装完成，如果有同学遇到重启终端，运行geth命令出

现找不到命令的问题，可以执行以下操作：

sudo gedit ~/.bashrc

在该文件的最后一行追加内容，

source /etc/profile

之后重启客户端geth相关命令应该都能正常运行。

### Geth客户端的简单操作（请截图保存在实验记录中）

在这一步中，我们将用geth工具构建我们的以太坊私链，步骤如下：

1. 进入桌面，新建创世块文件genesis.json，向其中写入内容如下：

{

"config": {

"chainId": 1337,

"homesteadBlock": 0,

"eip150Block": 0,

"eip155Block": 0,

"eip158Block": 0

},

"alloc": {},

"coinbase": "0x0000000000000000000000000000000000000000",

"difficulty": "0x400",

"extraData": "",

"gasLimit": "0xffffffff",

"nonce": "0x0000000000000042",

"mixhash": "0x0000000000000000000000000000000000000000000000000000000000000000",

"parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000",

"timestamp": "0x00"

}

对参数说明如下：

**chainId** : 以太坊区块链网络Id，ethereum主链是1，私有链只用不要与主链冲突即可。

**alloc** : 预留账户

**Coinbase**: 矿工账户

**Difficulty**: 挖矿难度，0x400，这个是简单。

**extraData**：相当于备注

**gasLimit**：最小消耗gas

**nonce** : 64位随机数，用于挖矿，注意他和mixhash的设置需要满足以太坊黄皮书中的要求

**parentHash** : 上一个区块的Hash值，因为是创世块，石头里蹦出来的，没有在它前面的，所以是0

**Timestamp** : 时间戳

1. 在桌面新建文件夹EthDBSpace，执行初始化网络命令

mkdir EthDBSpace

geth --datadir EthDBSpace init genesis.json

此时在EthDBSpace内应该已经多了两个文件夹geth和keystore，geth文件夹用于存放区块数据，keystore文件夹用于存放账户数据。

1. 启动geth客户端节点

geth --identity node1 --datadir ./EthDBSpace --rpc --rpcapi db,eth,net,web3 --rpcaddr 0.0.0.0 --rpcport 8481 --port 30301 --networkid 1337 console --dev --dev.period 1 --allow-insecure-unlock

对命令参数说明如下：

**- -Identity** : 节点身份标识，起个名字即可

**- -datadir** : 指定节点存在位置

**- -rpc** : 启用http-rpc服务器

**- -rpcapi** : 基于http-rpc提供的api接口。eth,net,web3,db…

**- -rpcaddr** : http-rpc服务器接口地址：默认“127.0.0.1”

**- -rpcport** : http-rpc 端口(多节点时，不要重复)

**- -port** : 节点端口号（多节点时，不要重复）

**- -networkid** : 网络标识符 尽量与链ID保持一致

**--dev**：使用开发者模式

**--dev.period**: 1表示自动开始挖矿，0表示只有当交易发生时才进行挖矿

1. 命令行操作私有链网络

输入命令，查看当前网络内的账户，应只有一个且为开发者账户。

>eth.accounts

现在该账户内应该还没有钱，输入命令，开始挖矿，并隔一段时间后让他停止。

>miner.start()

>miner.stop()

输入命令，新建一个账户，并将其密码设为“123456”

>personal.newAccount("123456")

此时我们继续查看网络内账户，发现变成了两个，随后我们分别查看两个账户的余额，输入命令如下：

> eth.getBalance(eth.accounts[0])

> eth.getBalance(eth.accounts[1])

发现新创建的账户余额为0，管理员账户因为之前的挖矿操作及初始化生成，已经积累了很多数字币，之后，我们可以进行转账交易，首先需要先解锁这两个账户，输入命令如下：

> personal.unlockAccount(eth.accounts[0]) ##管理员密码为空

> personal.unlockAccount(eth.accounts[1], "123456")

解锁完成后，我们正式开始转账交易，该交易由accounts[0]发起，由accounts[1]接收，交易金额为10ether输入命令如下：

>amount = web3.toWei(10,'ether')

>eth.sendTransaction({from:eth.accounts[0],to:eth.accounts[1],value:amount})

此时请同学们自行查看两个账户中的金额，应没有任何改变，我们输入命令查看交易执行状态。

> txpool.status

发现有一笔处于pending状态的交易，说明本次交易已经提交，但仍在处理中，只有当有矿工挖出新的区块后，该交易才会被写入区块链中，本次交易才会最终执行完成，输入命令开始挖矿，并隔一段时间停止。

>miner.start()

>miner.stop()

此时，请同学们再次查看交易执行状态和两个用户的账户余额信息，应已发生对应改变。

### Metamask钱包工具的配置使用（请截图保存在实验记录中）

Metamask是一款用于管理以太坊账户的钱包工具，主要作为浏览器扩展工具使用。

**3.1 Metamask安装配置**

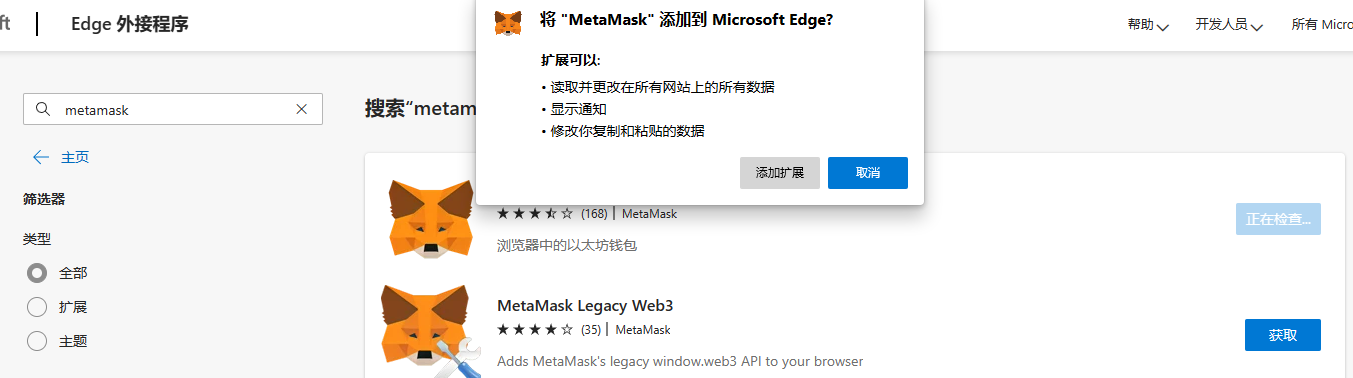
下面介绍一种Metamask的安装方式(以Edge浏览器为例)

1. 打开浏览器，选择设置-扩展-管理扩展-获取Microsoft Edge扩展

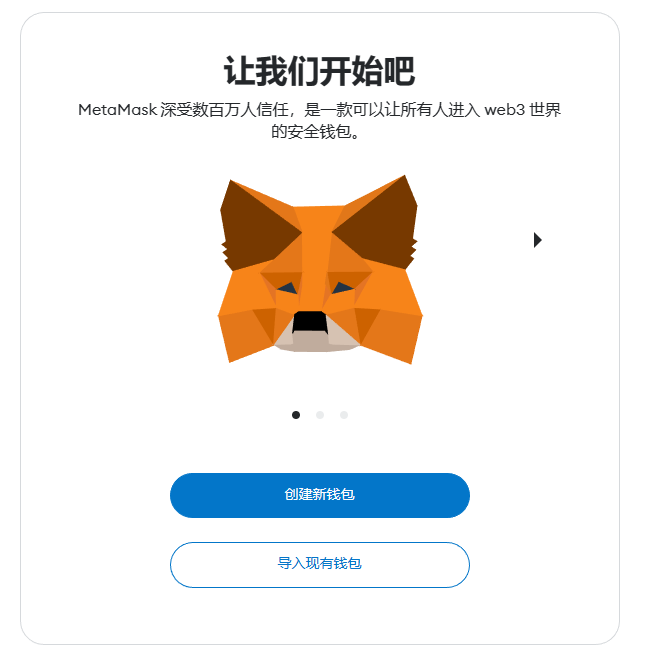




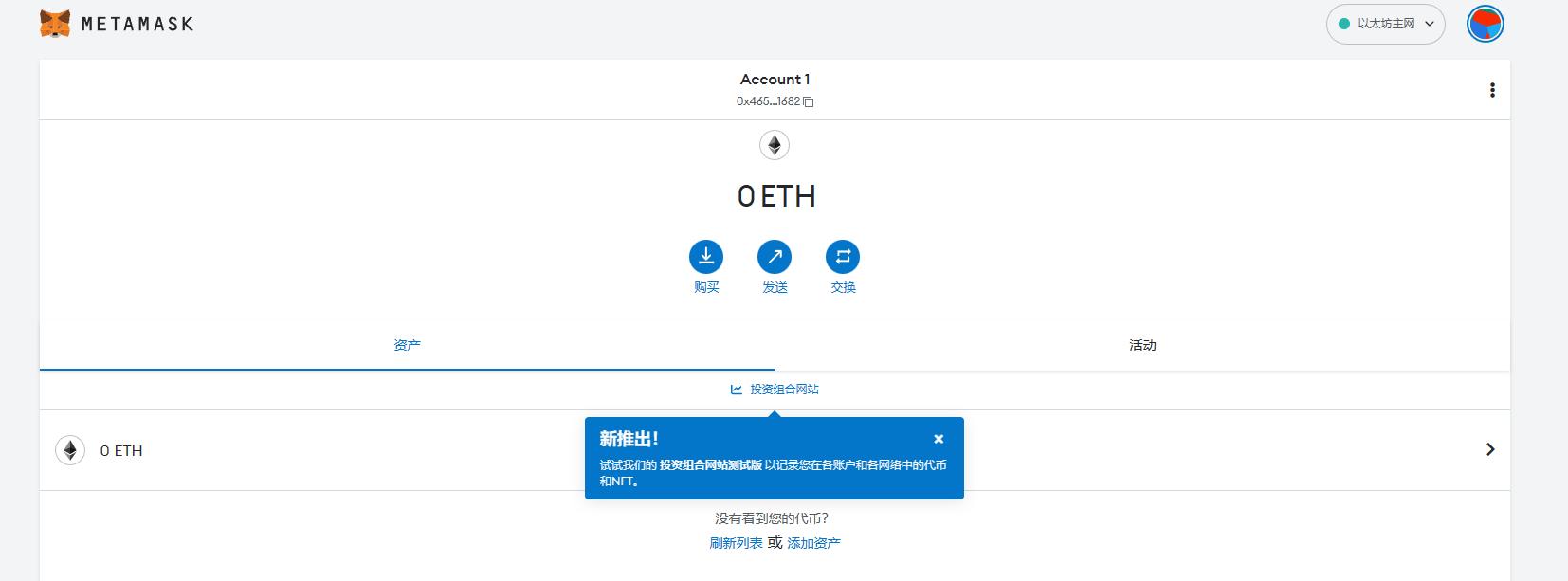
1. 搜索metamask并添加扩展



1. 添加完成后，依次选择创建新钱包-我同意-输入密码创建新钱包-账户助记词选不选都可以-



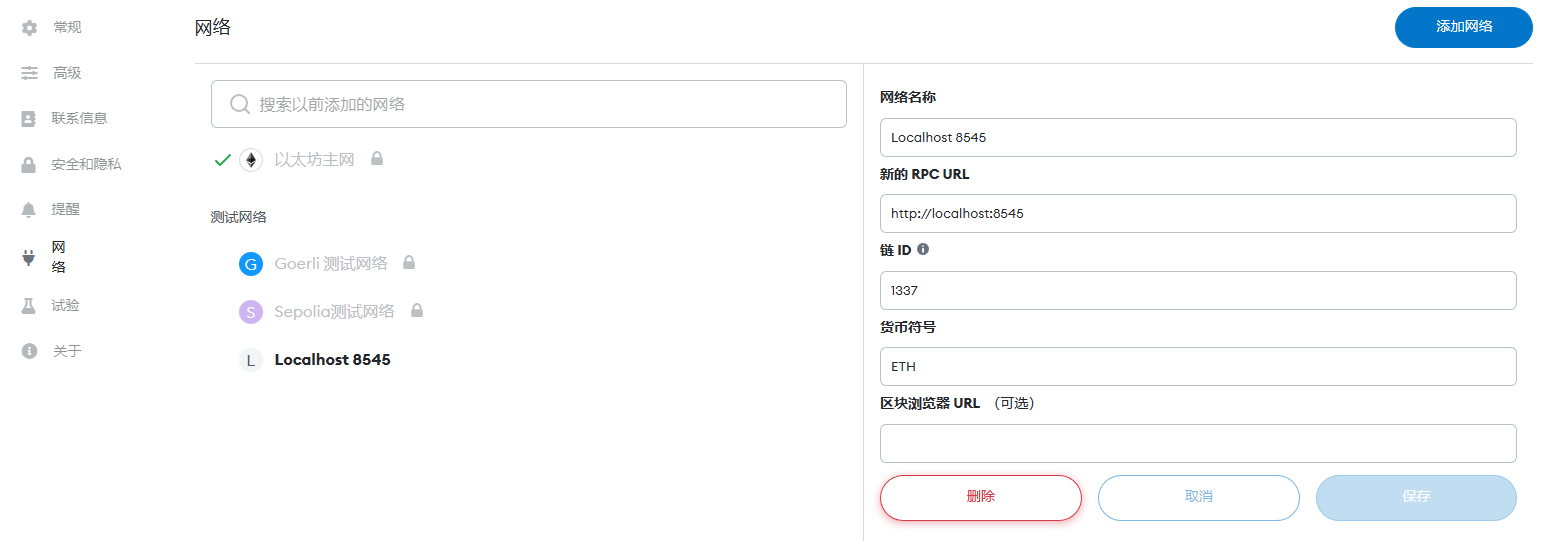
1. 钱包创建完成后，会自动进入到账户管理界面



下面我们需要配置自己的私有链网络信息，选择右上角-添加网络。



首先需将钱包自动生成的localhost 8545网络删除，防止其占用1337链ID。



删除后，选择右上角添加网络-手动添加网络，填写相应信息说明如下：

**网络名称：**随意命名即可

**RPC URL：**填写http://你的虚拟机ip:8481

**链ID:** 1337

**货币符号:** ETH

**区块链浏览器URL：**可不填

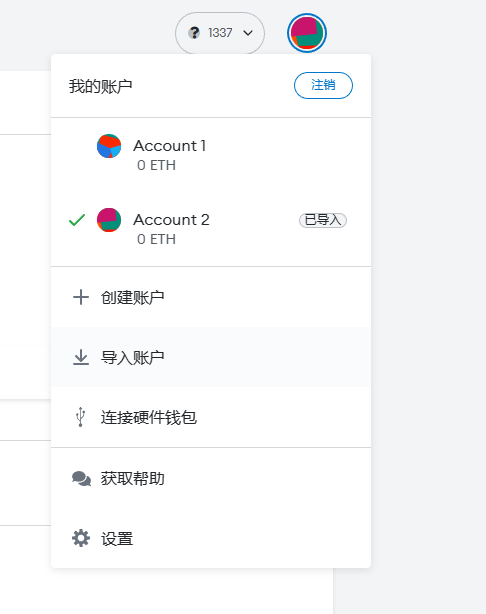
填写完成后，点击保存即可进入到自己的私有链网络中。

**3.2 账户导入**

接下来，我们尝试在metamask中导入在2.2.d步骤中，以密码110200的新建账户，我们需要将账户信息文件先从虚拟机中拷贝到主机中，文件地址为，

./EthDBSpace/keystore

其中应有两个文件，分别对应开发者账户和后续我们新建的账户，我们选择将文件名日期较靠后的文件复制到宿主机中，随后点击右上角头像选择导入账户，



选择json文件，并导入从虚拟机复制出来的账户信息文件，输入该账户的密码后即可完成导入。



导入完成后，查看到新建的账户中已经存在之前交易的10ETH，随后我们尝试用metamask向Account1进行转账。

选择发送—在我的账户之间转移—Account1—输入合适的数额—下一步—确认，回到钱包主页出现了一笔待处理的交易，但并没有完成，请同学们参考之前的实验步骤，在虚拟机的geth客户端进行相应的操作，让这笔交易完成，完成后请在metamask中分别查看Account1和Account2的账户余额，随后在虚拟机的geth客户端中分别查看管理者账户和新增账户的余额，分析交易产生的交易费的去向。

**作业上交内容与事项：**

1．实验报告文档：按照要求完成实验并将关键步骤实验结果进行截图记录。注意文档工整；

2．程序代码：如果有程序代码或其他相关材料，可汇总压缩所有文件并上传压缩包。注意文件命名；

**本期实验作业上交期限：**

请在实验设置的截止日期内提交实验报告，若逾期提交，成绩会被适当打折。

**本次作业上交内容：**

* 实验报告文档(.docx)

## 三、实验感受及记录

* 1. **实验感受（本次实验遇到的问题、主要收获等内容）**

通过本次实验，我对区块链技术的理解有了更深层次的认识，特别是在实际操作层面上得到了宝贵的经验。

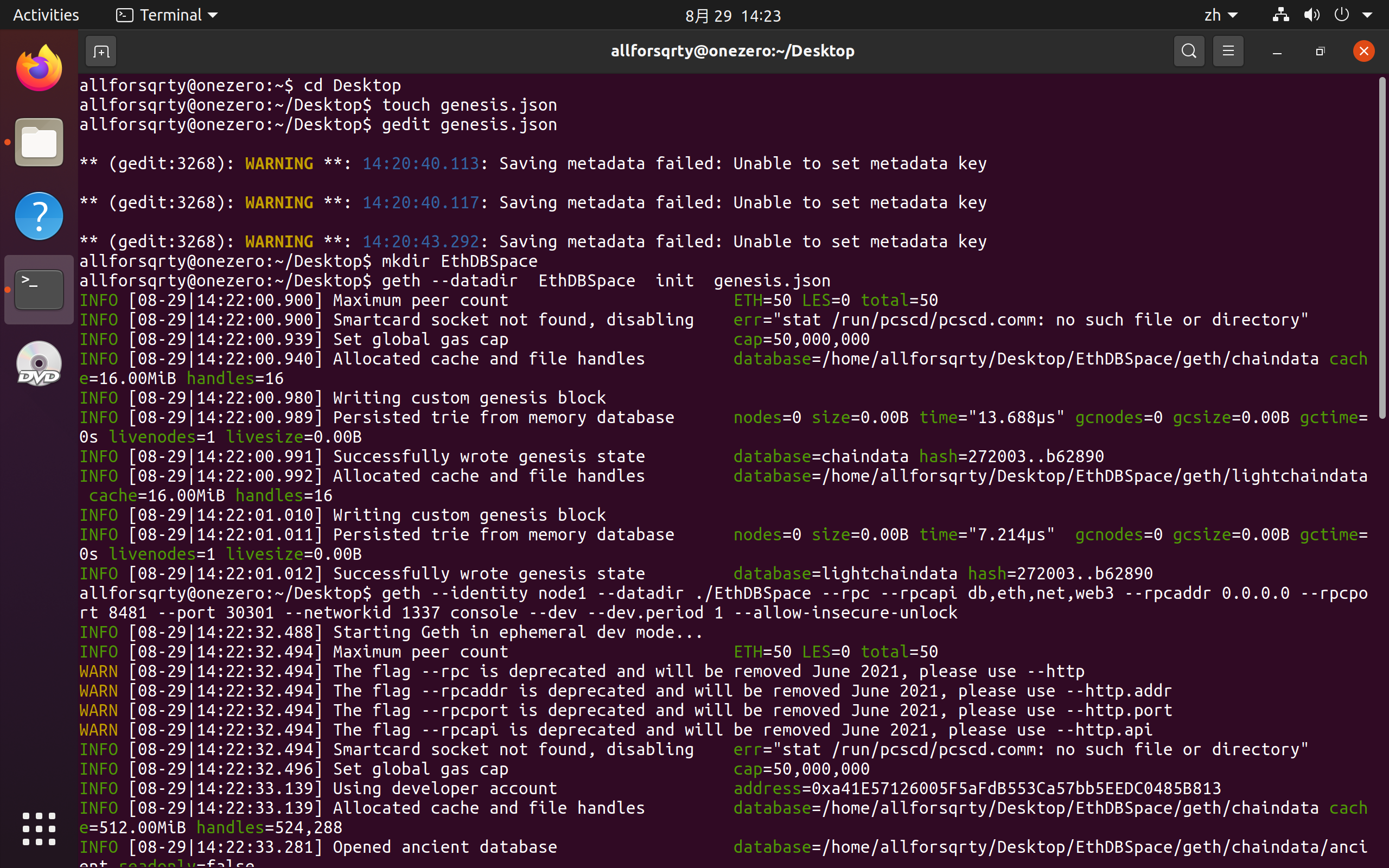
通过geth我初步掌握了如何在本地搭建一个以太坊节点。虽然之前在理论上了解过区块链节点的概念，但亲自动手操作让我对节点的同步过程、数据存储以及如何与以太坊网络进行交互有了更直观的感受。在安装过程中存在一些问题，比如节点同步时间长、资源占用较高等，但正是在解决这些问题的过程中，我对geth的工作原理有了更深入的理解。

在使用Metamask的过程中，我切实感受到了区块链技术在用户端的应用。Metamask作为一个浏览器插件，极大地方便了普通用户与以太坊网络的交互。通过它，我成功创建了一个以太坊钱包，并了解了如何管理私钥与公钥，如何进行简单的以太坊转账操作。这个过程让我对加密货币的安全性有了更深的认识，也明白了用户在使用区块链技术时需要具备的安全意识。

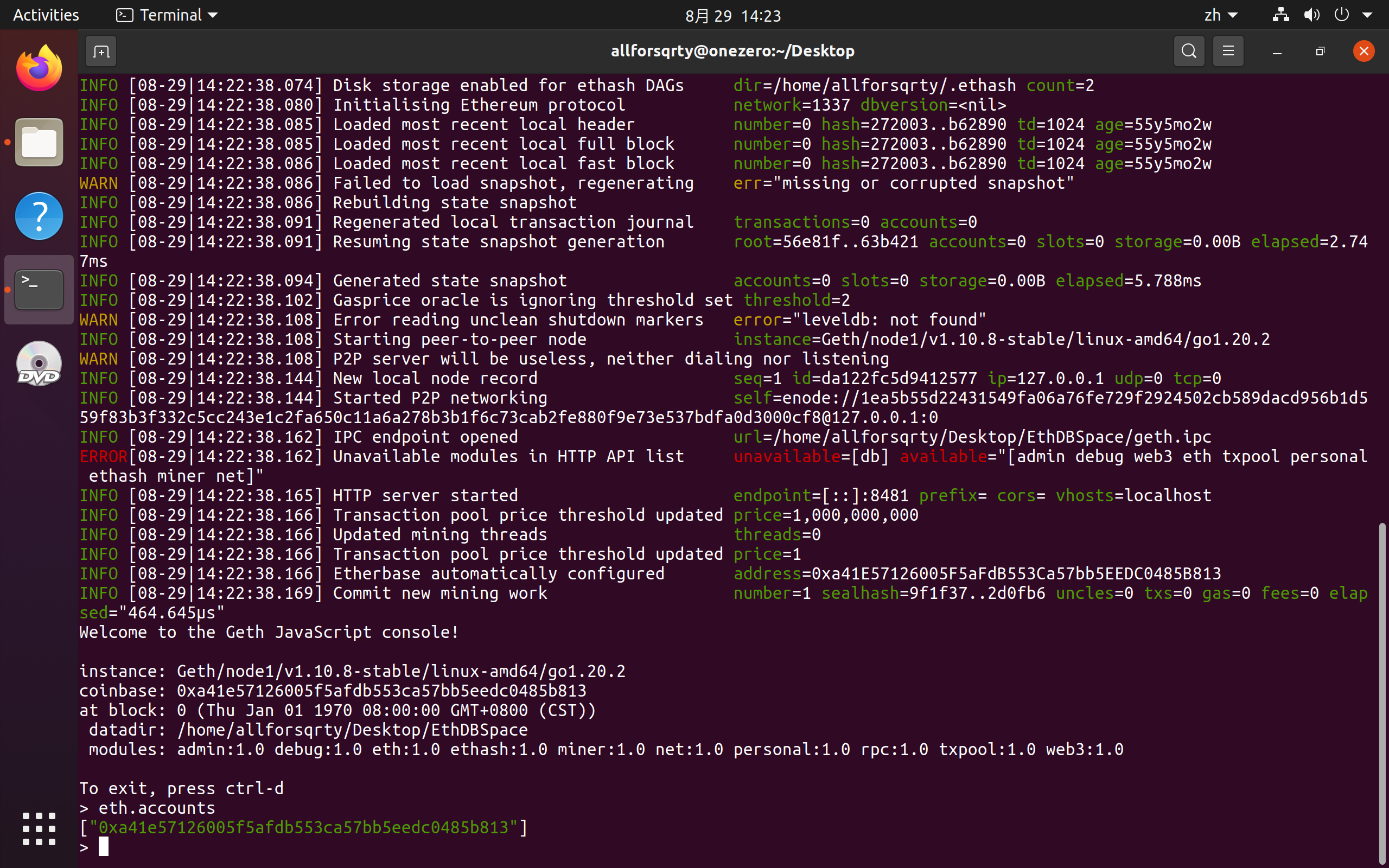
* 1. **实验记录（实验过程中关键步骤截图记录及文字描述）**

**1 Geth客户端的简单操作**

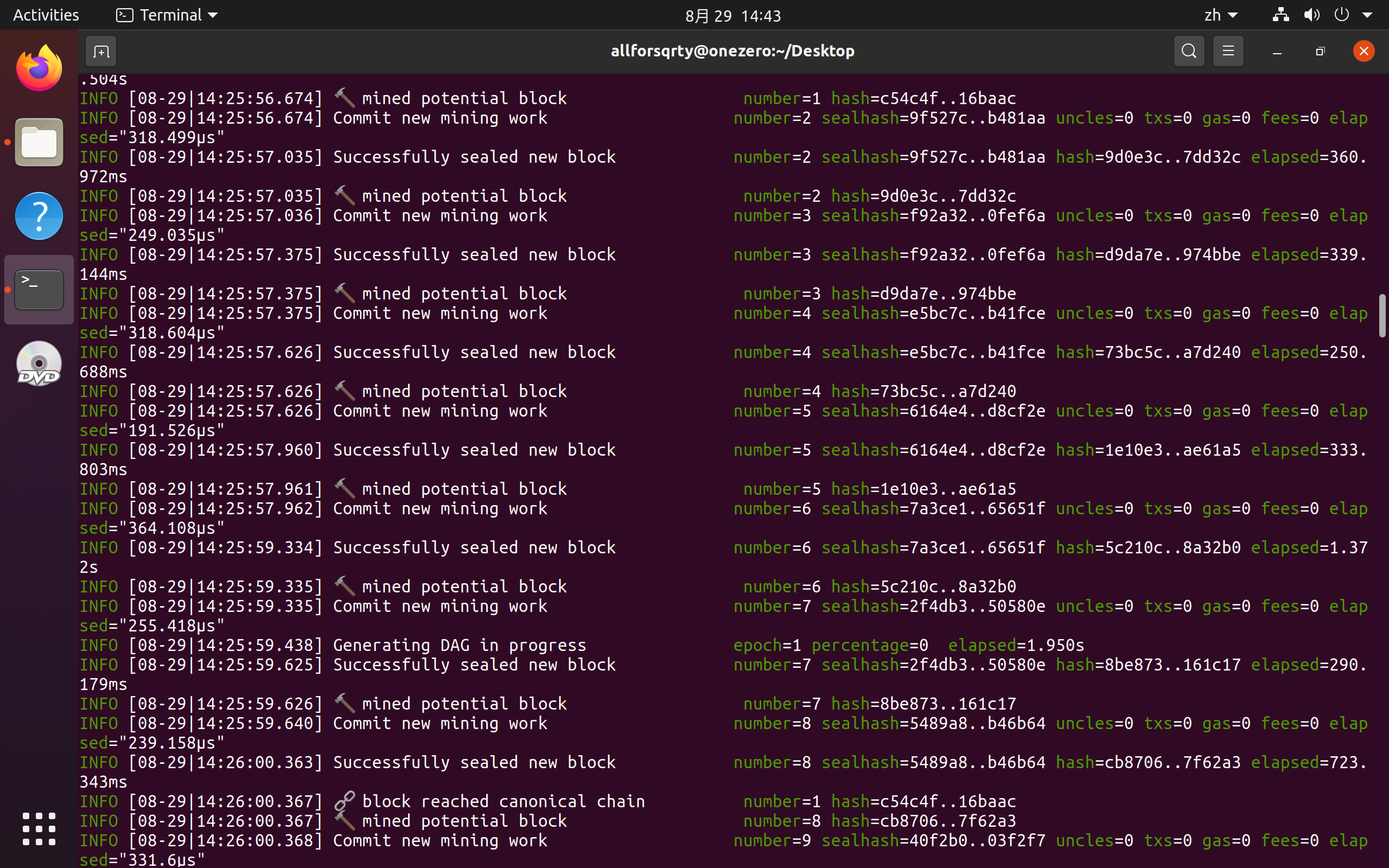
1. 首先构建以太坊网络，配置文件如文档中所示。

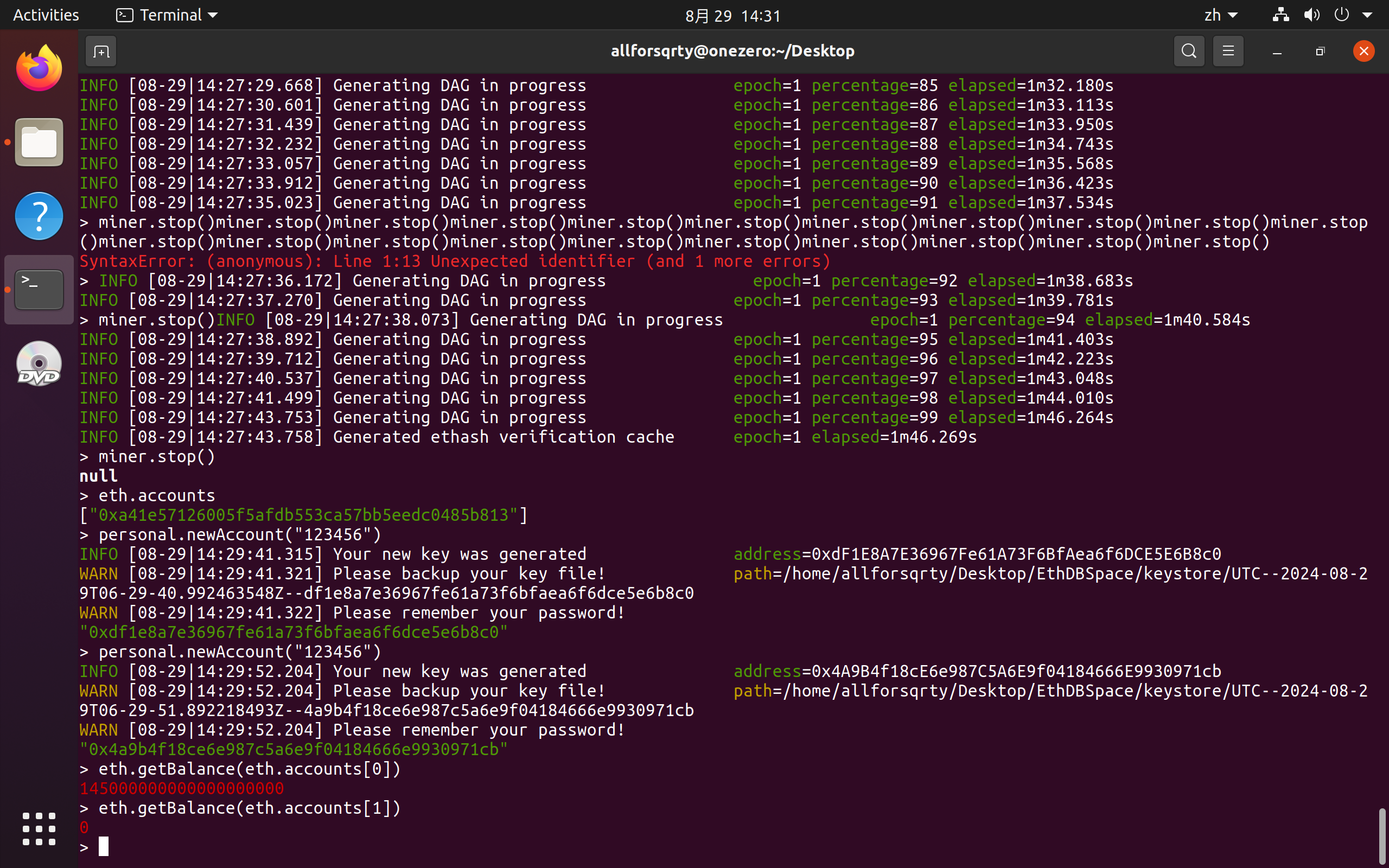


1. 之后启动geth客户端节点。查看账户发现现在仅有开发者账户。

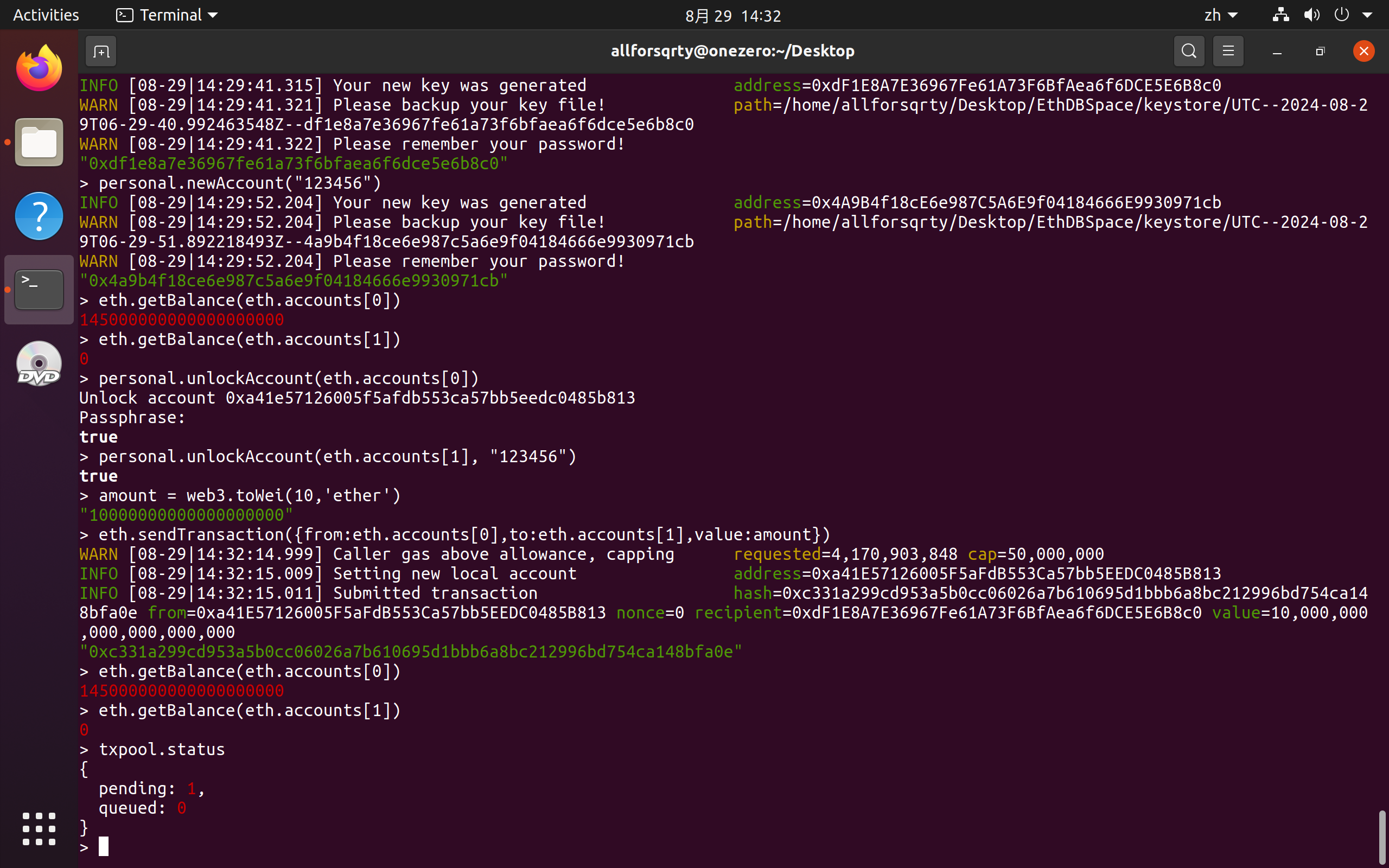


1. miner.start()开始挖矿，一段时间后，输入miner.stop()结束挖矿，并创建一个新账户。图中可看出，成功挖到矿后，原账户中存在资金。

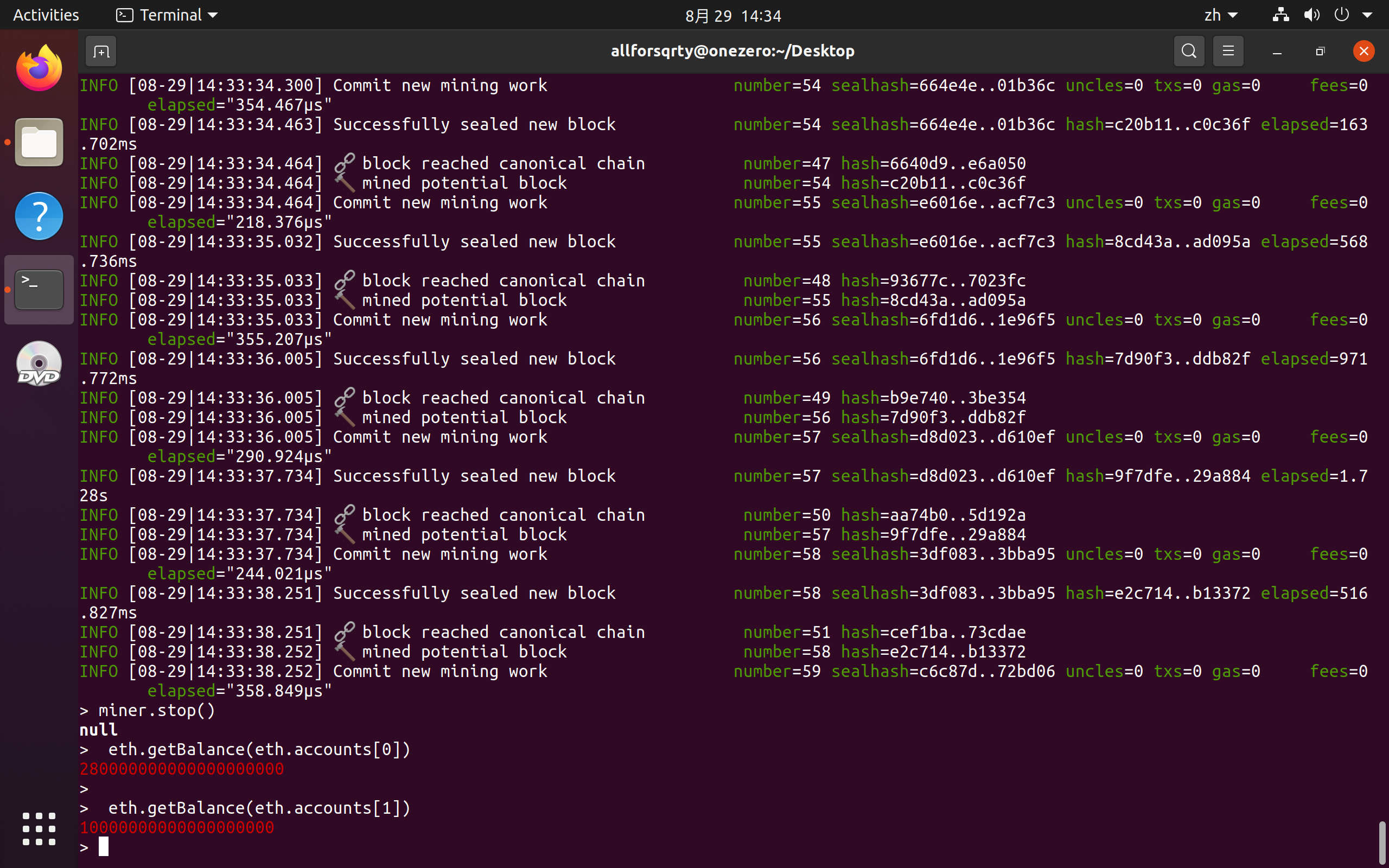




1. 解锁账户，并进行一笔交易，此时交易已经提交，但并未处理，两个账户中的资金保持不变。



1. 再次启动挖矿，区块被链接到区块链上后，交易完成，两个账户的资金都发生变化。



**2 Metamask钱包工具的配置使用**