第 3 章　区块链浏览器与区块链钱包

区块链技术的重要特点之一是具有数据不可篡改性。在基于区块链构建的公链应用（如数字货币、智能合约）中，良好的数据透明性使得经过区块链接收确认的所有数据变得公开可验证，这也是区块链技术可以被权威实体信任的根本原因。区块链浏览器作为区块链项目的关键基础设施，能够帮助大众在不需运行任何专用软件的情况下，对实时的区块链状态进行解析，获取其感兴趣的部分数据，也是学习区块链技术最为直观、便捷的工具。

本章实验的目标包括两方面：一方面，以比特币和以太坊的区块链浏览器为例，先介绍获取区块链数据的基本技巧，进而利用区块链浏览器解析并学习区块链账本层与合约层的构造，结合多个典型事务，加深读者对于多种区块链状态的认知与体会，最后学习批量获取区块链数据进行数据挖掘的相关技巧，该实验也是后续区块链实践的基本技能；另一方面，针对存储私钥的应用——区块链钱包，本章实验旨在让读者掌握主流数字货币系统生成密钥和地址、签发交易的基本方法，掌握区块链钱包的基本分类，了解并体验不同类型的区块链钱包，学有余力的读者可以组队展开进阶实验。

本章要求：掌握区块链浏览器的基本操作、功能、使用技巧（各类状态查询、简单 API 调用、数据可视化、钱包、测试链）；学会利用区块链浏览器解析并学习区块链账本层构造（地址、典型交易、交易费用、隔离见证、脚本构造等）；学会利用区块链浏览器解析并学习区块链合约层构造（合约状态、合约的相互调用、费用计算、ERC20 等）；尝试通过 vanitygen 工具用正则表达式生成比特币靓号地址，通过 bitcoin-core-testnet 工具体验私钥的冷存储，通过 bitaddress 工具体验脑钱包的工作原理；有能力的读者可以完成拓展实验，即批量获取并分析区块链元数据（API 调用、数据获取的使用、数据挖掘、部署开源区块链浏览器）。

3.1　区块链浏览器的基本操作

3.1.1　实验目的

（1）熟悉区块链浏览器的基本功能。

（2）掌握使用区块链浏览器进行基本查询操作的方法。

3.1.2 原理简介

区块链浏览器可以向外界提供区块链上的关键信息，包括但不限于：链状态、区块状态、交易状态、合约状态、账户状态，区块链浏览器还可能额外提供对于测试网络的支持（方便开发者进行测试应用的调试）、数据可视化服务（方便用户对区块链状态进行宏观认识）、钱包服务（方便用户管理数字资产）和开放 API（方便用户精确、批量地获取数据）。

一些主流且稳定的区块链浏览器包括：Blockstream、Blockchain、BTC、Blockcypher、Etherchain、
Etherscan。读者可以在网络上进行搜索（本教程提供的所有网络 URL 地址都由作者在教材编写时成功访问）。

3.1.3 实验环境

本实验在 PC 机上即可进行，操作系统不限。

3.1.4 实验步骤

本节以几种常用区块链浏览器为例，介绍区块链浏览器的基本功能。

1．浏览器反馈的几类区块链状态

（1）链状态

每条链的链状态以其链名作为唯一标识。例如，通过 blockchain.com 进行查询，其区块链浏览器的显示如图 3-1 所示。

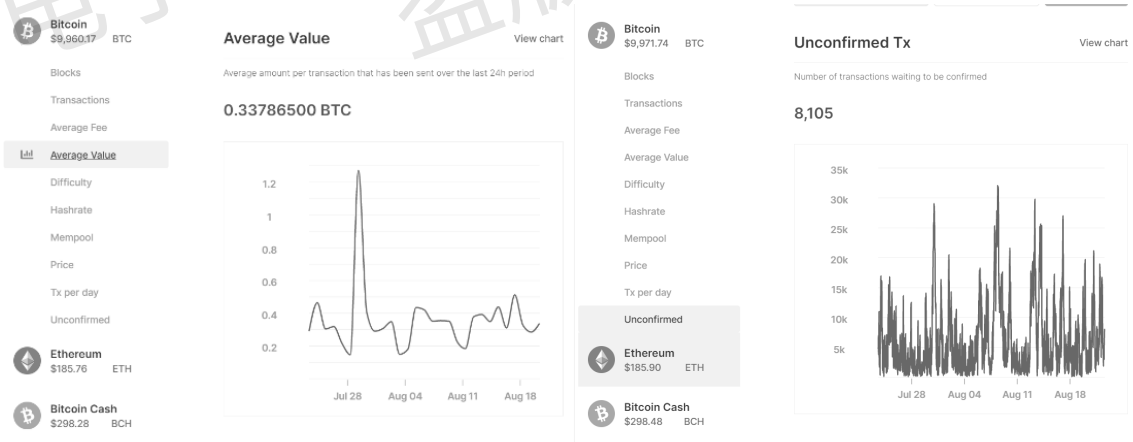

图 3-1　blockchain.com 提供的区块链浏览器主界面

可以访问 Bitcoin、Ethereum、Bitcoin Cash 三条链的状态，其主要特征包括：链的最新区块、最新确认交易池、平均交易费用、平均交易价值、实时挖矿难度、全网节点总算力、待确认交易池、代币价值、每日交易频率及积压的交易总数目。由此我们可以初步判断这条链的价

值、效率、安全性及交易的活跃度。

（2）区块状态

区块状态以区块地址和区块高度作为标识。例如，通过 blockstream.info 进行查询，以区块

```
000000000000000000136cf467d4d9ae8af79441d049d06b3e2ea03a83126ed1
```

为例，结果如图 3-2 所示。

图 3-2　使用 blockstream.info 查询区块状态

其主要特征包括：状态（有无分叉，确认深度），时间戳（并非一个精确的值，仅仅具有参考意义），实际数据大小，可见大小（一般为 Bitcoin 规定的 1 MB，节约的数据量由账本层隔离见证机制的施行带来，在 3.2 节着重解析），由隔离见证带来的额外数据（以 KWU 为单位，采用独特的换算标准），矿工节点版本号，区块的默克尔根，以及生成有效 PoW 所需的有效填充数 nonce。区块详情中包含了该区块容纳的所有交易信息，其中第一个交易固定为 Coinbase 类型，作用是将挖矿奖励支付给矿工指定的地址。

（3）交易状态

每个交易的交易状态以交易地址作为唯一标识。例如，通过 blockstream.info 进行查询，搜寻任何一个交易：

```
e75cc9e67a64d3974210da8480d3d80c0b5fb1a966b6451dd847754d4e82a5e1
```

其主要特征包括：确认状态、所在的区块信息（地址、高度、时间戳）、所付出的交易费用、交易的大小、节点版本号、锁定时间（用于定义该交易最早可入块的时间）、费用节约情况（如果激活隔离见证，则能够节省的费用）以及隐私情况（是否重用地址等）。进一步，通过单击交易的详情，我们可以观察交易的构造及其每个输入的赎回脚本 ScriptSig，以及每个输出的锁定脚本 ScriptPubkey 和输出的花费情况。

（4）账户状态

每个用户账户由其地址唯一标识。例如，通过 blockstream.info 进行查询，单击任意一个输入或输出中包含的地址字段，可以检索该地址相关的所有交易历史和地址的余额。例如，以地址

```
3NKtXY8ZpZe5XbE4YrjZogkid8hSBkDACw
```

为例，如图 3-3 所示。

图 3-3　使用 blockstream.info 查询账户状态

注意：对于比特币的账本交易，不建议通过地址重用的形式来管理账户，因为这样容易因为交易历史而暴露隐私。另外，由于地址上存储的资产在花销后，其对应的公钥也会随之泄露，一旦进入后量子密码时代，旧公钥密码算法的失效会直接导致用户的资产流失。目前建议的账户安全管理方法是：钱包记录并衍生一系列的用户地址，每个地址仅使用一次，钱包采用过滤器监听所有相关地址的交易并整理用户资产，避免上述问题的出现。

（5）合约状态

以太坊的合约状态以合约地址作为唯一标识。例如，可以访问以太坊的区块链浏览器 etherscan.io，选择其中的任意有效合约进行观察，观察中可以辨识的特征包括：合约名称、该合约的所有相关事务（初始的两笔事务分别用于创建合约、向合约地址付款，以启动合约）、源码、账户余额、创建者地址、编译版本、遵循的协议、状态变更历史，以及合约提供的接口

（API）等信息。例如，以

```
0x39e743fee400a5d9b36f1167b70c10e8f06440e5
```

为例，结果如图 3-4 所示。

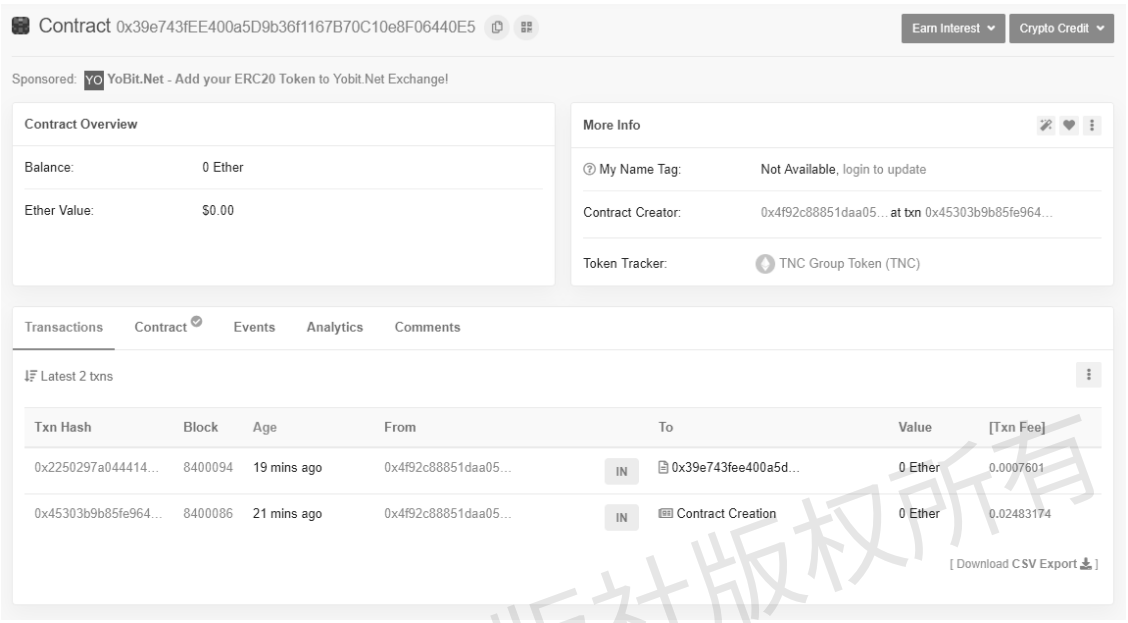

图 3-4　使用 etherscan.io 查询合约状态

以上状态皆可通过区块链浏览器的搜索栏，输入相应的标识进行查询，这也是区块链浏览器提供的最基本的功能。

2．区块链浏览器提供的 API 支持

（1）API 调用

下面以 blockstream 为例，其 API 调用的完整方法记录在 Github 上。

使用 API 的方法有两种：一种是使用 JS 包管理工具直接对该开源浏览器进行部署，另一种较为简单的方法是直接通过 URL 访问 API（前缀为 https://blockstream.info/api/）。我们建议使用命令行工具 CURL，这是一款利用 URL 语法在命令行方式下工作的开源文件传输工具，可以完成调用（采用浏览器直接访问亦可）。

我们以第一条 GET /tx/:txid/status 为例，尝试调用该接口。该接口的功能是根据交易的地址，返回交易的确认状态，包括：是否被确认（confirmed），被收纳的区块高度（block_height）、该区块的地址（block_hash）。

例如，查询地址为

```
6dcc37358d08b6adee18deb22f037326b5e659c2030189fbf774344c9fb3915
```

的交易确认状态，打开终端，输入以下命令：

```
curl https://blockstream.info/api/tx/6dcc37358d08b6adee18deb22f037326b5e659c2030189fbf7
```

```
74344c9fb39152/status
```

若所查询的交易的确认状态如图 3-5 所示，则所查询交易的确认状态以 JSON 的形式被返回，安装 Python 的读者可以使用“python -m json.tool”命令和管道对返回数据的格式进行修整，如图 3-6 所示。

```
C:\Users\vivid>curl https://blockstream.info/api/tx/6dcc37358d08b6adee18deb22f037326b5e659c2030189fbf774344c9fb39152/sta
tus
{"confirmed":true,"block_height":364292,"block_hash":"00000000000000003dd2fdbb484d6d9c349d644d8bbb3cbfa5e67f639a465fe",
"block_time":1436293147}
C:\Users\vivid>
```

图 3-5　所查询的交易的确认状态

```
C:\Users\vivid>curl https://blockstream.info/api/tx/6dcc37358d08b6adee18deb22f037326b5e659c2030189fbf774344c9fb39152/status|python -m json.tool
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   144  100   144    0     0    144      0  0:00:01 --:--:--  0:00:01   263
{
    "confirmed": true,
    "block_height": 364292,
    "block_hash": "00000000000000003dd2fdbb484d6d9c349d644d8bbb3cbfa5e67f639a465fe",
    "block_time": 1436293147
}
```

图 3-6　使用 Python 对返回数据的格式进行修整

（2）可视化

一些区块链浏览器对区块链的历史数据进行了挖掘，并提供了可视化的服务，方便用户对链状态的变化进行探究。下面以 BTC.com 浏览器为比特币提供的多项统计为例。

打开该浏览器，可以看到其对矿池份额、交易费用、脚本类型、难度变更等参数进行实时的统计，以矿池份额为例（如图 3-7 所示）。理论上，Nakamoto Consensus 能容限的恶意节点比例为 51%，在考虑自私挖矿后，该值可以降低至 25%，而现有最大矿池的算力占比已经接近这个值，矿池的扩张正在逐渐蚕食比特币的安全性。

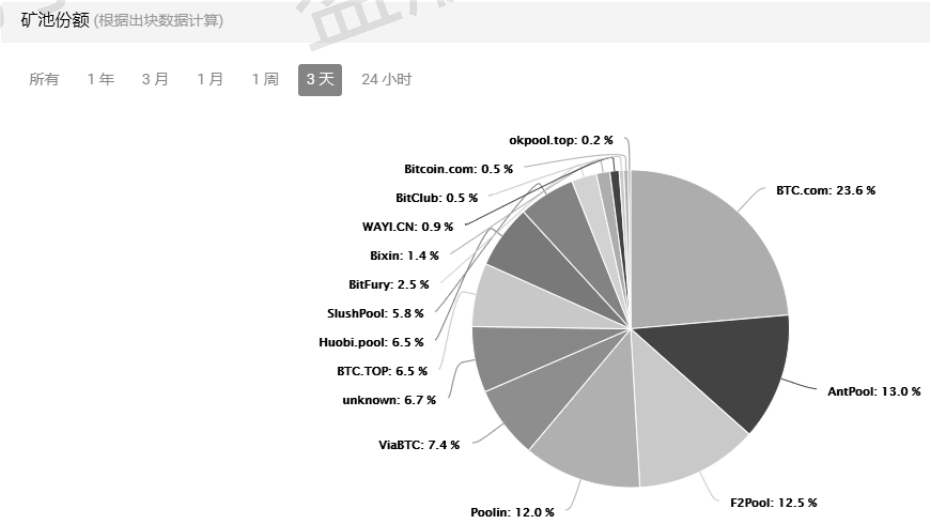

图 3-7　BTC.com 对矿池份额的实时统计

另外，去中心性的丧失也是令人担忧的。图 3-7 中非矿池的算力占比仅为 6.7%，整个比特币挖矿历史的该值为 37.4%。庞大的算力消耗与存储需求已使得个人节点不再适合作为全节

点维护比特币生态，更倾向于加入矿池，承担 Hash Puzzle 的运算外包业务。

（3）发布交易

一些区块链浏览器提供了网页端发布交易功能甚至数字货币钱包的功能。例如，blockstream 允许输入交易的 HEX 编码，浏览器可以代理广播该交易，如图 3-8 所示。

图 3-8　部分区块链浏览器可以代为广播 HEX 交易

（4）测试网络支持

考虑到代币的昂贵和未经测试的新技术部署风险，各区块链社区建立了测试网络，用于开发者对于区块链应用的测试。例如，比特币的测试网络 Testnet3 的特点是挖矿难度很低，代币没有经济价值，除此之外，其部署的技术一般先于比特币主网。

很多区块链浏览器提供了对于测试网络的浏览器支持，如 blockstream 也具备这个功能。

测试网络的代币可以通过从各类测试网络的 faucet 上输入自己的测试链地址来获取，如 coinfaucet.eu（**强烈建议**，在实践结束后，将该测试网络的代币发送回原 faucet 地址，便于更多人进行实验），如图 3-9 所示。

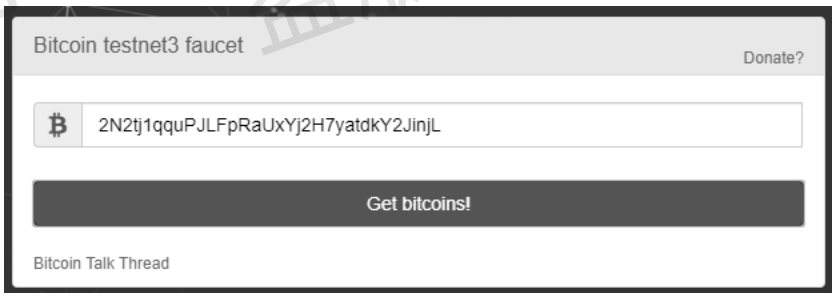

图 3-9　在 faucet 网站获取测试网络的代币

部分区块链浏览器还为测试网络提供了钱包的支持，我们可以在其上进行注册，并使用测试网络代币管理账户、发送交易、体验丰富的钱包功能、定制并观察自己的交易。

（5）读者练习

根据上述实验内容，请读者完成以下练习。请在区块链浏览器中查询区块

```
00000000000000003dd2fdbb484d6d9c349d644d8bbb3cbfa5e67f639a465fe
```

并对该区块进行分析。例如，该区块有何异常？造成该异常的原因是什么？这可能暗示了区块

链系统设计中的哪些问题？

观察浏览器对于比特币挖矿难度变化的可视化实时结果，如 https://btc.com/stats/diff；尝试回答：难度调整的间隔，难度变化的趋势和其带来的影响，以及推测平均算力的计算方法。

参考 blockstream API 的调用说明，调用 API 并回答：当前比特币待验证的交易数目为多少？数据量为多大？大概几个区块才能处理完这些交易？给出高度在 9991～10000 区块内包含的总交易数目。

3.1.5 实验报告

将上述实验步骤（5）的过程和问题解答写入实验报告。

3.2 利用区块链浏览器学习区块链账本层构造

3.2.1 实验目的

（1）熟悉区块链账本层的构造。

（2）掌握比特币脚本语言的原理。

（3）学会利用区块链浏览器对区块链账本层构造进行分析。

3.2.2 原理简介

比特币脚本是比特币使用的一种基于堆栈的执行语言。比特币将一系列带有特定功能的执行脚本 OPCODE 进行特定编码，通过合理组合后，按照“先入后出”的顺序执行，辅助完成交易的验证功能。

下面以常见的 P2PKH 为例，演示以其为核心脚本的账本交易验证过程。因为 P2PKH 是支付给收款者公钥的 Hash 值，所以交易的验证脚本需要分两步完成。首先，验证交易发布者所提供的公钥在 Hash 运算后是否匹配；然后，通过该公钥验证交易发布者提供的签名，若两者皆成功，则该输入的花销是有效的，如图 3-10 所示。

所有的比特币脚本都可以在比特币官方维基百科中进行查询。

3.2.3 实验环境

本实验在 PC 机上即可进行，操作系统不限。

3.2.4 实验步骤

本节实验利用区块链浏览器，让读者观察一些典型的账本交易构造方法，进而实现比特币脚本的编写。

首先，我们来熟悉一些典型的交易构造。

图 3-10　比特币 P2PKH 交易脚本的运行原理

1．Coinbase：创始块示例

创始块示例如下：

```
https://blockstream.info/block/00000000839a8e6886ab5951d76f411475428afc90947ee320161bbf18eb6048
```

作为一种特殊的交易，创始块固定作为每个区块的第一个交易，将挖矿奖励发送到矿工指定的地址或脚本，其输入脚本不需包含任何典型的赎回脚本，亦没有固定的格式，矿工通常使用的 OP_PUSHBYTES 可能嵌入具有一定含义的信息，也有可能在规定 nonce 尝试挖矿无果溢出后利用此种办法变相扩展 nonce 的范围，如图 3-11 所示。

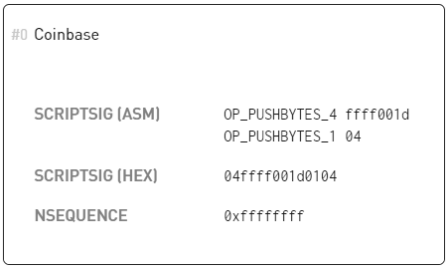

图 3-11 比特币 Coinbase 交易示例

2．P2PKH：示例

在 blockstream 中查询交易：

```
0de586d0c74780605c36c0f51dcd850d1772f41a92c549e3aa36f9e78e905284
```

在隔离见证激活前最为常用的一种支付脚本，我们已经在背景中介绍了其验证的过程，锁定脚本格式为

```
OP_DUP OP_HASH160 OP_PUSHBYTES_20 <hash> OP_EQUALVERIFY OP_CHECKSIG
```

其中推送的数据为 20 字节的公钥 Hash 值。赎回脚本格式为

```
OP_PUSHBYTES_72 <Sig> OP_PUSHBYTES_33 <Pubkey>
```

即先推送 71 或 72 字节的签名，再推送 33 字节的公钥。

类似的支付脚本有直接向公钥支付的 P2PK，但由于隐私的原因，被 P2PKH 所替代。

3．NullData（OP_RETURN）：示例

在 blockstream 中查询交易：

```
56a3de9926f1d1334b4f76ea9059d8357664d3ab72508b7c35efd9b511d82a01
```

作为一种特殊的输出脚本，NullData（OP_RETURN）可以在交易中嵌入最多 80 字节的任意数据。该输出不可花销，亦无法单独作为交易的输出，一般用作保证该部分数据的不可篡改性和时效性，作为存在性证明进行使用，辅助搭建更复杂的去中心化应用，如图 3-12 所示。

这里可以回顾账本层交易费用的设定。其费用与交易的容量成正比，也就意味着在 Bitcoin 这样一个以数字货币功能作为核心的生态内锚定数据是异常昂贵的，并且会影响其他正常交易的入块，这也是该脚本设置数据限定的原因。

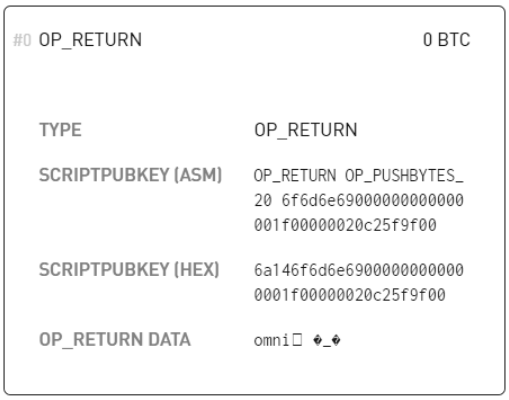

图 3-12 比特币 OP_RETURN 脚本示例

例如，示例交易为了锚定 20 字节的数据花费了 0.000446 BTC 的费用，价值 5 美金左右。

4．P2SH：示例

在 blockstream 中查询交易：

```
d3adb18d5e118bb856fbea4b1af936602454b44a98fc6c823aedc858b491fc13
```

比特币的脚本通过组合可以完成更复杂的逻辑，构造一些简单的合约，而 P2SH 脚本是实现这类功能最安全的方法。如图 3-13 所示，P2SH 的锁定脚本构造十分简单，仅包含数据段为赎回脚本的 Hash 值。

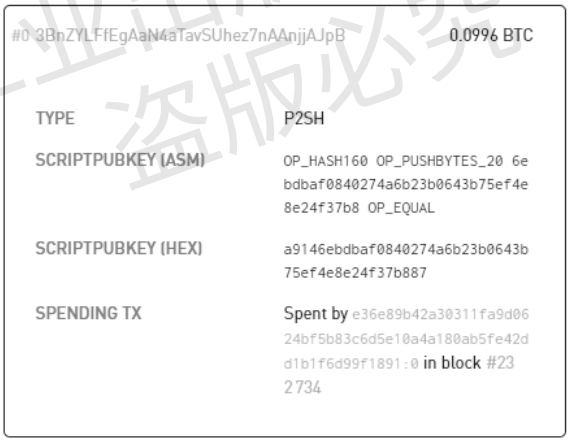

图 3-13 比特币 P2SH 交易脚本示例

但要合法地花费该输出，交易发布者不仅需要在下个交易的输入中嵌入完整的赎回脚本，还需要提供解锁赎回脚本的相应数据。例如，对于花费如下输出

```
d3adb18d5e118bb856fbea4b1af936602454b44a98fc6c823aedc858b491fc13
```

最终被交易，其赎回脚本为：

```
OP_PUSHNUM_2
OP_PUSHBYTES_65
```

```
        04f3d35132084eb1b99b6506178c20adb42d26296012e452e392689bdb6553db33ba24b900000892805de16
        46821c7b0fb50b3d879c26e2b493b7041e6215356a0
        OP_PUSHBYTES_65
        04ab4ecc9e8ea2da0562af25bcaede00c4d5a00db60edc17672376decf0a35a34fdc9f1ffad1fb74fd7b1b1
        98b9231c25df88e0769bec49975649b4b3f40adafb0
        OP_PUSHBYTES_65
        04f7149f270717c00f6cc09b9ce3c22791c4aab1af40a5107aacca85b6f644cc0d84459e308f998d801b8d9
        d355f8ec33b0e41866841e2870754cf667a9821703d
        OP_PUSHNUM_3
        OP_CHECKMULTISIG
```

上面定义了一个 2/3 门限交易，根据脚本 OP_CHECKMULTISIG 的定义，其脚本中包含 3 个公钥，而花费该笔输出需要提供两个不同公钥的合法签名，OP_CHECKMULTISIG 会利用所有公钥对输入的签名进行验证，若满足条件，则输出为真。

解锁脚本先推送解锁赎回脚本所需的数据，再使用 PUSHBYTES 脚本推送赎回脚本的 HEX 编码。

5．读者练习

请读者完成以下练习。观察某 P2SH 交易的赎回脚本：

```
        OP_3DUP OP_ADD OP_PUSHNUM_9 OP_EQUALVERIFY OP_ADD OP_PUSHNUM_7 OP_EQUALVERIFY OP_ADD
OP_PUSHNUM_8 OP_EQUALVERIFY OP_PUSHNUM_1
```

说明该脚本规定的解锁条件和运行机理，并拟写其解锁脚本。

3.2.5 实验报告

将上述实验步骤 5 的过程和问题解答写进实验报告中。

3.3 利用区块链浏览器解析并学习以太坊合约层构造

3.3.1 实验目的

（1）了解以太坊合约层的构造。

（2）学会利用区块链浏览器对以太坊合约层构造进行分析。

3.3.2 实验环境

本实验在 PC 机上即可进行，操作系统不限。

3.3.3 实验步骤

本节实验利用以太坊区块链浏览器 Etherscan 学习智能合约的基本构造，高阶技巧将在之后的合约实践课程中讲解。

1．合约层部署

单击任意区块的详情，如通过 etherscan.io 访问区块 8413441，可以观察到合约层部署带来的一些不同。

最显著的一点，区块的大小不再有固定上限，而是由事务费用上限 Gaslimit 决定的，事务费用则直接与矿工所执行合约的总复杂度相关联。对于每笔触发合约状态变动的事务，其复杂度由所执行程序的指令加权每个指令的复杂度求和得到，节点在发布事务前会附加相应的 Gas 并指定单位复杂度愿意支付的事务费用，矿工在执行该事务的过程中依次扣除执行费用。如果附加的 Gas 因不足而被耗尽，则该次执行不会造成区块状态变动，且不会退回所消耗的费用。

可以理解，由于智能合约的编程语言是图灵完备的，事务验证所消耗的算力不可被忽略，为了避免庞大的计算开销和恶意合约的影响，以太坊制定了以上经济模型，限定每个区块能处理的事务难度，这一举措也降低了生态丧失去中心性的风险。

2．探究合约和其触发事务的状态

下面以著名的 ERC 代币合约为例（如 ERC20、ERC621、ERC721）。以太坊考虑到每个分布式应用需要形成自己的生态，甚至发行自己的代币，所以允许应用以代币合约的形式发行自己的代币，并与其他种类代币进行价值流通。定义此类功能需要遵循 ERC 合约规范，并实现其规定的接口（以下为部分示例接口）。

- TotalSupply：代币发行总量。
- BalanceOf(address_owner) constant returns (uint256 balance)：查询余额。
- transfer(address_to, uint256 _value) returns (bool success)：发送相应代币数目到钱包地址。

应用开发者在此基础上再实现更复杂的合约功能。下面使用 Etherscan 查询以下合约：

```
0x06012c8cf97bead5deae237070f9587f8e7a266d
```

在其合约首页上，可以看到其代币的价值和合约定义的所有方法，以方便用户的学习、检测与调用，用户可以发布事务，通过自己的账户或驱动其他合约与该合约交互，如图 3-14 所示。

“Events”栏中显示该合约最近的状态变动、触发改动的事务地址、事务所调用的具体方法，如图 3-15 所示。

单击具体的事务地址，可以获得关于该次合约状态变动更加详细的信息，如图 3-16 所示。

Etherscan 同样开放了 API 接口提供对于以太坊的状态获取，但使用 API 需要申请相应的密钥，并受到每天 100 次访问的限制。另外，Etherscan 收集了一系列有助于合约开发和学习的在线工具，方便智能合约的开发学习和漏洞检测，有兴趣的读者可以自行查询。

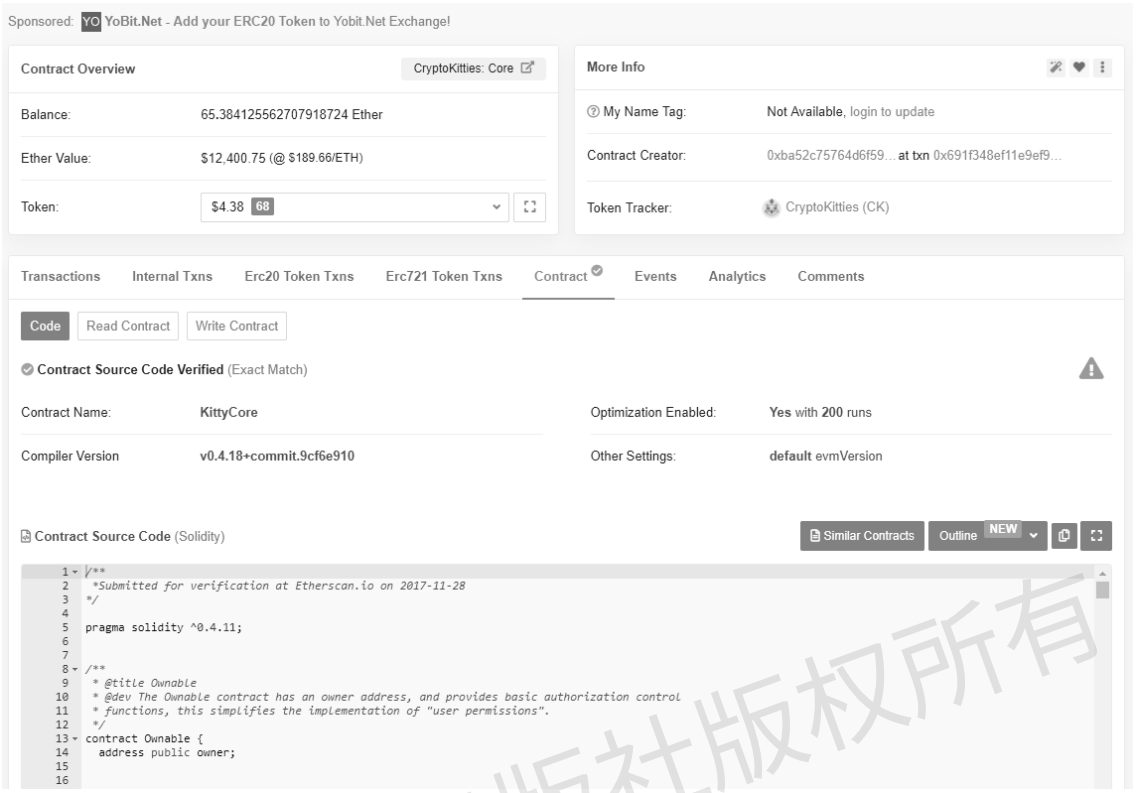

图 3-14　可以看到代币的价值和合约定义的所有方法

图 3-15　合约最近的状态变动

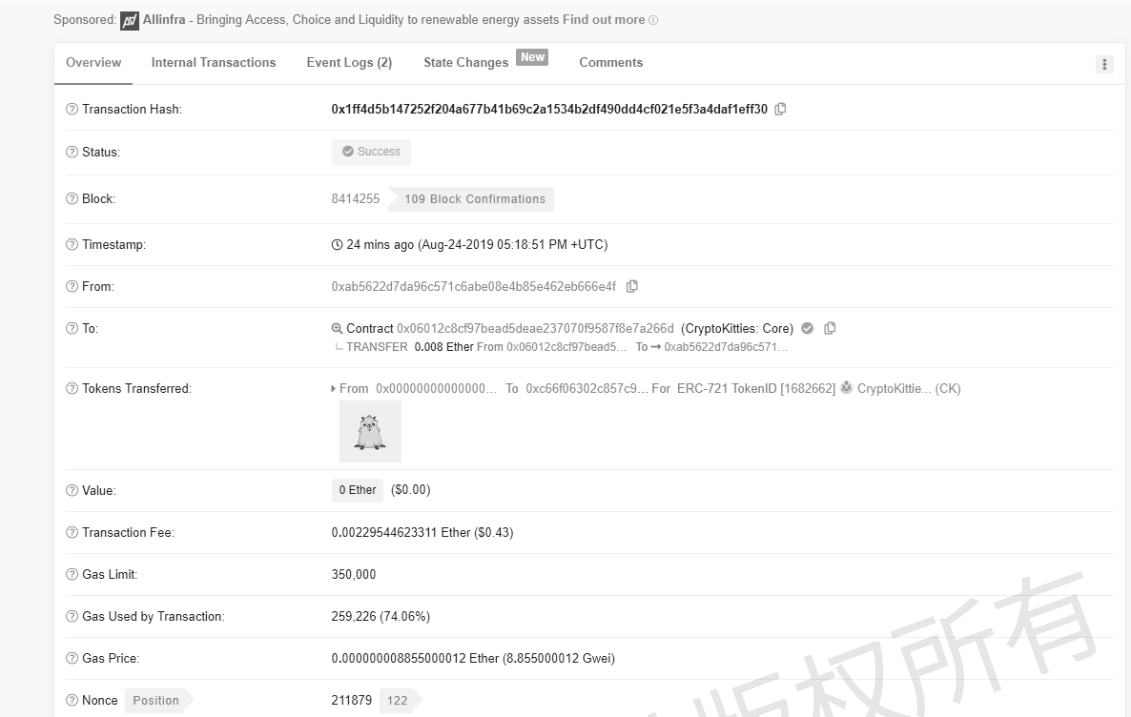

图 3-16　合约状态变动更加详细的信息

3.3.4　实验报告

在实验报告中总结主要实验步骤，并写出心得体会。

【思考题】

结合上述实验，总结以太坊合约层的构造方法。

3.4　体验区块链钱包原理

3.4.1　实验目的

（1）掌握区块链钱包的概念及分类。

（2）体验比特币靓号生成、冷钱包和脑钱包的制作过程，感受比特币钱包的奥妙所在。

3.4.2　原理简介

与普通钱包类似，作为数字货币，某些区块链也有“钱包”。只不过，区块链钱包中放的并不是现金，而是用户地址的私钥。换言之，区块链钱包和普通钱包都用来存放相应的货币系统中个人用户最重要、最害怕丢失的物件。

区块链钱包常按照下面几种方法分类：按照节点数据是否存储完整，可分为全节点钱包（完整存储区块链所有交易数据）和轻节点钱包（只保存了区块链钱包的基本功能）；按照区块

链钱包是否联网，可分为冷钱包（私钥在本地存储，不联网）和热钱包（联网）；按用户是否自行持有私钥，可分为中心化钱包（第三方机构代管用户私钥）和去中心化钱包（用户自行持有钱包的私钥）；按是否支持多种币种，可分为单币种钱包、多币种钱包、全币种钱包。

3.4.3 实验环境

本实验在 PC 机上即可进行，操作系统不限。

3.4.4 实验步骤

1．体验比特币靓号地址

本实验提供了工具包“vanitygen-0.22-win.zip”，解压缩即可使用。读者也可以自行搜索。

（1）打开 CMD，直接将 vanitygen.exe 的图标拖进 CMD 窗口中打开。

（2）输入一个空格和“1234”，这样就得到了一个以“1234”为开头的比特币地址，如图 3-17 所示。第一行是难度，第二行是限定筛选条件，第三行是碰到的地址，第四行就是私钥。

```
C:\Users\dell>D:\vanitygen-0.22-win\vanitygen.exe 1234
Difficulty: 78508
Pattern: 1234
Address: 12345iVbGMto4qm9aWBMsmRvWsavyNyW95
Privkey: 5Jo3VqjMCP7RUjETLzm3V8rKRn492GHLKTqgFMafu4pvXrpVEXb
```

图 3-17　以“1234”为开头的比特币地址

（3）随着指定字串的加长，难度是呈指数增长的，如果我们指定的是字母，那么忽略大小写可以降低难度，忽略大小写用参数“-i”，如图 3-18 所示。

```
C:\WINDOWS\system32>D:\vanitygen-0.22-win\vanitygen.exe 1zmzm
Difficulty: 264104224
Pattern: 1zmzm
Address: 1zmzmqC5221qw4BxAtMnCHgwHPHxanSCm
Privkey: 5Kd1WkqJQa5uP27JSA7q7cwqNa4w2EcxRHHz1HCMhPcKYNwG8dQ

C:\WINDOWS\system32>D:\vanitygen-0.22-win\vanitygen.exe -i 1zmzm
Difficulty: 16506514
Pattern: 1zmzm
Address: 1ZmZMB4DCNZjRgLqbc444WS8ahmSwgaE6
Privkey: 5K68sgARfjnmgi53vYpqkbWBbV78Np4hqLqTryJmiAmQhPGakTn
```

图 3-18　添加相关参数以降低难度

（4）请读者完成以下练习：

查阅相关资料，使用正则表达式，尝试生成满足以下条件的地址：

- 包含“abcd”的地址。
- 以 44 开头且以 99 结尾的地址。
- 以 5 个数字结尾的地址。
- 以 2 个数字再接“yyy”结尾的地址。

尝试碰撞刚才生成过的自己的某个地址，尝试逐渐减少输入的地址长度，体验并分析碰撞

难度。

2．使用 Bitcoin Core 进行私钥的冷存储

本实验提供安装包 bitcoin-0.20.1-win64-setup.exe，读者也可到官网自行下载其他版本。

（1）双击安装包，即可进行安装，除安装路径外，一直单击“Next”按钮，即可完成安装。安装完毕，暂时不要打开 Bitcoin Core 客户端。

（2）找到安装路径中的 bitcoin-qt.exe，右击，然后生成桌面快捷方式。

（3）在桌面找到 bitcoin-qt.exe 的快捷方式并单击右键，在弹出的快捷菜单中选择“属性”，在“快捷方式”中“目标”栏的“.exe”后输入一个空格和“-testnet”，单击“确认”按钮退出。

（4）完成上述所有准备后，通过桌面快捷方式进入 Bitcoin Core 的 Testnet 版客户端。初次进入客户端时，请设置好配置文件（包括区块）安装的路径并记下。

提示：区块自动同步非常缓慢，不必担心占用计算机空间，实验结束后将客户端卸载，并将上述路径中所有的文件都删除即可。

（5）在上述路径的 testnet3\wallets 中存有 wallet.dat 文件。断开网络，删除 wallet.dat，重新启动 Bitcoin Core 客户端，会发现 wallets 文件夹下重新生成了一个 wallet.dat 文件。

（6）单击“设置”中的“加密钱包”，输入密码，将密码和 wallet.dat 文件保存。这样就制作完成了一个冷钱包。

3．体验冷钱包地址的生成

（1）访问 bitaddress.org，等待网页跳转完毕，在链接的最后输入“?testnet=true”，回车后进行访问。随意滑动鼠标，直到显示生成地址的进度为 100%。

（2）将网页 bitaddress.org（测试网络）另存为网页文件，断开网络，在本地再访问一次。单击网页菜单的“Brain Wallet”，选中“Enter Passphrase”并输入“View”，这时可以用自己随机想到的一句话或几个单词。此过程重复一遍，就可以生成两对地址和私钥。

（3）单击网页菜单的“Vanity Wallet”，复制在步骤（2）中生成的两个私钥（不用复制地址）到网页的两个输入框；然后选择“Multiply”，单击生成按钮“Calculate Vanity Wallet”，再单击“Show”下的“View”按钮，生成一对新的地址和私钥。

（4）使用浏览器访问 faucet 网站，输入在步骤（3）中生成的地址，获取小额的测试代币，记录下交易 ID。

如果领取成功，那么网页将显示交易信息，如图 3-19 所示。

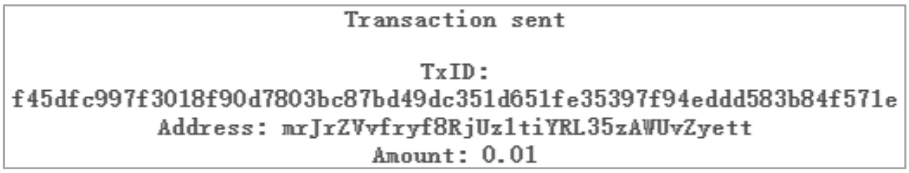

图 3-19　测试代币领取成功

3.4.5 实验报告

将实验过程和实验结果总结到实验报告中，并在实验报告中回答下述思考题。

【思考题】

成功领取测试代币后，这是否是一笔 UTXO？如果想要花费它，需要提供哪些信息？请具体列举，寻找合适的区块链浏览器，查询此笔交易并记录。

3.5 拓展实验：批量获取并分析区块链元数据

以下实验为拓展实验，感兴趣的读者可以尝试。

1．拓展实验 1：数据批量获取与挖掘

调用公开的 API 或其他方式获取 2020 年 8 月的所有区块数据，对所有交易发布者所使用版本号 version 的情况进行统计分析，并作统计图，对 locktime 字段的使用比例进行统计分析，过滤出所有 locktime 不为 0 的交易，并将该交易数据集写入一个文件，如图 3-20 所示。

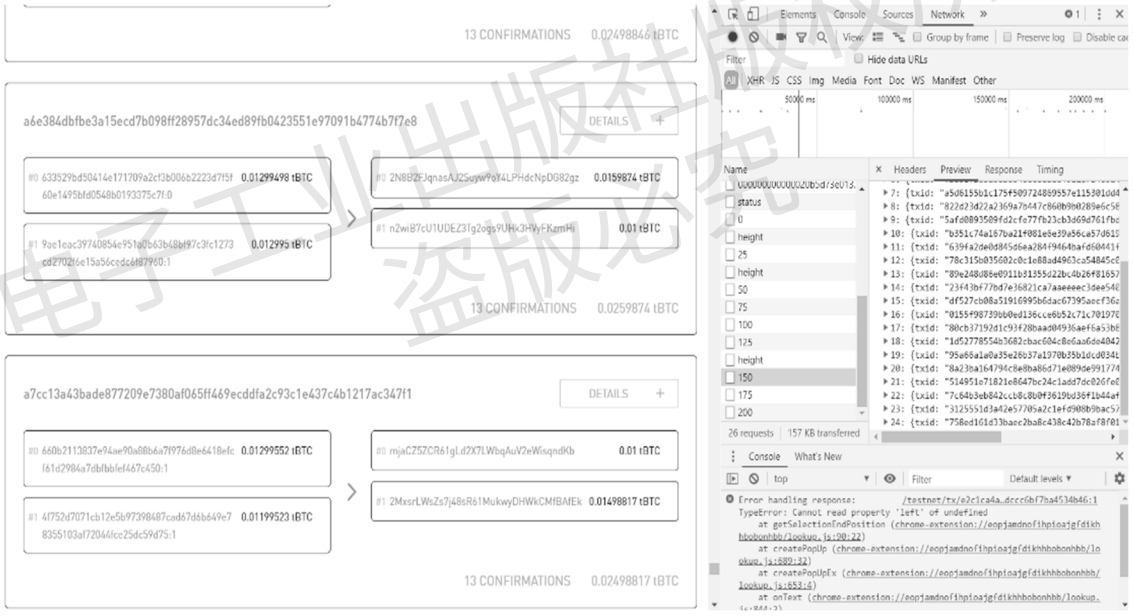

图 3-20　批量获取区块链数据

提示：可以利用程序模仿用户访问行为，根据 URL 呈现的规律，对区块链浏览器返回数据进行批量过滤与提取，一般在 API 调用受限时才会考虑使用。

2．拓展实验 2

利用开源项目，部署自己的区块链浏览器，并尝试调研区块链浏览器的构造方法。读者可以自行搜索区块链浏览器开源项目。

3.6 本章实验报告模板

读者在做本章实验时应及时记录实验结果，建议撰写实验报告，对实验进行总结和思考。本章实验报告模板如下。

<table>
<tr><th>类型</th><th colspan="2">实验报告内容</th></tr>
<tr><td rowspan="13">问答题</td><td colspan="2">1．简要介绍区块链浏览器的定义和主要功能。</td></tr>
<tr><td>定义</td><td></td></tr>
<tr><td>主要功能</td><td></td></tr>
<tr><td colspan="2">2．简述比特币 P2PKH 交易所涉及的脚本种类和交易验证时脚本语言的执行过程。</td></tr>
<tr><td>脚本种类</td><td></td></tr>
<tr><td>执行过程</td><td></td></tr>
<tr><td colspan="2">3．简述比特币 P2SH 交易所涉及的脚本种类和交易验证时脚本语言的执行过程。</td></tr>
<tr><td>脚本种类</td><td></td></tr>
<tr><td>执行过程</td><td></td></tr>
<tr><td colspan="2">4．总结根据赎回脚本写出解锁脚本的方法。</td></tr>
<tr><td colspan="2">5．比特币靓号地址生成的原理是什么？难度与什么有关？</td></tr>
<tr><td>原理</td><td></td></tr>
<tr><td>难度与什么有关</td><td></td></tr>
<tr><td colspan="2">6．简述 Bitcoin Core 属于哪一类钱包。</td></tr>
</table>

<table>
<tr><td rowspan="11">实验过程记录</td><td colspan="2">1．区块链浏览器的基本操作。</td></tr>
<tr><td colspan="2">（1）在区块链浏览器中查询区块
000000000000000003dd2fdbb484d6d9c349d644d8bbb3cbfa5e67f639a465fe
并对该区块进行分析，该区块有何异常，造成该异常的原因是什么？这可能暗示了区块链系统设计中的哪些问题？</td></tr>
<tr><td>结果截图</td><td></td></tr>
<tr><td>有何异常</td><td></td></tr>
<tr><td>造成该异常
的原因</td><td></td></tr>
<tr><td>暗示了哪些问题</td><td></td></tr>
<tr><td colspan="2">（2）观察浏览器对于比特币挖矿难度变化的可视化实时结果（可参考 https://btc.com/stats/diff），尝试回答：难度调整的间隔，难度变化的趋势和带来的影响，推测平均算力的计算方法。</td></tr>
<tr><td>结果截图</td><td></td></tr>
<tr><td>难度调整的间隔</td><td></td></tr>
<tr><td>难度变化的趋势
和带来的影响</td><td></td></tr>
<tr><td>推测平均算力
的计算方法</td><td></td></tr>
</table>

<table>
<tr><td rowspan="8">实验过程记录</td><td colspan="2">（3）参考 blockstream API 的调用说明，调用 API，回答：当前比特币待验证的交易数目为多少？数据量为多大？大概几个区块才能处理完这些交易？高度在 9991～10000 区块内包含的总交易数目是多少？</td></tr>
<tr><td>当前比特币待验证的交易数目</td><td></td></tr>
<tr><td>当前比特币待验证的交易数据量</td><td></td></tr>
<tr><td>处理完这些交易所需的区块数</td><td></td></tr>
<tr><td colspan="2">2．利用区块链浏览器学习区块链账本层构造。
观察某 P2SH 交易的赎回脚本：
OP_3DUP OP_ADD OP_PUSHNUM_9 OP_EQUALVERIFY OP_ADD OP_PUSHNUM_7 OP_EQUALVERIFY OP_ADD OP_PUSHNUM_8 OP_EQUALVERIFY OP_PUSHNUM_1
说明该脚本规定的解锁条件和运行机理，并拟写其解锁脚本。</td></tr>
<tr><td>解锁条件</td><td></td></tr>
<tr><td>运行机理</td><td></td></tr>
<tr><td>解锁脚本</td><td></td></tr>
</table>

<table>
<tr><td rowspan="12">实验过程记录</td><td colspan="2">3．利用区块链浏览器解析并学习以太坊合约层构造。</td></tr>
<tr><td>主要步骤</td><td></td></tr>
<tr><td>心得体会</td><td></td></tr>
<tr><td colspan="2">4．体验区块链钱包原理。查阅相关资料，使用正则表达式，尝试生成满足以下条件的地址。</td></tr>
<tr><td colspan="2">（1）包含“abcd”的地址，写出所用命令，并记录结果的截图。</td></tr>
<tr><td>所用命令</td><td></td></tr>
<tr><td>结果截图</td><td></td></tr>
<tr><td colspan="2">（2）以 44 开头且以 99 结尾的地址，写出所用命令，并记录结果的截图。</td></tr>
<tr><td>所用命令</td><td></td></tr>
<tr><td>结果截图</td><td></td></tr>
</table>

<table>
<tr><td rowspan="8">实验过程记录</td><td colspan="2">（3）以 5 个数字结尾的地址，写出所用命令，并记录结果的截图。</td></tr>
<tr><td>所用命令</td><td></td></tr>
<tr><td>结果截图</td><td></td></tr>
<tr><td colspan="2">（4）以 2 个数字再接“yyy”结尾的地址，写出所用命令，并记录结果的截图。</td></tr>
<tr><td>所用命令</td><td></td></tr>
<tr><td>结果截图</td><td></td></tr>
<tr><td colspan="2">（5）尝试碰撞刚生成过的自己的某个地址，尝试逐渐减少输入的地址长度，体验并分析碰撞难度。</td></tr>
<tr><td>结果记录
和分析</td><td></td></tr>
</table>

<table>
<tr><td rowspan="8">实验过程记录</td><td colspan="2">5．拓展实验：批量获取并分析区块链元数据。</td></tr>
<tr><td colspan="2">（1）拓展实验 1：数据批量获取与挖掘。
调用公开的 API 或获取 2020 年 8 月的所有区块数据，对所有交易发布者所使用版本号 version 的情况进行统计分析，并作统计图；对 locktime 字段的使用比例进行统计分析；过滤出所有 locktime 不为 0 的交易，并将该交易数据集写入一个文件。</td></tr>
<tr><td>实验原理</td><td></td></tr>
<tr><td>主要步骤</td><td></td></tr>
<tr><td>关键步骤
截图</td><td></td></tr>
<tr><td>实验结果
分析</td><td></td></tr>
<tr><td colspan="2">（2）拓展实验 2：利用开源项目部署自己的区块链浏览器，并尝试调研区块链浏览器的构造方法。读者可以自行搜索区块链浏览器开源项目。</td></tr>
<tr><td>实验原理</td><td></td></tr>
<tr><td>主要步骤</td><td></td></tr>
</table>

实验过程记录	关键步骤截图	
	实验结果分析	