区块链基础及应用 2022

Exercise 1

在本次作业中，你将创建多个交易，并将其发布到比特币测试网。我们将使用 python bitcoinlib 提供启动代码，python bitcoinlib 是一个用于操纵比特币交易的 python3 库。

1 项目背景

1.1 区块资源管理器概述

我们将使用在线区块管理器上传和查看交易，而不是下载整个 testnet 区块链并在本地机器上运行比特币客户端。我们将要使用的是 BlockCypher，它具有一个很好的 web 界面以及一个 API，用于提交原始事务，初学者使用它来广播创建的事务。在完成并运行每个练习的代码之后，BlockCypher 将返回新创建事务的 JSON 表示，并将其打印到终端。可以在 BlockCypher 的开发人员 API 文档中找到一个示例事务对象以及每个字段的含义 https://www.blockcypher.com/dev/bitcoin/#tx。在这个项目中，我们关注的是哈希值、输入和输出字段。

1.2 比特币交易剖析

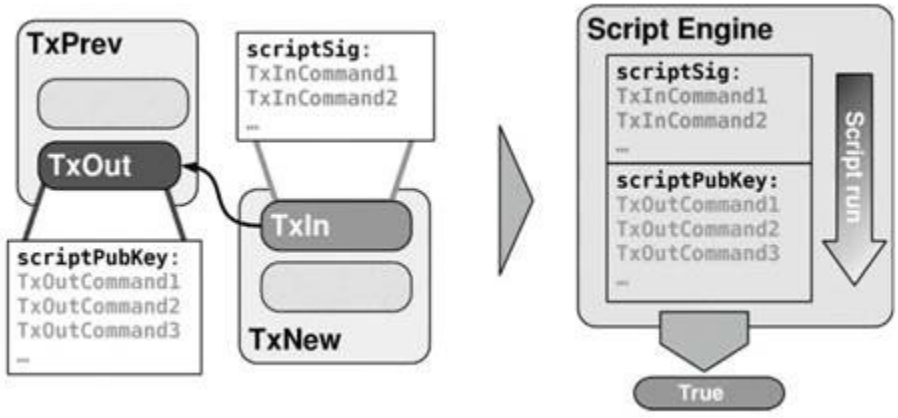

图 1：每个 TxIn 引用前一个事务的 TxOut，只有在 TxOut 的 scriptPubKey 的 scriptSig 输出 True 时，TxIn 才有效。

虽然除了设置一些参数和实现 scriptPubKey 和 scriptSig 脚本之外，不需要自己编写任何 Python 代码，但是了解这些代码和使用到的术语将非常有用。

比特币交易基本上是一个输出列表，每一个输出都与比特币的数量有关，这些比特币被一个叫做 scriptPubKey 的程序（有时也被称为“智能合约”）和一个输入列表“锁定”，其中每一个都引用一个现有转换的输出，并以一个称为 scriptSig 的程序的形式包含对该输出谜题的“答案”。验证 scriptSig 包括关联的 scriptPubKey，运行组合的脚本并确保它的输出为 True。大多数事务都是“PayToPublicKeyHash”或“P2PKH”事务，其中 scriptSig 是收件人的公钥和签名的列表，scriptPubKey 对这些值执行加密检查，以确保公钥哈希到收件人的比特币地址，签名有效。

每个事务输入称为 TxIn，每个事务输出称为 TxOut。图 1 总结了具有单个输入和单个输出的事务的情况。

交易输入中比特币的总和不得超过交易有效的输出总和，且总输入与总输出之间的差额隐含地被视为交易费用，作为一个矿工，可以修改一个接收到的事务，并将一个输出添加到他们的地址，以弥补差额，然后再将其包含在块中。对于这个项目，你创建的所有事务都将消耗一个输入并创建一个 PayToPublicKeyHash 输出，该输出将向 testnet faucet 发送一定数量的比特币。对于每个练习，在指定发送量时，你需要考虑交易费用，并从发送的输出量中减去一点，例如.001 BTC。如果不包括费用，你的交易很可能永远不会添加到区块链中，而且由于 BlockCypher 将删除一两天后仍未确认的交易，因此你必须包括费用并确保你的交易最终得到确认。

1.3 脚本操作码

比特币堆栈操作码记录在比特币 wiki[1]上，在为交易的 scriptpubkey 和 scriptsig 编写程序时，可以使用它们的名称来指定操作码。例如，下面是一个函数的示例，该函数返回一个不能使用的 scriptPubKey，而是用作任意数据块的存储空间，用户可能希望使用操作返回操作码。

```
def save_message_scriptPubKey（message）：
    return[OP_RETURN， message]
```

你可能要使用的一些操作码包括 OP_DUP， OP_CHECKSIG， OP_EQUALVERIFY， 和 OP CHECKMULTISIG，以及其他的操作码。

2 入门

1. 从课程网站下载入门代码，导航到目录并运行 pip install-r requirements.txt 安装所需的依赖项。确保使用的是 python3+。

2. 确保你已经了解比特币交易的结构，如果你想了解更多信息，请阅读推荐阅读部分的参考资料。

3. 实现下面练习的代码，使用比特币测试网络（testnet）测试代码。在这之前需要从 https://coinfaucet.eu/en/btctestnet/ 获取实验所需的比特币。每个练习都以将 coin 返回 faucet 为结束。

4. 必须通过编写 scriptPubKey 脚本来实现以下练习中指定的每个交易，该脚本将锁定一定数量的比特币作为第一个交易的一部分，以及一个 scriptSig 脚本，该脚本赎回第一个交易并将其发送回 faucet。必须在操作码级别显式地编写脚本（不调用 python bitcoinlib 函数来为执行操作）。

5. 可以使用事务哈希来跟踪块资源管理器工具上的事务，例如 https://live.blockcypher.com/btc-testnet/。

3 安装程序

1. 用 keygen.py 生成一个 testnet 私钥和地址，并在 config.py 适当位置复制并粘贴私钥。

2. 在 faucet(https://coinfaucet.eu/en/btctestnet/)粘贴第一步得到的地址，会得到一些 testnet BTC（注意 faucet 通常会根据比特币地址和 IP 地址对请求进行限制，所以尽量不要频繁请求）。注意保存 faucet 提供的 tx Hash，因为下一步需要它。在区块链资源管理器中查看事务还可以知道事务的哪个输出对应于你的地址，下一步也需要这些信息。

3. faucet 将给你一个可消费的输出，但我们希望有多个可花费的输出（至少为 3）来完成后面的多个练习，因此需要将其拆分。以防我们意外锁定一些无效的 scripts。编辑 split_test_coins.py.py，其

中 txid 是上一步中事务的事务哈希值，（如果你的输出是 faucet 事务中的第一个输出值，则 utxo id 为 0 如果是第二个输出值，则为 1）。split_test_coins.py 中的 n 是期望将测试硬币平均分割成的输出数。理想情况下 n=3，每个练习使用一个交易输出，但是如果你可能由于一个错误的脚本在每个练习中意外地锁定一个输出，那么你可能需要将 n 设置为更大的值，比如 10，这样你就不必等待再次访问 faucet 或使用不同的比特币地址。

4. 如果成功的话，你应该得到一些关于交易的信息。请注意事务哈希，因为每个练习都将使用此事务的输出，并将使用此哈希引用它。

4 练习

要发布每个练习的事务，请编辑 python 文件参数，以指定事务输出以及要在事务中发送的金额。注意所创建事务的事务哈希并将其保存至 transactions.py. 如果你编写的脚本无效，将在发布之前引发异常。

完成每个练习后，在区块链浏览器中查找交易哈希，以验证交易是否被网络获取。在提交哈希之前，请确保所有事务都已成功确认。

练习 1。打开 ex1.py 并完成标有 TODOs 的脚本，以赎回您拥有的输出，并通过标准 PayToPublicKeyHash 事务将其发送回 faucet。

5 推荐阅读

1. 比特币脚本：https://en.bitcoin.it/wiki/Script
2. 比特币交易格式：https://en.bitcoin.it/wiki/Transaction
3. https://privatekeys.org/2018/04/17/analytic-of-a-bitcoin-transaction/