第一版

First edition

MASTERING MONERO

The future of private transactions

精通门罗：隐私交易的未来

Written by SerHack

Co-authored by the Monero Community

作者：SerHack

校对：门罗币社区

第一版

MASTERING MONERO（精通门罗币）

The future of private transactions（隐私交易的未来）

SerHack and the Monero Community
原著作者SerHack 与门罗社区
（中文翻译 筋斗云& wookey极光&门罗中文社区）
（校对 Scott）

贡献名单

文案

- 4matter
- Amichateur
- Anonimal
- ArticMine
- Cryptochangements
- dEBRUYNE
- Isthmus
- Midipoet
- moneroexamples
- QuickBASIC
- Rehrar
- Sarang Noether
- Xeagu

设计

- Gustaf Baltsar Garnow
- TheMonera

插图作者

- Andrés Fernández Cordón

出版人

- Justin Ehrenhofer

编辑

- UncagedPotential

中文翻译贡献者名单

总翻译：头等仓区块链研究院 筋斗云
第五章：Wookeywallet：极光
第七章：生姜，时昕昱,剑灵山，逆羊羊
校对：夏洛特
特别贡献：中原愚人，二辉，叶子，大发村1618，江海潮,Jeremy,

感谢门罗币中文社区的每个参与人员，请关注微博超话门罗币和巴比特论坛:门罗节点。推特：PRMonero

本篇翻译社区做了尽可能的工作保持本文忠于原文，如果有能力还是请去阅读英文原著https://masteringmonero.com/。本文受版权保护，任何的转载，改写，翻译都要经过原著作者授权，请联系!press@masteringmonero.com

Chinese vision credits

Chief translator: First.VIP Research "筋斗云“
Chapter 5: Wookeywallet: 极光
Chapter 7: 生姜, 时昕昱, 剑灵山, 逆羊羊
Proofreading: 夏洛特
Special contribution:中原愚人，二辉，叶子，大发村1618，江海潮,Jeremy

Thanks to everyone in the Monero Chinese community, please follow us on WeiBO 门罗币超话 and https://www.chainnode.com/forum/145. Twitter: PRMonero

We have tried our best to keep this Chinese version as loyal to the original as we can. If it is possible please read the original English at https://masteringmonero.com/. Note that this book is copyright protected, and any reprinting, rewriting and translation must be authorized by the original author. Please contact press@masteringmonero.com.

感谢Monero 社区论坛FFS系统捐赠者的慷慨捐助，才使得免费分享这本免费的电子版书成为可能。 作者和社区已投入超过2100小时来制作本书，本书对新手和经验丰富的门罗币用户都是有用指南。 我们希望您发现此参考资料的价值并广泛分享。

由 LernoLibro LLC 发行.

第一版: 2018年 12月

免费的PDF版本公开于门罗币创造五周年纪念日,2019年四月18日.

本书是一项广泛的社区参与编辑的结果，这种冗长的资源可能包含错误.

如果你发现本书有任何知识上的错误，或者语法和拼写错误请发送内容到邮箱: support@masteringmonero.com

目录

精通门罗币的创作

关于作者

大家好，我是Nico（ID：SerHack），一名来自意大利的计算机安全领域的研究者，一名门罗（Monero/XMR）贡献者，同时也是这本书的作者。寻找优秀的资源和学习加密货币知识是很容易令人生畏的。对于新人来说，要找到深入浅出平白易懂的技术文档特别不容易。当我刚开始学习门罗的知识的时候，我需要花很多时间来寻找和评估相关的资源。

这本《精通门罗币》（Mastering Monero）就是为了陪伴你们的区块链之旅而写的。无论你是正在设置自己的第一个个人钱包，还是对那些深入的技术细节感兴趣，都不妨可以读一读。前几章是为那些对门罗感兴趣和希望使用门罗的人而写的，其中包含了浅显易懂的解释和示例，并辅以实际操作的指南。之后的章节会逐步过渡到更复杂深入的话题，专为有志于为门罗生态添砖加瓦的开发者而编写。

我的加密货币世界之旅，是从2016年1月接触比特币开始的。那是我总是在思考，如何在这个透明的公共账本上进行衍生。因为比特币和其它大部分加密货币的地址和交易记录都是透明、公开的，其交易信息在无意中将个人财务详情暴露出来。每一个地址余额都是公开的信息，这使得任何人都可以研究你的收入，消费习惯以及加密货币资产量。

这可能导致一些人们不希望发生的影响，比如基于钱包余额的价格操纵。

在2017年5月朋友向我介绍门罗之前，我一直以为比特币就是加密货币里的“唯一”。直到我被门罗币优美的全新范式所震撼：一个账户余额、交易信息，发送方和接收方这些脆弱的信息被隐藏保护起来的新世界。因为其隐私功能被设置为默认且永久执行，整个门罗的区块链如同蒙上了一层面纱。用户甚至连意外发送一个公开交易的机会都没有。

在认识到这个项目的重要性以后，我开始寻找向它提供贡献的途径。很快我看到了一个机会：通过创建商业在线支付的网关来进行门罗的大范围推广，于是我作为先锋加入了门罗集成项目（Monero Integrations）。这个开源的代码库围绕着门罗以隐私保护为中心的精神进行设计：无需签约，无需第三方，因为资金直接发送到收款人的地址中。门罗社区对该项目非常支持，项目的所有资金都来自于门罗论坛众筹系统（Monero Forum Funding System ）(FFS)的捐赠。

在门罗集成项目工作的时候，我认识到终端用户或者新贡献者的一大障碍，是缺少一份全面的门罗指南。这个需求也激励着我来编写这本《精通门罗币》，希望它可以作为全球社区了解门罗的通用资源。我非常感谢来自于FFS的慷慨捐助，他们使得我可以把这本书以免费电子版（和纸质书！）形式出版出来。

无论你是一页页地细度，或是直接跳到你感兴趣的部分，我都希望你在了解门罗以及其社区内令人振奋的项目时，感到发自内心的愉悦。

这本书的结构是什么样的？

前两章将对一些关键主题和技能做非技术性的介绍。对于那些想要了解背后的细节的读者来说，第三章和第四章将提供对门罗隐私功能和区块链的非数学化的概念解释。之后的章节将带你潜入更为复杂的技术细节世界，以便你理解，开发和集成门罗。

第一章：（加密货币&门罗简介）：这章将对区块链和加密货币的一些关键概念和想法做一个大众化的非技术性的介绍。我们将会回顾加密货币的历史和基础概念，并阐述区块链技术如何解决传统主流的金融系统（特别是银行业）存在的几个问题。不幸的是，大多数加密货币都面临着隐私性不足这个问题——我们将会讨论这些缺点的个人思考，并且了解门罗是如何降低这些风险和保护你们敏感的财务信息。

第二章：起步—（接收、储存和发送门罗币）：这章将介绍一些使用门罗的实用技能。我们将会介绍一些必要的术语，并了解各种类型钱包的优点和缺点。你将会习得，如何创建你的第一个钱包，你甚至还可以参考这本书的钱包示例做练习！

第三章：（门罗是如何工作的？）：我们将会讨论门罗的四项核心技术：环形机密交易（RingCT），环形签名（Ring Signatures），一次性/隐蔽地址（one-time/stealth addresses)和Kovri（译者注：一种基于洋葱路由的变种I2P的，用以保护IP地址隐私的工具）。这些解释将不掺杂任何代码或数学，所以你可以在概念上理解每种技术的工作原理，以及它们对门罗的作用。

第四章：（门罗网络）：这章将概念化地描述门罗的网络以及矿工如何在区块链上处理交易。我们将讨论矿工的激励机制（出块奖励+交易费）和所提供的服务（确认交易、保障去中心化和无需信任的网络的安全）。我们还将介绍关于专业化挖矿设备的"热点话题"，并描述门罗社区的平等主义哲学（egalitarian)和对ASICs（编者注：Application Specific Integrated Circuit，特定用途集成电路）的积极抵抗。

之前的章节，我们一直专注在如何概念化、直观地理解门罗，之后的章节我们将深入门罗内核，深入它的数学和代码。如果你选择继续钻研这些高深的主题，那么你将真正地"精通"门罗币！

第五章：（深入门罗和密码学）：本章将带你深入第三章提及的隐私技术细节。我们将不再使用类比，而是开始接触数学，以及门罗改进后的CryptoNote协议的细节。

第六章：（社区与贡献）：这章为那些希望有志于为门罗社区贡献时间和才华的人提供一些信息帮助。

你的才华在这里总有施展的舞台：翻译、外联、开发、应用、等等。

第七章：（面向开发者的门罗集成）：这一章中，我们将讨论支付方案，以及通过OpenAlias（人类可读）和门罗URI（机器可读）。商家支付方案的开发者，将学习如何通过OpenAlias创建简化的地址。开发者还将学习如何通过与Monero daemon的远程过程调用（Remote Procedure Calls，RPC）来与门罗区块链进行交互。另外，开发者还可以写的如何使用Python来实现一些基本的功能。

第八章：(钱包指南、答疑和小贴士):这章包含了关于图像化（GUI）和基于终端的（CLI）钱包的从设置到常见问题的解答等各方面的信息。

第一章

加密货币和门罗简介

Maria从George处购买汽车，我们在这一章将介绍她可以选择的3种支付方式：传统银行，公开的加密货币（如比特币）和门罗币。

1.1 通过银行支付

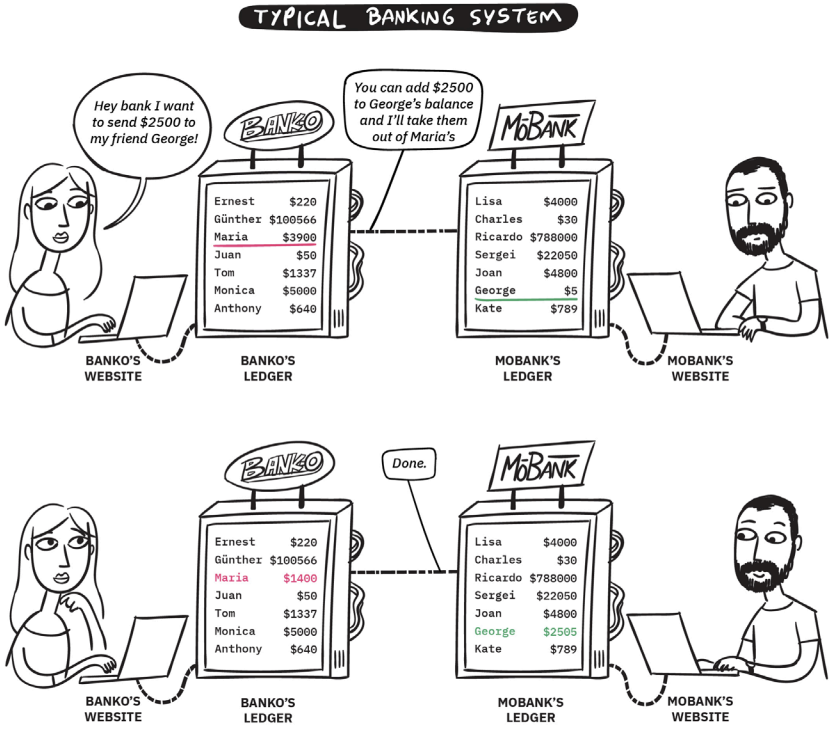


图1.1：Maria通过传统银行系统向George汇款。

如果Maria通过传统银行系统向George汇款，那么他们需要对两个中介方（也即是他们各自的开户行）产生信任以完成资金的象征性转移。

在这个过程中，没有任何实体钞票或资产的转移；两个银行只是在各自的数据库中简单地进行编辑，以表示他们的资金进行了转移。当Maria向银行提交了转账（无论是通过电汇，网络银行，或是app），她的开户行在账本中将她的账户余额扣除掉$2,500，然后联系Geroge的开户行，请求他们往George的账户上增加￥2,500。

这个做法有一些缺点和风险，而且人们必须对银行有充分的信任。Maria，George以及银行的行为都依赖于一个前提：交易合情合理，账本数据真实。这种第三方中介的信任是有风险大，因为不法分子可以在账本上篡改数据来凭空“创造”财富。

此外，Maria实际上并不拥有$3,900，她只是拥有一张存在银行的借据，并且她必须确保这个借据是可以提现的。但是她的开户行到底有没有足够的资金（$3,900），这个答案无从知晓，因为她没有审计的途径。

I实际上，银行真的有可能满足不了客户的提现需求。大多数银行目前都采用部分储备金制度，换言之，政策允许他们只持有远少于向客户承诺的资产量。

根据汇款方式的不同，Maria向George发起的汇款，可能只需数分钟，也可能需要几天才能到款。此外，George也无法参与发起汇款后的环节，所以无从知晓和监视整个过程。

大部分人并没有经历过经济危机，所以认为银行和借据肯定不会出现大的问题。只有少数人担心把一生的积蓄都托付给一个不透明的机构，无异于把所有鸡蛋都放在一个篮子里。当下列情况发生时，个人的资产将会受到损失:

- 过失：银行出现纰漏
- 财务问题：银行放账过多，业务岌岌可危
- 内部的作恶：银行内部的高层或个人意志主导的用户资产挪用、偷盗
- 外部的作恶：银行被抢劫，或被黑客入侵

幸运的是，区块链技术的出现，可以减轻这些可能的损失。区块链是一个所有人都可以平等使用，查看和验证的分布式账本。它使得互不相识的人可以就某一共享的信息达成共识，我们把这一点称作【去中心化共识】。这项卓越的功能在过去十年中不断地演化。

在接触区块链的初期，特别是当一下子听到特别多的行话和专业概念，人们就很难理解这些术语的含义。你可以这样理解：区块链是一项可以使网络达成【去中心化共识】的科技。它使得陌生的人和人可以安全地共享一个账本的时候，并在此基础上打造一种具有电子现金功能的【加密货币】。正如法币世界有像欧元、美元、日元这些货币一样，加密货币的世界也有着像门罗币、以太坊、比特币等不同特点的币种。

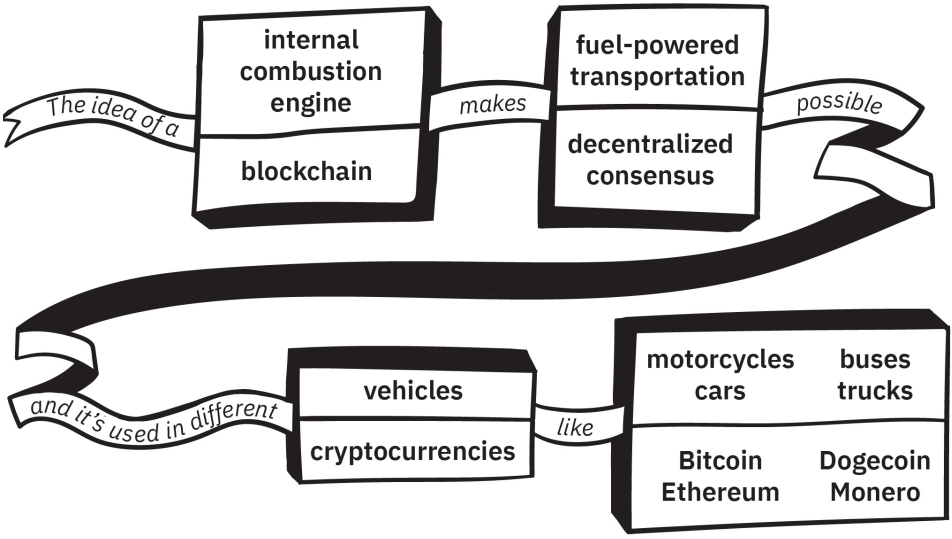


图1.2：就像引擎驱动汽车行驶，区块链技术使得去中心化共识变成可能。不同的汽车（轿车，摩托，巴士，卡车）使用不同的引擎，不同的加密货币（比特币、以太坊、狗狗币，门罗币）也使用着不同的区块链。

1.2 区块链简介

对于任何想了解门罗和其区块链工作原理的人来说，底层的数学和加密学知识都不是必须要了解的（就像一个人不需要懂DNS服务器和IPv6协议，也可以在互联网上穿梭自如）。这一章将在不涉及技术细节的层面上，解释一些关键概念和词汇。如果你想深入了解加密学框架，可以直接跳到第4、5两章。

1.2.1 什么是区块链?

【区块链】是指安全地保障共享数据库中的数据的一种方式。它的革命性突破在于，构建了一个人与人之间无需信任，却可完全掌控自己的资金，易验证，易审计，没有中心权威的系统。

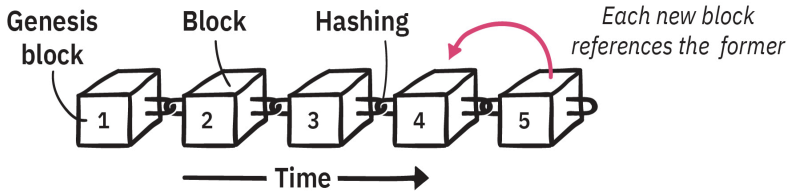


图1.3：每过几分钟，网络都会往区块链上添加一个用于存储信息的永久性区块，并且通过哈希值使其与上一个区块牢牢连接起来。

区块链欢迎世界上每一个人参与其中，成为其网络的维护者并对其它维护者进行验证，以确保他们是诚实的。当有人试图在区块链上传播并记录信息，网络维护者会将这些信息按区块分组，并使用加密学工具，

使其中的数据不可篡改，且永久地添加到区块链中。

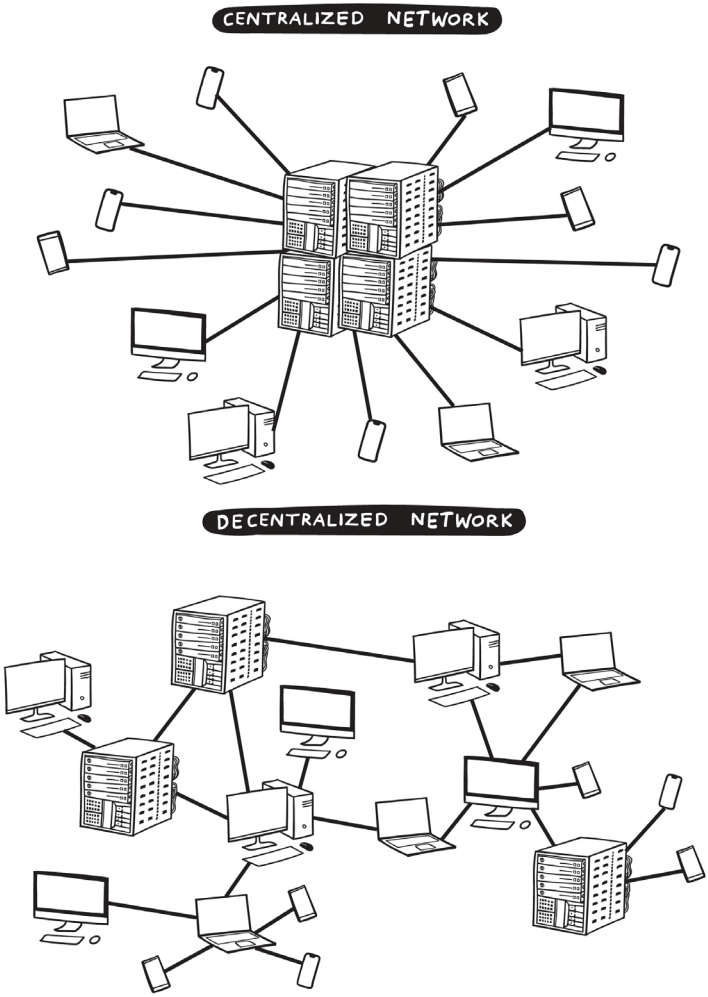
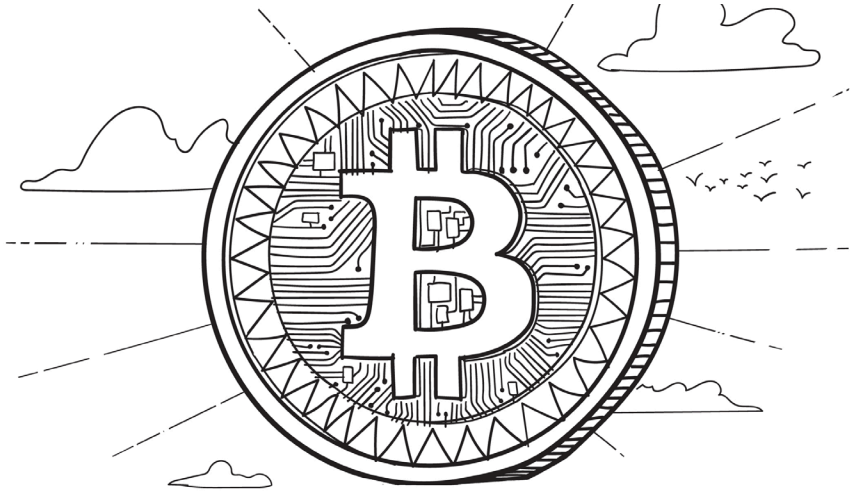


图1.4：在传统的中心化网络中（如第一张图所示），所有用户必须链接至一个第三方设立和维护的特定设备（服务器组）。在去中心化的网络中，用户互联，自组成网。门罗使用的就是后者：无特殊的中心化服务器，全由志愿者组成高弹性的点对点网络以共享信息。

一旦数据被记录在区块链上之后，它就无法被删除，转移或者改变。记录无法篡改，且网络上的每个参与者都会持有相应的备份以做验证用。大多数区块链都使用高明的挖矿模式，来激励节点参与其中，确保信息被城市地记录下来并同步。这些类型的去中心化非常地强健，不会被任何单个机构或中心化服务器恶意地攻击或操纵。

这些去中心化系统也无需信任，因为网络的参与者维护和验证自身备份的账本，无需依赖第三方。这样一个全球化的，不可篡改的记账系统，特别适合记录财务数据。第一个现代化的分布式区块链，作为比特币的底层机制，于2008年首次在世人前亮相。



2008年10月的最后一天，一个化名为Satoshi Nakamoto（国内译名：中本聪）的个人/组织，发表了一篇白皮书“比特币：一个点对点的电子现金系统”（*“Bitcoin: A Peer-to-Peer Electronic Cash System.”*）。在这份日后改变世界的文档中，

作者阐述了名为【比特币】的开源去中心化加密货币的设计框架，以及它所基于的名为【区块链】的革命性技术。

如图**1.1** 所示，若要传统的银行系统中顺利完成转账，系统需要发起数笔交易，多个账本以及对多个银行的信任。

而在下图**1.5**中，Maria通过区块链系统，向George转了10.5个比特币（从Maria的地址1BuUygisXY发往George的地址1eK5FSywkp。为了方便起见，该图展示的是比特币转账的场景，实际上几乎所有的加密货币都使用这种类型的公共账本，也经受着它的利与弊。

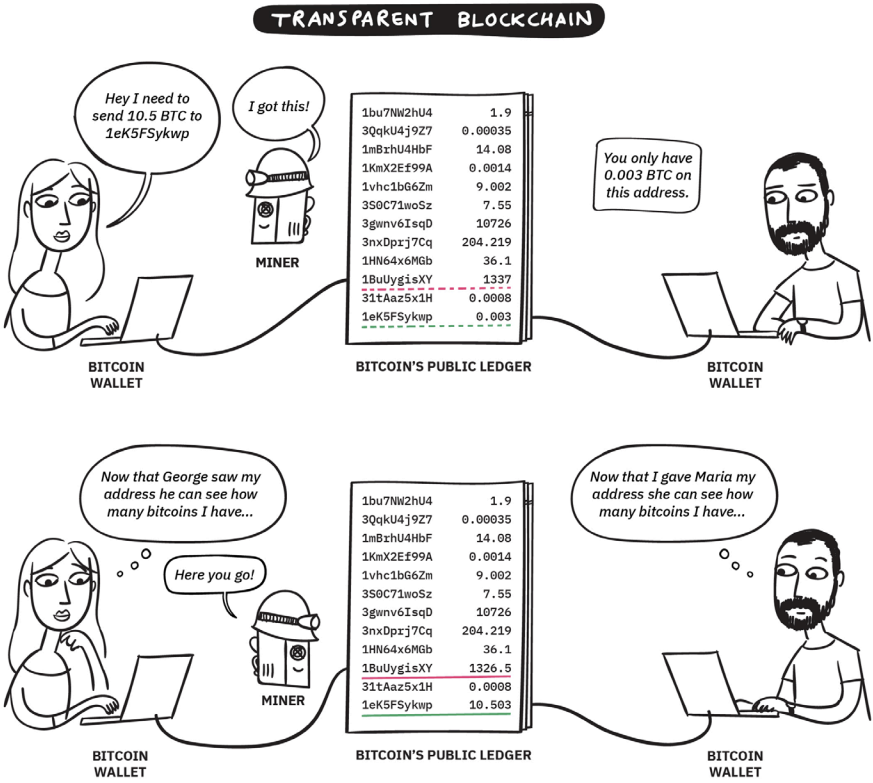


图1.5：Mariz通过公开的公共区块链向George进行加密货币转账（例如比特币）。

1.2.2区块链的优越之处

以下是区块链显而易见的几点优越之处：

- 简约（&快速）：Maria转账给George这个过程，只需对一个账本做更新，且步骤简捷，只需数秒钟或数分钟即可完成（交易的确认时间）。而银行和电报汇款往往需要数天，甚至是数周。

- 无第三方风险：Maria和George将自己的资金存在一个以密码学作为安全保障，无需他人维护的系统里，而不是需要信任的第三方。

- 假名：与传统银行不同，加密货币账本从来不会记录在账户中你的真实名字，如“Maria”和“George”。建立加密货币钱无需个人信息。Geroge使用假名（1eK5FSywkp）来接收来自Maria（1BuUygisXY)的汇款。

比特币等一众加密货币的革命仍然在进行中。在去中心化网络中，任何人都可以按自身地意志来存储和转移资金。之前，如果没有值得信任的银行和信用机构，大规模的资金存储将是一个问题。同样的，我们的转账也依赖于支票、电汇、借记卡/贷记卡等第三方支付工具。

这是有史以来第一次，我们在无需银行和其它外部机构许可的情况下，也可以自由行使自身的财务权。

这一切归功于加密货币。在不久的将来，任何设备（电脑，手机、平板）都可以作为加密货币的全功能钱包，用以接收、存储和发送资金。创建钱包无需任何形式的身份信息、手续费和授权，因为系统是通过地址（一串“随机的”字符串）来识别用户，而不是依靠姓名、住址和手机号等身份信息。

1.2.3 区块链的不足之处

大部分的加密货币都是假名的，因为用户是通过由数字和字母组成的“毫无意义“的字符串来识别的，而不是个人信息。当你收到一笔加密货币汇款，你无法获知发送者的姓名。你只知道它来自地址，如：1A1zP1eP5QGefi2DMPTfTL5SLmv7DivfNa。

虽然假名能在一定程度上保护个人隐私，不过还是让人有机可趁的。任何一个去中心化区块链系统的参与者，都可以拥有一个完整的账本。在加密货币的语境下，这个账本可以用来查明任何一个（如比特币）地址的资产。

在这个公开共享的账本上，所有的账户资产和历史交易都是公开的！借助一些网站的帮助，你可以轻而易举地查询任何地址和交易的信息。

假设你开了一家商店，一个顾客购买一袋面包，并从比特币地址3P3QsMVK89JBNqZQv5zMAKG8FK3kJM4rjt付款给你。你可以很快查到（You can instantly check）这个账户居然共收到超过5,000个比特币！当你知道眼前这位客户最近经手过五千万美元，你也许会对他个人进行商品提价，甚至心生歹意。

简言之，隐私保护的不足将导致个人安全隐患。

除了知道他的资产总量以外，你还可以浏览他的历史交易明细：金额、时间戳和参与双方的地址。对这些交易的分析，可用以描绘个人画像：消费习惯、收入、储蓄和互动对象。

如果你的区块链身份和实际身份关联起来的话（例如交易所账户注册和网购），你的个人敏感信息将会暴露无遗。只需简单几步，就可获知账户的实际拥有者，比如你动动脑，就会发现上面的两个地址分别属于中本聪和菠萝慈善基金（Pineapple Fund charity）：https://www.blockchain.com/btc/address/1A1zP1eP5QGefi2DMPTfTL5SLmv7DivfNahttps://www.blockchain.com/btc/address/3P3QsMVK89JBNqZQv5zMAKG8FK3kJM4rjt

有几个独立的机构在追踪和反匿名化这些公开的区块链。比如Elliptic提供了一个交互式的浏览器interactive explorer，来呈现中本聪和支付处理商，交易所，论坛，市场，博彩机构，

图1.6：Elliptic的比特币Bitcoin Big Bang explorer.浏览器

慈善组织和其它个人、服务商的资金往来。

图1.6：Elliptic的比特币Bitcoin Big Bang explorer.浏览器，呈现2010年早期的比特币资金流，关联方包括矿池，Mt.Gox和丝绸之路。

停下来想想你日常生活中生成的宝贵又敏感信息：信用卡交易，你的每一次搜索，你浏览或购买的商品，互动的社交媒体页面，等等。你的银行、支付处理商、科技/数据行业的巨头和政府一直都记录着这些信息，并以此获利。

如此大规模的数据收集，导致了你的个人隐私信息以较中心化的方式储存起来，而这些信息宝库，恰好是黑客垂涎的目标，也常在黑市上被转卖。你的个人信息很有可能在你不知情的情况下进入公众视野，并与你的人口或消费档案相关联。

看下最近的Equifax，Target，Home Depot，和Uber，Panera数据操作违例，个人和财务信息泄露的例子屡见不鲜，个人和银行卡都存在安全隐患。

数据的意外泄露并不是唯一的担忧。 大数据和科技公司在网络上仔细记录下你的行踪，以便他们你的偏好画像，从而可以向你提供更好的服务。通常，这只是为了市场定位和推广；然而，这些数据可以被利用来操纵你的感受（manipulatingyour feelings）或者你的

投票行为（your voting behavior.）

任何你被机构所记录下来的信息最终都可能被盗窃，售卖和不正当地使用。你需要对自己的数字指纹格外小心，因为信息一旦暴露则永久暴露。

如今，隐私在主流的经济和商业系统中难有容身之处。传统的支付处理商，银行和加密货币会留下非常明显的足迹，他人可以研究，监视这些信息并从中获利。一旦信息被收集走，你通常无法再去掌控它的传播，也无法遏制个人安全信息和隐私被卖给未知组织而到导致的风险。

唯一可保障个人财务隐私的方法就是不要泄露它们！因此，我们需要一种安全的交互方式，一个交易无法跟其他交易、身份、资产关联起来的系统。而门罗就是你的最佳选择。

1.3 门罗简介

MONERO（发音为：/mōněrō/，复数为Moneroj）是一个专注于交易隐私保护和抵制审查的优秀加密货币。大多数加密货币（如比特币和以太坊）公开可验证的性质，使得任何人都可以追踪其账本上的资金流动。此外，交易记录和个人身份的关联会危害你的人身安全。

为了杜绝这些危险，门罗使用密码学技术创造了一个不会泄露**发送者、接收者和交易金额**信息的互动网络。同其它加密货币一样，门罗也拥有一个任何人都可以下载和验证的去中心化账本。

门罗应用了一些数学手段来隐藏敏感信息和阻断区块链追踪。门罗的隐私保护功能允许网络对交易进行验证，查看发送者的资产是否大于发送额，同时又可以确保发送者的交易金额和资产量不被任何人所知。没有人可以查看他人的账户资产，并且交易资金的源头也无法追溯。

门罗的核心特质和理念之一是默认的强制化隐私保护。用户任何有意或无意发起的不安全交易将会在初始阶段被杜绝。

门罗通过对公开交易的零容忍，来确保每一个用户都可以安心地使用这个网络。

图1.7：该图描绘的是Maria使用门罗从George处购买汽车。该过程大体上和其它的加密货币支付相似，但是其中敏感的信息被隐藏了。我们将在第3章和第5章分别对这些技术进行概念上和技术上的解释。

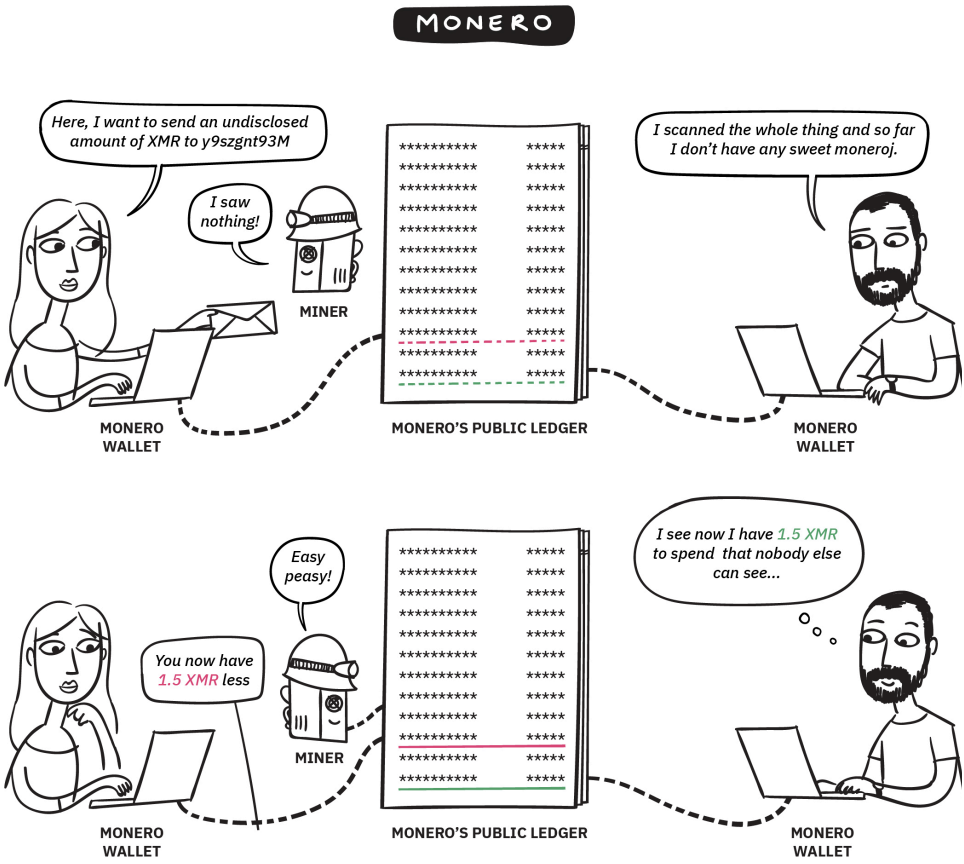


Figure 1.7 - Maria 发送 George 门罗币. " ****" 代替了账户余额，收款钱包等信息，让门罗网络更安全.

1.3.1 门罗的理念

门罗的设计遵循以下的理念和原则:

- **网络去中心化：门罗的网络和账本没有国家和地域的限制。它也不存在那种可以被黑客恶意攻击，掌控和监视的中心服务器和数据库。单个政府的任何（禁掉门罗的）意图，如关闭境内所有节点，限制门罗的发送和接收，都将注定是一场徒劳。因为在这个国家之外，还会有很多力量来维护门罗网络、处理交易。**

- 财务安全：Monero网络由不可破解的加密机制自我保护，因此无需信任和您资金交易关联负责的第三方。每个Monero参与者都可以自己验证分类帐的有效性，因此您甚至不需要信任节点运营商！ （您可以在第5章中了解有关保护Monero的加密技术的更多信息。）

- 财务隐私：大多数区块链牺牲隐私性来换取强大的安全性。然而，门罗在将隐私性列为高优先级的同时，也确保了很强的安全性。交易金额，发送者和接收者的身份都已在区块链上隐藏，以防止你的门罗资产信息被追踪。

- 货币可互换性（Fungibility）：它指的是：单位资产难以区分，并且可以互换。举例来说，你借给邻居1公斤面粉用以做蛋糕。过了一个星期，邻居还你了一公斤的面粉。虽然这一公斤归还的面粉不是你原来借出去的那袋面粉，也不管它源于何处，但这对你来说没有差别，因为两袋面粉是可互换(fungible)的。 不过，机动车就不是可互换的了，如果你借给邻居一辆车，你通常会希望他还来的是你当初借出去的那辆。

对于门罗来说，货币可互换性是其高端复杂的隐私技术的体现。交易记录被隐藏的同时，所有门罗币的历史也被隐藏了。如果你借给朋友1个门罗币，他还任何一个门罗币给你都没有问题，因为门罗币之间是没有区别的。大部分加密货币保持公开透明，使得可互换性这一点，显得微不足道；然而，它却是所有货币在使用过程中必不可少的一点。

1.3.2 门罗的实际生活"案例"

这一部分我们将谈到一些使用不安全的加密货币将遇到的困难和风险。为了方便，我们就以比特币为例，因为它相当于所有公开透明区块链的原型。但是，以下提到的缺陷是所有透明公开的区块链都面临的。

- **抬价：Sofia是小镇上唯一的机修师。一名顾客向她支付比特币，随后Sofia查看了他的地址，发现上面存的比特币足够买一台兰博基尼。当这名顾客再次需要维修服务时，Sofia加倍要价。如果这名顾客使用的是门罗，那么Sofia将对他的资产一无所知，也不会有抬价的行为了。**

- **财务监管： Oleg的父母给了他一些比特币来购买教科书，同时他们也一直查看着儿子的比特币地址。几个月后，Oleg将剩余的一些比特币捐给了一个公共慈善组织，但该组织的政治观点与他父母不合。直到收到充满了父母的愤怒和呵斥的邮件后，Oleg才发现自己的地址一直被监视着。如果Oleg使用的是门罗，那么也不会有后来的这些事情了。**

- **供应链隐私：Kyung-seok拥有一个小型企业，主要业务是承办当地活动的酒席。一个大型餐饮公司通过区块链追踪并确认了他大部分的客户身份。他们联系了这些客户，承诺将以更低的价格（95折）提供同等服务。如果Kyung-seok使用的是门罗，那么它的客户信息将不会被竞争对手窃取。**

- 歧视：Ramona公司附近的小区找到了非常理想的公寓。她每月通过比特币支付房租给房东。然而房东注意到，这些比特币来自于在线博彩。因为他本人对于赌博非常鄙视，所以决定不再续租给Ramona。如果Ramona使用的是门罗，那么房东也不会因为对其合法资金收入有歧视，而中断她的续租。

- 交易安全/隐私：Sven卖了一把吉他给陌生人，并把自己长期的储币地址作为收款地址给了对方。这个陌生人看到地址上仍有价值不菲的比特币后，便持枪抢劫了Sven。如果Sven使用的是门罗，那么资产便不会暴露出来。

- 被污染的币：Loki通过在线出售一些自己的艺术作品，攒钱上大学。当他攒够了学费并支付后，竟然收到了来自校方的“支付无效”的回复。Loki有所不知，他收到的一笔比特币，竟来自去年交易所被盗的那批比特币。而校方拒绝所有来自黑名单上的“被污染“的比特币，所以不接受Loki的支付。Loki只能面对这个窘境：他的比特币已经从账户中转出去了，但是校方却拒绝认可这笔付款。如果Loki使用的是门罗，货币可互换性将使他免陷于这样的窘境。

从以上的例子可以看出，门罗的隐私性将使用户免受监视、污染币和不道德的商业行为带来的负面影响。但是，所有的加密货币都还属于新兴科技，尚无加密货币可以做到“完美的隐私性”。

当你面临一笔生死攸关的交易，任何加密货币都不能保证万无一失。

1.3.3 门罗：开源去中心化社区和软件

门罗是一个由全球各地的密码学和分布式系统专家积极开发运维的开源项目。这些开发者自愿将他们的时间贡献给门罗项目。其余的项目是由门罗社区资助的，以保证他们可以全身心地投入在门罗主体项目上。

T门罗去中心化的开发团队相较统一的公司或组织，有几点优势。因为其网络和团队遍布全球，门罗不会被任一单个国家关停，它比任何单个个人和机构都有更强的生命力。

开源，指的是源代码（软件蓝图）对外公开，可以被任何人查看。相反地，闭源指的是开发者只发布最终编译好的产品（二进制文件比如.exe），这类产品的代码不公开，也无法被研究。如果你使用闭源软件，你就必须信任开发者和发布者。问题是，即使一个怀着善心的开发者开发的软件，也会有被黑客发现并利用的漏洞存在。我建议各位只使用经过独立第三方审计过的，并且证实为无恶意代码，意外错误和执行缺陷的开源加密货币软件。

加密货币社区从一开始就以拥抱的姿态接纳开源软件：比特币最初是以白皮书（white paper https://bitcoin.org/bitcoin.pdf）和开源、社区共建代码的形式面世的。这与决策结构集权且不透明的（政府背书）法币截然相反。当然，开源理念的历史远比加密货币的历史要悠久。在过去25年中，超过5000位开发者对开源的Linux内核做出贡献，而LInux被认为是最安全的操作系统之一。

对于任何加密货币来说，开源的可信任和安全性是至关重要的。门罗即是完全开源的。开发者们使用GitHub（https://github.com/monero-project/monero）进行版本控制，使所有人都可以方便地查看那些被建议添加、移除和修改的代码。至今，有超过240位开发者参与到门罗代码的审核、测试中。这大大降低了门罗代码错误被忽视的可能性。在第6、7章中，开发者可以获得更多与门罗代码库交互的信息。

开发团队透明度对于获得社区信赖来说，是非常重要的，特别是在加密货币领域。门罗开发内容的讨论，可参见openIRCchannels,频道（https://web.getmonero.org/community/hangouts/），门罗项目网站Monero Project website （https://web.getmonero.org/）和会议记录public archives of meeting logs（https://web.getmonero.org/blog/tags/dev diaries.html）。

1.3.4 门罗的历史

在2013年，Nicolas van Saberhagen发表了“CryptoNote”协议。日后，这份协议有多个加密货币实现，第一个实现是Bytecoin。如同发明比特币的中本聪一样，Bytecoin的创始人以匿名的身份，通过Bitcointalk来宣传Bytecoin。

仔细地审查Bytecoin，你会发现它在某些方面似乎很可疑。Bitcointalk上的成员“thankful_for_today”，调查了它的货币发行曲线，注意到了大概82%的货币早已经发行出来，并且很有可能非常集中地被持有。

最终，Bytecoin贪婪的预挖导致它的公信力丧失，实用性大大降低。好在，thankful_for_today认识到了CryptoNote的宝贵之处，并将其注入一个有着强大的开发团队，并且由社区驱动的新项目：门罗。门罗由先锋thankful_for_today构建，在2014年4月发行。它本来的名字是“BitMonero”（在世界语中意为“比特币”），不久就在社区的投票决议下改成了“Monero”（在世界语Esperanto中意为“币”）。

1.3.5 伦理 讨论

门罗在精妙地设计下，具有了如货币可互换性和交易隐私性等特点。这些特点对于所有的货币（加密货币或其它）的使用都是必要的。如“门罗的实际生活用例”这部分所描述，当今的金融系统无法保护用户的隐私，因此出现了很多问题。

可惜的是，门罗的这些特点，也被用于掩盖违法及不正当活动。这显然不是门罗的设计初衷。实际上，任何一种货币都存在着这样的问题，因为金钱的概念早已在数千年前孕育出来了。并且，使用加密货币进行非法交易的规模，

跟法币非法交易的规模相比，仅仅是九牛一毛，例如欧元，卢布，日元，美元。

门罗挖矿被设计成与电脑、手机、平板和网页浏览器兼容，这使得参与门罗挖矿的硬件门槛变得极低。不幸的是，黑客利用了这一点，开发了可悄悄调用他人计算资源来进行门罗挖矿的恶意网站和软件。这种未经他人允许的挖矿是资源上的一种窃取，对此，门罗社区最近成立了一个专门帮助“受害者”的志愿者团队：恶意软件响应工作组（Malware Response Workgroup： Malware Response Workgroup 他们提供反恶意软件和勒索软件的知识教育，工具和在线支持。

《精通门罗币》的作者们对于货币在个人、零售和商业上的广泛应用感到非常兴奋。我们希望读者们能够道德地使用门罗，频繁地使用门罗！你可以探索接受门罗币支付的在线商城Project Coral Reef（https://www.projectcoralreef.com）。如果你希望无偿贡献算力，并将挖矿收益捐赠给非营利性的慈善组织，欢迎使用：UNICEF AustraliaUNICEF Australia（澳大利亚联合国儿童基金会：https://www.thehopepage.org/）、Bail BlocBailBloc（https://bailbloc.thenewinquiry.com/）和Change.org.（https://theminingscreensaver.com/）。

起步 – （接收、储存和发送门罗币）

上一章，我们阐述了为什么要使用门罗，在这一章，你将了解如何使用门罗。具体来说，第二章会讲述接收、存储和发送门罗币这些实用技能，至于复杂的技术细节，我们会留到之后的章节。

在一开始，我们会接触一些与门罗钱包和软件相关的关键概念和术语。之后，我们将手把手教你使用门罗官方的免费开源软件：门罗命令行界面（CLI: command line interface）和图形化用户界面（GUI: graphical user interface）。

2.1 什么是钱包？

在你收到门罗币之前，你必须计划好在哪里 接收和存储门罗币。这时候你就需要一个钱 包，来存储和花费门罗币。你的纸钞（比如 欧元和美元）可以存储在多种实体钱包中。同样地，你也有多种门罗钱包可以选 择，并进行资金的转移。

钱包会帮助你处理那些复杂的加密进程，

所以使用钱包并不需要任何高深的数学知识。你只需要保管好你的种子密语（seed）和地址，学会如何使用钱包的功能即可。至于公钥、私钥等钱包背后的技术细节，我们将在第五章开始涉及。

你的门罗种子密语，是钱包用以定位和花费你的门罗币的一串秘密数字。为了方便起见，这串数字被转化成12~25+个词组，它就像一张藏宝图一样，能够帮你在区块链中找到属于你的资产。所以，你必须极其小心地生成和保存你的种子密语。请不要在咖啡店这样的公共场所设置钱包，因为可能关键信息会被他人或摄像头记录下来。以电子化的形式（如保存在文本文件或电子邮件中）存储种子密语也是危险的，因为恶意软件或服务可能窃取到它，并转走你的门罗币。

种子密语生成地址，以供你接收门罗币。就像你的邮箱地址一样，你把地址分享给要向你寄东西的人。大多数钱包有两种形式来显示你的地址：① 由字母和数字组成的字符串；② QR码。你可以选择任一一种形式，将地址分享给他人。

如果你的门罗钱包的实体部件损坏了，你只需要在一个新的钱包中导入种子密语即可恢复钱包。只要你持有种子密语，你就随时可以获得的资产。如果你丢失了种子密语，那么你就永远丢失了这些资产。你可能知道密码丢失后，可以通过管理员来重制密码。**但是种子密语不是密码**，没有人知道你的种子密语，

也没有人可以在不凭借种子密语的情况下，将你的资金转移到新的账户。

大多数软件都会建议你在初始化钱包的时候记录下种子密语。但有些软件没有这样的提示，你就必须主动完成这个步骤，并确认好备份功能。请在一开始就立即做好这件事情，否则一次损坏将造成永久性的资产损失。

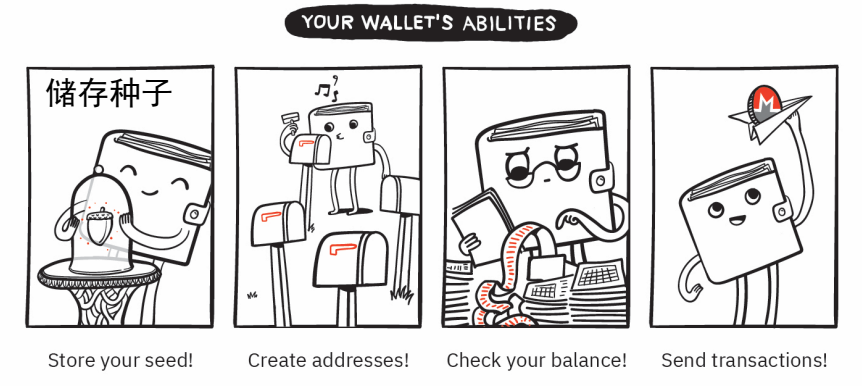


图2.1：钱包的主要功能：存储种子密语，生成地址，查看资产余额和发送交易。

2.2 选择最适合你的钱包

在这一部分，你将接触多种不同的钱包。

通常你将大部分法币资产（如欧元或美元）存储在银行或保险柜里，只随身携带一部分现金做日常用途。同样地，许多人也选择采用两种互补类型的钱包来存储加密货币：① 便捷的热钱包来存储少量的资金做日常用途；

② 更安全的冷钱包来做长期或大额资金存储。

市面上有不同的存储方案，它们在便捷性、隐私性和安全性上各有不同。你的需求决定了最适合你的钱包。接下来我们先看下，这些钱包是如何以不同方式存储种子密语的。

2.2.1 软件和移动钱包

软件钱包（桌面端或移动端）非常便捷。许多门罗用户在手机上安装热钱包，以用于日常支付。一个最佳实践，就是只随身携带和日常现金等值的加密货币。因为软件钱包将种子密语存于你的设备中，一旦你的设备中毒或被安装了键盘记录程序，你的门罗币将被盗走。

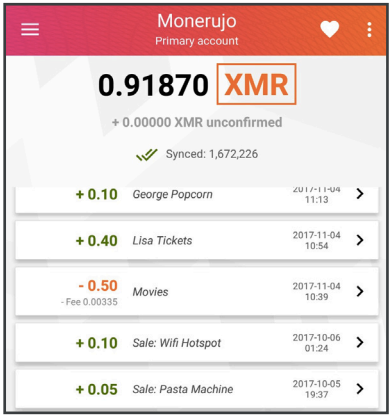


图 2.2.a - Monerujo (安卓系统门罗轻钱包)

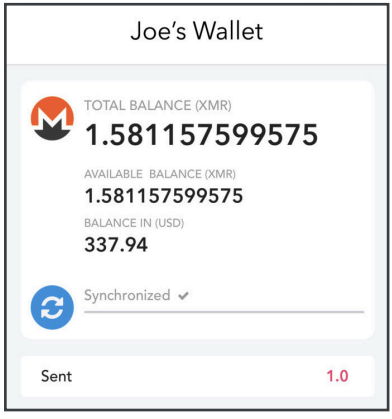


Figure 2.2.b - Cake wallet (iOS 苹果系统门罗钱包)

2.2.2 硬件钱包

硬件钱包是具有机密钱包功能的硬件设备，和手　机或电脑断开连接后便完全隔离。硬件钱包拥有　配套的屏幕以显示你的种子密语和交易信息，不　需要将这些信息传送给外部设备。

虽然硬件钱包不如软件钱包便捷，但是它们非常安全！即使你怀疑或确定一台设备已经被恶意软件感染了，你仍然可以使用硬件钱包从这台设备上发起交易，因为你的种子密钥是存储在硬件钱包中，且以非常安全的方式被保护着。目前，门罗社区的正在开发他们的第一款开源硬件钱包：Kastelo。

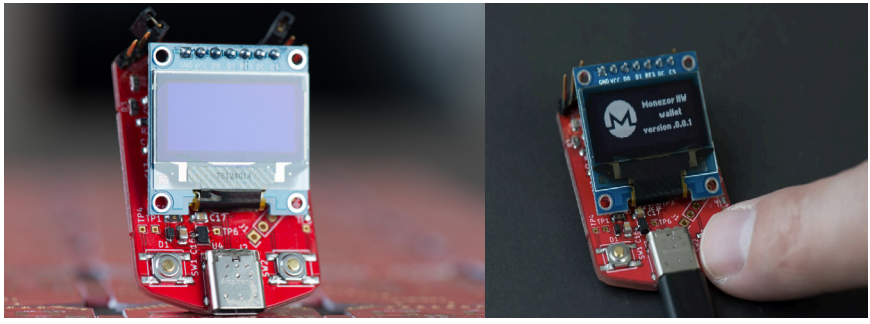


图2.3：门罗的开源硬件钱包Kastelo

2.2.3纸钱包

如果你不经常转移你的门罗币，那么纸钱包是非常低成本的存储方式。你只需要将你所需的信息（可公开的或私密的）打印在纸上即可。因为你的种子密语只存储在纸上，所以你不需要担心被遭遇病毒或数据泄露。不过，纸钱包不适合频繁的资金转移，因为你每次都必须将纸上的信息录入设备中。

图2.4：纸钱包是门罗密钥的纸质备份。一定要确保没有其它人接触这些信息。

2.2.4 网页钱包

你可以通过第三方提供的网页来登录你的门罗钱　包。这些在线钱包极其便捷，但这是在安全性和　隐私性上有所牺牲换来的。网页钱包分为两
种，核心区别是作为用户的你，知晓或不知晓你 的种子密语。

第一种网页钱包将你的资金存在他们自己控制的账户中，只给你提供一个登录用的账户名和密码（注意：你在交易所里的钱包也是如此）。实际上，你不拥有这个账户的种子密语，所以你无法掌控个人资金。你必须信任这个替你保管资金的机构。因为他们实质上提供的是银行服务，所以你必须十分谨慎。不管是由于意外或者盗窃，他们随时都可能遗失你的资金。一旦他们的网站关闭，你的账号密码也没有任何用处了，因为你没有种子密语，就无法掌控自己的资金。

第二种网页钱包将种子密语和资金交由你自己保管。诸如MyMoneroMyMonero（https://mymonero.com/）这种设计优秀的钱包，使得你可以在不将自己的种子密语发送给第三方的情况下，安全地访问账户。因为你的种子密语没有保存在设备和服务提供商那里，所以每次登录时，你都必须输入它。这一类型的网页钱包（相对来说）更安全，因为第三方不掌控你的资金。他们只是为浏览器提供一个你与账户交互的界面。即使这个类型的钱包网站无法再访问，你也可以在其他钱包中输入种子密语，重新访问你的账户。

虽然网页钱包使用起来非常便捷，

但是以上两种都不建议作为长期存储或大额资产存储的选择。两者都在安全性方面有所不足（需要信任第三方/需要频繁地在浏览器中输入种子密语），并且在隐私性方面也有所牺牲。

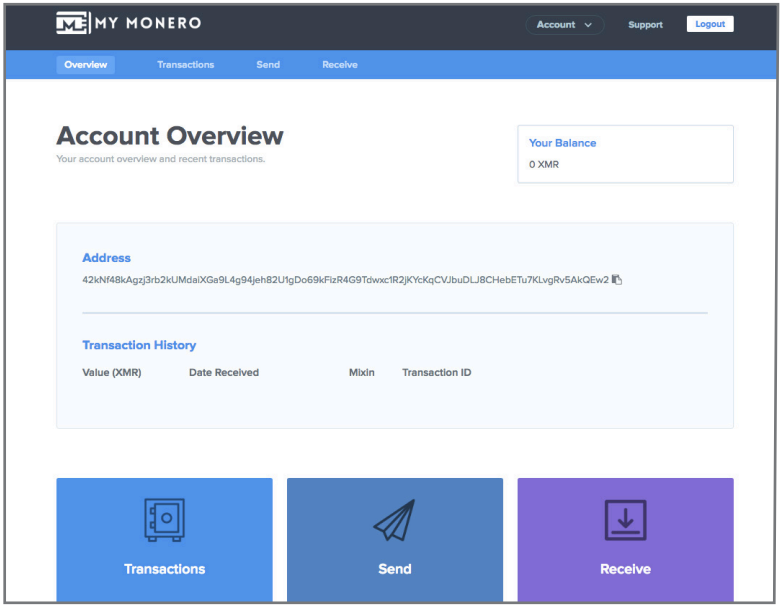


图2.5：MyMonero交互界面

2.2.5 冷钱包

“冷钱包”这个词是对纸钱包和其它离线存储方式的统称。例如，只有需要使用加密货币时才启动和连网的电脑或手机，是一种电子冷钱包。这个设备可以使用任意操作系统，但请务必采用高规格的安全措施（比如防火墙、杀毒软件和只访问/使用你信任的网站/软件）。种子密语保存在电脑上，所以请尽量保证它尽可能地“与世隔绝”。

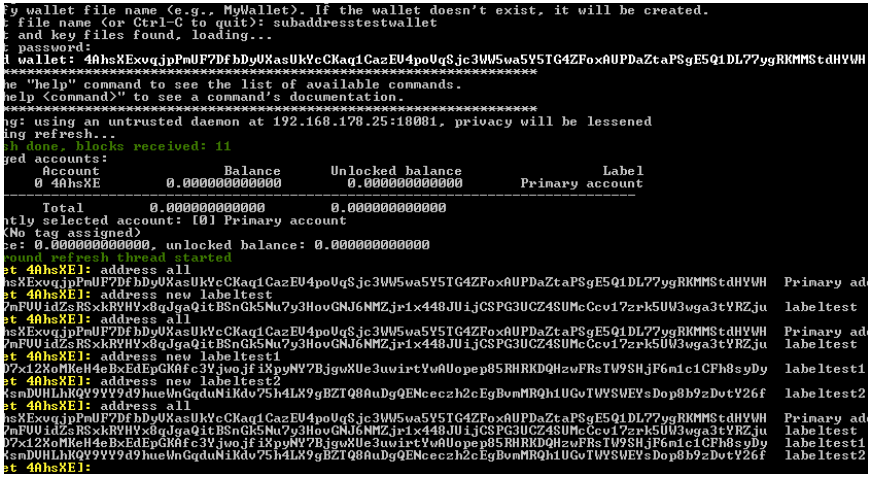


图2.6：冷钱包指的是专门用于加密货币的存储和汇款的设备。上面这台运行门罗CLI的电脑就是一个冷钱包。

2.2.6 门罗钱包链接

不管你选择哪种钱包，请务必使用来自可信渠道的经过审查的软件，因为钓鱼软件和诈骗钱包数不胜数！如果你在一个恶意钱包中输入了种子密语，那么当你还没反应过来的时候，你的门罗币就已经消失无踪了。

以下包含一些社区开发或信任的开源钱包的下载地址。

轻钱包：

- Monerujo - Android安卓系统
- Cake Wallet - iOS 苹果手机系统
- Mymonero.com - 网页端钱包, 微软电脑端, 安卓手机端 and 苹果手机 iOS

官方软件:

- 图形化用户界面 (GUI) - Windows, Mac and Linux
- 命令行界面 (CLI) - Windows, Mac and Linux

2.2.7连接至远程节点（可选）

如果你想要缩短同步时间和减少磁盘空间的占用，那么你可以选择连接至远程节点，而不必在设备上保存完整的区块链。大多数钱包，比如上述的轻钱包，都自动设置为连接到某个默认的远程节点。如果你想要手动设置远程节点，你可以使用node.moneroworld.com（端口/port 18089）这个社区网站上的资源。

节点是指那些下载了完整区块链的电脑，

它们还可协助其他用户同步钱包和中继交易。运行本地全节点不仅是保护隐私的首选，也可以通过共享全节点来帮助他人维护网络的安全。如果你不想下载全节点，或者想快速使用门罗钱包，那么你可以选择便捷的远程节点连接模式。

运行节点不等同于挖矿。挖矿过程是资源密集型的，我们将在第4章中涉及这个话题。只要区块链同步完成，那么运行本地节点就不会占用太多CPU或者网络资源。

2.3使用门罗币

这个章节将分享发送和接收门罗币的相关知识。本书中所有的示例都使用下面这个种子密语：

```
MASTERING MONERO DEMO SEED: lamb hexagon aces
acquire twang bluntly argue when unafraid
awning academy nail threaten sailor palace
selfish cadets click sickness juggled border
thumbs remedy ridges border
```

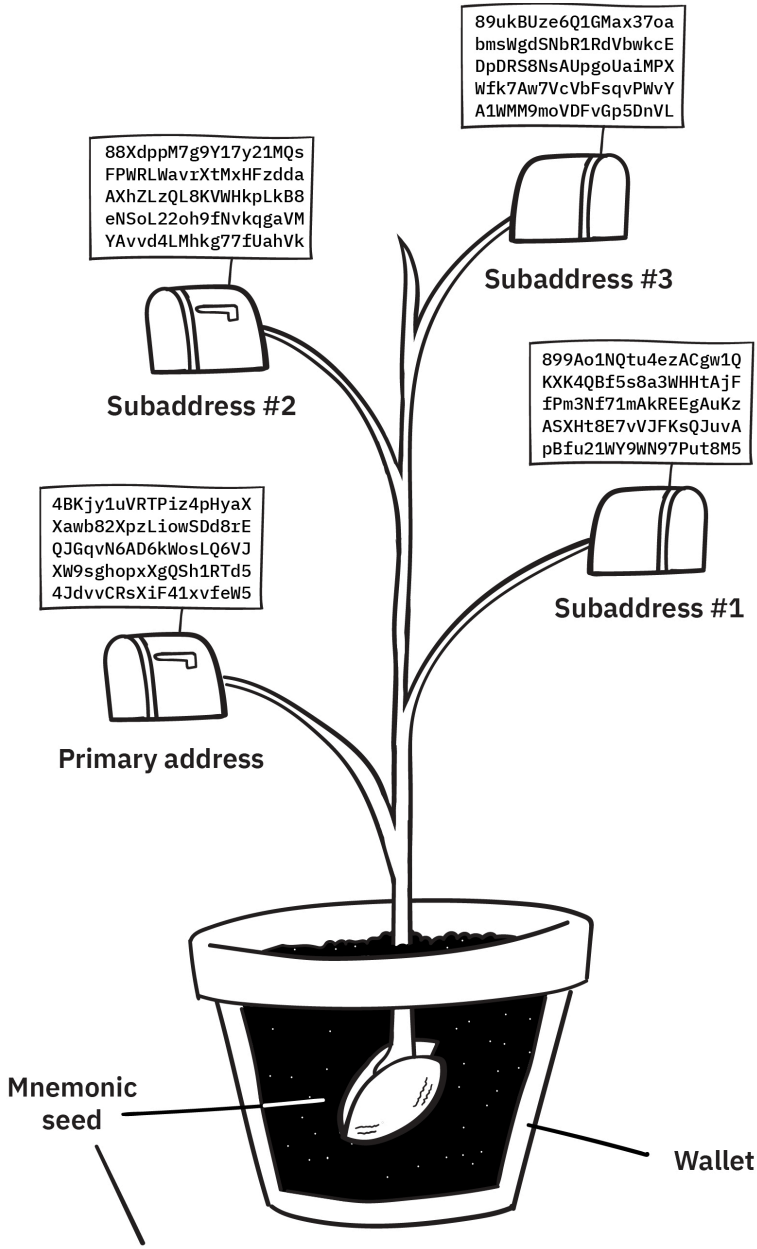


图2.7：钱包通过种子密语生成用以接收门罗币的地址。

你可以导入这个种子密语来练习生成地址，检查历史交易记录和验证交易。你可以使用这个种子密语练习本书中的所有操作，但是请不要往该地址上发送门罗币！因为任何阅读本书的人都可以花费它！

2.3.1 接收门罗币

接收门罗的方式很简单，你只需将你的钱包地址分享给向你发送门罗币的人。大多数的钱包将使用两种格式来显示你的地址：一个数字和字母组成的字符串（方便复制粘贴），和QR码（便于用摄像头扫描）。下面是使用示例种子密语产生的两种格式的地址：

Your address for receiving Monero can be represented as a text string, or QR code. You can share whichever is more convenient. In the example, we have 4BKjy1uVRTPiz4pHyaXXawb82Xp-zLiowSDd8rEQJGqvN6AD6kWosLQ6VJXW9sghopxXgQSh1RT-d54JdvvCRsXiF41xvfeW5.

你所分享的这个地址永远不会存储在区块链上（这得归功于门罗的隐蔽地址[Stealth Address]功能，我们在第三章会进行它的概念性讲解，第五章将深入其技术部分）。你可以在门罗中使用一个种子密语生成多个子地址(subaddress)，这些不同的子地址收到的汇款都将存在同一个钱包。

每一个门罗账户都有一个主地址 (以“4”开头)。考虑到使用的便利，我们可以生成无数个子地址 (以“8”开头)。打到这些地址的资金都将存在钱包的主账户中。你在第五章中可以习得如何管理多个地址。

当别人向你发起一笔汇款，钱包可能需要等待10~20分钟的确认后，才标记你收到汇款，并可以花费它（第四章会讲到为什么这样设定）。这是一个非常普遍的安全措施， 钱包也会显示待确认的交易。如果你的钱包在等待一笔0.06 XMR交易的确认，你会看到如下信息：

账户资产：0.075 XMR，可使用资产：0.015 XMR）

发生这样的情况请不要担心。半小时之内，0.06 XMR的交易将会被确认，并变成可用资产。

门罗还将提供一种“只可查看”(view-only)的模式，在该模式下，你可以查看所有的收款信息，但是无法查看汇款信息，也无法发起交易。这个功能有很多重要的使用场景：使慈善捐款完全透明，向授权的审计机构提供财务信息访问权，开发有限权限（只可查看收款信息）的监管设备。启用“只可查看”模式需要用户对外分享查看密钥(secret view key)，它与种子密语不同。这个话题将在第五章中讨论。

2.3.2 发送门罗币

发送门罗币也很简单，你只需输入/扫描收款方的地址和想要发送的金额。点击“发送”就可以发起交易了。

如果你向一个商家发送门罗币，他们可能会要求你提供交易ID (Payment ID)，以关联你的订单。如果你向你自己或朋友发送门罗币，你可以选择不填写交易ID。有些服务使用"集成地址"（integrated address)，它将交易ID和地址结合成一个字符串表示，既更方便又更保护隐私。

在2018年，门罗添加了新功能，使得每个钱包都可以生成大量的子地址来收款，因此交易ID和集成地址的使用率大大降低。传统的做法是向每一个顾客提供相同的地址和不同的交易ID，但现在商家们只需给顾客一个唯一的子地址即可（这样做法更加直接，使用也更少出错）。

任何使用免费的OpenAlias系统的人，都可以提供便于阅读的门罗地址 (例如"donate.getmonero.org")来代替原始字符串地址(44AFFq5k……)。两种地址只是格式不同而已，不管把门罗币发往哪种格式，都没有差别。不过，设置新的OpenAlias地址是一个技术活，我们会在第七章中介绍。

你的钱包会设置一笔小额的交易费，用以回报那些广播和处理你交易的网络。你的钱包会根据当前门罗币网络的负载，你的交易紧急度和一些其它因素来推荐一个合适的交易费。你可以在第四章中了解交易费是如何设定的，以及它们对于网络维护的必要性。

2.3.3 如何查证交易

考虑到门罗币的隐私性，你也许会

疑惑一个人如何证明他成功发送了一笔交易。除了交易ID这个选择外，门罗还有一个功能可允许用户性选择性地证明交易发送成功。这个过程需要用户分享只有它自己才可以生成的交易密钥 (transaction key)。

案例

假设你朋友 Khan 和 Maria 每个人欠你 `0.06552376` XMR 的餐费. 你只收到了一笔交易，如下图所示:

Amount（金额）: `0.06552376` XMR
Transaction ID （转账ID）:
`4b540773ddf9e819f0df47708f3d3c9f7f62933150b90edc89103d36d42ca4b7`

Received to (your) sub-address（收款子地址）:
`899Ao1NQtu4ezACgw1QKXK4QBf5s8a3WHHtAjFfPm3Nf71mAkREEgAuKzASXHt8E7vVJFKsQJuvApBfu21WY9WN97Put8M5`

这是DEMO钱包中在2018年4月20日收到的一笔真实的汇款。你可以在区块链浏览器blockchain explorer,中查看一些相关信息，但是无法知道汇款发送者的身份。Khan和Maria都说自己已经汇款了，这时你可以要求他们提供交易密钥：

Khan:
`OutProofV1N4Y5pUJEnRACJyB5C3zK1zTqAihdnN8MfVZhEWfD13Z2N7Npt1uxa1EY7N7jnvuJF76tXUwKrakvZSxTj4Zux5SpavFb4X1jRcLAJ2b5hqviQPiS58j2qH53QL44CJEgHtY5`

Maria:

```
OutProofV1To53Qu2gegZbUevosKCTwrEdqiECgFyUygutX-
MEdhrHg1EtXMrFNaszWYFjdU4aXFZ2iPF8G8jzoDJzCoW5d-
sWvb4mVN65abAya3U47cGXs7WABrTzG5aPfV4YBANhwPgwD2
```

当你查看两者的交易密钥（transaction keys）时，你发现Maria向你交易发起的汇款已被确认，而Khan的交易因为“无效签名”（bad signature）被退回。你可以使用上述地址和交易密钥练习该操作。

2.4 操作安全

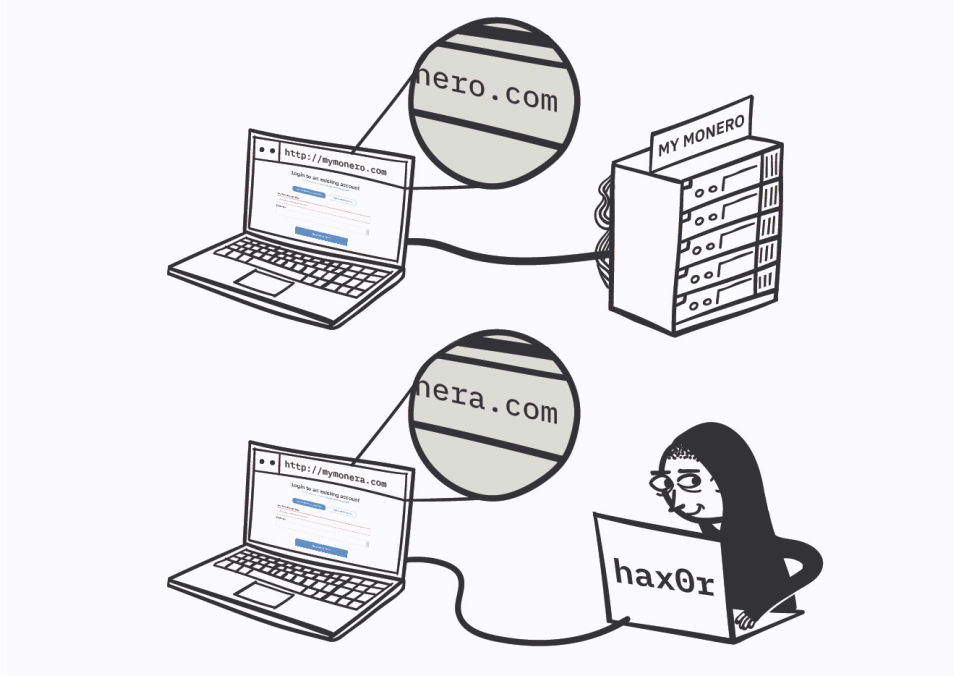


图2.8：钓鱼攻击 (Phishing attack) 通常使用一个非常相似的网址(URLs)，试图让用户以为自己正在访问正确的网站，从而骗取用户的种子密语和密码 (如钓鱼网站www.\mymonera/.com，试图伪装成www.mymonero.com)。请务必仔细检查网址，特别是那些可直接链接的网址。

门罗币是你自己的银行，隐私除了你以外，没有人可以控制你的资产。这一带有草根理念的金融赋权是加密货币带来的最大福音之一。能力越大，责任越大。保证操作安全也是保证你和资产的安全。

2.4.1 永远不要透露你有多少门罗币

“祸从口出”这样的话是有道理的。当你在公共场合讨论你有多少门罗币的时候，你就无意中让自己成为诈骗或盗窃的目标了，这在网络论坛和社交媒体中尤其普遍。

诈骗者和盗窃者潜伏在网络上，寻觅着那些泄露了个人资产信息的人作为目标。

大多数都知道不应该在社交媒体上晒自己的银行账户或退休金投资组合。这种做法不仅不安全，也很粗鲁，甚至还可能将人机关系变得非常尴尬。然而，很多人却经常晒自己拥有的比特币或门罗币。

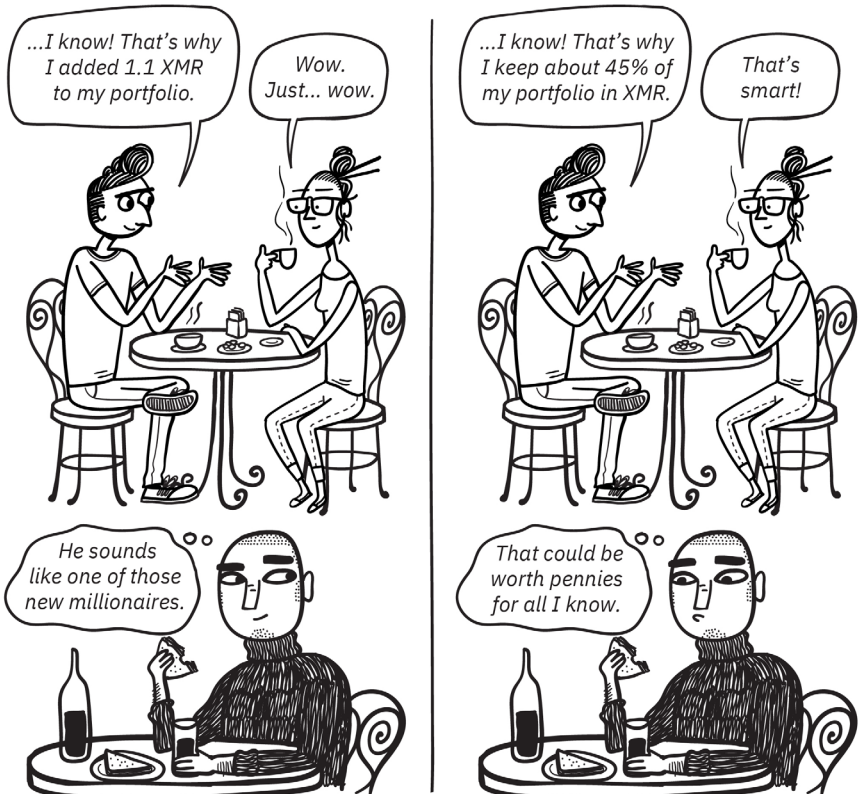
请记住加密货币价格波动性强，可能出现戏剧性的暴涨。一个2012年的帖子“我只花了50美金买比特币”当时看起来没有什么特别的，但到了2017年末，不到5年的时间里，50美金已经变成上百万美金。信息一旦留在网络上了就很难被清除干净，所以避免这种情况的最好方式就是永远不要泄露你的资产信息。

鉴于加密货币投资意愿很普遍，会有很多关于持仓和投资组合的对话。请记住，永远只用百分比而不是绝对金额来表达。以下图表显示如何计算你的资产组合，以便你在与他人讨论投资策略的时候不会泄露敏感信息。

PORTFOLIO

Cryptocurrency	Quantity		Exchange Rate		Value		Percentage
ABC	0.1	×	1150€	=	115€	=	23%
XMR	1.1	×	200€	=	220€	=	44%
WYZ	3300	×	0.05€	=	165€	=	33%
					500€		100%

HOW TO TALK ABOUT IT



2.4.2保管好你的种子密语

种子密语安全的话，资产才安全。种子密语主要有2个风险：①意外丢失②失窃。

为了防止意外丢失，请务必对种子密语进行备份并保存在安全的地方。问自己:"如果我的手机坏掉了，网站崩溃了，我还有办法访问我的资产吗?" 你应该考虑进行额外的备份，并放在另一个安全的地方。你不会想因为住宅失火或遭遇洪水而损失掉设备和备份的种子密语吧。

为了防止失窃，请永远不要与他人分享你的种子密语或密钥。任何拥有你的种子密语的人，能窃取你所有的资产，并且门罗的隐私性使你无法确定资产的去向。

2.4.3 交易相关的安全措施

当你往一个新的个人或交易所地址发送大额资产时，请先使用小额资产来测试下，对方是否成功收到汇款，该收款地址/服务是否可行。这是一个非常重要的习惯，可以帮助你提早了解可能的意外，记住，加密货币的世界里没有“撤销”这个选项。

对于每一笔加密货币交易，都要重复确认地址是否正确。即使你是复制&粘贴，也要在确认下地址被完整无误地拷贝过来。黑客已经开发出在剪切板中可以将地址偷换为攻击者的地址的恶意软件。

如果你做好了重复确认工作，就不用白白给黑客“捐款”了。

2.4.4 交易所安全性

交易所为你创建钱包，但是通常不会把种子密语分享给你。这是有风险的，因为如果交易所被攻击、关闭或发生其他意外情况，你没有办法找回你的资产。加密货币领域的一句名言是““Not your keys? Not your Bitcoin!” ”(密钥不在手，这比特币就不是你的)，特指那些掌控着你的密钥的钱包和服务商，他们掌控着你的资金。

在2.4.2中我们说过，永远问自己:”如果我的手机坏掉了，网站崩溃了，我还有办法访问我的资产吗?" 一个黄金原则是除非你短期内想交易门罗币，不然就都转移到你所掌控的钱包中。

2.5 门罗币与商业

2.5.1 门罗币是商家的理想选择

在这一章，我们谈及了门罗币的通用使用技巧。这一部分将介绍一些方便商家在自身的系统和服务中集成门罗的工具。如果你没有从事商业支付，你可以直接跳到下一章。

接受门罗币支付的商家可以享受到快捷、保护隐私和不可逆转的交易。

有几个工具专注于提升门罗币的线上/线下接收体验。

当然，你也可以使用前面部分的知识来创建钱包、接收门罗币。但是这一部分所涉及到的工具，可以实现商家的自动化交易集成，以及发票和收据的生成。

2.5.2 接收门罗币的工具

Monero Integrations 支付网关允许任意在线商城添加门罗支付选项。商家只需安装其中一种插件即可，这些插件是为几大受欢迎的内容管理系统而设计的。Monero Integrations的解决方案(方案发明人即本书作者）和门罗的理念一致：整个项目都是免费、开源open source、去中心化和保护隐私的。交易将直接指向你的钱包，所以你无需信任和依赖第三方来处理交易，因此在隐私性和安全性上不用做让步。

Kasisto是第一个接受门罗币的支付系统，它同时也是开源的，无需任何第三方介入。这个应用是为手机或平板上的应用内购买而设计的，它独特的交易探查机制使得它几乎可以即时接收汇款。你可以前往Kasisto GitHub.试用它的demo。

此外，GloBee也是一个选择。它允许商家可以接受加密货币和信用卡支付。GloBee是第一个三方公司，因此可以提供一些额外的功能，比如接受多种类型的加密货币支付，并即时转成门罗币、其它加密货币甚至是法币(如欧元或美元)。

这给了你接收门罗币并瞬间转成当地法币，以避免价格波动的选项。

如果你想了解更多代码层面的细节，或者开发自己的支付工具，你可以在第七章中学习后端的相关知识。

门罗是如何工作的

前

两章分别介绍了为什么和如何使用门罗币，这已经包含了使用门罗币所需的一切知识了。

本书剩下的部分，将为那些想要了解门罗背后运作原理的人提供更多的细节。第三章和第四章将讲述门罗隐私保护功能，区块链和挖矿的技术。我们重在概念性的讲解，不会涉及高级数学。剩余的章节将讲述本质和原理以供开发者和加密学极客参考。

3.1 交易和账本

作为讲解门罗隐私保护技术的知识铺垫，我们会先讲述如何在账本系统中接收和发送门罗币。这一章我们会专注于区块链的功能性——天生地可以保存交易且不可篡改的共享式数据库。第四章将会设计挖矿、哈希等话题。

当你第一次设置钱包的时候，钱包会生成一个新的种子密语。你需要将它秘密保存好，并使用它来支配区块链上的门罗币。这个初始化过程只在你设备中执行，它甚至可以离线执行，无需在网络中广播和记录。

具体来说，你的钱包从种子密语中计算出两组密钥。你的私钥（private keys）需要严格保密，因为它代表着你的身份和门罗币使用权。你的公钥（public keys），顾名思义，是可以让其他人知道的。公钥和私钥是按组生成的，两者之间有着特定的数学上的联系。

为了接收门罗币，你需要把地址（由公钥生成）发送给发送者。当有人（可以是顾客，交易所或朋友）向你发送门罗币的时候，他们将向网络广播一条交易：向账本上的一个新账户发送特定金额的门罗币，并且只有你的密钥才可以解锁这笔门罗币。

用技术术语来说，交易的输出（output）存储在区块链上，你使用私钥来访问和花费它。这个术语可能有点让人困惑，因为加密货币语境中的“输出”和通常我们所说的“输出”意思不一样。

每当你收到门罗币的时候，你其实是获得一项输出；每当你花费门罗币的时候，你消耗了一些输出，并创建一条新的输出给他人。实际上，你所“拥有”的门罗币，其实都是区块链上你可以解锁的输出。除非有人向你发送门罗币，不然你在区块链上没有可以解锁的输出。

你的钱包扫描或同步的过程，实际上是用你的私钥在区块链上检查所有的交易和输出，辨别相关的账户。你的账户资产是你可以解锁和花费的所有输出的金额总和。

当你从钱包中发出门罗币的时候，你消耗了一些输出，并将其转变为交易的输入广播到网络中。区块链可以简单理解为这些交易的记录，每一条交易将发送者的输出消耗为输入，并为接收者创造新的输出。

以上部分对于核心概念（公钥/私钥、交易、输入/输出）的解释可能略显简单。接下来我们将对门罗的核心隐私保护功能进行非技术性的讲解。

3.2 隐私保护技术一览

图3.1：该图表示了门罗的各项技术如何互补地一同保护敏感交易信息

环形机密交易（RingCT）隐藏交易金额

环形签名（Ring signatures）通过混淆真实的输出来隐藏发送者身份

隐蔽地址（Stealth address）隐藏接收者身份· Kovri（注：一种独特的洋葱路由变种，但目前Kovri项目似乎已停止开发）通过混淆广播源头和隐藏门罗的网络活动痕迹，来阻止他人通过交易跟踪到你的物理地址

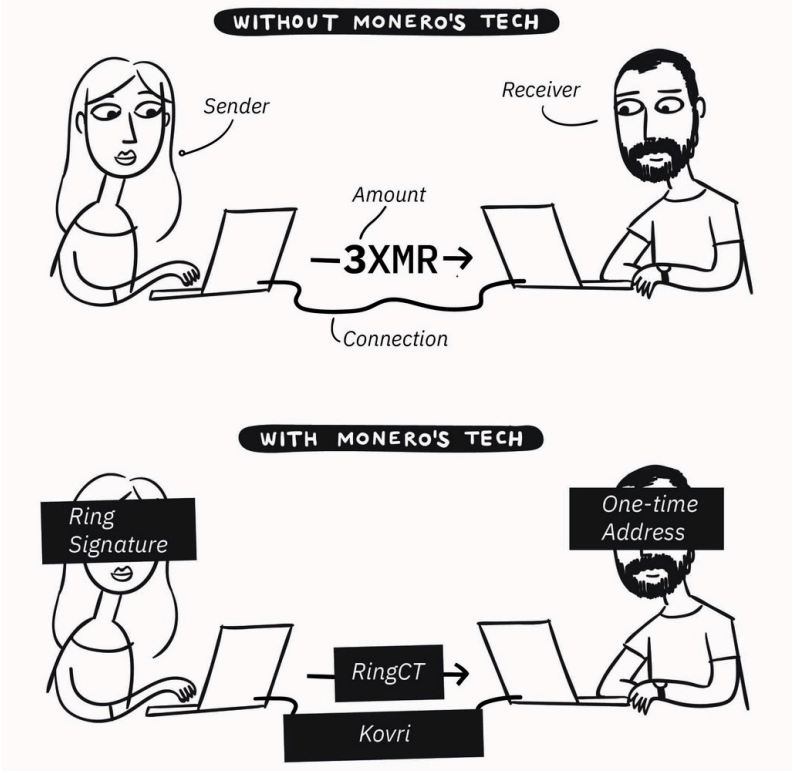


图3.1：该图表示了门罗的各项技术如何互补地一同保护敏感交易信息

3.2.1 环形机密交易

环形机密交易是一种用以隐藏任意交易的金额的密码学技术。对于大多数加密货币来说，任何人都可以查看交易的金额。其原理是发送者不再需要提供真实的交易金额，只需证明自身拥有足够的门罗币来发起这笔交易。这要归功于密码学中的承诺（commitments）和范围证明（range proofs）技术。

当你发送门罗币的时候，你以一种私密的方式来进行“承诺”：你只需透露部分的信息，足以让网络确认交易的正当性即可，

而不用透露具体的交易金额。一个有效的承诺保证交易无法被伪造，门罗币无法被双花。

范围证明是环形机密交易的另一项重要机制。它确保承诺的值介于0和特定值时间。这有助于防止发送者承诺一个为负或极高的值。承诺和范围证明结合起来，确保门罗币的供应真实，不会被伪造或超发。

在应用环形机密交易之前，门罗币的交易被拆分成特定面额 (比如12.5 XMR的交易会被拆分成10 XMR+2 XMR+0.5 XMR)，且金额对外界可见。环形机密交易自2017年1月激活后，迅速获得广泛的使用。激活后的1个月内，近98%的新交易都自愿选择使用环形机密交易协议！

为了贯彻门罗的强制默认隐私保护方针，自2017年9月开始，所有的门罗交易都必须使用环形机密交易。如需花费以往的非环形机密交易的输出，用户必须先转化成环形机密交易输出以隐藏金额。

3.2.2 隐蔽（一次性）地址

所有的门罗交易都使用隐蔽地址来保护接收者的隐私。为了避免接收者的钱包地址别记录在区块链上，每一笔门罗交易都将发送至一个惟一的、用完即可丢弃的、一次性的地址。接收者可以访问发往隐蔽地址的资金，并且不会暴露与钱包公共地址和其它交易的关联。

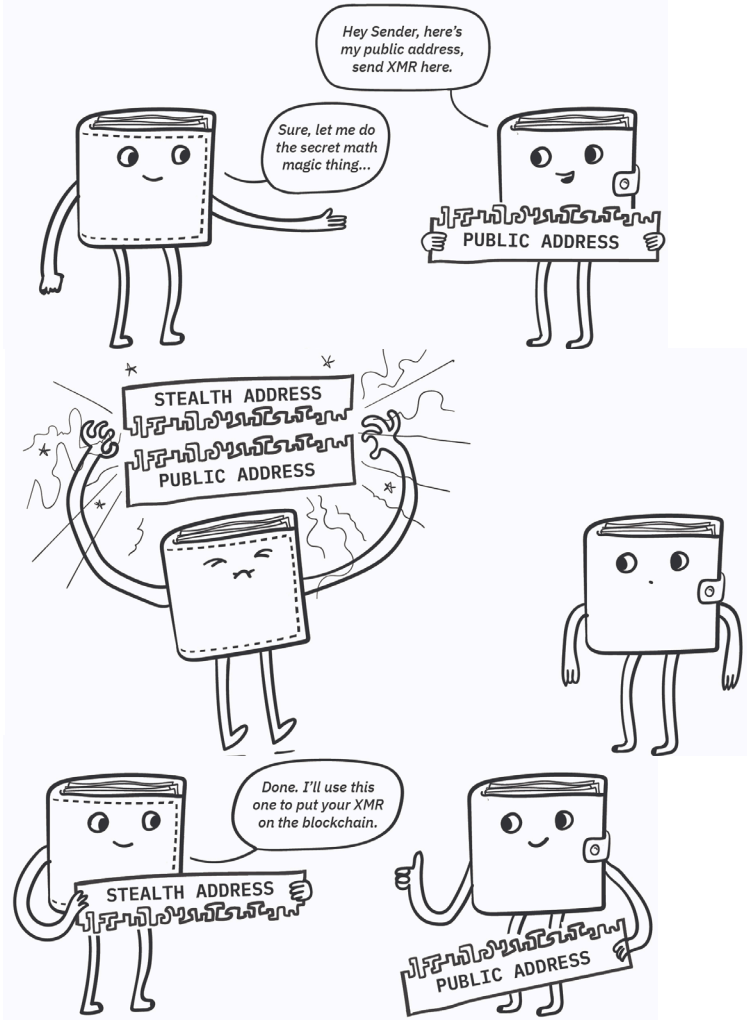


图3.2：门罗的发送者钱包将从接收者的公共地址中，生成一次性的隐蔽地址。交易和区块链中只会记录隐蔽地址。

为了方便理解随机一次性的地址是如何保护接收者的身份的，想象你想给你的朋友André几本有关于治疗某种敏感疾病的书。不巧的是，André下周才能过来，而你刚好有事要出远门。或许你可以考虑请邻居帮忙暂时代为保管和移交这些书。

你的邻居需要对任何前来认领的人进行验证，以确定他们是你预期的接收者。鉴于你朋友很注重隐私并且当前处在敏感阶段中，你不便直接告诉邻居接收对象的真实身份是André。那你怎么在保护André隐私的情况下完成这件事情呢？其实你可以给房东一个一次性的随机的密码，并且告诉你邻居说，只要对方出示这个密码，就把书给他（例如把书给出示"PolarComboTango357"）。如此一来，你的邻居便可以把书给到André，且不会泄露接收人的身份。

和上述例子相似，门罗也是使用一次性密码系统来防止网络获悉你的身份信息。门罗区块链不会公开记录接收者的地址（类似"把书给André)，资金总是被发送到一次性的隐蔽地址（类似"把书给出示PolarComboTango357的人"）。隐蔽地址的技术会在第五章详细介绍，以下是一些关键点。

这些一次性地址是如何生成的？你的门罗钱包公共地址四一个95位的字符串，其中包含了从种子密语衍生出的两个公钥（查看公钥 [public view keys] 和花费公钥 [public spend keys])。

当他人像你发送资金时，他们回到使用你地址中的公钥并与一些随机的数据结合起来，生成一个唯一的一次性公钥。这些记录在去看看了交易中的一次性公钥被称为隐蔽地址，因为网络或外人无法通过这些随机密码来追溯到原始的钱包。

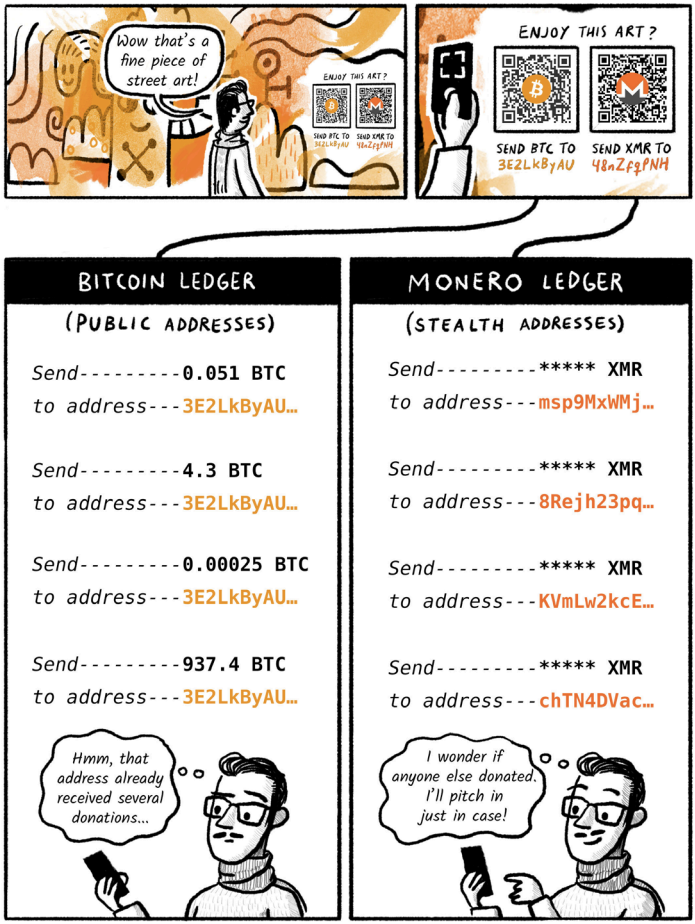


图3.3：在如比特币这样的透明区块链上，所有发送至同一公共地址的所有交易都很容易被查看和关联起来。门罗的隐蔽地址是无法关联，无法复制的，所以接收者的财务信息不会被其它用户（包括发送者）所知晓。

需要注意的是，子地址跟隐蔽地址不一样。子地址是可以重复使用的公共钱包地址，且不会被记录在区块链中。发送至单个子地址的所有交易都会指向不同且不可关联的隐蔽地址。

实施隐蔽地址的一大好处就是不会在区块链上记录钱包地址，从而增加隐私性。它的更重要的一个启示是，使用一次性唯一的密钥可以防止发往同一个地址的交易被关联起来。

假设你创作了一些公共艺术作品，并且贴了一个地址用以接收加密货币捐款。如果你使用的是透明区块链的加密货币 (如比特币)，那么每一笔发往那个地址的交易都会被记录下来，并且可以被搜索，可以被链接。任何使用区块链浏览器的人，都可以查看你接收到了多少比特币，以及这些比特币是否被转出。任何一笔汇入的交易都可在账本上通过地址进行索引。

如果你贴的是门罗币地址，你的捐款将不会暴露在公众视野中。每一个捐赠者都会生成一次性的唯一的隐蔽地址，并将该地址记录在区块链上。你贴在项目旁的公共地址不会和交易关联起来，隐蔽地址也不会提供任何接收者的信息。因为每一个捐赠者都在混入了独有的随机的信息以生成隐蔽地址，所以捐赠者无法识别其它捐赠者的地址。

所有的门罗交易都必须使用隐蔽地址，来保证整个网络的隐私性。你的钱包将会在生成交易的时候自动从公共地址中生成隐蔽地址。

3.2.3 环形签名

环形签名是门罗的一项通过混淆花费的门罗币来源，用以保护交易发送者的技术。在讲解环形签名之前，我们先介绍下数字签名。

数字签名是用以确保消息真实性和来源的通用密码学技术。人们可以使用公钥来对签名进行验证，以确认签名者的身份，验证签署的消息是否是完整、未被篡改。哪怕签署的消息被改动了一个字符 (无论是刻意的篡改或意外的通信错误)，签名都将显示无效。

加密货币很重要的一个部分就是数字签名的实施。为了花费你的输出，你需要创建一条消息。这条消息描述当下的交易，然后用你的密钥来对消息进行签名，再将结果广播至网络中。在执行交易之前，网络将检查签名的有效性，以验证消息是否被不拥有相应密钥的第三方篡改或伪造。

对于透明的加密货币（如比特币），每条消息都直白地指明被花费的输出是哪一个。这对于简易记账来说是很有用的，因为网络只需简单地维护好未花费输出（UTXO）的记录即可，这些未花费输出将作为是新交易的有效输入。如果有人试图对一笔比特币输出进行双花，第二笔虚假交易将会立即被拒绝，

因为网络知道这笔比特币的主人已经花费掉这笔输出了 (他们签署并广播了第一笔交易)。可惜，这种对资金来源和所花费输出进行确定性指示，直白地提供所有权证明的方式，不利于隐私保护。

门罗使用的是完全不同的技术：环形签名。这个群签名方式允许一个成员代表一群人对消息进行数字签名。在这个过程中，群里其它成员的公钥被混合在一起，因此无法判断真正的签名者是谁。在密码学上，我们只能验证有一个环成员签名了消息，但是无法知道是哪一个。

门罗的环形签名将多个输出混合起来，以混淆真实的被花费输出。假设Maria想要花费她的一笔门罗币输出。她的钱包将半随机地选择区块链上的几个（非Maria的）历史输出，并将这些输出混入环形签名中作为诱饵（decoys）。网络只能验证其中一个输出被花费，而诱饵和实际的被花费输出在密码学上无法被识别。

环形签名保护所有交易的发送者，因为接收者和门罗网络无法查明哪一个环成员是资金的真实来源。环形签名的一个显着效果就是让外人无法确定真实的被花费输出。外人也无法辨别任何一个输出到底是真实的被花费输出或诱饵。

鉴于此，你也许会疑惑，门罗币如何防止投机分子重复花费同一个输出？这对于像比特币这种一个输出对应一个花费的透明公开的区块链来说，不成问题。

然而，门罗的输出在花费之前和之后都可以加入到环形签名中，所以必须采取其他方式防止输出的重复花费。

这个将由密钥镜像（key images）来完成。密钥镜像只能够由真实被花费的输出生成，并且每一笔交易都会生成和记录它。网络无法确定与密钥镜像关联的环成员，但其他参与者只需检查这个密钥镜像之前是否被使用过。如果一个恶意用户试图双花一笔输出，

门罗网络最初并没有强制执行环形签名，所以导致了无诱饵混入（zero-mixin）的交易出现，危害了整体的隐私性。这些早期的交易和透明区块链中的交易拥有同样的结构和弱点，发送者和接收者暴露无遗，真实的被花费输出也容易被他人知晓。从2016年开始，网络要求每笔签名至少拥有两个环形成员，以推动发送者隐私保护默认化。在2017年晚些时候，环值（ring size）要提高至5，并在2018年年初提高至7。

需要注意的是，在2016~2018年间，环值规则只限定每次交易的最小混入数量，而并没有对最大值做限定，

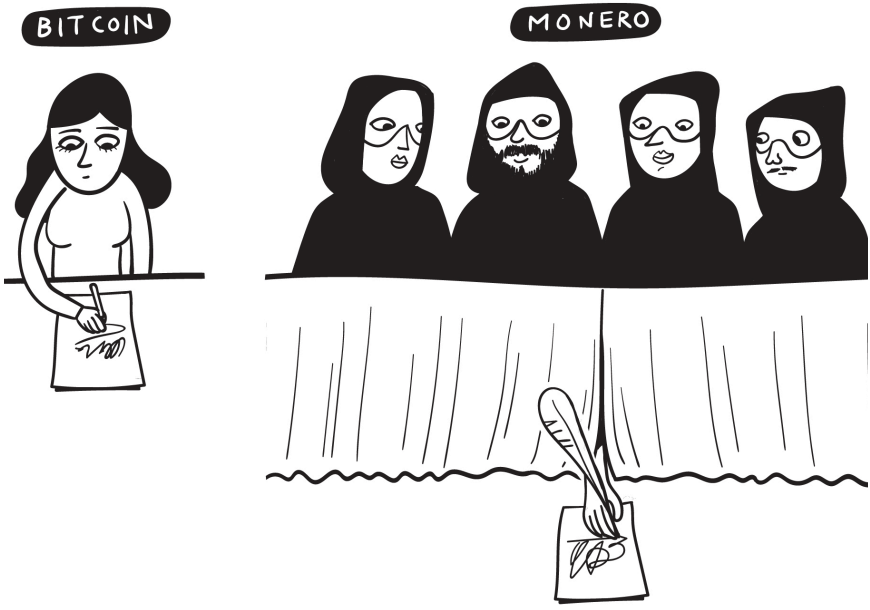


图3.4：每一笔门罗交易都由一群“环签名”授权，以保护发送者的隐私。

所以用户可以按自己的意愿混入更多的诱饵。理论上，大家可能会认为使用更多的诱饵可以获得更高的隐私性。然而，实际上有一个问题需要纳入考量：当大部分的交易都使用最小环值，使用大环值的交易就会显得特别引入注目，这其实恰恰不利于保护隐私。

这个问题在2018年的网络升级中得到了解决：网络不再限定环值的最小值，而是限定一个固定值。在写稿的此刻，所有的交易都必须使用环值为7的环形签名。随着威胁的统计模型不断地调整，最佳隐私保护实践也将不断优化，这个数值在未来也有可能会增加。

3.2.4 Kovri & 流量分析

任何连接至互联网的设备都会被分配一个IP地址，以引导流量流向正确的用户。但是，IP地址很容易与用户的物理地址和身份关联起来。

门罗币的交易等事件若和IP地址关联起来，将带来很多负面影响。对节点连接日志中的IP地址的分析，有可能会使我们之前提到的保护门罗用户隐私的部分加密学措施前功尽弃。

我们有必要考虑当门罗币网络活动和物理地址、身份相关联时会产生的不利局面。

在向门罗币网络广播的时候会显示IP地址，所以收到广播的交易的节点有可能可以确定交易发送者的IP地址。门罗的其它隐私保护措施在防止他人利用链上数据来追踪交易的时候，监视型节点如果观察到多笔交易源自同一IP地址，便可推断出他们之间的关联性。

除了隐私方面的顾虑外，IP地址的暴露也会引发潜在的监管问题。一个恶意节点可以选择不中继广播的交易给特定的个人或群体。更糟的是，对你怀有敌意的人可以通过IP地址暴露出来的地理信息，亲自登门拜访你。

IP地址和门罗网络活动的关联性不仅仅对广播交易的用户造成威胁：

通过网络服务提供商和其它监管方，可以看到经过节点的网络流量，这意味着如果当地的政府或网络服务提供商对加密货币比较反感，那么节点拥有者将受到很大的风险。

矿工也有可能因此受到不公正的对待：攻击者可以通过监视矿工的区块寻找攻击机会，或由于意识形态上的分歧，限制非政府或非公司的挖矿实体。

防止从IP地址中泄露网络活动（以及物理地址/身份）显然对门罗币生态的各个部分都有益处。如果你需要保护这方面隐私的话，目前有几种选择。

一个选择是使用虚拟专用网络(Virtual Private Networks: VPN)。用户通过VPN以加密的连接方式发送流量，使网络服务提供方和政府无法监视。其原理是在你和VPN服务器之间建立一条安全的通道，把你的流量和其它用户的数据混合在一起，并从另一个IP地址中广播出去。不过需要注意的是，向你提供VPN的服务方可能会记录你的日志（logs），所以请做好尽职调查，使用可信赖的服务。

另一个选择是使用洋葱路由(The Onion Router: Tor)。TOR使用由数个中继节点组成的私密网络来定向发送用户的流量。Tor起初是由美国海军实验室研发制成，专供记者、情报人员等需要避开监视和审查的人员使用。Tor是免费、开源、去中心化的私密网络，

其设计初衷是让网络中的任何人无法识别出任一广播的来源。你可以通过专用浏览器来访问Tor，对隐私性需求更高的用户，可以使用Linux的应用，如Whonix，它可以将所有流量都纳入Tor。

此外，门罗社区也支持Kovri的开发。Kovri是基于去中心化的隐形互联网项目(Invisible Internet Project: I2P)的一个专注隐私性的实现。Kovri使用加密技术和复杂的路由技术，创造一个分布在互联网中的私密网络，同时它也可供其它应用使用。

在Kovri或者相似类型的功能集成至门罗之前，对于网络流量分析有顾虑的用户，请使用Tor或可靠的VPN来保护你们的IP地址和连接隐私。

3.3 结语

门罗币使用数种多的隐私保护技术来保护网络中所有交易、所有参与者的多种信息元素。环形机密交易隐藏每笔交易的金额信息。环形签名隐藏发送者的信息，隐蔽地址隐藏接收者的信息。即将到来的Kovri路由技术则防止他人将你在门罗网络中的活动和你的物理地址/身份关联起来(注：可惜的是Kovri项目目前似乎处于停滞状态)。

在这些技术的共同作用下，门罗币的用户将保持匿名，资金无法追踪。通过在加密学上杜绝所有链上交易分析的可能，门罗实现了货币可互换性(fungibility)，

这是实用型现金的必备属性之一。读到这里，你应该能够明白在第一章中所描述的门罗是如何保护个人的。

第四章

门罗网络

在

这一章，你将接触区块链科技的核心概念，并了解门罗矿工是如何维护账本的安全。我们会先带你了解区块的构造，以及他们链接成不可篡改的链的方式，以及矿工是如何使用工作量证明以对合意的区块链版本达成共识。关于矿工，我们将讨论新币的产生和在生态中的发行方式。最后，我们会涉猎一些加密学概念（哈希值 [hash] 与随机数 [nonce]）以更深入地了解挖矿的过程。

4.1 简单剖析区块构造

前面的章节讲述了系统构建交易的方式。概括来说，你的钱包发起了一条消息，指示系统将你的一个输出移交给他人。在你的钱包使用你的私钥对消息进行签名以授权该消息之前，消息中的敏感信息（发送者、接收者、金额）在密码学上已经被隐藏起来。

在这一章中，你将习得交易消息是如何被处理以发起实际的转账。当你的钱包广播一条消息的时候，门罗币网络会暂时将它存储在待处理的交易系列表中，也就是我们说的记忆池(memoery pool)。矿工们从记忆池中收集待确认的交易，并将它们打包到区块(blocks)。下图是一个区块的简易表示：

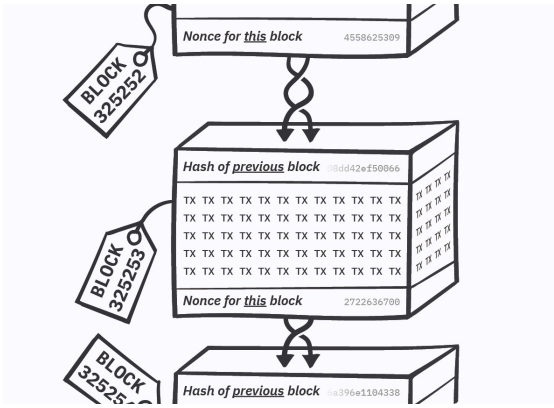


图4.1：每一个序列数字化的区块都包含随机数和指向上一个区块哈希值的参考值。

每一个区块都包含一组交易，一个指向上一个区块的加密学链接（哈希值），以及一个供矿工来生成(include)以使区块完整的特殊数值(随机数)。

如果你想要了解哈希值和随机数的工作原理，这一章末尾将提供易读的讲解和示例(如果你对加密学真的非常感兴趣，那么你可以直接先跳到本章最后一部分)。你只需要了解以下两个概念，便可以理解这些技术是如何保障区块链的安全的：

1. 哈希值是一项安全措施，以保障每一个情况都链接至未经篡改的上一个区块。如果攻击者想要篡改账本的任意一处，即使是极其细微的改动，每一个区块的哈希值都将发出警报，使这种尝试昭然于众。

2. 随机数是一个补足区块的特殊字符串，并将区块标记为就绪，以加入区块链。要想找到定型和封装区块所需的随机数，是计算上的一个巨大难题。矿工付出大多数的时间和精力来搜寻有效的随机数。

随机数无法提前计算，必须等待新区块出现才可以开始搜寻工作。随机数在数学上没有太多意义，只是一次性的，包含随机字符的字符串。

4.2 节点是网络的主干

4.2.1 节点在网络里中继数据

之前我们提及“门罗币网络”的时候，都对它的细节一笔带过。你的交易是如何在这个星云状的网络里传达给矿工和其它用户的？分散在这个星球上的数以千计的门罗节点互联成网，互通信息。

这些点对点网络中的节点，为门罗用户提供高效弹性化的通信。运行节点无需特殊的设备或技能，你只需下载门罗download https://web.getmonero.org/downloads/ 和安装门罗软件即可(过程非常方便，比读这本书快多了)。

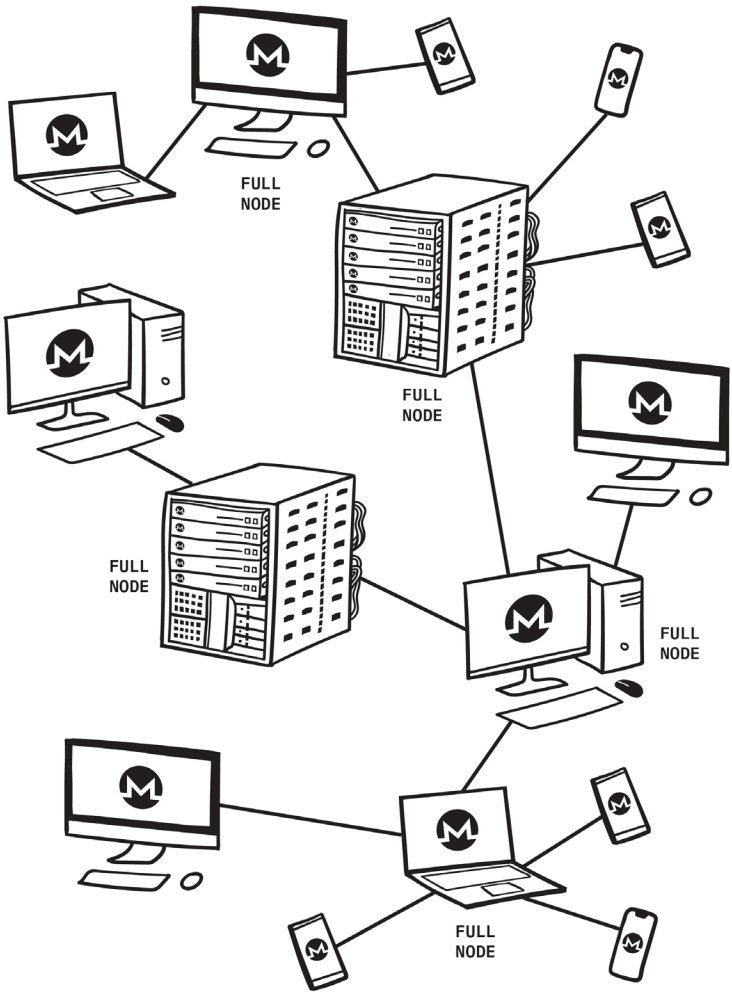


图4.2：门罗币使用分布式点对点网络，该网络由自愿加入和共享数据的设备组成。保存完整的区块链副本的节点便是这个网络的主干。因为运行节点需要大量的硬盘空间，所以有些设备（特别是移动钱包）只能选择连接至远程节点来获取信息和广播交易。

在门罗的点对点网络中，不存在任何“特殊”或“超级”节点的概念；所有节点都是平等的，平等地分享资源，平等地劳作协同。

节点可以运行在各式各样的计算设备上：笔记本，台式机，服务器甚至是虚拟机。

4.2.2节点保存区块链

要启动一个新节点，必须下载完整的区块链并验证加密学链接，如哈希值和随机数。初始化同步过程可能需要数小时，因为节点需要在本地建立自己的区块链副本，同时确认每一笔交易和每一个区块的有效性。门罗节点不会只从单一的中心化源头下载区块，而是接收多个同伴传播过来的信息。节点间无需辨别身份和信任，因为数据的有效性是建立在加密学上的。

所有门罗钱包软件（如GUI、手机钱包等）必须获得访问区块链副本的权限，否则无法完成诸如检索历史交易记录，计算资产余额和发起交易等任务。没有连接至已同步好的节点的钱包无法发起交易，因为钱包需要找到并计算所有相关的未花费输出。然而，不管你有没有连接至已同步好的节点，你都可以接收门罗币(只是你需要钱包下载和验证那个区块后，才会将收款显示在你的资产余额中)。

4.2.3“本地节点"vs"远程节点“

运行本地节点（local node）意味着你需要在本地储存和验证整个区块链，以保证你的钱包可以和你自己的账本副本交互。在这种配置情况下，你的钱包只与你个人的区块链账本交互。运行本地节点需要较多的硬盘空间 (截止写稿时，大约需要60GB)，这不适用于智能手机等设备。

好在钱包还可以连接至远程节点(remote node)。这同时也意味着你的钱包会连接至他人的节点，并请求获取关于你的输出的信息。鉴于轻应用的特性，大部分门罗的移动端钱包都默认设置为连接远程节点。门罗GUI和CLI钱包都可以设置为本地节点或连接远程节点。

使用远程节点没有多少安全上的风险；你的种子密语和密钥并不会泄露出去，远程节点的运营者也无法控制你的资产或破解任何被环形机密交易、隐蔽地址等门罗技术保护的信息。

不过使用远程节点需要在一些小问题上做让步，因为节点运营者会知晓你广播交易和连接更新的时间和设备的IP地址。即将到来的Korvri隐私技术将显着降低这方面的风险。如果你使用自己的本地节点，你的钱包只用扫描你自己的区块链账本里的历史交易，而不会依赖第三方去检索这些信息。

4.3矿工生成区块

4.3.1矿工在最长链上添加区块

矿工在记忆池中收集待确认的交易，通过检查其密码学证明和签名的有效性，来验证其交易真实性，并检查密钥镜想是否被花费过

(回顾3.2.3环形签名来理解这么做的重要性)。

为了准备一个区块，矿工需要起草一份希望打包进区块的交易列表，以及上一个区块的哈希值以作为密码学上的链接。最后，矿工投入到搜寻补足区块所需的随机数的劳动中。

门罗网路中时时都有数以千计的矿工，各自（或组成矿池）来寻找补足当前区块的随机数。一旦矿工或矿池找到了定型当前区块的随机数，他们将向网络中的其他人公布自己准备的区块版本。其它矿工和节点收到这个完整的区块后，将其添加进自身的区块链副本中，这时候，区块高度上升"1"。已被这个新区块打包的交易将从记忆池中移除，其它矿工放弃在这区块上所作的工作，投身至下一个区块周期中。

4.3.2 高难度的工作保证稳定性和公平性

门罗生态遍布全球，网络延迟导致的无法预测的广播中继延期，可能使另个矿工在同一时间同一高度各自补足好一个区块，这时区块链出现短暂的分裂。假设一个南美洲的矿工率先完成一个区块，欧洲的另一个矿工此时尚未收到来自南美洲的这条广播，且正好完成了自己的区块。在这种情况下，西半球和东半球可能暂时使用的是不同的区块链。在很短的一段时间内，两个略微不同的门罗账本在互相竞争中(差异取决于矿工从记忆池中收集的交易不同)。有人可能会觉得这是灾难性的事情。

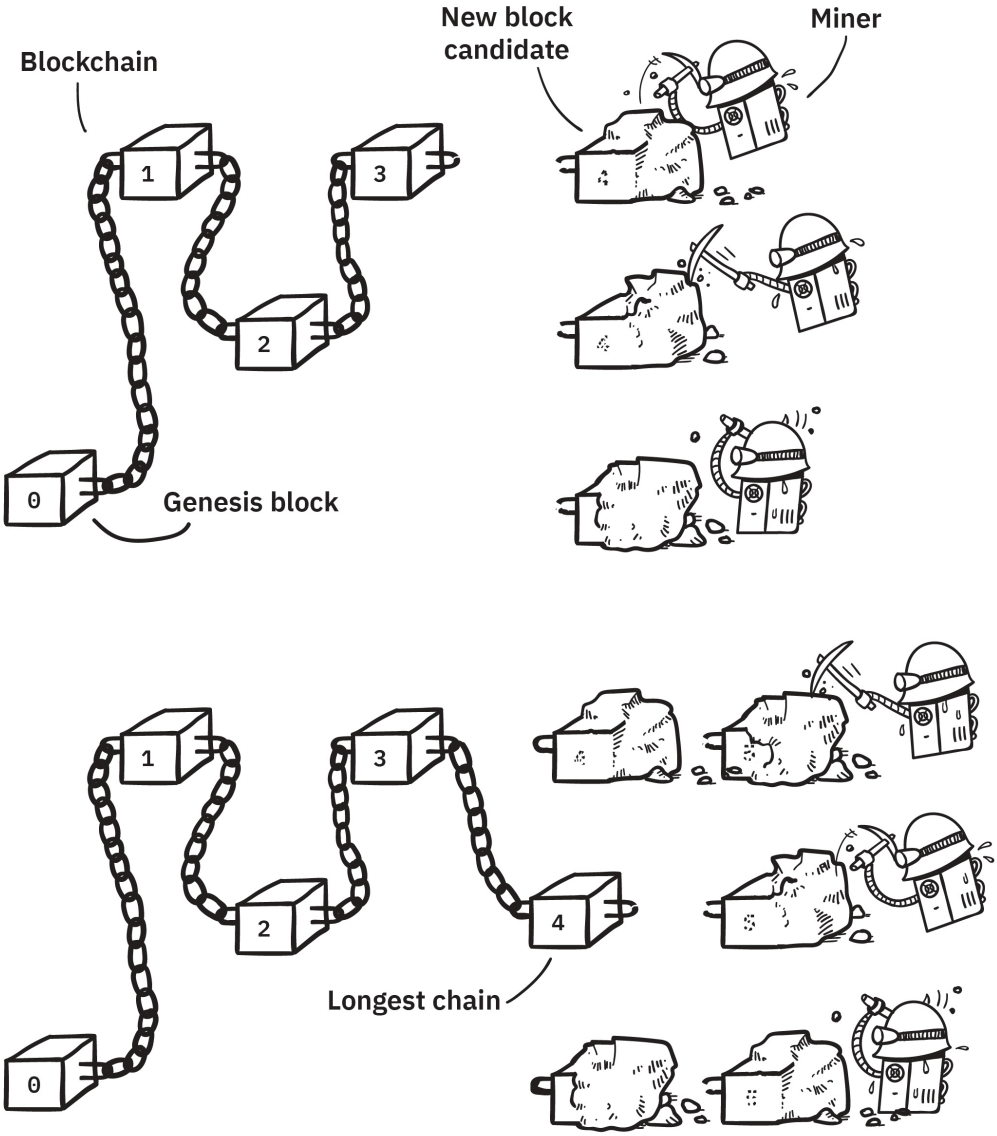


图4.3：矿工相互竞争，给链添加新的区块。上图中，区块链的高度为"3"，所有的矿工都在努力完成区块"4"。下图中，中间的矿工率先完成他的区块并将其加在了区块链上，所有的矿工便转向寻找区块"5"的随机数了。

实际上，可以执行一个简单的原则来优雅地解决这个问题：所有的矿工必须统一在最长链上继续下一个区块的开采工作。这是门罗去中心化共识协议的核心之一，它将使门罗网络从以外的分裂中快速恢复成一条链。其原理并不是尝试直接修复差异化的区块，而是让矿工继续竞争下一个区块。

几分钟过后，将会有一个矿工开采出区块并添加至链上，这条链也就变成了最长的链，所有的矿工和节点迅速同步其副本，并放弃前一个竞争区块，使之成为孤块(orphaned block)。孤块中的交易返回到主链的记忆池中，会被打包进之后的区块。简单地执行最长链原则，区块链网络的任何分裂都将化解，并在唯一通用的账本上达成共识。

4.3.3 门罗“出租车”使用难题来确保公平性

验证一组交易和将他们打包进区块，在计算上并不是什么难事。对于矿工来说，真正耗时的任务是需要寻找一组能够补足当前区块的随机数。这个难题极具挑战，且只能通过暴利破解来不断尝试；寻找有效随机数的过程中，不存在任何捷径，在数学上也无法缩小搜寻范围。矿工只能随机挑选数字，然后看是否能补足当前区块，不断试错，循环反复。

这种无理的障碍设定起初看似非常古怪！

对于矿工来说，验证交易这样的至关重要的工作， 在计算上却十分简单，而对于寻找并提交随机数这种无用的工作上，却极其困难。

为了理解这背后的原因，想想一个假设的门罗出租车网络。这个网络里只有几辆出租车，但有很多司机。主要提交了受认可的路线，他们将获得短暂的出租车使用权。一天到晚，城市各处的潜在乘客都打电话叫车。所有还未接通的乘车请求呼叫被放置在一个实时的待处理出行“池”中。

图4.4：在准备出行计划的第一阶段，每个司机注视着待处理出行池，起草着路线。

传统做法是一个中心化的出租车管理处为每一个出行订单指定车子或司机，但门罗出租车网络的做法是：每一个司机观察池子并规划一个可以在半小时之内完成5~10单的方案。有经验的司机可以快速地完成这个规划过程。司机完成规划后，将方案中的出行订单打包在下一个出行"区块”中，并开始下面所描述的任务，完成后提交他们的路线至门罗出租车网络。

如果司机提出了一个有效的路线，那么他的方案将被认可。该出行方案中的出行订单将会从记忆池中移除，司机检查好出租车，便开始将乘客送往目的地。司机从各个乘客那里收取车费，并还将从门罗出租车网络中获得作为第一个提交完成规划的奖金。

以上的比方对于你们来，应该还是比较直观的。未完成的出行订单存放在记忆池中，当某个司机成功地提交了受认可的订单区块，这些乘客将被送往目的地，相应地，这部分订单也会从记忆中移除。

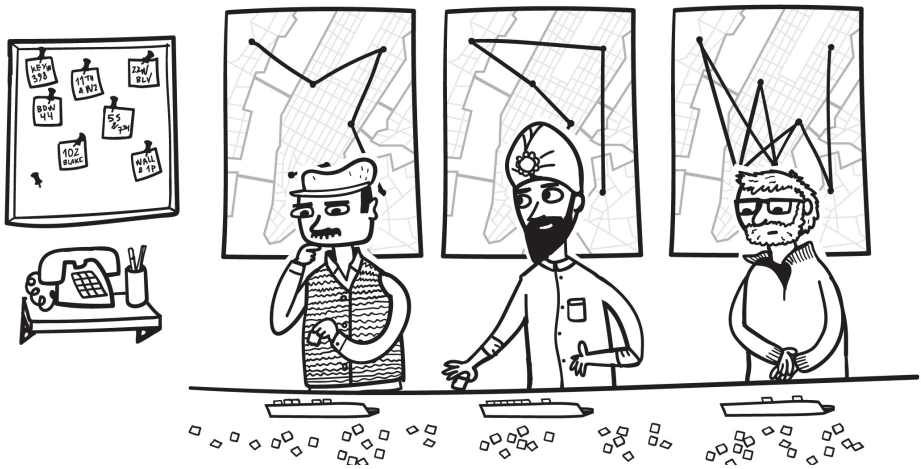


图4.5：当司机们完成了（相对）简单的路线规划，他们必须解一道难题：从地址中抽取字母拼凑成句子。

然而，门罗出租车网络公司有一个非常特殊的规则：让驾驶员将他们的乘车计划提交给门罗网络前，他们必须首先完成一些困难的无用任务。 想象一下，地址抽取的所有字母被随机打乱，并使用其中一些字母生成五个句子（总共大于 50个单词），可以说出任何内容，但必须使用本地语言进行正确的语法/拼写。

司机发的计划路线必须包括游乐设施列表和与目的地中的字母匹配的无意义（“随机数”）句子，否则他们的路线将被自动拒绝。 有多个有效的随机数短语可以用大多数字母组构成（例如{a，e，e，g，i，m，n，r，r，s，t，o，o}可以重新排列形成 两两组句子例如“Rims enrage too！”或者“Monero is great！”），除了将这套游乐设施提交给门罗出租车网络公司之外，所得到的句子绝对无用。

在门罗加密货币中，有着类似的特点：**寻找随机数很难，但是验证随机数很简单**。在这个出租车场景中，要从一打地址中重组出50词的有效句子很难。但是，让一个人来评估结果，并验证它是否可以补足当前区块是很容易的。当一个司机像下图所示一样提交随机数（造句），你可以快速判断“Apple jam is very bad”（苹果酱不好吃）是一个有效句子，以及这个句子是否是从乘客的目的地中重组而来的。对于造句来说，验证的过程几乎是瞬间可以完成的。

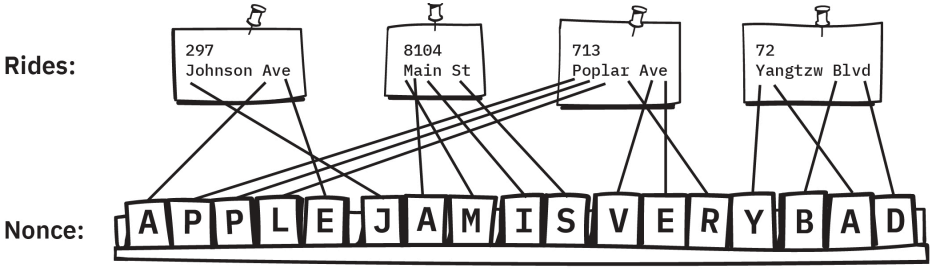


图4.6：从地址中重组句子是耗时且棘手的。然而，验证的过程却十分简单。

富有经验的司机可以在60秒内完成路线规划，然而重组句子可能需要数分钟。

实际上，为了提交区块，重组句子这种无意义的工作占用了大部分的时间。

从司机的视角下回顾下这个过程：在上一个区块被认可之后，你立即开始从记忆池中手机一笔出行订单并完成一份出行规划方案。然后你急忙地从地址中抽取单词充足成句子。在那几分钟里，你和其它所有司机都在对同样一份出行订单列表做着同样的事情。突然，有个司机提交了出行规划方案和句子。你的出行规划中的所有订单瞬间从池子中消失了。你必须立即放弃手上正在忙的事情（因为乘客已经在行程中了），开始收集下一组未出行订单，并重复造句的过程。

为什么门罗出租车网路要向司机施加这么难又无用的任务呢？这其实是他们确保所有乘客一定会被公平对待的原则。想象有几个品行不端的司机，他们会选择性地忽视从某个区域发出的所有订单，或只选择因公出行乘客下手敲竹竿。如果没有造句的过程，这些恶意的司机个人或群体会不断地提交他们（不公平的，排外的）的路线图规划。他们会对某些乘客群体造成系统性的威胁，这与门罗出租车网络的核心理念”人人平等“截然相反。

出租车司机间的造句竞赛对于确保出租车和订单的公平对待来说至关重要。假设所有的出租车司机造句的速度相差无几，那么司机提出方案和提交出行订单区块的可能性是随机的。单个司机连续包揽数个区块的情况不太可能发生，因为每一个司机都在和所有其他的司机竞争，并且无法在这样的竞争中持续获胜。

大多数的司机都是比较诚实的，提交公平的出行订单区块来保证整个城市的交通顺畅。如果有少数的恶意司机试图提交不公平的区块，造句任务将防止他们控制整个系统。统计上来说，他们偶尔会率先完成造句并提交排他性的路线规划。然而，剩余的司机大多为诚实的，他们会开始下一个区块的竞争并提交公平的出行订单区块。因为区块的竞争结果是比较随机的，那么下一个区块大概率上会由一个诚实的司机竞得，这样一来，之前被可以排除在出行区块之外的订单将有机会被纳入下一个区块中。通过施加“无意义”的工作，来使门罗出租车系统的订单分配随机化，保障少数的恶意司机无法针对个人或群体，将其订单拒绝在区块之外。

门罗出租车系统没有负责管理和分配出行的中心化权威，而是通过让司机来参与“无意义”的造句竞赛来随机地分配订单。从统计上来讲，出租车大多数情况下被分配给诚实的司机，

所以门罗出租车网络以公平至上的服务理念闻名。

此刻你或许想问这个出租车的比方跟加密货币有什么关系，你可能已经想到，去中心化的门罗出租车系统就是门罗加密货币网络的类比，在无中心化权威的情况下提供公平的全球化服务。

每一笔出行订单对应门罗交易，在记忆池中等待被收集进出租车/区块。出租车司机代表矿工，两者都进行着简单却又重要的工作（司机规划路线；矿工收集和验证交易），并被要求和其它司机/矿工进行造句/随机数竞赛。这使得出区块权的分配随机化，（在统计学上）使得大部分的出租车/区块被分配给诚实的司机/矿工。任何一个率先提交路线规划的司机都将获得乘客的出行费用和来自门罗出租车司机的额外奖励。同样地，矿工也会获得区块中的交易手续费，以及一笔佣金（也称为币基 [coinbase]和区块奖励[block reward]）。

4.3.4 矿工提供服务并收费

每个一个矿工成功挖出区块（即第一个找到随机数，补足最长链上的下一个区块），他们将收到两种报酬：

1. 首先，矿工收到一笔对其补足包含经过验证的交易的区块的奖励。这个区块奖励与出租车网络付给司机的额外奖励相似。所有矿工收到和确认新挖出的区块后，

将刚刚出炉的币基添加到那个率先找到有效随机数的矿工地址中。

2. 其次，矿工从被打包进区块的交易费收取手续费。门罗用户提供的手续费越高，其交易越有可能被尽快打包进区块中。

通常很多人误认为矿工在“寻找”或“铸造”加密货币。实际上，矿工只是验证交易，并因自己的工作收取加密货币。这种引入新的门罗币方式也称作货币发行曲线(coin emission)。

当门罗启动时，门罗币的发行曲线速率超过30 XMR/2分钟。区块奖励平稳减少至2022年的0.6 XMR/2分钟。相对其他加密货币的区块奖励的瞬间减半，门罗这种持续减产的设计是为了向矿工提供一种更为稳定的经济环境。在2022年以后，门罗将进入速率恒定的尾部发行(tail emission）阶段，每个区块奖励将一直为0.6 XMR。

大多数加密货币都会设置一个货币发行的固定上限，以控制总供应量。当供应量触及上限后，矿工就不再获得新币，收入来源只有手续费。如比特币，矿工盈利模式将在2140年改变，届时比特币供应量达到2100万枚且永久不再增发。这种模式常常被奉为“通缩”，然而相关的讨论也常常基于将通胀性货币供应（inflating monetary supply）的概念与“通胀”（inflation）一词的其它用法混为一谈，来描述大众不希望看到的货币购买力的减少。

进入尾部发行阶段后，门罗的供应年增长率不超过1%（0.6 XMR/2分钟）。因为矿工一直可以收取手续费，币基奖励在长期可以向他们提供更好的财务稳定性。这项社会契约激励着矿工继续使用他们的设备来维护门罗网络的安全。

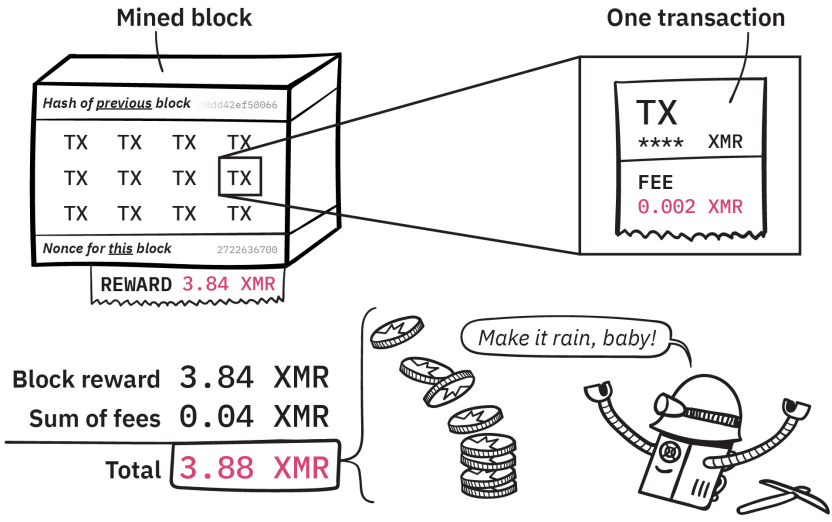


图4.7：一个矿工在挖出区块时，将收到两种形式的报酬：1）每笔交易的手续费；2）刚刚出炉的新门罗币作为区块奖励。

4.4 工作量证明系统

现在我们将不再讨论出租车的比方，而是直接进入确保门罗公平性的共识系统中。这个将网络中重要的功能和无意义随机数的搜寻整合在一起的过程，被称为工作量证明(Proof of Work, PoW)。许多加密货币都打造于工作量证明的共识机制上，并且他们的属性和实施各有不同。不过，他们的本质是一样的：通过提交随机数进行确认，以推动去中心化。有时候，随机数自身被称为“工作量证明”，即非常难以寻得/创造的，但易以验证的一小段数据，比如出租车司机重组而成的句子。

矿工用“每秒能生产的哈希值数量”来衡量其挖矿速度，简写为H/s。每个矿工都可以衡量自身的哈希率（hashrate），且哈希率根据挖矿设备不同而不同。网络哈希率（network hashrate）指的是所有在当前区块进行挖矿的矿工的哈希率总和。

4.4.1 益处

4.4.1.1 抗监管

在PoW系统中，随机数的搜寻竞赛使得最长链上的出块权变得随机化。在出租车的比方中，PoW框架可以抗强监管。门罗网络中的一些恶意矿工可能试图对记忆池中的交易有倾向性或排他性的对待，然而诚实的门罗矿工挑选交易时不会有这么多心思。

4.4.1.2 防止双花攻击

这是在出租车比方中没有展现的，但是是可被PoW系统解决的另一大区块链问题。一个恶意矿工可能试图对一笔输出进行双花(double spend)，即矿工生成替代性的区块，试图逆转他们过去的交易，并将资金切回他们手中。攻击将按照以下流程发动：· 恶意矿工Martin广播了一条交易，发送他的门罗币至受害者“Valerie”· 当该笔交易（Martin>>Valerie）被打包进区块链后，Valerie认为她收到了这笔资金· Martin也从Valerie手中得到了交易物品· 然后Martin挖出了一个区块，该区块与包含原交易（Martin>>Valerie）的区块版本不同· 在Martin的替代版本中，不包含原交易（Martin>>Valerie）！ 而是包含一笔自己向自己转账的交易（Martin>>Martin)。· 如果Martin能在短时间内挖出足够多的区块，使自己的链成为最长链，那么网络将接受他的替代性区块。（实际上，正是这一步的不可行性防止了双花的可能。）· 因为Martin的输出的密钥镜像在链上出现过（现在又和向自己转账的交易相关联），网络则不再认为原交易（Martin>>Valerie）是有效的，因为与此输出关联的密钥镜像已经被花费掉了。

如此一来，Martin货财两得，潇洒离去。

好在PoW系统可以通过限制攻击者在生成区块的速度，来防止他们进行这类型的双花攻击。

回想一下，因为矿工总是跟随最长链，所以恶意矿工需要以足够快的速度，将要撤销的交易所在的区块和之后所有区块都重新替换，超越原主链的长度。因为恶意矿工将独自生产替代性区块，他们对抗的是维护原始账本的整个网络，所以他们常常落败。这类型的攻击只能在一种情况下可行，那就是网络中的恶意矿工必须掌控和剩余网络同样多的算力。所以，51%攻击（51% attack）常用来表示需要掌控全网大多数算力才可发动的恶意攻击类型。

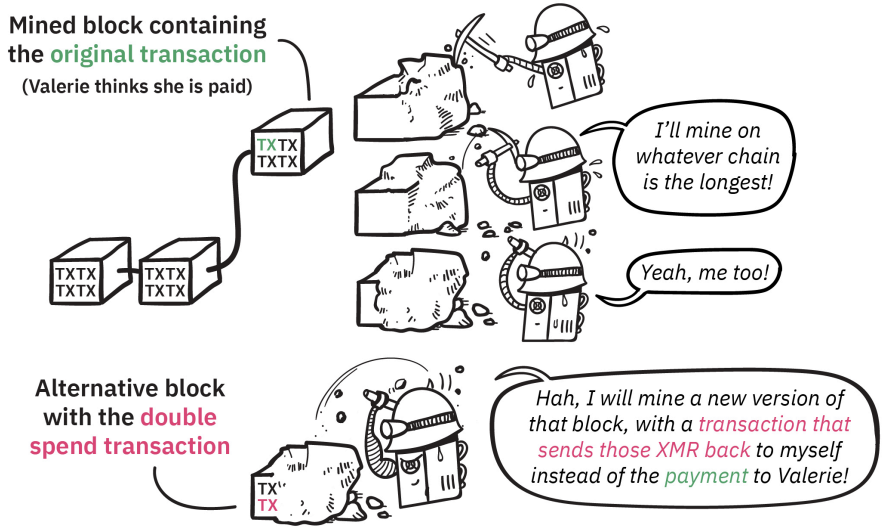


图4.8.a：最下面的不诚实矿工试图通过创造替代性区块来发动双花攻击。

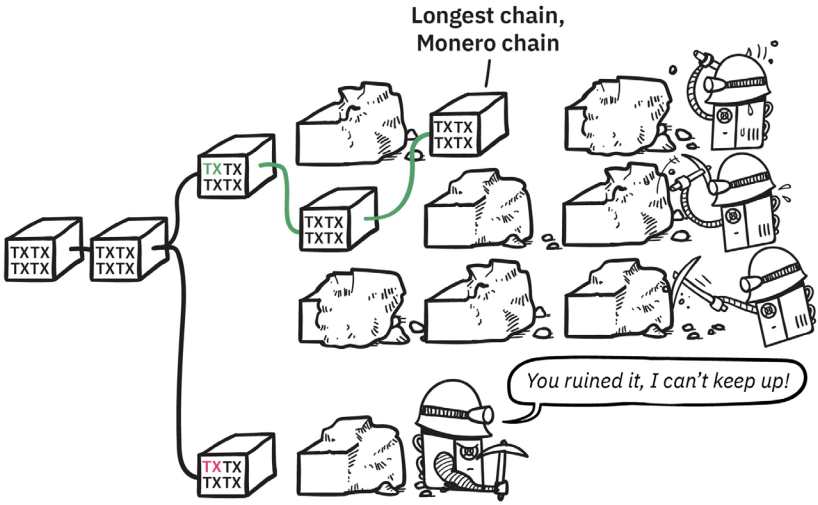


图4.8.b：所有的诚实矿工合力生产区块的速度快于恶意矿工的速度，所以攻击者的替代链被抛弃了。

因为恶意矿工总是需要掌控比剩余网络更多的算力，所以拥有更多矿工、更多算力的加密货币越难被攻击。通过增加矿工来增加算力，可以使网络更安全，更抗攻击。

4.4.2 “难度”调整出块时间

门罗网络大约每两分钟向链上添加一个新的区块。随着每个新的区块被挖出，记忆池中的部分交易被添加进已确认的区块链上。如果平均的出块时间太长，交易的确认就会很慢。如果平均出块时间太短，网络常常无法及时同步。

网络通过调整随机数谜题的难度（difficulty）来影响出块时间。随着越来越多的矿工加入到网络中，他们的猜谜能力（哈希率）的总和将使得出块时间变短（类似于，如果你投入双倍的人力寻找某样东西，可以节省差不多一半的时间）。

这在统计上，将使得矿工挖出区块的时间小于2分钟。为了稳定出块时间，谜题的难度将增加，寻找符合要求的随机数的耗时也将增加。同样的，在挖矿算力下降的时候，难度也可以用来调节，使出块时间不至于太久。

在之前的出租车网络比方中，重组造句的难度可以通过增加或减少词数来调整。如果有一天，有20%的司机（矿工）缺席，那么平均的路规划区块提交时间将变长，意味着部分出租车将被闲置。为了优化，出租车司机可同意将词数从50降到40，如此一来，司机和出租车的供需管理将重新得到平衡。

难度根据网络哈希率的增加而增加，以保持区块流的稳定。

4.4.3CryptoNight算法

与其他大多数加密货币大不相同，门罗使用CryptoNight PoW算法的一个变种。门罗的标志性属性之一，是其PoW对于专业挖矿设备的抵制。

在大多数情况下，“最优化”都是一件好事，所以你可能会不解：为什么门罗的PoW算法刻意地阻止挖矿效率的最优化。

这是因为制造太过强大的挖矿设备的能力，可导致矿工的中心化，这是一种危险的境况。比特币的挖矿历史就是完美的例子。

4.4.3.1 背景：比特币挖矿历史

当加密货币跟着比特币在2009年进入大众视野的时候，大家都只是使用CPU来挖矿。因为网络挖矿难度根据当时的哈希率而调整，CPU挖矿的收益在早些时候还是可观的。CPU矿工的比特币算力的数量级大概在1,000,000 H/s左右，简写为1 MH/s。

很快，显卡（GPU）也加入了挖矿大军，并将挖矿效率提升至100 MH/s数量级。随着网络难度因为GPU矿工的加入而提高，CPU矿工开始入不敷出（开采出来的比特币抵不上购买设备和电力成本）。

之后，为挖比特币而专门开发的应用专用集成电路（application-specific integrated circuits, ASICs）出现了。这类特殊的设备较为昂贵，但是可以达到GPU上千倍的挖矿效率，数量级为1,000,000 MH/s。到现在为止，比特币网络的难度已经随着ASIC的加入而不断攀升，CPU和GPU再无用武之地。

比特币最初发行时的愿景是世界上任何拥有电脑的人都可以参与挖矿，维护网络，并获得相应的奖励。不幸的是，ASIC的出现和扩散终结了这个梦想。如果现在你想要参与比特币挖矿，你需要耗费数百或数千欧元来获得一台ASIC。

ASIC驱逐了绝大部分的比特币矿工。比特币网络的安全最初是由分散在全球的众多电脑极客使用自身的个人电脑和显卡来维护的。可惜，比特币的真正去中心化时代已经离去了。如今，整个网络被几家大型公司、 ASIC矿场所掌控，他们也成为了比特币的主干。

4.4.3.2 ASIC导致危险的中心化

因为许多加密货币都被ASIC所掌控，所以我们有必要思考这个话题和其风险。中心化以两种形式展现：只有少部分公司生产ASICs（制造的中心化）和少部分矿场挖矿（挖矿的中心化）。

ASIC制造和挖矿的中心化，使得黑客、攻击者和政府可能是会对网络的运行造成过于强大的影响。这将使去中心化的众多益处不复存在。比如：

- 全民皆可挖矿：在CPU和GPU时代，大多数人使用不受监管的通用硬件。挖矿现在需要专业硬件，而专业硬件越可能成为监管和控制的目标。某些政府也有可能禁止、或对ASIC制造/持有进行许可制。

- 抗监管：如果门罗大部分的哈希率被少数几个大型矿场所掌控，那么他们可能会被施压，以确认和监视某些特定的交易。

而如果是分散在全球各地的挖矿爱好者，那么就没这么容易被施压了。

- **网络生命力：如果一个恶意的矿机制造商（或遵循政府的命令）在ASIC中秘密部署了一个远程控制或关闭装置，那么网络岌岌可危。秘密装置的激活可瞬间杀死几乎整个网络，使之突然陷入哈希率急剧下降的十分脆弱的状态。ASIC的制造商越少，它的风险就更高。**

比特币完全被ASIC占领了。在大型矿场支配比特币网络哈希率的时候，仍然有少数小型ASIC矿工组成矿池在挖矿。比特币早期，矿工们使用各种品牌、型号与配置的CPU和GPU挖矿，但如今完全不同，大部分的比特币ASIC都由一家制造商设计、生产和配送。如果ASIC市场能够获得多样化的充分竞争，那么ASIC普遍化的风险也就会少很多。

4.4.3.3 门罗与 ASIC 的积极抗争史

平等主义是门罗的根基，因此门罗社区不认可 ASIC 以及它所带来的不可避免的算力中心化。比特币使用的 "CPU-密集型 " 哈希算法inevitable centralization of min-ing power**（SHA-256）可以进行 ASIC 最优化，门罗抵抗 ASIC 的办法是使用 " 内存-密集型 " 算法（**CryptoNight**）减弱 ASIC 加速效果。**
即使到了 2018 年，大家仍然可以用 CPU 和 GPU 来挖门罗。如今有数十亿台设备（任意现代化 x86

CPU 以及很多 GPU）可以支持门罗挖矿，任何人只要连网，皆可挖矿。实际上，使用电脑或者手机的网页浏览器都可以支持门罗挖矿。

在 2018 年的 3 月初，网络中的 CryptoNight ASIC 矿机的悄悄生产和部署震惊了门罗社区。这些设备声称比最先进的显卡的挖矿效率还要高 25 倍，对哈希率进行追溯的分析显示，2017 年年底至 2018 年年初，门罗网络中一半的哈希率来自 ASIC。

CryptoNight 算法本是设计成 " 内存-密集型 "，”以缩小大部分矿工使用的 CPU 和小部分矿工使用的 GPU/FPGA/ASIC 的性能差距“。ASIC 的出现无疑是众人不想看到的事实。CryptoNote 的作者们观察到，“部分用户拥有优势是可以接受的”，他们进而提出“他们的投资在算力上至少应得到线性的回报”。诚然，新一代的电脑或更高级的显卡将比旧设备更具效率，但是 ASIC 的出现导致哈希率的分布极不均衡。

在还未完全确认“ASIC”存在时，门罗社区就已经快速做出反应，积极采取措施对抗“ASIC”。2018年春的常规升级包括对CryptoNight 算法的轻微调整，以对 ASIC 造成不同于 GPU/CPU 矿工的影响。这个轻微的改变并没有影响算法的难度和执行，所以 CPU/GPU 矿工能轻易地适应升级后的网络。

ASIC 则根本无法适应或少或多的新改动。你们可以把 ASIC 想象成一种被训练成能且只能把一件事情做得非常好的员工，但是无法学习做其它事情。ASIC 电路按照算法进行物理蚀刻，所以他们无法重新编程或应用于其它目的。

当 CryptoNote 在区块高度 1546000 进行调整，ASIC 瞬间无法与网络兼容，网络中几乎近半的哈希率消失。因为 ASIC 无法调整以处理新算法的区块，任何他们新生成的区块都会被门罗网络认为是无效的。

门罗看起来暂时消除了预期之外的 ASIC 威胁。为了永久性地抵制 ASIC，门罗将在每次网络升级中都进行挖矿算法微调。门罗每 6 个月进行常规硬分叉升级的方案，应该可以永久性地消除生产门罗 ASIC 矿机的念头，因为 ASIC 的再设计昂贵且耗时，生产出来以后又可能变成一堆废铁。（注：因每半年一次的硬分叉给团队带来极大负担，且 ASIC 研发生产周期似已压缩至 3 个月，2019 年 3 月，团队决议开发一劳永逸的抗 ASIC 算法 RandomX，如若该方案失败，则将专项 ASIC 友好型的 SHA-3 家族算法，以最大限度地促进 ASIC 市场竞争。）

4.4.4 PoW 的替代性方案简介

除了 PoW 之外的，还有一些可以维持公平性的替代性系统，例如权益证明（Proof of Stake），空间证明（Proof of Space），带宽证明（Proof of Bandwidth），甚至是多种证明的结合体。每一个系统都有自身的优缺点。PoW 目前是使用最广泛，经过实地验证的共识机制，也是门罗唯一采用的系统。

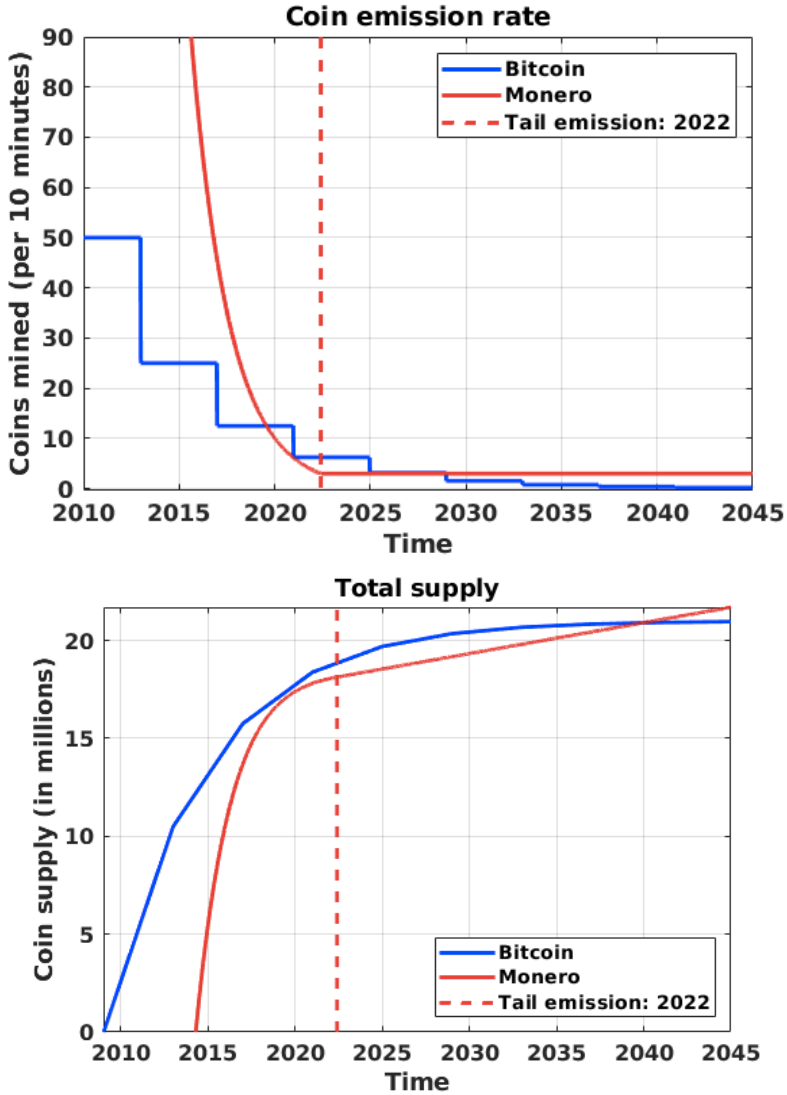


图 4.9：这是比特币（蓝）和门罗币（红）货币发行政策的可视化图。红色虚线代表门罗进入尾部发行阶段的时间（2022 年）。

上图表示的是货币发行速率（以每 10 分钟计算）。比特币的区块奖励会“减半”，而门罗币会平稳地减少直至进入尾部发行阶段。

下图表示的是货币总供应量。比特币的供应量将无限接近 2100 万（2140 年），而门罗将以固定速率持续增长。

4.5 PoW 中的加密学概念

在这章中，我们将详解哈希和随机数的功能，并辅以比方。如果你想了解它们的工作原理，剩余的章节将带你一览加密学机制。

4.5.1 哈希值（一般概念）

哈希函数是一种可以将任意输入的数据，转变为唯一的输出的加密学工具。这些算法使得输入中的任意改动，哪怕是极其细微的改动，都将产生完全不同的输出。“哈希“这个词，既可以指代这个函数，也可以指代某个输入的输出值。

增加、移除或者改变一个字符，都会产生完全不同的哈希值。例如，”Please send 50 euros to

Jen."（请发送 50 欧元给 Jen），我们可以选择其中一种哈希算法来生成它的哈希值 a2d2a9059ed8d323。下列的表格将向你展示，输入中的任意变动如何造成输出的显著不同：

Input	Output hash	Comment
Please send 50 EUR to Jen.	a2d2a9059ed8d323	True message
Please send 500 EUR to Jen.	05cbdd8dd96718ac	Added an extra '0' to amount
Please send 60 EUR to Jen.	f5087a90b63b1777	Changed '5' to '6'

Input	Output hash	Comment
Please send 50 EUR to Jon.	ffd424b7077a3c58	Changed recipient to 'Jon'
Please send 50 EUR to Jen.	a2d2a9059ed8d323	Same input = same output!

此处输出值为采用 SHA-256 算法生成的哈希值的前 16 位，$ echo {input} | sha256 sum | cut -c1-16

哈希值在多个加密学安全环节有着重大作用。哈希加密函数具有抗碰撞（collision-resistant）的特性，即很难找到两个拥有同样输出值的输入值。这对于区块链的不可篡改的核心原理所在，因为历史区块中的任意改动，都将使那个区块和之后所有区块的哈希值发生改变。

这种由哈希值连块成链，只可添加并不断延展的数据库，是区块链革命背后的关键概念。

4.5.2 Nonces (general concept)

4.5.2 随机数（一般概念）

随机数是指在物理或数学上没有内在意义的字谜。看下下面的填空题：

· A）门罗币在世界语中意思为“c___”？

答案：”coin“（货币）

· B） 1 公斤相当于“_____”克？

答案：1000

- · C）没有重复数字的 3 位素数“3___”？

答案：可以是{307, 317, 347, 349, 359, 等等…}

· D）没有重复数字的 5 位素数“7___”？

答案：可以是{71263, 72169, 73609, 74869, 等等…}

题 A 和题 B 都是有意义的，并且每个问题都只有一个正确答案（A: “coin” B: “1000”），学生可以记住以应对将来同样的问题。这样的答案不能称之为“随机数”。

然而，题 C 和题 D 都是复杂的工作，并且对任何实际问题都没有帮助。满足条件的答案有多个，比如对题C说，“359”和“307”是等价的。

一个学生花费一个小时尝试各种数值以得出“359”这个题 C 的答案，不得不根据题目的调整重新搜寻随机数，例如“没有重复数字的 3 位素数“6___”？”

如果你只有一张纸和一支笔（加上计算器好了），你愿意尝试解答题 C 和 D 吗？可能你会选择题 C，因为找一个有效的 3 位素数比找 5 位的素数要快一些。你现在应该可以感受到，调整素数位数是如何影响解题的难度。

4.6PoW概念总结

加密货币通过每个区块的哈希值来判断它的内容有没有被篡改，因为哪怕是单个字符的改变会使哈希值发生显著的变化（并且将影响后来的所有区块）。

区块的哈希值包含了它所有的内容：交易、区块头，前一个区块的哈希值以及随机数字段。

为了补足一个区块，矿工们必须猜测随机数的值，以生成一个低于特定基准条件的输出，这个基准条件又由当前网络的难度决定。因为无法预测输入的改变如何影响输出，矿工只能通过暴力破解的方式，不断试错，直到生成符合条件的输出。

网络通过提高或降低基准来影响挖矿难度，以保证无论全网哈希率如何，出块时间维持在 2 分钟。

深入门罗和密码学

在计算机问世之前，数学和密码学就已经是通信与信息交换的核心学科。
密码学最早可以追溯到罗马共和国时期恺撒Caesar's time使用的简单密码（一种广为人知的替换加密技术，后世称之为恺撒密码）,现代密码学则开始于世界大战时期，用于重要机密信息加密理。最开始投入研究密码学的都是各国政府与军队，以保护国家机密为主要研究方向。

现在，密码学已不再局限于间谍和军事活动中使用，而是互联网时代通信和安全的支柱，世界各地的学术与行业研究人员都在对密码学进行广泛与深入的研究。

作为现代十分普及的幕后工具，密码学在保密、管理、通信等很多连接层面上改善着我们日常生活。例如：Secure Socket Laye（安全套接层，简称 SSL，一种网络安全协议，不推荐使用TSL）采用了数据加密签名技术。医院、银行、政府和企业均采用密码学技术保护用户数据。

在本章，我们将主要讨论：加密工具如何应用一个去中心化金融数据库生产出加密数字货币（重点介绍门罗币）。

5.1 数学基础

以下先简要介绍/回顾密码学的几个核心数学原理。

5.1.1 欧几里德除法（A/B）

将任意一个数 A 除以另一个数 B（写为 A/B 或 A÷B）会得到一个结果，该结果既可写为商与余数，也可单独写为小数。

通常：

$$\frac{a}{b}$$

A/B=q，余数为 r
例：
12/4=3，余数为 0，可写为小数 3.0
13/4=3，余数为 1，可写为小数 3.25
27/5=5，余数为 2，可写为小数 5.4

5.1.2 质数

质数是只能被“1”和自身整除的整数，例如：20 不是质数，因为可以被 2、4、5 和 10 整除，得到整数，
例：20 ÷ 4 = 5 或 20 ÷ 10 = 2

7 是质数，因为除了 1 和自身外，除以任何整数均不会得到整数，
例：7 ÷ 3 = 2.3333

其他简单的例子包括 3, 5, 7, 11, 13, 97, 223, 997, 3413, 4421, 17837, 145601, 428567, 1171967, 还有很大的数字例如“2074722246773485207821695222107608587480996474721117292759925899121966847505496583100844167325500 77”或者孪生质数 2,996,863,034,895 × 2^1,290,000 ± 1, 单个质数均超过 350,000 位！

5.1.3 模算术

模算术描述的是会绕过特定整数的数字。一个直观的示例为 12 小时制。如果在晚上 11 点以后熬夜 5 个小时，那么，你不会经历下午 16 点！因为在午夜，时间会转而绕到零点（因此晚上 11 点过去 5 个小时后，是第二天早上 4 点）。

给定任意两个正数，A（被除数）和 B（除数），，

模数 B=A/B 的余数 r

就时钟而言，晚上 11 点后熬夜 5 小时可表示为：

（晚上 11 点 + 5 小时）mod
12=…
=16:00 mod 12
=4:00（早上）

5.1.4 整数表示

整数可用许多不同的编码表示，其中一些在计算机科学中经常遇到。

大多数人都非常熟悉 base-10“十进制”数制，该数制用 10 个字符表示数字：
0,1,2,3,4,5,6,7,8,9。

对于一个 base-16 集合，“十六进制”编码额外增加了 6 个字符：0,1,2,3,4,5,6,7,8,9,a,b,c,d,e,f。

以十进制写为 11719682 的整数可以用十六进制表示为 B2D402。请注意，较大的字符集通常需要较少的位数（较短的字符串）来表示相同的数字。

计算机以 base-2“思考”时，只使用字符“0”和“1”。这称为二进制，例如数字11719682（base-10）可表示101100101101010000000010。

门罗币以 base-58 打印最终地址和密钥，其中使用阿拉伯数字和大多数拉丁字符集（大写和小写）。base-58 类似于另一个 Base64 方式，但其已经修改，以避免数字和字母在打印时混淆不清。门罗币使用这种格式，目的完全在于方便人们使用，因其经常须人工阅读或转录长地址。

base-58 字母表包括：

123456789ABCDEFGHJKLMNPQRSTU-

VWXYZabcdefghijkmnopqrstuvwxyz

注：该 Base58 字母表不包括零（0）、字母 I（i 的大写）、字母 O（o 的大写）和字母 l（L 的小写），因其彼此间易混淆不清。

5.1.5 椭圆曲线

5.1.5.1 概述

椭圆曲线被定义为满足以下等式的一组二维（x,y）点：

$y^2=x^3+ax+b$.

例如，在固定系数 a=2 和 b=3 的情况下，该等式变为

$y^2=x^3+2x+3$,

这由许多对点来满足，例如：

x = 3 and y = 6
x = 3 and y = -6
x = -1 and y = 0.

5.1.5.2“Ed25519”扭曲爱德华兹曲线

门罗币使用特殊的扭曲爱德华兹（Twisted Edwards）椭圆曲线 Ed25519 进行加密操作，

该曲线是蒙哥马利曲线 Curve25519 的
二元等效物。

ed25519 曲线可以代数形式表示为

$$-x^2 + y^2 = 1 - (121665/121666)\, x^2 y^2.$$

回想我们的一般椭圆曲线等式，该扭曲爱德华兹曲线是一个使用以下参数的特例：

$$a = -1 \text{ and } b = 121665/121666.$$

近期发现，NIST（National Institute of Standards and Technology 的简写，意为：美国国家标准技术研究所）支持的 PRNG（Pseudo-random Number Generator Algorithm 的简写，意为：伪随机数发生器算法）明显带有缺陷，含有潜在的后门potential backdoor。由于 NIST4 标准算法最近出现一些问题，因此密码界的许多问题选择采用扭曲爱德华兹曲线来解决。

从更广泛的角度来看，NSA（National Security Agency 的简称，意为：美国国家安全局）也在暗中支持 NIST 选择的曲线。鉴于之前 NSA 利用其对 NIST 的权威削弱由后者所提出的算法一事，密码和密码货币界对这些背书持怀疑态度。

扭曲爱德华兹曲线 Ed25519 不受任何专利的限制，其背后团队已开发和调整了基本加密算法，并考虑到了算法效率。据认为，该曲线目前是安全的。

5.1.5.3 椭圆操作

椭圆曲线点加法和标量乘法是椭圆曲线密码方案的基本运算。在深入研究门罗币的计算机制之前，

我们应对这些概念有一个基本了解。

椭圆曲线点加法不同于日常算术中遇到的典型加法。如需将椭圆曲线上的两点相加，则须找到这两点之间的直线，然后找到曲线与该直线的交点。然后，沿 x 轴映射该点直至到达最终点。

向自身添加点（所谓的点加倍）时，必须找到起点切线，以到达该切线与曲线的交点。然后，沿 x 轴映射该点直至到达最终点。

标量乘法利用曲线上的一个点和一个整数。如需将一个点 P 乘以一个整数 S，则将该点以 S 次添加至其自身。许多加密方案（例如门罗币使用的方案）使用椭圆曲线上的公共基点作为生成器点，从私钥生成公钥。

当曲线生成器点被多次添加至曲线自身时，所得点不能用于确定操作发生的次数。该问题通常称为椭圆曲线离散对数问题。这种标量乘法被认为是单向函数，因其反转运算非常困难。

5.2 密码学基础

由于具有面向隐私的独特加密特性，

门罗币成为一种领先的安全的且不可追踪的加密数字货币，这一点我们会在本章中进行更深入地探讨。由于密码学的数学本质，书中列有更多关于密码学的技术性章节。更加复杂的技术建立在被称为加密原语的简单原理之上。

加密原语是一种算法，用作加密协议的构建模块。门罗币为各种用途使用了不同的加密原语，我们在第 3 章和第 4 章中会从概念上讨论其中部分内容。门罗币对隐私和（ASIC 抗性）工作证明的有意方法需要比许多其他加密数字货币使用更复杂的加密工具。

5.2.1 对称和非对称密码学

对于加密数据，根据所使用的密钥类型，可将其算法描述为对称或非对称。

对称加密要求参与者共享一个秘密，例如，你用密码 “hunter2” 加密信息，而接收人用密码 “hunter2” 解密信息。如需以该方式交流，双方必须事先就共享（对称）秘密达成一致。这个实际问题限制了对称加密在许多应用中的利用。

不对称加密允许双方在不共享特定秘密的情况下安全交互。这种类型的密码学已深植在互联网安全、端到端通讯工具和加密数字货币的框架中。

比特币使用两个密钥的不对称加密：

• 私钥 — 用于签署交易和解密数据

• 公钥 — 用于签名验证和加密数据

门罗币密码框架更复杂，其需要有四个密钥：

• 查看公钥 — 用于验证地址的有效性

• 查看私钥 — 用于查看余额、费用和交易金额等数据（查看密钥无法创建或签署交易）

• 支付公钥 — 交易验证的另一个公钥

• 支付私钥 — 用于签署交易，即发送门罗币

这对公钥可直接表示为你的公共门罗币地址，而比特币（和复制体）使用其单个公钥的散列。EdDSA 密钥（私有和公开）长 256 位，或有 64 个十六进制字符。并非每个 256 位整数均为有效的 EdDSA 标量（私钥）；其必须小于 Ed25519 函数部分等式中描述的“曲线顺序”。

5.2.2 散列

第 4 章讨论了散列的概念及其使用范围，从确认数据保真度到在工作证明中分配奖励。散列示例显示于第 4 章末尾的密码学部分。

选择一个良好散列算法对于以安全方式生成地址和密钥至关重要。如果两个不同的输入产生相同的散列输出，则称其为冲突。散列凭借其有效唯一性在区块链系统中通常被用作标识符。

此外，种子生成过程中的冲突会导致多个个人拥有相同的密钥和地址；显然，这会出现严重问题！

门罗币使用 CryptoNight PoW 系统，该系统采用一种建立在 Keccak 散列基础上的特殊 CryptoNote 散列算法。Keccak 算法赢得 NIST 竞赛，被指定命名为 SHA3，由非 NSA 工程师设计。门罗币对交易和区块散列均使用 32 字节输出的 Keccak-256 散列函数。

5.2.3 门罗币伪随机数生成（PRNG）

当用户和计算机创建新的密钥时，找到别人无法猜到的新密钥是至关重要的一点。这实际上是一项非常困难的任务，因为硬件和软件通常均为支持再现性而设计。如果计算机以可预测的方式产生随机性，则输出可能在表面上为随机性，但更容易猜测。

例如，考虑到一个 PRNG，其简单地将当前时间的数字洗牌，生成一个4位数密钥。因此，在“10:34”时，它可能会输出“0413”或“1403”或“0134”……

如果想对输出密钥保密，则这会是一个糟糕的方法，原因如下：

- 当攻击者知晓你在中午 12:45 左右开始工作时使用了密钥时，便会知晓数字“1”和“2”会出现，从而将选择范围缩小到明显更少的选项。

- 一天中不会出现三个“9”的 HH:MM 时间。事实上，不会出现从 中所选任何三个数字的时间，因为 17:89、18:78 等都是不可能出现的时间。

这个规则消除了许多 4 位数的密码，让攻击者从更小的范围中进行猜测。

上述基于时钟的随机数生成器非常糟糕，因为使用一天中的时间作为初始种子是可预测的。最初的种子对于攻击者而言应该更难猜测。良好随机数生成器会引入大量熵，使其输出不可预测。简单地移动4位数不会引入太多熵，这也是上述 PRNG 不安全的另一个原因。

生成钱包时，用户的操作系统提供初始种子/熵源。然后，门罗币反复应用 Keccak 散列函数，以导致不可预测和不可复制的输出。每轮散列产生一个输出，用作下一轮散列的输入。

5.3 生成门罗币密钥和地址

5.3.1 挑选种子

在第2章中，我们谈及钱包的核心：也就是种子。你的钱包生成了这个秘密，用来派生你所有的密钥，以及存取/支出你的资金。在概述中，我们简单地考虑了25个单词的种子助记词。

在幕后，种子是一个唯一的256位整数，从中派生密钥和地址，例如：

```
112699108505435943726051051450940377552177626778909564691673845134467691053980
```

这些数字通常表示为64位的base-16数字，例如：

```
f9296f587419f1cdede67de160fca14d1069ecaa4c-
52f012af031eeA09ee039c
```

对于助记词类型密钥，种子的这种表示实际上就是支付私钥本身！

写下上述任何一种密钥类型都是相当困难的一件事，大多数人都容易犯至少一个错误。转换成种子助记短语是另一个步骤，只是对于人们来说有可解释性和可用性。助记短语本质上将上述256位数字转换为24位（24个单词）base-1626“数字”（因为种子字典中有1626个单词）。这种长种子字符串的表示法更易于阅读：

（译者注：羔羊 六边形 王牌 获得 鼻声 迟钝 争论 何时 无惧 遮阳
篷 学院 钉子 威胁 水手 宫殿 自私 学员 点击 疾病 歪曲
边界 拇指 补救 山脊 边界）

```
lamb hexagon aces acquire twang bluntly argue when
unafraid awning academy nail threaten sailor palace
selfish cadets click sickness juggled border thumbs
remedy ridges border
```

当你的钱包显示24单词种子时，便会添加第25个单词作为校验和，以便以后可检测到打字错误或差错。门罗币的助记词法编码的比例最小为4:3。换言之，四个字节创建三个单词，加上一个校验和单词；八个字节创建六个单词，加上一个校验和单词；依此类推。

查看私钥是通过Keccak-256散列种子派生，产生第二个256位整数，然后将其发送到称为sc_reduce32的函数，以确保它与椭圆曲线兼容。通过这种方法创建的种子始终是有效的标量，

因为它们首先被发送至sc_reduce32函数。

5.3.2 密钥派生

5.3.2.1 所有密钥

上述门罗币种子实际上是支付密钥，所有其他密钥均从该密钥派生而来。查看私钥是支付私钥的简化散列，转换为ed25519曲线的有效标量。

这两个私钥乘以生成器点，得到钱包的两个公钥（支付公钥和查看公钥）。这种派生密钥的方法称为确定性方法。

5.3.2.2 仅查看钱包

你可通过使用查看私钥（而非支付私钥）设置钱包来授予对门罗币帐户的仅查看权限。这些仅查看的钱包可查看所有收入交易，但不能支付门罗币或查看支出交易。

有几种情况下，在没有发送访问权限时检查收入交易非常有用。例如，拥有冷钱包的人可使用查看密钥来检查资金是否到达，同时将他们的支付密钥安全地封存起来。同样，开发人员可构建能够检测和响应收入支付的系统，而无需具有转移这些资金的能力。

这项功能对慈善机构尤其有价值，因为慈善机构可分享自己的查看密钥，以确保捐赠的透明度和责任感。如果向公共地址捐款，

则可使用查看密钥来验证慈善机构是否收到资金。

例如，考虑主要门罗币捐赠地址：44AFFq5kSiGBoZ4NMDwYtN18obc8AemS33DBLWs3H7otXft3XjrpDtQGv7SqSsaBYBb98uNbr2VBBEt7f2wfn3RVGQBEP3A.

由于混淆地址会阻止公共地址在区块链受记录或搜索，因此社区还会发布查看密钥（f359631075708155cc3d92a32b75a7d02a5dcf27756707b47a2b31b21c389501），以便公众可查看捐赠活动。

由于任何持有查看密钥的人均可看到钱包收到的总金额，因此被授予100 XMR的透明慈善机构不能挪用90 XMR并声称他们只收到10 XMR。对于必须达到特定捐赠门槛的众筹情况，此功能尤其有用。

无法从仅查看钱包中查看支出交易是一个特性，而非错误！如果支出交易公开，则其将揭示何时花费输出。这存在严重问题，因为环签名依赖于支出状态的模糊性。假设一个慈善机构揭示了何时花费了输出资金；所有未来（和先前）环签名中的表现均可受识别为诱饵。因此，不公开支出交易对于维护整体网络隐私的完整性而言是必要因素。

5.3.3 地址生成

门罗币钱包的标准地址由上一节中派生的两个公钥（支付公钥+查看公钥）组成。其还包含一个校验和和网络字节，用于标识网络和地址类型。

5.3.3.1 网络字节

网络字节用于区分各种加密货币和网络。例如，CryptoNote类型的币在文件src/CryptoNote_config.h中指定适当的值，例如：

```
uint64_t const CRYPTONOTE_PUBLIC_ADDRESS_BASE58_
PREFIX = 18;
```

门罗币的主网络使用“18”来表示主地址（这便是门罗币的主地址以“4”开头的原因，这是ASCII表示）。
门罗币开发人员使用有自己独特网络字节的测试网（用于开发者）和阶段网（用于用户体验）：

Name	Code value	ASCII value for prefix
Main net primary address	18	4
Main net subaddress	42	8
Test net primary address	53	9
Test net subaddress	63	B
Stage net primary address	26	5
Stage net subaddress	36	7

5.3.3.2 级联公钥

支付公钥和查看私钥被连接并附加到网络字节，以产生原始地址（除校验和之外的所有内容）。尽管该地址仍然是原始格式，

但其包含所有密钥信息：用于创建交易的密钥和网络元数据，以确保交易得以传达至正确的网络。

5.3.3.3 校验和

由于门罗币交易不可逆，因此将付款发送至正确的地址至关重要！为帮助避免打字错误和小错误，地址包括校验和。如果发送人输入错误或没有捕获整个地址，则校验和将不匹配，表明输入的字符串并非有效地址。

该校验和是由Keccak对上一节中收集的地址信息进行散列处理而生成。哈希摘要被缩短为前4个字节，并用作校验和。

5.3.3.4 将所有此类信息结合起来：地址最终确定

最后，根据门罗币规范连接网络字节、密钥和校验和：

		Description
0	1	identifies the network and address type - '18' for mainnet and '53' for the testnet (in base-58, '4' and '9' respectively)
1	32	public spend key
33	32	public view key
64	4	checksum (hash created with Keccak function of the previous 65 bytes, trimmed to first 4 bytes)

最后，该69字节的输出字符串受编码成门罗币base-58格式。这种转换会将长度增加到95个字符串，便于读写。全部过程便是如此！门罗币主地址仅包含：

[网络字节+支付公钥+查看公钥+校验和]标准地址示

例：4BKjy1uVRTPiz4pHyaXXawb82XpzLiowSDd8rEQJGqvN6AD6kWos LQ6VJXW9sghopxXgQSh1RTd54JdvvCRsXiF41xvfeW5

以下伪代码描述了生成公共地址的过程，使用Hs（）表示Keccak散列，使用“||”表示字符串连接。

```
    Checksum = Hs(Varint(Prefix) || public spend key  || public view key)
    SerializedString = Base58(Prefix || public spend key || public view key || checksum)
```

第7章包含实际的Python代码，用于自身生成密钥和地址！

5.3.4 子地址

门罗币交易的隐私通过三个主要构建实现：环签名、混淆（一次性）地址和RingCT。这些减轻了交易因与那些可以被分析的区块链数据链接的风险。然而，必须考虑“链外”链接风险（即，从除区块链数据本身之外的其他来源收集的信息）。

例如，假设主地址收到几个不同个人的付款。

由于门罗币的混淆地址技术，你的公共地址从未在交易中得到明确记录，因此无人可通过分析区块链（包括消费者）来链接这些交易。但是，如果两个发送人互相通信，并发现他们都将门罗币发送至同一个地址，则这种加密隐私便完全被规避了！

你可通过生成多个子地址、与每个发送人共享一个唯一的子地址来避免这种风险。该子地址与你的主地址来自相同的密钥，因此收到的任何子地址的资金都将归集至相同的总钱包余额。然而，不同的子地址在加密上不可链接，因此多个发送门罗币到同一个钱包的人，也无法通过比较他们的地址列表来识别这一点。

5.3.4.1 创建子地址

回想一下，每个钱包都有两对密钥。查看私钥（pV_0）和支付私钥（pS0）受保密，而查看公钥（PV_0）和支付公钥（pS_0）被编码至每个地址中。如前所述，公钥是通过将私钥乘以椭圆曲线上的生成器点（G）生成，即 $(PV_0, PS_0) = (pV_0, pS_0)G$。

你的钱包可创建大量子地址，每个子地址都有不同的索引 “i”（通常从i=1开始）。每个子地址在每个索引处均有自己的密钥集，含有唯一的私钥 (pV_i, pS_i) 和公钥 (PV_i, PS_i)。

为第i个子地址创建支付公钥的公式是：

$$PS_i = Hs(pV_0||i)G+PS_0$$

这个过程从连接索引“i”到主地址查看私钥（pV_0）开始，并通过hash_to_scalar函数传递该结果（注：在实践中，引用客户端钱包也将字符串子地址连接到数据，作为散列的常用盐）。所得标量乘以曲线生成器点，并通过椭圆曲线点加法添加到主要支付公钥。

该子地址支付公钥乘以主要支付私钥，得到子地址查看公钥：

$$PV_i = pV_0*PS_i$$

子地址公钥按照与主地址相同的惯例被编码至公共地址中：

$$Subaddress\_i = base58(network\ byte\ ||\ PS_i\ ||\ PV_i\ ||\ checksum)$$

然而，子地址主网的网络字节为0x42，这便是其均以数字“8”开头的原因。

5.3.4.2发送至子地址

这种识别第一个网络位的不同方法是至关重要的一点，因为交易到子地址的构建必须与正常情况略有不同。

构建交易时，钱包通常会生成32个随机字节作为私钥。当发送到主地址时，

该随机密钥通过椭圆曲线标量乘法与椭圆曲线生成器点相乘，以产生交易公钥。但是，当发送到子地址时，交易私钥反而会乘以接收子地址的支付公钥。

5.3.4.3 接收到子地址

由于门罗币区块链的模糊性质，钱包必须扫描每笔交易，以确定其是否属于所有者。为确定给定的输出X（带有支付公钥R）是否被发送至主地址，钱包根据其查看公钥和支付公钥检查计算。如果等式$X==Hs(pV_0*R)G+PS_0$为真，则可解锁和花费输出！

然而，检查哪些输出属于子地址的过程略有不同。除了从输出中减去hash_to_scalar项并与子地址支付公钥进行比较之外，计算基本相同。如果等式$PS_i == X — Hs(pV_0*R)G$为真，则钱包知道它建立在自己的输出上。

5.3.5密钥派生的其他方法

更令人困惑的是，目前在门罗币中使用的至少有三种不同的私钥派生方法（比特币也是如此）。这些方法在几种“密钥”方面有所不同：

- 原始（非确定性类型）：支付私钥和查看私钥均独立和随机选择，以形成一个帐户。除了保留每个文件的副本之外，

并无好的方法来备份不确定的帐户。由于有更好的选择，因此不再推荐这种笨拙方法。

• 助记词（确定性或“Electrum”）类型：在这种类型中，所有的密钥均是从一个单独的支付私钥中派生而来，该支付私钥被称为种子。查看私钥是通过用Keccack-256散列支付私钥以产生有效的EdDSA标量而派生。这些帐户是很容易备份的，因为你只需要写下种子（通常用base-1626助记短语表示）。

• MyMonero类型：MyMonero钱包系列使用类似于Electrum惯例的方法，但是种子短语是13个单词，而非通常的25个单词。这13个单词转换为128位整数，用于支出和查看密钥派生。用Keccak-256对种子整数进行散列，并转换为支付私钥。用Keccak-256对该支付私钥再次进行散列，并转换成查看私钥。

你可能已注意到MyMonero类型和Electrum种子类型之间的一个重要区别。MyMonero通过散列一个随机整数来创建查看私钥，而Electrum类型散列支付私钥。这意味着13单词和25单词的种子并不兼容 — — 不可能创建一个与MyMonero类型帐户相匹配的Electrum类型帐户（反之亦然），因为查看密钥对总是不同。

5.4 隐私技术

5.4.1 混淆地址

第 3 章从概念上描述了一次性地址（也称为混淆地址）如何允许交易在不暴露接收人真实地址的情况下发布到网络上。本节将更深入地解释一次性公钥背后的加密技术。

5.4.1.1 发送

CryptoNote 协议根据公式 X=Hs(r*PV|i)G+PS 计算接收一次性地址。让我们逐步了解这些符号的含义，以及 Maria 向 George 付钱时会如何生成一个一次性地址。

变量 r 是交易私钥，这是一个 256 位的伪随机标量。Maria（发送人）是唯一知道该密钥的人；甚至 George（接收人）也从来不知 Maria 钱包里选的 r 的随机数。

Maria 然后将 George 的查看公钥 PV 乘以 r，然后附加输出索引 i。该量 (r*PV|i) 随后通过 hash_to_scalar 函数 Hs() 运行。该函数使用 Keccak-256 算法对其输入进行散列，然后将得到的散列结果对质数进行取模

```
2^255 + 27742317777372353535851937790883648493.
```

将上段中计得的 Hs(r*PV|i) 项乘以 ed25519 基点 G。最后，Maria 将该量加到 George 的支付公钥 PS 中

，以产生最终输出 X，即混淆地址。

这一错综复杂的过程使得 Maria 能够使用一个随机生成的一次性地址向区块链的 George 进行隐秘交易，但无人能够与他联系。

5.4.1.2 接收

鉴于 Maria 将发送给 George 的门罗币隐藏得那么好（受 George 不知的交易私钥所掩蔽），你可能想了解 George 如何能在区块链找到它！

如第 3 章所述，George 必须扫描区块链，寻找属于他的输出。这个过程与 Maria 用来生成地址的方法非常相似。

George 从区块链获得交易公钥 R，并将 R 乘以他的查看私钥 pV。按照与 Maria 相似的步骤，George 附加输出索引 i，然后将 hash_to_scalar 函数应用于 (pV*R|i)。然后，他将结果乘以 G，并加上自己的支付公钥 PS。如果该值与输出匹配，则其属于他。

换言之，George 的钱包扫描了区块链的每一笔交易，以确定 X=Hs(pV*R|i)G+PS 的输出。

5.4.2 环机密交易

环机密交易（RingCT）模糊了交易中发送的门罗币的数量。RingCT 于 2017 年 1 月实施，并于 2017 年 9 月之后成为所有交易的强制性规定。

只有创造新门罗币作为 coinbase 奖励的交易才具有可见的数量，且不会受 RingCT 所掩蔽。这是一个审计功能，允许任何网络参与者计数并验证已生成多少门罗币。在币公开释放之后，这些交易在进一步使用之前被转换成 RingCT。

所有非 coinbase 交易均使用 RingCT 加密交易金额。每笔交易的金额以两种不同的方式加密，这两种方式均包含在消息中。

首先，该金额由从接收人地址中的公共信息导出的密钥加密。该版本记录在 ecdhInfo 字段中，只能由接收人使用交易共享秘密解密和读取。

其次，该金额被纳入佩德森承诺，如此可允许其他门罗币用户自己验证交易的有效性。无人可从佩德森承诺中检索交易金额，但是任何人均可检查结果，以数学方式验证输出与输入是否平衡。如此防止任何试图伪造门罗币的交易。

RingCT 的验证有两个关键方面：

1. 发送人使用范围证明，可验证性证明所有输出均包含正数。范围证明表明，掩蔽数可被生成为 2 的正幂之和，而不会揭示这些幂是什么。如果没有范围证明，

则一个拥有 5 XMR 的恶意的用户可用一对包含 +13 XMR 和 -8 XMR 的输出创建一个交易。

2. 发送人还能证明输入与输出平衡，这是十分厉害的技能，因为环签名包含诱饵，以防止验证方知悉输入资金的真正来源！同态的佩德森承诺使发送人能够证明其中一个潜在的输入与输出之间的差异为零，而不会揭示过程中的数量。

为进行简单的类比，请考虑以下等式示例。类似掩蔽交易金额，你可验证每个等式是否有效，而无需知道 A 值。

A = 我们的输出, 无人知晓A值

`5A + 1A + 4A = 10A` 真实! 可以在不知道A的情况下证明等式验证，

`6A + 4A + 2A = 14A` 假！未通过验证，拒绝！

5.4.3 环签名

门罗币利用环签名技术来保护每个交易发送人的隐私。环签名是一种加密签名，允许一个有效参与者代表一个组对消息进行签名。有效签名者拥有的私钥与来自其他成员的公钥信息混合，以产生单个签名。任何人均可对照公钥验证签名的消息，以验证某个环成员发起了签名，但是无法确定哪个成员贡献了私钥。

在门罗币背景下，消息是由环签名授权的交易。实际支出的输出是真正的签名者，来自其他输出（来自过去的交易）的公钥受混合作为诱饵签名者。实际签名者和诱饵签名者在数学上同等有效；不可用密码检查得到的环签名来确定哪个成员主动发起了签名。因此，任何外部方（包括接收人）均无法确定交易中引用的输出中的哪一项被实际支出了。

每个环签名产生一个单一密钥镜像，该图像是从实际支出的输出中派生而来。这是一个加密安全的过程：每个输出对应一个密钥镜像，且生成密钥镜像并不能揭示环中真正的签名者。

当输出的所有者在新的交易中支出该输出时，网络将存储由环签名产生的密钥镜像。由于网络无法识别哪些输出被支出了，反而会跟踪那些被支出的密钥镜像！如果所有者试图欺诈性地再次支出该输出（双花），则将会产生相同的密钥镜像，因此网络会拒绝交易。

让我们深入研究生成环签名的实际数学运算。在本示例中，假设 HS 是返回标量（在适当的字段中）的散列函数，HP 是返回点（在适当的曲线组中）的散列函数。我们有意避免正式定义这些域和上域，以避免复杂化。让 G 成为各方均知晓的固定点。

你将使用环签名在交易消息M上签名。门罗币目前要求每个签名有 11 个环成员，但是应考虑一个有 3 个环成员的简化示例。你拥有所支出的输出的密钥对（公开和私有），

并选择另外两个输出（及其公钥）作为诱饵。自然而然，环成员的索引应为随机性，因为如果真正的签名者总是在 1 号币口中，则密码匿名将被规避。对于有 3 个环成员的简化示例，假设你的钱包已随机选择将资金的真正来源放在 2 号币口中。

你从区块链检索诱饵（P_1和P_3）的公开输出密钥，且你拥有所支出的输出的私钥（p_2）和公钥（$P_2=p_2G$）。你先选择一个稍后会丢弃的随机数 u。首先你构建以下承诺，从你为密钥选择的一个索引开始：

$c_3=Hs(M,uG,uHp(P_2))$

为构建其余的承诺，你还可选择稍后需要的随机数s_3和s_1：

$c_1=Hs(M,s_3G+c_3P_3,$

$s_3Hp(P_3)+c_3p_2Hp(p_2))$

请注意，此处包含了几条信息：你从区块链提取的公钥 P_3、你想出的随机数 s_3、之前的承诺c_3以及由你自己的密钥形成的值$p_2Hp(P_2)$。继续：

$c_2=Hs(M,s_1G+c_1P_1,s_1Hp(P_1)+c_1p_2Hp(P_2))$

但是你还未完成！为隐藏实际密钥，你巧妙地定义 $s_2= u-c_2p_2$。你发送到区块链和世界的签名包含几个数量：（c_1 , s_1 , s_2 , s_2 , J），

其中 $J=p_2Hp(P_2)$ 是每个承诺中使用的关键图像。我们在此将其重新命名，以强调一个事实：公众对构建承诺的碎片不了解。

这便是高明之处：通过设置 $s_2=u-c_2p_2$，你可重新排列来得出 $u=s_2+c_2p_2$。这意味着公众认为你构建的第一个承诺 c_3 如下：

$$c_3=Hs(M, s_2G+c_2P_2, s_2Hp(P_2)+c_2p_2Hp(P_2))$$

这看起来和其他承诺完全一样！尽管你从未广播 u，但你用它巧妙地让每个承诺在观察者眼中看起来都一样。这便是环签名的力量。无人能确定哪个承诺隐藏了你真正的密钥，但是每个人均可采用数学方法来证明：

1. 发送人知晓由公钥表示的一个私钥

2. 密钥镜像计算正确

观察到密钥镜像 $J=p_2Hp(P_2)$ 是从真实输出的密钥对中唯一计算出的、无任何随机数或诱饵的公钥。因此，任何试图欺诈性地第二次支出这笔输出（双花）的情况都将生成相同的密钥镜像。由于网络可跟踪使用的密钥镜像，因此任何重用输出的尝试均会被轻易地检测到并被拒绝。

请注意，上述 Back 类型 LSAG 环签名示例是出于培训目的而纳入，不应用作产品实现的参考文档。

5.4.4 更多资源

如果想更深入地研究这些技术背后的计算，可以看看《Zero to Monero》，这将是一段高技术性地数学旅程，也是一个社区资助的免费 PDF。

5.5 门罗币区块链

现在你已熟悉区块链作为分布式公开账本的重要性和实用性。这些区块被结构化并排序到不可变的只加数据库中，由防止任何篡改或欺骗的加密工具进行保护。对于该独一无二的门罗币区块链，我们将会在本节中讨论其技术和规格。

5.5.1 闪电记忆映射数据库

门罗币使用闪电记忆映射数据库（LMDB）系统来存储其区块链。LMDB 是一个软件库，可以密钥值存储形式提供高性能嵌入式交易数据库。这意味着其非常高效，且易于搜索。

- 任意密钥/数据对存储为字节数组
- 基于范围的搜索能力
- 支持具有多个数据项的单个密钥
- 在数据库末尾添加记录的高级方法，与其他类似存储相比，

该方法大大提高了写入性能

5.5.2 区块结构

CryptoNote 标准定义了在区块内和区块链上存储和描绘数据的规范。区块结构包含三个主要部分：

- **区块头**
- **基础交易（一种产生新币的交易）**
- **交易标识符列表（在区块中挖掘出的交易散列）**

5.5.2.1 区块头

每个区块均以包含密钥元数据的区块头开始。字段“major_version”定义了区块头解析规则，因此其可获得正确解释。字段“minor_version”定义了与主区块头解析无关的解释细节。

即使“minor_version”未知，用特定“major_version”解析区块头也始终安全。用未知的“major_version”解析区块头有风险，因为区块头内容可能会受到误解。

Field	Type	Content
major_version	varint	Major block header version
minor_version	varint	Minor block header version

»

timestamp	varint	Block creation time (UNIX timestamp)
prev_id	hash	Identifier of the previous block
nonce	4 bytes	Any value which is used in the network consensus algorithm

5.5.2.2 基础交易

每个有效区块包含一个单一的基础交易，该交易将其 coinbase 奖励发送给矿工。基础交易必须遵循发币规则，并包含区块高度字段。

5.5.2.3 交易标识符列表

Field	Type	Content
version	varint	Transaction format version
unlock_time	varint	UNIX timestamp.
input_num	varint	Number of inputs. Always 1 for base transactions.
input_type	byte	Always 0xff for base transactions
height	varint	Height of the block which contains the transaction
output_num	varint	Number of outputs
outputs	array	Lists of outputs as array

基础交易之后是交易标识符列表。这些标识符是通过获取交易本身的 Keccak 散列来计算。列表开头为标识符数量，后面为标识符本身（如果区块不为空）。

5.5.2.4 区块标识符的计算

区块标识符通过用 Keccak-256 散列下列数据产生：

- 区块头的尺寸
- 区块头
- Merkle 根哈希
- 交易数量（变量）

Merkle 根哈希将区块主体中引用的交易“添加”到区块头：一旦 Merkle 根哈希固定后，交易便不能被修改。该安全特性使区块链免受篡改或任何形式的追溯性修改。

5.5.3 挖矿经济

第 2 章和第 4 章从概念上提及区块奖励和费用。现在，你将真正了解区块大小、奖励以及与费用关系的复杂性。5.5.3.1 挖矿 coinbase 奖励

正如第 4 章所讨论那样，所有门罗币均是作为对矿工成功完成区块奖励而产生。

该 coinbase 奖励支付的规模取决于当前供应量（A）和原子单位的初始数量（$S=2^{64}-1$）。原子单位是门罗币目前可由网络（$1x10^{-12}$ XMR）识别的最小单位

$$基础奖励 = 2 * ((S - A) * 2^{-20} * 10^{-12})$$

门罗币有一个“尾部发行”，这是一个小的固定基础奖励，将在大部分供应开采后继续进行。门罗币的最低基础奖励是每区块 0.6 XMR，因此矿工们将永远不必仅依靠交易手续费维生。

5.5.3.2 动态区块大小

与许多使用静态（固定）区块大小的加密货币相比，门罗币动态区块大小可随着网络的增长而不断调整。例如，比特币最初的 1 MB 固定区块大小会通过限制每个区块中可包含的交易数量（从而限制网络的总交易量）进行缩放。2017 年，该瓶颈导致了极高的费用和交易处理的延迟。因此针对该问题提出了各种拟议的解决办法，导致在一段时间内出现了有争议的辩论。

为避免这些问题，门罗币采用了动态区块大小机制，允许矿工使用更大的区块来容纳增加的流量。但是，如果区块大小完全不受限，则门罗币网络可能容易受到垃圾交易攻击，即，会有许多小额交易通过使区块链扩张过快来耗尽网络和存储资源。

为防止区块大小过度增长，

门罗币挖矿协议包括一个惩罚函数，以降低对超大区块的 coinbase 奖励。最初的 CryptoNote 作者纳入了该共识规则，以限制区块大小扩展的速率，避免快速的区块链膨胀。

如果所开采区块的大小（B）大于最后 100 个区块（M_N）的中值大小，将会根据以下条件扣留部分基础奖励：

惩罚 = 基础奖励*((B/M_N)-1)^2

矿工可获得全部奖励只要区块尺寸在 300 kB 以内；对于任何更大的区块，惩罚函数“开始生效”。最大区块大小为 2*M_N，此时所有 coinbase 奖励都会被扣缴。

5.5.3.3 费用

当交易量较低、区块较小时，矿工会以最低的费用获得全部 coinbase 奖励。

但是，想象一个不同的场景：当最后 100 个区块的中值大小变得大于无惩罚区块尺寸（300 kB）时，将会发生什么？此时，动态费用算法开始发挥作用！

费用根据交易的权重（kB）计算。较大（“较重”）的交易会产生较高的费用。考虑到门罗币生态系统的几个因素和交易的优先级，动态费用的计算较复杂（发送人可通过追加更高的费用来激励矿工快速纳入紧急交易）。为在即将到来的区块中具有竞争力所需的费用根据以下公式计算：

每 kB 费用=(R/R₀)*(M₀/M)*F₀*(60/300)*4

• R 为基础奖励

• R₀ 为参考基础奖励（10 XMR） • M 为区块大小限制

• M₀ 为最小区块尺寸限制（300 kB） • F₀ 为 0.002 XMR

• 60/300 是一个调整系数说明无惩罚区块尺寸限制的提高（2017 年从 60 kB 调整到 300 kB）

• 4 是一个调整系数说明默认费用乘数（最低费用水平使用 x1 的乘数，正常优先级交易使用 x4）

因此，费用考虑了相对于最小区块大小的中间区块大小的增加。例如，600 kB 区块大小（最小区块的两倍）可将费用降低一半。

理想情况下，提高门罗币的汇率和使用率会导致绝对费用（即，依据 XMR）减少。这种费用减免机制在价格大幅上涨时效率较低，价格涨幅远远大于交易量的涨幅（因此也大于区块大小的涨幅）。

动态费用算法设计为在中值区块大小始终高于 300 kB 时起作用。尽管该系统旨在解决价格上涨的问题，但使用率与价格并不完全相关，因此这并非一个完美的指标。

5.5.4防弹协议

防弹协议是一种新技术，其极大地减小了交易尺寸，

进而降低了每笔交易的总费用！门罗币交易尺寸曾经相当大（通常大于 12 kB），因此防弹协议曾是一个备受期待的增强功能。

为防止滥用和垃圾交易，门罗币的隐私特性需要在交易验证期间进行几项复杂的“测试”。这包括核实隐蔽金额、查询费用以及确保没有发生双花。

大多数开发人员都遇到过“溢出”错误，即操作创建值超出了可表示范围。不幸的是，“无限”对于电子产品而言是一个抽象的概念，其会遇到许多巨大的障碍。

由于 RingCT 隐藏了交易金额，因此需要复杂的计算来验证输入和输出是否正确地平衡。承诺的有用代数性质对于实现任何参与者均可确认其有效性的屏蔽交易而言具有价值性。

但是，确保每个金额均为正值，而不会导致溢出，这也是至关重要的因素。这便是范围证明的作用，即通过允许任何人验证一个承诺，此承诺代表一个特定范围内的金额，验证不会透露这个值的任何其他信息。每个范围证明过去需要大约7 kB，因此它们构成了交易的大部分。大多数交易有两个输出（目的地和更改地址），需要至少约 12 kB。

防弹协议采用一些巧妙的数学技巧clever mathematical tricks，以更高效的机制more efficient mechanism构建范围证明。这可将单个范围证明大小减少到 2 kB 左右！

在防弹协议之前，具有多个输出的交易需要有多个独立的范围证明。

因此，交易尺寸与输出数量成线性比例（例如，1 个输出=7 kB，2 个输出=14 kB）。凭借防弹协议，交易尺寸反而会随着输出增加（例如，1 个输出=2 kB，2 个输出=2.5 kB）以对数形式缩放。

通过减小每个范围证明的大小，并允许其以更有效的方式组合，防弹协议极大地减小了交易规模，从而降低了交易费用。2018 年 10 月，门罗币 v0.13.0 网络升级启用了防弹协议，作为一个可选的功能，在后继的升级中将会强制实行。

社区和贡献

我们欢迎您对代码或生态系统的其他方面做出的贡献。 本章提供了我们的去中心化结构的详略论述，并包含了参与的方式和链接。

6.1 社区文化

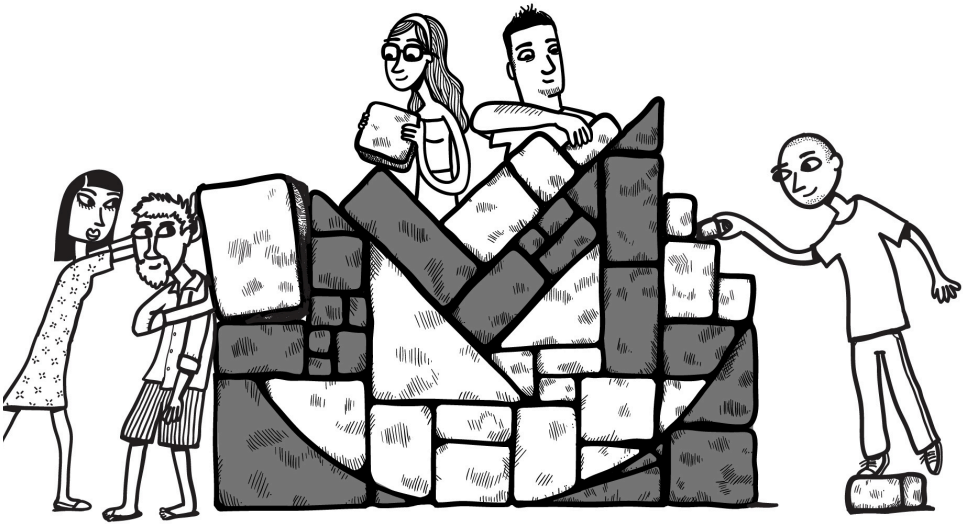
6.1.1 开放

虽然门罗币本身是隐私币的典范，但其社区是建立在透明与协作这一核心价值观之上的。用户、开发者和研究者所在的中继聊天频道（IRC channels）也是完全开放给公众的。你也可以通过Slack、Mattermost和Taiga等平台与门罗的用户和开发者交流互动。所有重要的会议记录都可以在官方网站上找到official website。

门罗开放协作的文化需要从门罗的渊源讲起。门罗源自ByteCoin的代码分叉，而Bytecoin的开发者一直保持着神秘的身份，制定决策也全然无视社区的反馈。这不仅导致了开发出错，更过分的是其引火自焚的预挖设置。

门罗社区在对幽暗隐蔽的ByteCoin进行代码分叉的那一刻诞生，在去中心化、协作性和多元化的光辉下成长。这无疑在方方面面促进了门罗的成长，

门罗社区也从中领悟到协作共荣的理念。如果说加密学是门罗的技术支撑，社区则是门罗发展的力量来源。



6.1.2江山代有才人出

门罗项目是由全球数百人齐心协力，倾心奉献的结晶。在写稿的此刻，累计有超过500位人士对门罗的代码做出过贡献，其中仅去年就有200位。门罗采用非政府性治理（un-governance）的机制来协调开发等工作。项目由几大分支组成：门罗核心团队（Monero Core Team），门罗研究实验室（Monero Research Lab），门罗工作组（MoneroWorkgroups）和社区。

门罗核心团队负责以下关键事项：·作为论坛众筹系统（Forum Funding System，FFS）的社区代理人和首要仲裁人：

- 作为门罗收到的捐款资金的管理人，确保这些资金用于门罗的发展。
- 管理门罗项目的代码库：合并代码，保存备份，保障安全和开放性
- 担任普通捐赠基金的管理员，指导其资金进一步推进Monero项目。
- 担任Monero软件和相关技术的可信赖的签名者和分发者。
- 与社区合作，识别领导Monero项目的愿景和路线图。

门罗研究实验室从基础理论和应用两个层面，研究和分析最前沿的机密货币技术。实验室中有数位学者和研究人员，所有的研究也在https://lab.getmonero.org/这一公众平台上发布。https://lab.getmonero.org/

门罗工作组（**The Monero Workgroups**）是专门为一个个任务的协作而运行的，成员围绕数个任务组成数个小组。例如，门罗硬件工作组正在全力打造第一社区驱动的、开源的硬件钱包。另一个例子是门罗集成工作组，他们开发开源支付网关。你还可以加入门罗翻译组，门罗见面会组织小组等。（translating Monero, crafting kits for Meetups）

门罗的发展离不开它的社区，我们诚挚欢迎各位加入社区，为门罗贡献代码，提案，众筹fund proposals，宣传help with outreach和撰文。

6.2 编程文化

6.2.1 创建拉取请求来改进门罗

我们欢迎所有有意为门罗代码库做贡献的人。如果你想要修复或修改代码，请创建一个拉取请求（pull request）并提交至“master”分支。你可以参照下列步骤，通过引用代码副本，并在编辑后提交，来修缮门罗代码：

1. 引用（fork）GitHub上的程序库（repository）
2. 复制程序库至本地
3. 创建分支，做必要的修改
4. 对改动进行清晰的描述
5. 执行git push origin brach-name，将本地的改动同步至引用的程序库
6. 创建拉取请求（包含清晰的描述和档案），并提交至原始程序库

如果你的改动比较轻微，或者没有影响到代码库的其它部分，那么你的改动通常会在短时间内被通过。如果你的改动很大很复杂，那么则需要经过社区充分的讨论。

在GitHub上提交拉取请求时，请确保您的分支已重新定位。 避免离开流浪并合并您将提交的分支机构中的其他程序员的提交。 如果存在冲突（即使它们可以轻易解决），可能会要求您进行变更。

6.2.2 打补丁的礼节

希望大家按照以上步骤，以拉取请求的方式提交补丁。如若不行，请以git format-patch的格式发送补丁（例如，粘贴至fpaste.org，并分享链接给#monero-dev on irc.freenode.net）。

补丁应该是独立自足的。一个优秀的经验法则是为每一个问题、功能或逻辑改变创建一个补丁。当你要对某部分代码进行修缮时，请遵循它原本的代码风格，并且不要进行不必要的编辑，比如空格(whitespace)或行首缩进(reindentation)，请进行压缩精简（squash）（例如，如果你补丁中要修复的问题也包含在下一个补丁中，请合并两个补丁）。

6.2.3 通用教程

提交的消息（commit message）应该是清晰且合乎逻辑的。标题中必须对补丁进行描述，另外可利用更多文字来提供细节，纪录等。富有意见性的代码能够积极地引导他人理解你所要表达的信息，并于之互动。如果你的改动中有新增功能，那么你可以在拉取请求中附上测试结果，这对他人非常有帮助。

如果你做了一些随机的不相关的改动（比如过于热心的编辑），你可以通过使用git add -p来选择要包含的改动，单步调试每一条编辑。这将有助于你创建一个干净简约的补丁。Git diff表示你在树（tree）中的改动，git diff --cached表示目正要被提交的改动。

添加了git add -p的hunks，将从git diff输出中迁移至git diff --cached输出，这样一来你就可以明了地看到你的代码提交会是什么样的。

有关通用流程的更具体指导原则在Monero项目的官方存储库official repository中有所描述。

6.2.4 Monero的存储库

Monero Project GitHub上托管了许多不同的存储库。 其中一些包含我们已经在Mastering Monero中讨论过的组件，例如：

- Monero: 门罗币主网 （包含 CLI钱包, 用 C++ 语言完成）
- Monero-site: 社区网站 https://getmonero.org 的源代码
- Monero-GUI:Monero的图形用户界面，使用Qt库
- kastelo: 社区硬件钱包
- kovri: Kovri匿名路由器

这些项目都被很好地纪录下来，所以你可以熟悉它们的代码并进行改进。此外，门罗还有很多子项目，你可以参与其中。欢迎随意浏览repositories一个程序库，看看仍待处理的问题open issues中，有没有是你可以参与贡献的。

```
$ This text is a terminal command. Don't run this
command if you don't know what you are touching.
```

注意：在写稿的此刻，*门罗生态*的部分程序*库*正从*GitHub*迁移至*GitLab*。

6.3门罗开发简介

因为开发门罗代码是一个复杂的过程，我们在这里提供一些建议和总结。因为Linux系统拥有内置的壳（注：shell，用来区别于核，是指“为使用者提供操作界面”的软件），来帮助开发门罗内核，所以请优先考虑使用基于Unix的操作系统。门罗是通过C++编写的（C++11风格referenciation）。

6.3.1下载门罗源代码

门罗使用Git作为版本管理的工具；Git允许开发者追踪代码中的改动和修缮，使得共享文档的协作变得简单容易。执行下面的命令，以下载门罗源代码：

```
$ git clone --recursive https://
github.com/monero-project/monero
```

6.3.2 Dependencies

从源码构建门罗，系统路径需要包括以下依赖组件（如表格所示）。部分依赖库已经包含在本代码库（标记为外部编译[vendored]）。默认情况下，项目构建将优先使用系统自带的库，除非在系统路径中没有找到对应的库，才会使用外部编译的库。

GCC	libunbound	ldns
CMake	libsodium	expat
pkg-config	libminiupnpc	GTest
Boost	libunwind	Doxygen
OpenSSL	liblzma	Graphviz
libzmq	libreadline	pcsclite

6.3.4 开发指南

门罗使用CMake构建系统和一个顶级的生成文件（Makefile）来调用需要的cmake命令。一旦你安装了依赖部件（dependences），改变源代码目录的根（root）并执行make命令以开始构建。这个过程将耗费1~2小时，当代码完成构建后，你可以在构建文件夹中找到门罗的二进制文件。

6.3.5 Build故障排除

如果你碰到出错，输出会指明到底是哪里出错了。以下是一些常见的bug：

- 过期的boost版本（你可能需要手动安装一个现在的版本）

- 过期的gcc/g++
- 丢失 libzmq3-dev
- 丢失 libreadline-dev
- OpenGL 出错

你可以（可选）输入make debug来编译一个除错构建（debugging build）。有众多社区可以帮助你排除故障。通过搜索引擎查询构建错误，你可以获得解决方案或他人的帮助。

6.3.6 构建门罗图形化用户界面（GUI）

门罗图形化用户界面是通过C++和Qt library构建的，两者缺一不可。有了依赖，你可以通过以下命令克隆和构建GUI：

```
$ git clone --recursive https://github.
com/monero-project/monero-gui
$ cd monero-gui
$ ./build.sh
```

面向开发者的门罗集成

本章内容涉及包含一系列门罗币公链的标准和协议，开发人员可以用本章知识来参与门罗币公链的交互以及构建新工具。本章将首先将介绍，OpenAlias别名系统和Monero URI格式，这部分内容是达成高效通信和获得其他关键细节方法的关键。本章的其余部分通过列举在C ++语言和python环境里的真实案例来探讨远程程序调用（RPC）。

7.1 OpenAlias: (**对人类而言**)**易用的文本型地址**

除非是过目不忘的人，不然直观读取和记忆加密货币的地址是非常困难的。与诸如“ 45ttEikQEZWN1m7VxaVN9rjQkpSd- mpGZ82GwUps66neQ1PqbQM no4wMY8F5jiDt2GoHzCtMwa7P- DPJUJYb1GYrMP4CwAwNp ”这样的门罗币地址相比，生活中的街道名称，比如“ 某某某大街123号 ”，或者电子邮件地址“ donate@masteringmonero.com ”，要容易识别和方便使用的多。

加密货币的地址包含了许多信息，但是对人类而言，太粗苯繁琐了。而事实上也已知有一个著名的三元悖论——既Zooko不可能三角*Zooko's triangle*：当设计命名系统时很难同时在以下三个方面同时满足该命名系统：安全性；去中心化；（人性化)易于理解

类似上面那样的门罗币地址，虽然可以满足Zooko不可能三角中的安全性与去中心化

，但却是人类难以直接理解的，不够人性化。门罗币公钥地址至少有95个字符长，难以阅读，几乎不可能记住。所以一定得有一种简化的解决方案！

门罗币核心开发团队发布了OpenAlias标准，通过关联人类易于识别的名称与门罗币原始地址，来突破Zooko不可能三角。OpenAlias这套标准是一个记录在“完全限定的域名（fully-qualified domain name FQDN）”上的DNS文本。每个DNS文本仅需包括两部分信息：前缀和收件人地址。也可以加上收件人名称键值对(可选项，不是一定得加)。如下是一个典型的OpenAlias文本记录：

```
oa1:xmr

recipient_address=45ttEikQEZWN1m7VxaVN9rjQkpSd-
mpGZ82Gw Ups66neQ1PqbQMno4wMY8F5jiDt2GoHzCtMwa7P-
DPJUJYb1GYrMP4C wAwNp

recipient_name=MoneroFFS
```

这个例子中的“oa1:xmr”，说明本条记录是基于OpenAlias第1版、其目标地址是属于门罗币。“recipient_name”收款地址可以更改成指定收件人名称，在本例中为“MoneroFFS”。

字段名称	字段长度(byte)	字段说明
oa1:	4	DNS文本记录总是以“oa1:”开头，以表明遵循OpenAlias第1版规范。如果没有这个开头，那么就忽略这段文本，因为在此种情况下，这可能是一条SPF记录，或者其他什么无关文本。

»

Name	Size in byte	Description
symbol	3	加密货币的代码，应遵循ISO4217规范。例如：门罗币的缩写是xmr，而比特币的缩写是btc。
recipient_address = address;	17 + address + 1	接收地址。 格式为recipient_address = your_address; 其中your_address是您的加密地址。 对于Monero，它将是一个95个字符的字符串(key-value pairs)。 键值（key-value）对由分号(semi-colon)分隔，并且可选地可以用空格以便于易读性。必须存在此键值。 OpenAlias的存在是将FQDN别名为任何类型的“ 地址 ”，并以此值表示。
recipient_name = description;	14 + description + 1	这个键值对不是必选项，但可以帮助用户确认接收方，或者便于用户维护一个地址簿。

对于开发者而言，OpenAlias是一项可扩展的标准。而且对用户很直观，可以在去中心化与中心化系统之间互操作。该标准可以用于任何加密资产，事实上已经在门罗币、比特币(bitcoin (Electrum))和HyperStake项目中实施。

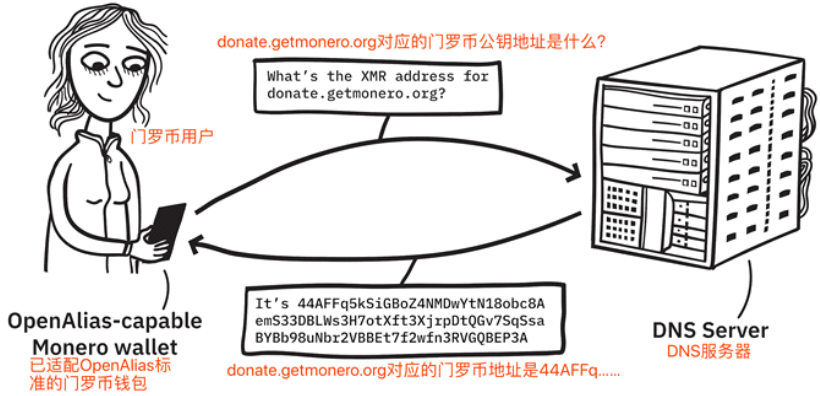


图7.1-用户可读的donate.getmonero.org由DNS服务器解析，该服务器发回捐赠地址44AFFq5kSiGBoZ4NMDwYtN18obc8AemS33DBLWs3H7otXft3XjrpDtQGv7SqSsaBYBb98uNbr2VBBEt7f2wfn3RVGQBEP3A。

7.2 Monero_URI: 便于(计算机)处理的文本信息

门罗币统一资源标识符(URI)标准规定了关键数据的格式。交易票据和交易本身的确认就依赖于准确传递确认关键数据字段。统一资源标识符URI特别适用于商品推销，例如利用URI生成付款的QR码。

门罗币URI语法遵循RFC 3986，其中空格依据x-www-urlencoded格式编码为%20。以下URI示例的含义是为“Mastering Monero book”支付0.0413XMR到公钥地址4BKq...feW5。

```
monero:4BKjy1uVRTPiz4pHyaXXawb82XpzLiowSDd8rEQJGqvN6AD6kWosLQ6VJXW9sghopxXgQSh1RTd54JdvvCRsXiF41xvfeW5?tx_amount=0.0413&tx_description=Mastering%20Monero%20Book
```

参数	数据类型	说明
address	字符串	接收门罗币的原始地址(raw address)
tx_payment_id	字符串	(如果有需要的话)建议关联交易识别码(payment id)
recipient_name	字符串	(如果有需要的话)建议关联收款方名字
tx_amount	浮点型	原子货币单位的交易金额
tx_description	字符串	描述哪项交易应当被发起

用此二维码向门罗开发者基金捐助0.05XMR，检测URL应该会显示以下信息：monero:44AFFq5kSiGBoZ4NMDwYtN18obc8AemS33DBLWs3H7otXft3XjrpDtQGv7SqSsaBYBb98uNbr2VBBEt7f2wfn3RVGQBEP3A?tx_amount=0.05&tx_description=Donation%from%Mastering%20Monero%20Book

7.3 门罗币RPC

开发者对门罗币项目的集成，可以通过使用 Monero's C++ API (C/C++)或远程过程调用(RPC)接口。任何能发出HTTP请求的编程语言，都可以用来访问RPC，因此使用RPC方法十分灵活， 这种灵活性会带来一系列好处和便利，我们将举例一些常见的任务如何利用这种灵活性编程代码。

调用门罗币守护进程(monerod)的RPC接口，可以执行检查余额与支付等关键操作。门罗币钱包RPC(monero-wallet-RPC)则允许开发者通过传入JSON字符串来使用其全部功能。

在RPC中门罗币数量通过目前门罗币最小单位"原子单位（atomic unit）"表示，换算关系如下:

1 XMR = $1x10^{12}$ atomic units

7.3.1 初始化与配置(安装与安全)

首先，启动monero-wallet-rpc(门罗币钱包RPC)，并指定所用端口与钱包文件对应地址。

```
$ ./monero-wallet-rpc --rpc-bind-port 18082
--disable-rpc-login --log-level 2 --wallet-file
your-wallet-file --prompt-for-password
```

如果你希望使用远程节点， 只需添加`—daemon-address`标识所要连接的地址，

例如:

```
--daemon-address node.moneroworld.com:18089
```

因为门罗币钱包RPC没有指定默认的IP地址或端口， 需要加上`--rpc-bind-ip ${你的ip}` 才能进行远程链接。(实际上还需要加上—confirm-external-bind —译者注)

建议开发者采取一些安全预防措施，因为使用开放式RPC接口进入程序生成阶段的（风险程度）就像是在没有保护的情况下进行野外探险！ 请务必确认在暴露你的节点之前设置了用户名与密码。如果你遵循上面步骤并采取了适当的安全设置， 你的 API应该是安全的。

`--restricted-rpc`标识对于保持链接私密与避免潜在的不当使用非常有用。例如，这种模式将保证你的节点不会通过RPC反向传输敏感数据，并阻止外部使用者使用你的设备挖矿。

7.3.2 JSON-RPC 格式

JSON-RPC是一个无状态，轻量的RPC协议，使用JSONRPC4672格式。这个格式主要包含几个数据结构与相应的处理规则。这个协议是“传输方式无关”的，也就是说它的功能不取决于通信方式。所以，无论是进程内通信，通过socket通信，通过HTTP链接通信或者其他任何通信方式中都可以使用同一规则。

为了正常使用钱包RPC，你需要通过POST方法提交数据。JSON-RPC API接收如下格式的消息

```
{ “jsonrpc” : version , “method” : method, “params”: params, “id”: id }
```

各项的解释如下:

Field	Description
域	解释
version	JSON RPC 协议版本 (门罗币使用v2.0)
method	声明要调用的功能
params	其他调用功能所需信息
id	用来追踪回复的数字(从零开始的整数)

7.3.3 RPC调用示例

门罗币RPC可以如下面的例子一样，直接在终端中调用。门罗币官网通过wallet RPC和

daemon RPC文档给出了规则与功能定义。

7.3.3.1 获取余额

可以通过以下方法查询余额:

```
$ curl -X POST 127.0.0.1:18082/json_rpc -d
'{“jsonrpc”:”2.0”,”id”:”0”,”method”:”getbal-
ance”}' -H 'Content-Type: application/json'
```

操作返回两个结果，（总）余额(balance)与可用余额(unlocked_balance)。unlocked balance是已经在主链上获得足够多的全节点确认，可安全支付的余额(比方说已获得6个确认的交易结果)。

```
{“id”: “0”,“jsonrpc”: “2.0”, “re-
sult”: {  “balance”: 140000000000, “un-
locked_balance”: 84000000000} }
```

在这个例子中，这个钱包包括0.14个XMR，并且只有0.084个可用。

7.3.3.2 获取地址

查询钱包的地址:

```
$ curl -X POST 127.0.0.1:18082/json_rpc -d
'{“jsonrpc”:”2.0”,”id”:”0”,”method”:”getad-
dress”}' -H 'Content-Type: application/json'
```

返回:

```
{“id”: 0,“jsonrpc”: “2.0”,“result”: {“ad-
dress”: “42uMGYwvLuUGJzqdWZvr47CGCBz1qNNEx-
ZeegcjLPMbaFkBb3XG g6Y1bUwaMbovzGWDXtaASxS-
BYtaiBB4wuDmrAMCygexH”, “addresses”: [{
“address”:  “42uMGYwvLuUGJzqdWZvr47CGCBz1qN-
NExZeegcjLPMbaFkBb3XG g6Y1bUwaMbovzGWDXtaASx-
SBYtaiBB4wuDmrAMCygexH”, “address_index”: 0,
“label”: “Primary account”,“used”: false
            },
            {
                “address”: “894PaGJyxRjZU8nP-
7Dh4FuAyzr2dK3VT9ZZX95MxdAGP3HoHEpA bNb8Htg-
p5LKzc1pXQ8zhpokTZtcUTnzeU823oUPUGSpv”,
                “address_index”: 1,
                “label”: “”,
                “used”: false
            },
                        ]
   }
}
```

7.3.3.3 生成地址

为一个账户生成新的地址。（可选功能）标记新的地址。

```
$ curl -X POST 127.0.0.1:18082/json_rpc -d '{“json-
rpc”:”2.0”,”id”:”0”,”method”:”create_address”,
“params” : “{“account_index:0,”label”:”Secondary
account”}}' -H 'Content-Type: application/json'
{
   “id”: 0, “jsonrpc”: “2.0”, “result”: {
    “address”: “86KoCQsZHQvSUnp9fFn-
92e5QGUiZtH1qZ1nNx1Jv5eJs94ywbLR2k 11CjZ-
Tq5o4v8j9bx3CEAturCheJqJR7cYdQKT4xE3w”,
“address_index”: 9
   }
}
```

7.3.3.4 生成新账户 ———生成一个新账户

```
$
'{“jsonrpc”:”2.0”,”id”:”0”,”method”:”cre-
ate_account”,  “params”:{“label”:”Secondary ac-
count”}}' -H 'Content-Type: application/json'
```

```
{
  “id”: “0”,
  “jsonrpc”: “2.0”,
  “result”: {
    “account_index”: 1,
   “address”: “88bV1uo76AaKZaWD389kCf5Ef-
PxKFYEKUQbs9ZRJm23E2X2oYgV9b Q54FiY-
6hAB83aDXMUSZF6KWyfeQqzLqaAeeFrk9iic”
  }
}
```

7.3.3.5 转账

转账（发送）一定量的门罗币时，需要以“原子单位”为单位。

```
$ curl -X POST http://127.0.0.1:18082/json_rpc -d
' {“jsonrpc”:”2.0”, ”id”:”0”, ”method”:”transfer”,
”params”:{“destinations”: [{“amount”:100000000,
 ”address”:”9wNgSYy2F9qPZu7KBjvsFgZLTKE2TZgE-
pNFbGka9gA5 zPmAXS35QzzYaLKJRkYTnzgArGNX7T-
vSqZC87tBLwtaC5RQgJ8rm” }, {“amount”:200000000,
 ”address”:”9vH5D7Fv47mbpCpdcthcjU34rqiiAYRCh1tYy-
wmhqnE k9iwCE9yppgNCXAyVHG5qJt2kExa42TuhzQfJbmb-
peGLkVbg8xit” }],”mixin”:4,”get_tx_key”:
true}}' -H 'Content-Type: application/json'
{
  “id”: “0”,
  “jsonrpc”: “2.0”,
  “result”: {
    “fee”: 48958481211,
“tx_hash”: “985180f468637bc6d-
2f72ee054e1e34b8d5097988bb29a2e0cb 763e4464db23c”,
    “tx_key”: “8d62e5637f1fcc9a8904057d6bed-
6c697618507b193e956f77c 31ce662b2ee07”,
    “amount”: 300000000,
    “tx_blob”: “”,
    “tx_metadata”: “”,
    “multisig_txset”: “”
  }
}
```

7.4门罗币的集成实践（Python和C++语言环境）

挑选这些示例挑选程序语言是棘手的，每一个程序员都知道世上没有一个程序语言适用于所有场景。然而Python是一个对初学者很友好且易于理解的脚本语言，并且python免费、开源，用来学习门罗币再适合不过了。

下面的例子都是基于最新的Python3编写的。
大多数基于Debian的Linux发行版预装了Python 2和Python 3。在开始之前，你应该更新升级你的软件，确保接下来使用的程序都是最新的版本：

```
$ sudo apt-get update && sudo apt-get -y upgrade
```

以下教程的代码可在公共存储库中免费获得。 您可以使用以下命令通过功能强大的'git'控制版系统直接下载练习：每个教程都位于一个文件夹中：

每个教程都有一个单独的文件夹，例如：“Tutorial 1”（教程1）的文件夹被命名为“tutorial-1”（教程—1），你可以使用版本控制系统“GIT”来下载教程资源，执行以下语句：：

```
$ git clone https://github.com/monerobook/code
```

7.4.1 教程1-获取你的余额

这个程序会通过RPC的方式连接后台进程，查询并打印出账户余额.通过教程1的学习，你应该还记得get_balance这个函数（这个函数也可以用get_balance表示）。

我们先导入“requests”和“json”这两个指令到Python库，它们用于构造HTTP的POST请求十分有用。

```
# Mastering Monero Tutorial. This is a comment
import requests
import json
```

为了避免使POST请求过于杂乱，我们先定义一些变量来存储部分信息:

```
## Import Setup variables
## Url for JSON RPC interface. We assume that your RPC
interface is running on localhost port 18082
url = “http://localhost:18082/json_rpc”

## JSON headers . Required
headers = {'content-type': 'application/json'}

## RPC input . Adding method name , at the moment we don't
need variables.

rpc_fields = {
       “method” : “get_balance”
}
```

在一个RPC调用过程中，应该包含以下字段:

```
# Adding the JSON RPC version and id. Id is a int variable
which should be incremented each request. First request
is 0 , second is one and ...
rpc_fields.update({“jsonrpc”: “2.0”, “id”: “0”})
```

万事俱备，只欠东风！现在只需要通过requet库的post方法把所有的变量发送到接口就大功告成了:

```
# execute the rpc request
response = requests.post(url,data=json.dumps(rpc_input),-
headers=headers)
# print the response as JSON
print(json.dumps(response.json()))
```

将上面所有的代码保存为tutorial.py（你也可以自己命名）并执行：：

```
$ python tutorial.py
```

此时你应该会得到以下的输出:

```
{
    “id”: “0”,
    “jsonrpc”: “2.0”,
    “result”: {
        “balance”: 0,
        “multisig_import_needed”: false,
        “unlocked_balance”: 0 }
}
```

尽管这些输出包含了我们所需要的所有信息，但是输出RPC语法让人们读起来不够人性化，这么多的“{}”括号会让我们眼花缭乱!

为了看起来更简洁，我们可以在底部添加几行代码运行教程脚本，以便它只打印余额（或可用余额，如果你愿意的话）

```
# Get the balance from response array and convert to a
string.
balance = str(response.json().get('result').get('balance'))

print(“Balance is “ + balance )
```

现在，运行以下代码

```
$ python tutorial.py
```

只会打印给我们想要的余额（balance）的值:

```
Balance is 426700000
```

你可以使用这样的方法开发属于自己的门罗币钱包客户端!

7.4.2 教程2-生成伪随机地址

在第五章中，我们介绍了伪随机地址的概念。在这一小节，我们将通过python来实现伪随机地址的生成，以便更深的理解它是怎么通过数学方法来实现的。

首先，我们如下图所示，导入必要的库并将它们添加到路径中。

```
# Import libraries. Hexlify for hex code, utils for the
utility, etc.
import os, sys
from binascii import hexlify, unhexlify
sys.path.append('../libraries')
import utils
import ed25519
import base58
```

编写函数generate_random_address，必须包含几个步骤:

1) Create your seed　创建你的seed。随机生成一个32 byte（256bit）的随机数来创建你的seed。使用hexlify库将你的seed转换成一个十六进制的字符串，将其保存在变量seed中。

2) Record your secret spend key　生成你的secret spend key。通过utils库中的方法sc_reduce32从seed中生成secret spend key，这样可以使其符合椭圆曲线签名算法ed25519。secret spend key是seed的简化版。

3) Calculate your secret view key　生成你的secret view key作为您的secret spend key的简化哈希值。 hash_to_scalar函数对输入进行哈希处理，然后将其转换为ed25519椭圆曲线的有效标量。

4) Derive public keys 生成你的public keys。通过方法publickey_to_private_key从你的secret keys中生成你的public keys。比如，你的secret view key被用来生成你的public view key。

5) Begin building your public address 通过连接网络字节（公共Monero地址为0x12），公共支出密钥和公共视图密钥，开始构建公共地址。 这些是每个Monero地址中包含的关键信息。

6) Calculate the checksum 通过获取其Keccak-256哈希的前4个字节（8个十六进制字符）来计算将附加到上述字符串的校验和。

7) Encode the info + checksum 进行编码。为了增加易读性，使用Base 58将变量data和校验和进行编码。至此，我们要做的已经做完了！就像我们在第五章讨论的那样，门罗币地址包括：

[network byte + public spend key + public view key + checksum]

```
def generate_random_address():
    ## Generate 32 bytes (256 bits) of pseudo-random data
    seed = hexlify(os.urandom(32))

   ## Reduce random data to make it a valid ed25519 scalar
    secret_spend_key = utils.sc_reduce32(seed)

    ## Use a reduced hash of the secret spend key for the
deterministic secret view key
   secret_view_key = utils.hash_to_scalar(secret_spend_key)
```

下一页内容继续 »

```
    ## multiply by the generator point to get public keys
from private keys
     public_spend_key = utils.publickey_to_privatekey(se-
cret_spend_key)
     public_view_key  = utils.publickey_to_privatekey(se-
cret_ view_key)
    ## the network byte, public spend key, and public view
key are all concatenated together
    ## 0x12 is the Monero mainnet network byte
    network_byte = “12”
    ## Concatenate the three strings
   data = network_byte + public_spend_key + public_view_key
    hash = utils.keccak_256(data)
    ## checksum is the first 4 bytes (8 hex characters) of
the hash of the previous data
    checksum = hash[0:8]
    address = base58.encode(data + checksum)

    ## Printing the keys

    print(“Secret_spend_key : “ + secret_spend_key)
    print(“Secret_view_key : “ + secret_view_key)
    print(“Public_spend_key : “ + public_spend_key)
    print(“Public_view_key : “ + public_view_key)

    ## Returning address generated
    return address
```

结尾

7.4.3教程3-vanity address生成器

Vanity address的特点是你可以自定义门罗币地址的前缀。如果你希望你的地址包含“cat”，你可以使用这个教程中提到的方法生成一个以“4cat”开头的public address。门罗地址的格式会有一些局限性：你不能移除开头的4（八进制为0x12）和一些由于base 58编码产生的字符（I,l,0,o）。

永远不要相信网站和第三方帮你生成的vanity address。因为你无法保证生成vanity address的key是否被他们偷偷备份了。

你可以通过下面的Python小脚本来安全地生成自己的vanity address。方法很简单，只需要不停地重复生成地址，直到你满意为止。自定义前缀时，推荐使用短的字符串，因为随着时间的推移，可选前缀的长度会随着门罗币地址数量的增长而显著的变长。

这个程序通过循环语句while（1）来实现。每一次循环都会调用前面教程中出现过的方法generate_random_address，用来生成一个新的地址。

新的地址生成以后，程序会检查是否满足你的自定义条件。一旦发现符合条件的地址，程序会将其打印出来，并退出循环。

```
import sys
sys.path.append('../libraries')
import address

if (len(sys.argv) != 2):
   print(“usage: python vanity_address.py [desired_prefix]”)
    exit()

if (sys.argv[1][0] != “4”):
   print “Monero addresses must start with the character 4”
    exit()

## create random addresses until one of them matches the
desired prefix
## bruteforcing takes a while
while(1):
    rand_address = address.generate_random_address()
    if (rand_address[0:len(sys.argv[1])] == sys.argv[1]):
        print(rand_address)
        exit()
    else:
        print(“searching”)
```

7.4.4教程4-生成stealth address

在第五章中介绍的stealth address生成方法稍微有点复杂，这一小节将借助一个Python的小程序来帮助我们理解。通过这个程序一步一步地帮助理解其内部的数学原理。

教程4的目的是使用一个public view key，一个public spend key，和一个随机private TX key（256-bit）来生成一个stealth地址。

首先，从libraries文件夹中导入必要的库。

```
import os, sys
# library for hex
from binascii import hexlify, unhexlify
sys.path.append('../libraries')
# utils and ed25519 libraries
import utils
import ed25519
```

根据generate_stealth_address算法定义，通过执行一些必要的数学运算来生stealth address。这个运算过程会用到public keys和一些随机生成的信息。

```
def generate_stealth_address(publicViewKey, privateTxKey,
publicSpendKey, index):

## multiply r*A
derivation = utils.generate_key_derivation(publicViewKey,
privateTxKey)

## concatenate index to derivation then hash and reduce
## Hs(rA|i)
scalar = utils.derivation_to_scalar(derivation, index)

## multiply by base point
## Hs(rA|i)G
sG = ed25519.scalarmultbase(utils.hex2int(scalar))
## interpret the public spend key as a point on the curve
pubPoint  = ed25519.decodepoint(unhexlify(publicSpendKey))

## add the public spend key to the previously calculated
point
## Hs(rA|i)G + B
output = ed25519.edwards(pubPoint, sG)
## convert the point to a hex encoded public key
return hexlify(ed25519.encodepoint(output))
```

这个小脚本可以通过这种方式调用:

```
print(generate_stealth_address("be90718b250a06b4b-
cffca6af948240ad6d8951b730a9711f78d4c9decefb4bd",
"12b793b002ed168f36c9dc8d13c0e820546359452f67136f03087e-
b18208710e", "6b48d1c30a640b0b33d0062188df2edd4e6acac-
7282b215e86701a644a9f70ba", "01"))
```

每次生成的Stealth address都是不一样的，因为在整个过程中加入了一些随机信息。下面是一个输出stealth address的例子：

```
a2bd788a63555e0847800b56051072d-
b3558ac2f97b58b8021e57c67125b4411
```

7.5 Monero C++ API

虽然Monero通过RPC接口进行交互简单易行，可以生成地址和子地址，也能转账，但是RPC方法的拓展性不好，对于大型的企业应用可能会有瓶颈限制。

那么有其他的选择吗？当然有，Monero也可以用C++ API来实现所有功能，包括钱包管理和发送交易。

因为C++ API使用起来比RPC接口复杂，一般人可能不愿意在开发环境中使用它除非对Monero的集成很熟悉，使用C++开发中的任何错误和问题都可能破坏隐私和安全性。

7.5.1 Monero 库

Monero Core是Monero几个基础库的集合，包含了Monero运行的一些必要操作，比如Boost, Ed2559和 Crypto

这些集成库能够简化开发者的开发流程，比如，编码者可以直接从MoneroCore库中调用base58_decode函数，

而无需重新实现一个函数。

要使用库首先得用Monero Core编译库文件，编译完成之后，会生成一个拓展名为.a或.so的输出文件。

7.5.2 开始使用C++

要集成Monero Core的代码首先要编译它的库文件，只需按照上面的说明进行操作，并检查第6章中的依赖表。熟悉C++(特别是C++11标准的基础知识)对以下的教程很有帮助。

7.5.3 教程 5 - 使用花费私钥(private spend key)恢复所有密钥

本教程介绍了如何用C++ API和CMake方法从花费私钥(private spend key)中恢复Monero所有密钥。本教程适用于基于Linux的平台，因为苹果和Windows的系统有他们自己的库(比如OpenSSL或Boost)。

首先，将所有环境变量和库设置在CMakeLists.txt文件中，在本教程中，我们会把Monero Core编译在 /opt/monero 文件夹中。

```
cmake_minimum_required(VERSION 3.5)

set(PROJECT_NAME tutorial-5)

project(${PROJECT_NAME})

set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11 -ldl")

if (NOT MONERO_DIR)
    # Path of Monero source code
    set(MONERO_DIR ~/monero)
endif()

message(STATUS MONERO_DIR ": ${MONERO_DIR}")

set(MONERO_SOURCE_DIR ${MONERO_DIR} CACHE PATH "Path to
the root directory for Monero")

# set location of Monero build tree
set(MONERO_BUILD_DIR ${MONERO_SOURCE_DIR}/build/Linux/
master/release/ CACHE PATH "Path to the build directory
for Monero")

set(MY_CMAKE_DIR "${CMAKE_CURRENT_LIST_DIR}/cmake" CACHE
PATH "The path to the cmake directory of the current proj-
ect")
list(APPEND CMAKE_MODULE_PATH "${MY_CMAKE_DIR}")

set(CMAKE_LIBRARY_PATH ${CMAKE_LIBRARY_PATH} "${MONERO_
BUILD_DIR}" CACHE PATH "Add Monero directory for library
searching")

# find boost
find_package(Boost COMPONENTS
        system
        filesystem
        thread
        date_time
        chrono
        regex
```

下页继续 (1/3) »

```
        serialization
        program_options
        date_time
        REQUIRED)

# include boost headers
include_directories(
        ${Boost_INCLUDE_DIRS}
 )

include_directories(
              ${MONERO_SOURCE_DIR}/src
              ${MONERO_SOURCE_DIR}/external
              ${MONERO_SOURCE_DIR}/build
              ${MONERO_SOURCE_DIR}/external/easylogging++
              ${MONERO_SOURCE_DIR}/contrib/epee/include
              ${MONERO_SOURCE_DIR}/version
              ${MONERO_SOURCE_DIR}/external/db_drivers/
liblmdb)
# Specify source files
set(SOURCE_FILES main.cpp)

# Make executable
add_executable(${PROJECT_NAME} ${SOURCE_FILES})

set_target_properties(${PROJECT_NAME} PROPERTIES LINKER_
LANGUAGE CXX)

set(LIBRARIES
        wallet
        blockchain_db
        cryptonote_core
        cryptonote_protocol
        cryptonote_basic
        daemonizer
       cncrypto
        blocks
        lmdb
        ringct
        device
```

下页继续 (2/4) »

```
        common
        mnemonics
        epee
        easylogging
        device
        pcsclite
        sodium
        ${Boost_LIBRARIES}
        pthread
        unbound
        crypto
        ringct_basic)

if (Xmr_CHECKPOINTS_LIBRARIES)
    set(LIBRARIES ${LIBRARIES} checkpoints)
endif()

set(LIBS common; blocks; cryptonote_basic; cryptonote_core;
cryptonote_protocol; daemonizer; mnemonics; epee; lmdb;
device; blockchain_db; ringct; wallet; cncrypto; easylog-
ging; version; checkpoints; ringct_basic; )

foreach (l ${LIBS})
       string(TOUPPER ${l} L)
       find_library(Xmr_${L}_LIBRARY
                    NAMES ${l}
                    PATHS ${CMAKE_LIBRARY_PATH}
                    PATH_SUFFIXES "/src/${l}" "/src/ringct"
"/src/" "/external/db_drivers/lib${l}" "/lib" "/src/crypto"
"/contrib/epee/src" "/external/easylogging++/"
                    NO_DEFAULT_PATH
                    )

       set(Xmr_${L}_LIBRARIES ${Xmr_${L}_LIBRARY})

       message(STATUS " Xmr_${L}_LIBRARIES ${Xmr_${L}_LI-
BRARY}")
       add_library(${l} STATIC IMPORTED)
set_
property(TARGET ${l} PROPERTY IMPORTED_LOCATION ${Xmr_${L}_
```

下页继续 (3/4) »

```
LIBRARIES})
endforeach()
target_link_libraries(${PROJECT_NAME} ${LIBRARIES})
```

End (4/4)

现在库已经添加完毕，开始开发我们这个程序。用花费私钥(private spend key)生成所有密钥是生成和恢复钱包所必需的常用操作。

```
// main.cpp file for Tutorial 5 - Mastering Monero
// https://github.com/monerobook/code/tutorial-5/main.cpp

#include “cryptonote_core/blockchain.h”
#include “common/base58.h”
#include “crypto/crypto-ops.h”
#include “crypto/hash.h”

// Converts crypto::hash into crypto::secret_key or cryp-
to::public_key
template <typename T>
T get_key_from_hash(crypto::hash & in_hash){
       T* key;
       key = reinterpret_cast<T*>(&in_hash);
       return *key;
}

int main(){
       // Put here your private spendable key!
       std::string str_spend_key = “f8f2fba1da00643bbf11f-
fec355a808d2d8ca4e4de14a10476e116abd8dd7f02”;
       // Specify the network type. It could be cryp-
tonote::nettype, where nettype is MAINNET, TESTNET or
STAGENET
       cryptonote::network_type nettype = cryptonote::-
MAINNET;
       crypto::public_key public_spend_key;
```

下页继续 (1/3) »

```
        // Convert hex string to binary data
        cryptonote::blobdata blob;
        epee::string_tools::parse_hexstr_to_binbuff(str_
spend_key, blob);
        crypto::secret_key sc = *reinterpret_cast<const
crypto::
secret_key *>(blob.data());
        std::cout << “Private spend key : ” << sc << st-
d::endl;

        // Generate public key based on the private key
        crypto::secret_key_to_public_key(sc, public_spend_
key);

        std::cout << “Public spend key : “  << public_spend_
key  << std::endl;

        crypto::hash hash_of_private_spend_key;

        crypto::cn_fast_hash(&sc, sizeof(), hash_of_pri-
vate_spend_key);

        crypto::secret_key private_view_key;
        crypto::public_key public_view_key;

        // Generate keys from hash_of_private_spend_key
        crypto::generate_keys(public_view_key,private_view_
key,get_key_from_hash<crypto::secret_key>(hash_of_private_
spend_key), true);

        std::cout << “\n” << “Private view key : “  << pri-
vate_view_key << std::endl;
        std::cout << “Public view key  : “  << public_view_key
<< std::endl;

        cryptonote::account_public_address address {pub-
lic_spend_key, public_view_key};
        std::string public_address;
        // Get account address as a string
        public_address = cryptonote::get_account_address_
```

下页继续 (2/3) »

```
as_str(nettype, false, address);
       std::cout << “Monero Address:” << public_address <<
std::endl;
       return 0;
}
```

结束 (3/3)

要编译这些代码，需要切换到代码所在目录并执行cmake。如果在教程代码所在位置的根目录，执行：

```
$ cd tutorial-5 && cmake .
```

执行结果大概如下图：

```
$ cd tutorial-5 && cmake .
-- The C compiler identification is GNU 6.3.0
-- The CXX compiler identification is GNU 6.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compil-
er: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
.....

-- Configuring done
-- Generating done
-- Build files have been writ-
ten to: /code/tutorial-5
```

```
$ make
Scanning dependencies of target tutorial-5
[ 50%] Building CXX object CMake-
Files/tutorial-5.dir/main.cpp.o
[100%] Linking CXX executable tutorial-5
[100%] Built target tutorial-5
```

如果遇到任何错误，请先确认编译相关文件的版本是否正确，CMake (>= v. 3.5.2)和GCC (>= v. 5) 。 CMake会创建一个makefile文件，然后我们执行如下命令：

最后，通过运行./tutorial-5 来启动程序，如图：

```
Private spend key : <f8f2fba1da00643bbf11f-
fec355a808d2d8ca4e4de14a10476e116abd8dd7f02>
Public spend key : <fffb624bd31dfafb015b01c-
beaef28cbff3b2d77af01c54b77d6e1cef04d5f1e>
Private view key : <9227a05c665f684f5b8fef-
815cedd8a911b426c9fa07554c70daacf87757b302>
Public view key : <d79eaf3acfd1f7a93526d2eec-
5bec5b76b880177e2610b69716b4f0577950308>
Monero Address: 4BKjy1uVRTPiz4pHyaXXaw-
b82XpzLiowSDd8rEQJGqvN6AD6kWosLQ6VJX-
W9sghopxXgQSh1RTd54JdvvCRsXiF41xvfeW5
```

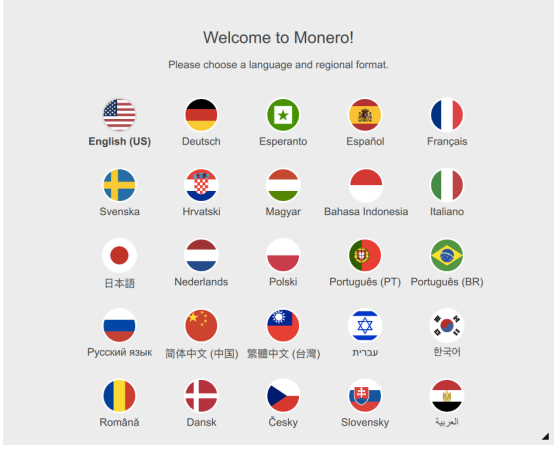
第八章：钱包指南、答疑和小贴士

8.1 门罗官方GUI的详细使用指南

以下部分将指导你如何使用门罗图形化用户界面（GUI），如果你使用其它钱包，你可以跳过该部分。

1. 选择语言

官方的门罗GUI可以从 https://getmonero.org/downloads. 下载。当你解压缩和启动该应用后，你将来到语言选择界面：



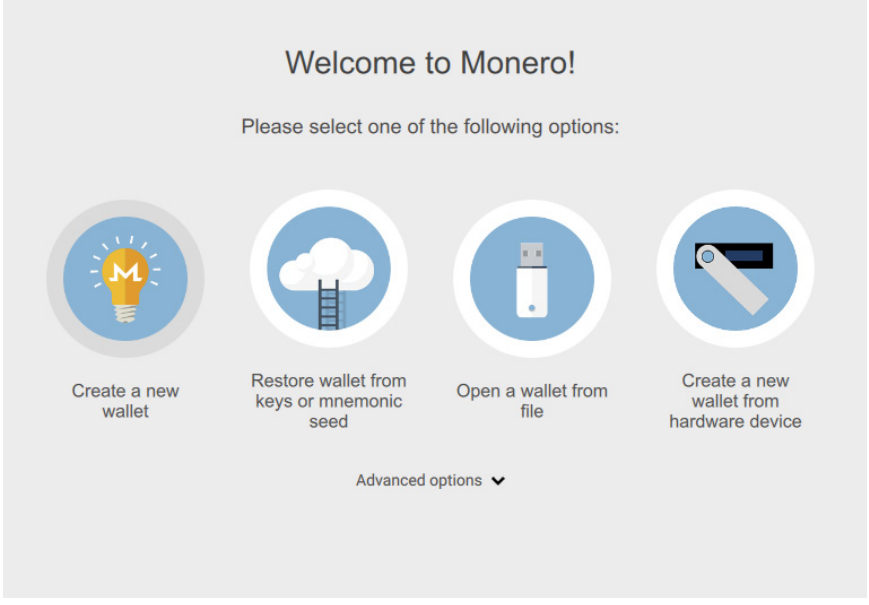
如果其中没有你使用的语言，你可以参与翻译，以帮助到与你使用同种语言的朋友。

2. 选项设置

门罗GUI提供3种类型的钱包：主网、测试网（Testnet）和Stagenet网络

如果你想使用真实的门罗区块链，请选择主网（默认选项）。测试网和Stagenet是两个独立的网络，供开发者开发和测试新代码。这两个网络中的门罗币没有任何实质价值，也无法转移至主网上。

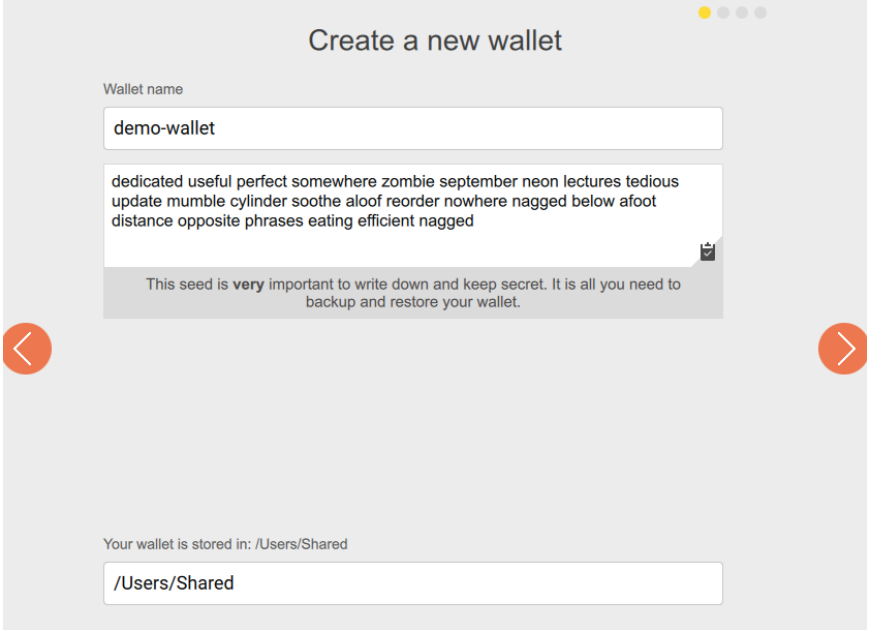
如果这是你的第一个门罗钱包，请选择“创建新钱包”。门罗钱包会为你生成一个新的种子和25词的种子助记词。



3. 写下助记词

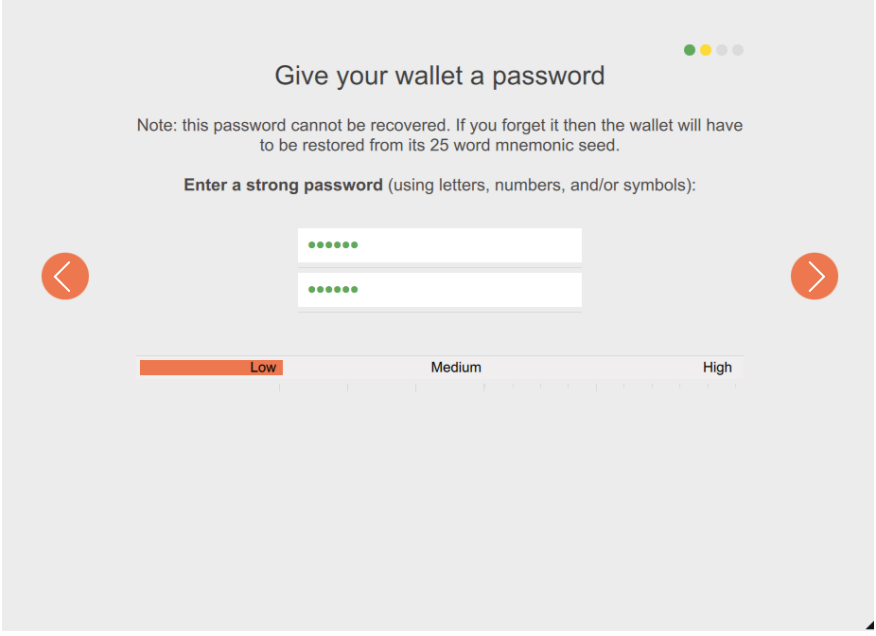
请记住，种子和密码不一样！如果你丢失了种子，网络无法恢复你的资金访问权限。

请确保你写下这些助记词，并将其保存在其他人无法找到的安全的地方。



4. 输入密码

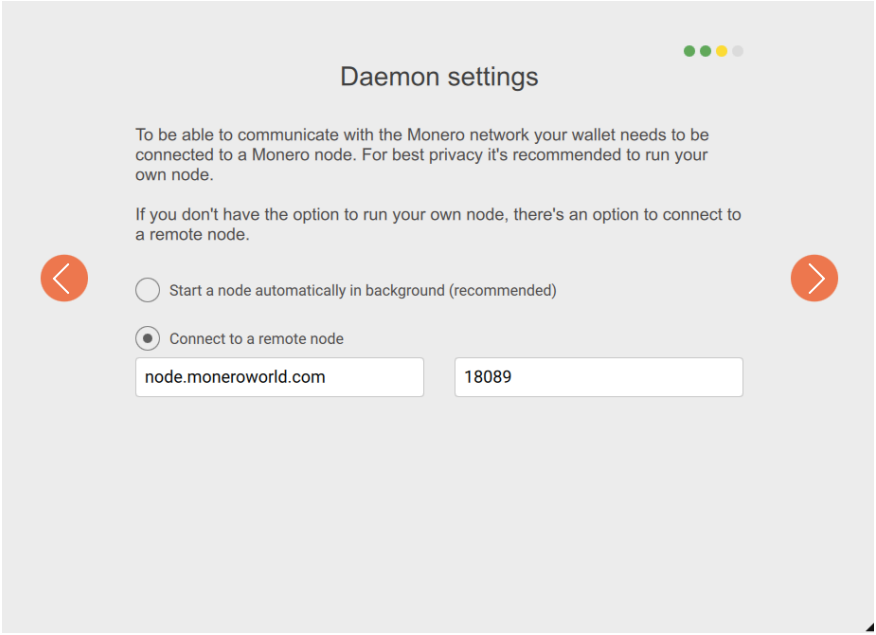
如果有多个人使用你的电脑，你可以设置一个钱包密码来保证资金的安全。钱包密码是一项本地安全措施，就像PIN解锁码一样。它与加密学或你的资产如何在区块链保存无关，所以使用种子恢复你的钱包，将无视本地密码这一环节。

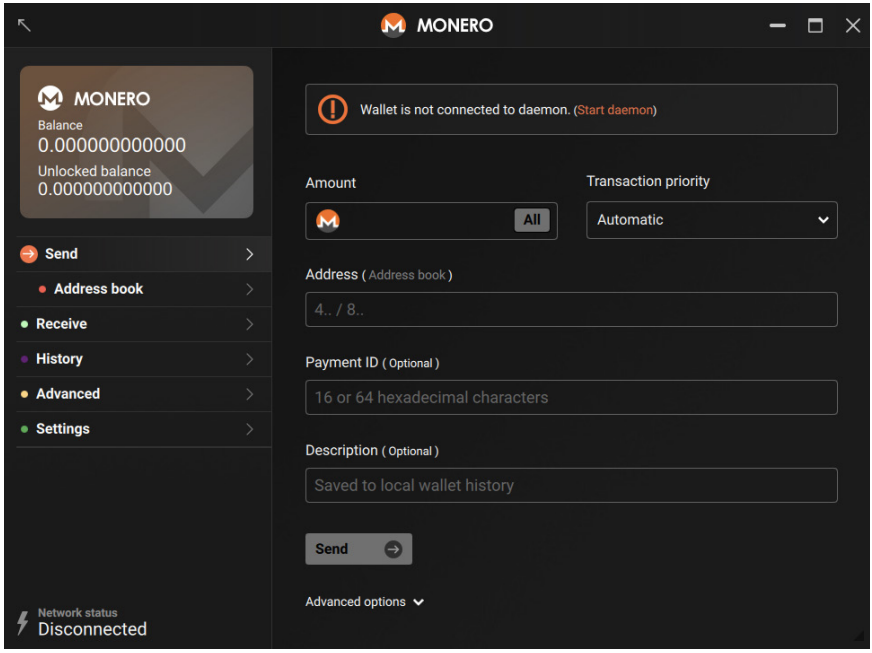


5. 下载门罗区块链

现在，你可以选择建立自己的节点，或者连接至远程节点。

运行自己的节点至少需要60GB的硬盘空间来储存区块链。如果你的设备资源有限，你可以选择“连接至远程节点”，创建一个轻量级的钱包，访问存储在其它地方的数据。你可以在4.2.3中了解到远程节点和本地节点的优劣。

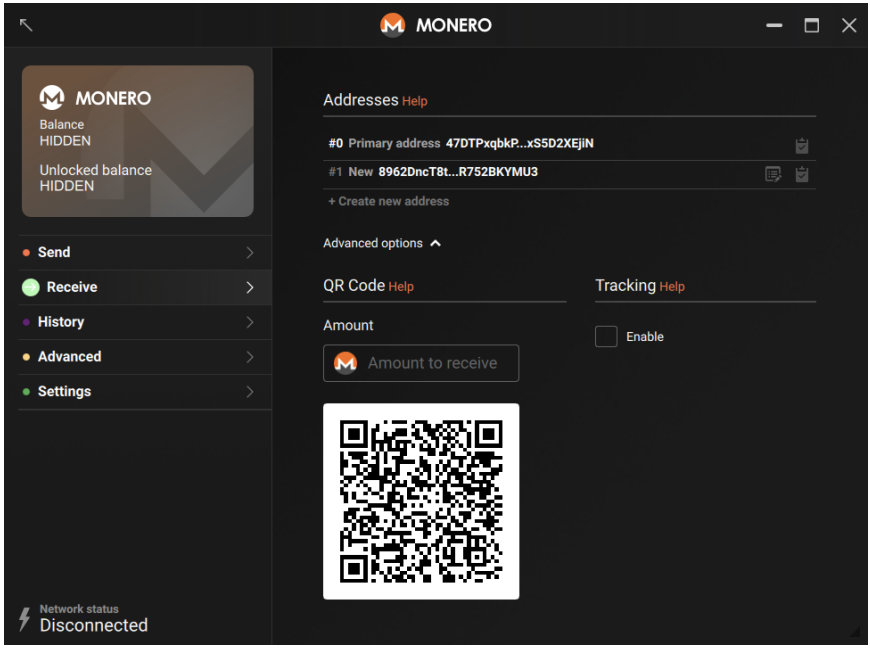




那么..欢迎使用门罗GUI！

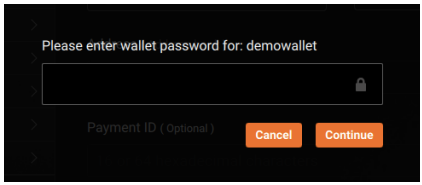
8.1.2 使用GUI接收门罗币

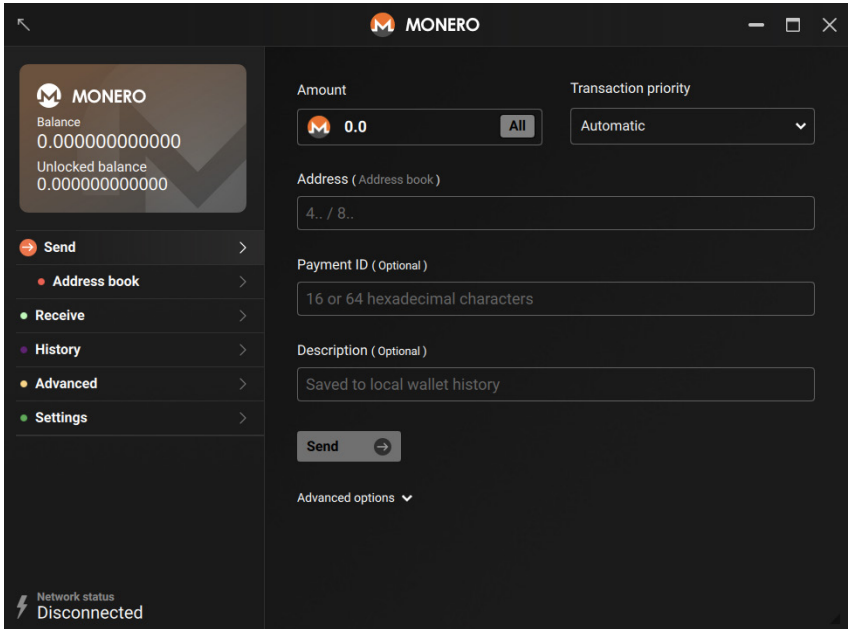
GUI中的“接收”界面包含2种形式的收款地址：文字和QR码。点击”创建新地址”可以生成多个子地址，这些地址都指向同一个钱包（种子）。如果你想向某人发起收款请求，你可以在“金额”中输入金额，该金额也会自动编码至QR码中。



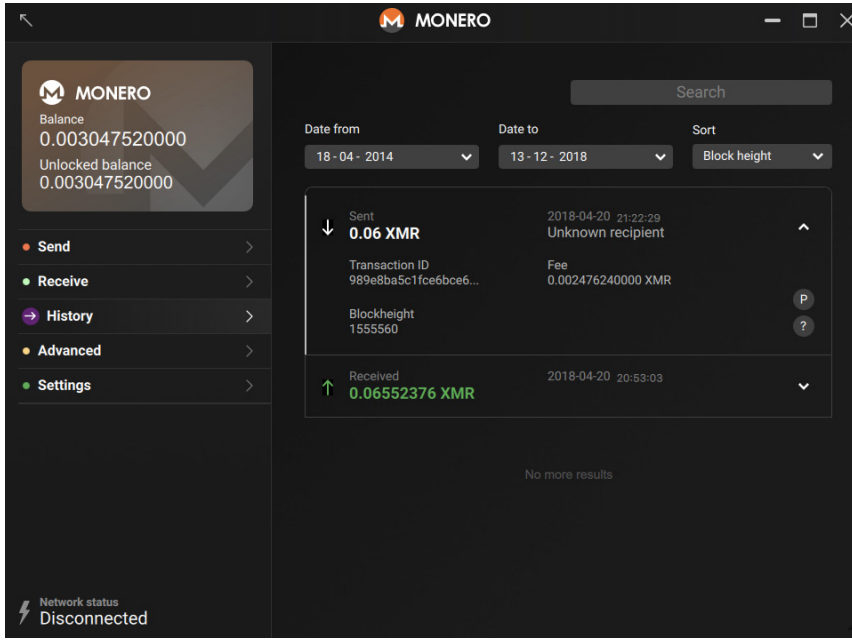
8.1.3 使用GUI发送门罗币

若要发送门罗比，你只需输入要发送的金额和收款人地址即可。你可以选择不填写支付ID（Payment ID），除非你的收款人要求你这么做。描述（Description）这一项的内容是存储在本地的，所以你可以留下一些备注，不过当你使用种子恢复钱包的时候，这些备注不会被恢复。

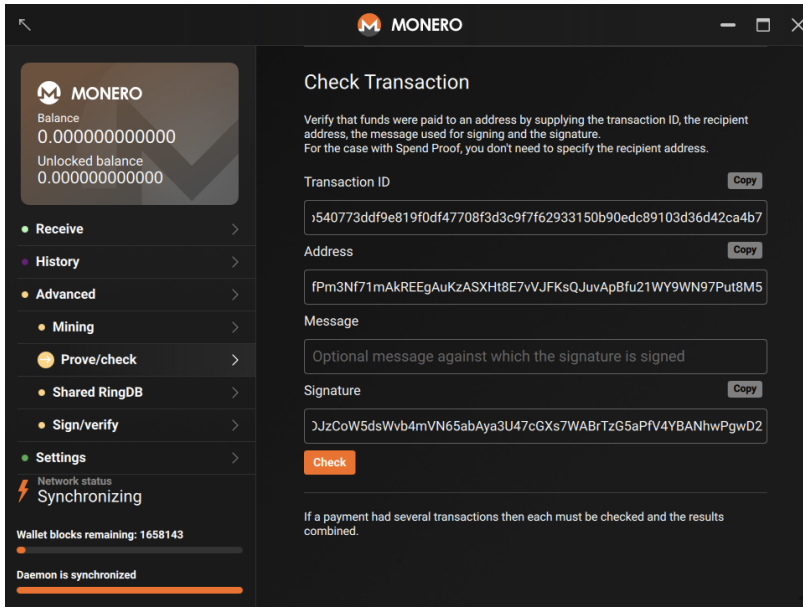




交易记录


MONERO
MONERO
Balance
0.003047520000
Unlocked balance
0.003047520000
Send
Receive
History
Advanced
Settings
Network status
Disconnected
Search
Date from
18 - 04 - 2014
Date to
13 - 12 - 2018
Sort
Block height
Sent
0.06 XMR
2018-04-20 21:22:29
Unknown recipient
Transaction ID
989e8ba5c1fce6bce6...
Fee
0.002476240000 XMR
Blockheight
1555560
P
?
Received
0.06552376 XMR
2018-04-20 20:53:03
No more results

8.1.4 GUI的支付证明



通过“证明/验证"（Prove/check）界面，你可以对支付证明进行验证。以上的截图显示第二章中Maria和Kahn的交易的交易ID、地址和交易密钥。

8.2 门罗官方CLI钱包的详细使用指南

以下部分将指导你如何使用门罗命令行界面（CLI），如果你使用其它钱包，你可以跳过该部分。

8.2.1 使用CLI设置钱包

门罗的官方CLI可以在https://getmonero.org/downloads https://getmonero.org/downloads**下载。**

运行CLI的命令会根据你的操作系统不同而不同。在Linux中，可直接从程序文件夹运行./monero-wallet-cli。如果你想要连接至远程节点，隐藏你的IP地址或者使用其他高级功能，你可以添加额外的标记（flags，记录在该章中）。

```
$
```

```
2014-08-12 (490 MB/s) - 'li-
nux64' saved [45719102/45719102]

$ tar jxvf linux64

./monero/
./monero/monero-wallet-rpc
./monero/monero-blockchain-import
./monero/monero-blockchain-ancestry
./monero/monero-blockchain-usage
./monero/monero-wallet-cli
./monero/monero-blockchain-depth
./monero/monero-gen-trusted-multisig
./monero/monerod
./monero/monero-blockchain-export
./monero/monero-blockchain-blackball

$ cd monero && ./monero-wallet-cli

2018-10-24 18:58:11,024 INFO  [de-
fault] Page size: 4096
This is the command line monero wallet. It needs
to connect to a monero daemon to work correctly.

WARNING: Do not reuse your Monero keys on anoth-
er fork, UNLESS this fork has key reuse mitiga-
tions built in. Doing so will harm your privacy.

Monero CODENAME (vX.X.X-release)

Specify wallet file name (e.g., MyWallet). If
the wallet doesn't exist, it will be created.
Wallet file name (or Ctrl-C to quit): (enter the
name of your wallet you want to create) testwallet
```

Continues on next page (1/3) »

```
No wallet found with that name. Confirm cre-
ation of new wallet named: testwallet
(Y/Yes/N/No): Yes
Generating new wallet...
Enter a new password for the wal-
let: (enter your secret password)
Confirm password: (confirm your password)
List of available languages for your wallet's seed:
0 : Deutsch
1 : English
2 : Español
3 : Français
4 : Italiano
5 : Nederlands
6 : Português
7 : русский язык
8 : Japanese
9 : Chinese
10 : Esperanto
11 : Lojban
Enter the number corresponding to the lan-
guage of your choice: (from 1 to 10)

Generated new wallet: 4BKjy1uVRTPiz4pHyaXX-
awb82XpzLiowSDd8rEQJGqvN6AD6kWosLQ6VJX-
W9sghopxXgQSh1RTd54JdvvCRsXiF41xvfeW5

View key: 9227a05c665f684f5b8fef815ced-
d8a911b426c9fa07554c70daacf87757b302
**********
**********
Your wallet has been generated!
To start synchronizing with the dae-
mon, use the "refresh" command.
Use the "help" command to see the
list of available commands.
[...]

lamb hexagon aces acquire twang bluntly ar-
gue when unafraid awning academy nail threat-
en sailor palace selfish cadets click sickness
juggled border thumbs remedy ridges border
**********
```

下页继续 (2/3) »

```
**************************
Starting refresh...
Background refresh thread started
[wallet 433bhJ]:
```

结束 (3/3)

8.2.2 接收门罗币

你可以通过命令address显示你的地址。如果你想要合并地址和支付ID，你可以使用命令integrated_address来生成一个随机的支付ID，或者你可以指定某个特定ID作为输入的一部分，例如

```
[wallet 433bhJ]: integrated_ad-
dress 12346780abcdef00
```

通过命令show_transfers，你可以查看所有的流入资金。你也可以查看特定区块高度的交易记录，如： 只显示区块高度 650000的记录:

```
[wallet 433bhJ]: show_transfers in 650000
```

8.2.3 发送门罗币

使用命令transfer来发送门罗币。若将门罗币发往单个地址，你无需指定环值（从2018年10月的硬分叉开始，环值固定为11），你需要输入收款人地址以及金额。

使用CLI发送0.6 XMR：

```
[wallet 433bhJ]:  transfer 4758W1dAkifB2G1wQK-
mPWRvPs9zdsb5ctRFW2ttQbkQxYHRuPRdHZ9ijq-
J7oxcns9SvtpiH8ti8BRjL3LUHaBURpiz4KF  0.06
```

8.2.4 支付证明

默认情况下，CLI不记录交易密钥，不过你可以使用命令set store-tx-info 1来启用这个功能。你可以通过指定交易ID来定位交易密钥：

```
[wallet 433bhJ]: get_tx_key
4b540773ddf9e819f0df47708f3d-
3c9f7f62933150b90edc8910 3d36d42ca4b7
```

使用命令check_tx_key TXID TXKEY ADDRESS来验证密钥，例如验证第二章中Maria的交易：

```
[wallet 433bhJ]: check_tx_key
4b540773ddf9e819f0df47708f3d-
3c9f7f62933150b90edc891 03d36d42ca4b7
```

如果已知支付ID，那么你可以使用payment命令来搜寻该笔交易：

```
[wallet 433bhJ]: payments 12346780abcdef00
OutProofV1To53Qu2gegZbUevosKCTwrEdqiECgFyUygutXMEdh
```

8.3 常见问题解答

8.3.1 问题：我往自己的钱包中发送了门罗币，但资产仍然显示为0 XMR

以下为可能的情况：

1) 【永远】检查你复制的门罗地址是否正确（有时恶意软件会试图修改地址）；

2）按照以下步骤，检查交易是否发往你的钱包/地址：
A 在GUI的“设置”页面，选择“显示种子&密钥”，复制查阅私钥；
B 使用区块链浏览器，如https://xmrchain.net/explorer
xmrchain.net
C 输入交易ID/哈希\
D 在“Decode outputs”下，输入查阅私钥和地址
E 点击“Decode outputs”
F 如果结果显示“output true”，则说明交易已经发送成功并记录在区块链上

3） 如果在步骤2）中显示你的交易发送成功，说明你的钱包还没有同步该项输出。门罗GUI使用本地缓存，每个几秒钟刷新一次，尤其是在Windows操作系统中。如果这个问题持续存在，尝试点击GUI设置页面的“重新扫描输出”（Rescanning for outpus），或者尝试去社区频道中（见第六章）寻求帮助。

8.3.2 问题：我的GUI钱包总是出错/一直很卡

首先最重要的是，保证你运行的是最新版本的GUI。你可以在“设置”页面中（Debug info处）看到当前的版本号，如果你运行的不是最新版本，请更新。

在初始同步阶段，GUI钱包相应速度较慢是正常的，因为monerod守护进程（daemon）需要消耗大量的CPU资源来验证区块和交易。

你可以参照一下步骤限制monerod的CPU使用率：

1. 进入GUI“设置”页面
2. 在“daemon startup flags”中添加--max-concurrency 1
3. 停止守护进程，退出GUI
4. 重启GUI和守护进程

你必须标记--max-concurrency并重启，才可是的monerod只是用1 CPU进程。

词汇表

Account | 账户

账户是子地址系统中的一部分。一个钱包带有一个种子，种子衍生出主地址的花费私钥和查阅私钥。这些私钥又进而延伸出子地址，子地址归类到不同的账户中。

主地址指的是钱包中第一个账户的第一个地址。

每个账户显示各自的资产，并可以拥有多个子地址。账户实际上只是子地址的集合，所以没有所谓的账户地址（除非你把某个账户中的第一个子地址称为“账户地址”）。

一个钱包可以拥有多个账户，每个账户可以拥有多个子地址。因为账户和子地址都是从种子中衍生出来的，所以你只需要知道种子，即可恢复钱包中的所有账户/子地址（不过你给账户/子地址设置的标签需要重新标注）。

Address | 地址

当你向他人发送门罗币的时候，你只需要一个信息：对方的地址。门罗的公共地址是一串以“4”开头的，95位的字符串。

Airgap Air gap，air wall或者air gapping

是一项部署在单台或多台电脑上的网络安全措施，确保安全的计算机网络与不安全的网络在物理上隔离开来，例如公共网络或不安全的本地网络。

该技术因为像是在网络间建立一道抽象的空气墙，从而得名Air gap。Air gap也不是完全字面上的，因为网络使用专用的加密设备与不受信任的网络建立通道以交换数据包，同时防止数据包速率和大小被墙掉，被墙掉的计算机之间无法通信。

ASIC | 应用专用集成电路

Application-Specific Integrated Circuit (ASIC)，即应用专用集成电路，是一种为特定用途而专门定制的集成电路，而非通用目的的电路。例如，转为数码音频录制器而设置的芯片，或者高效的比特币矿机，都是ASIC。

ASIC Resistance | 抗ASIC

抗ASIC指的是某些加密货币为了确保自身的挖矿算法与ASIC不兼容而采取的一系列措施。你可以在第4、5、6章中了解更多关于门罗社区积极保护CryptoNight算法，使之只与CPU、GPU相兼容的信息。

Base32 Address（Kovri）| Base 32地址（Kovri）

Base32地址是一个缩减的，编码过的I2P地址，它是.b32.i2p主机名的第一部分。

BitMonero | 门罗币

BitMonero是门罗币原先的名字（见第一章），一些比较老的信息中仍然使用这个名字，如默认情况下，区块链和日志存储在~/.bitmonero文件夹中。

Block | 区块

区块是交易的容器，平均每过2分钟就有一个新的区块添加到区块链上。区块中同时也包含一种特殊类型的交易：币基（coinbase）交易，即门罗网络中新币的产生。区块通过挖矿生成，成功挖出区块的节点向相连的节点广播这个区块，收到广播的区块再次循环这个动作，直至整个网络都收到该广播。

Blockchain | 区块链

区块是一个分布式的数据库，它因记录其加密货币的所有交易而持续增长。因为其记录的数据为大量的交易记录，我们也将该数据库称作账本。

在门罗中，平均每隔2分钟，这些交易就被打包进区块中，网络中所有的矿工和节点都将拥有这些区块的副本

Bulletproofs | 防弹协议

防弹协议是一种用以验证隐藏交易金额的新型数学系统。它能够将交易大小减少80%左右，从而大大地降低交易费用。

Change | 找零

你发送给他人的门罗币，其实是该笔门罗交易的一部分，另一部分作为找零，发送回你的账户中。

Coinbase Transaction | 币基交易

币基交易是每个区块中存在的一中特殊类型的交易，包含发送给矿工的区块奖励

Command Line interface | 命令行界面

命令行界面（CLI）是一个基于文本的，通过终端输入命令的界面。你可以在这里下载免费且开源的CLI：https://getmonero.org/downloads/

Consensus | 共识

共识指的是在类似门罗这样的去中心化网络中，多数诚实参与者防止少数恶意参与者作恶的过程。

Cryptocurrency | 加密货币

数字货币的一种，应用加密技术来确保货币生成的规范性，验证交易的有效性，并通常不受中央银行控制。

Cryptographic Signature | 加密学签名

用以证明信息所有权的加密技术，同时也可证明信息在发送后没有被篡改

Decoys | 诱饵

诱饵指的是，进行一笔门罗交易时，从区块链中伪随即选取，用以混合真实输出，组成环签名的输出，详见5.4.3

Denominations | 面额

面额指的是货币的金额，也通常作为货币的子单位。例如，美元这个货币的一个面额是1美元，1美分是它的子单位，为其价值的1/100。

为了方便使用，门罗的面额的名字去掉“mo”，并加上国际单位制前缀。最小的面额是1 piconero（0.000000000001 XMR）。门罗币“monero”的复数单词是“moneroj”

Name	Base 10	Amount
piconero	10^{-12}	0.000000000011
nanonero	10^{-9}	0.000000001
micronero	10^{-6}	0.000001
millinero	10^{-3}	0.001
centinero	10^{-2}	0.01
decinero	10^{-1}	0.1
MONERO	10^{0}	1
decanero	10^{1}	10
hectonero	10^{2}	100
kilonero	10^{3}	1,000
meganero	10^{6}	1,000,000

Diffucluty | 难度

难度是网络中的一个参数，通过升高或降低满足特定条件的哈希值的门槛，影响矿工挖出新区快的时间。当越来越多的矿工加入网络时，难度就会增加以防止区块过快被挖出（当哈希率降低时，难度也会随之下降）。

Encryption | 加密

在加密学中，加密是一种将消息或信息进行编码，确保只有授权方才可以解码和理解的过程。加密本身不能防止窃听，但可以防止其内容被窃听。

Fees | 手续费

每一笔交易都包含手续费，支付给将这笔交易打包进区块的矿工。对于高优先级的交易，y用户可以提高手续费来促使矿工更早地打包这笔交易。

Fungibility | 货币可互换性

在经济学中，货币可互换性指代的是，一种物品或商品，其基础单位是可以互换的。透明账本中的加密货币缺乏该性质，因为每一枚货币的历史都是公共可见的。门罗使用了多项隐私保护技术来确保其区块链上不会记录任何使门罗币贬值的信息，一次保证所有的门罗币价值相等，可以互换。

Fluffy Blocks | Fluffy 区块

区块由区块头和交易组成。Fluffy区块只包含区块头，交易目录和任何节点在接收区块时可能丢失的交易。Fluffy区块有助于节省带宽，因为节点可以知道区块中大部分或全部的交易，无需再次发送它们。

I2P I2P网络

为互联网通信提供强力的隐私保护。任何容易将你的隐私暴露在公共互联网的行为都可以使用I2P来匿名化。

Integrated address | 集成地址

集成地址是将普通地址和加密过的64位支付ID合并起来的地址。原始的集成地址长达106位。

Kovri

Kovri是I2P网络的C++实现。Kovri现在正在紧密地开发中，尚未集成至门罗（注：Kovri项目目前已经停滞，团队将使用I2P-Zero代替Kovri）。当Kovri集成至门罗节点时，你的交易安全性将更上一层楼。

Mining | 挖矿

挖矿是一个加密学上的计算过程，用以补足产生新区块。

它也是在区块链上分布式地确认交易的过程，也就是产生区块链的新区块过程。门罗节点使用区块链来区别正当交易和试图双花的交易。 .

门罗是由工作量证明驱动的。它使用一种数以亿计的设备（任何现代x86 CPU和多种GPU）都可以参与的挖矿算法：CryptoNight。

它的精妙之处在于允许在个人电脑上进行CPU挖矿，

而不是矿场或矿池的中心化挖矿，这是中本聪愿景中的真正的点对点货币。

Mnemonic Seed | 种子助记词

种子助记词是一串13或25词的词组，用以备份门罗账户，适用于多种语言。只需拥有这个25词的词组（MyMonero是13词），便可查阅和花费门罗资产。

Monero | 门罗币

隐私性最强的加密货币

Node | 节点

运行门罗客户端并下载有完整门罗区块链的设备称之为节点。节点可以积极地保护门罗网络。

OpenAlias

从最基本的来讲，OpenAlias是一个在FQDN（fully qualified domain name，完全认证域名）记录的TXT DNS。OpenAlias标准由门罗核心团队发布，可以提供更易读的地址，并在Zooko的基础上更进一步。OpenAlias已经集成至门罗中，并且也适用于其它加密货币。

Payment ID | 支付ID

支付ID是一组32字节（十六进制下为64位）或8字节（集成地址中）的字符，

可以选择添加在交易中。

支付ID通常被用于向商家或交易所标明某笔交易：因为鉴于门罗内在的隐私性，通常使用单个公共地址来收款，这时候支付ID有助于将汇入的资金和发送者关联起来。

自从v0.9（Hydrogen Helix）版本开始，支付ID可被加密并嵌入至集成地址中（集成地址由一般门罗地址和支付ID结合而来）。该类的支付ID应为64位，并且结合只有发送者和接收者才知道的一次性随机密钥进行加密。

我们建议你使用官方钱包的指令来生成包含支付ID的集成地址。你可以参照以下命令使用命令行来生成支付ID：

```
$ openssl rand -hex 8
```

生成8字节的支付ID

```
$ openssl rand -hex 32
```

生成32字节的支付ID

Pedersen Commitment | Pedersen承诺

Pedersen承诺是一种加密学算法，证明者借助它可以对某个值进行承诺（commit），并且不会泄露该值，同时也无法改变该值。

当你发送门罗币的时候，你所花费的输入和你所发送的输出的金额都被加密隐藏起来了，除了收款人以外，其余人都无法看见。

Pedersen承诺允许你在不泄露交易金额的情况下发送门罗币。Pedersen承诺同时也允许他人验证区块链上交易的有效性，并防止凭空伪造门罗币。

Pedersen承诺意味着可以验证门罗输入和输出的总和是否相等，但无法计算两者的总和分别是多少，Pedersen承诺还意味着一个输入对于另一个的比例，或者一个输出对于另一个的比例也是无法计算的。

因为真实输入和诱饵输入都在环形签名中，所以我们无法判断哪个是真实的输入，也无法判断Pedersen承诺要将哪个输入加入到求和运算中。不过这没有关系，因为环形机密交易和环形签名至需要证明输入和输出的组合总和与输入的总和相等即可。

鉴于它的数学原理，这无法被伪造。

Ring Signatures | **环形签名** 在加密学中，环形签名是数字签名的一种，可由组内任意拥有密钥的组员发起。因此，带有环形签名的交易实际上是由该组里的某个人签署的。从计算的角度上来讲，我们是无法判断真正提供签名的密钥是哪个，这是环形签名的安全特性之一。

例如，环形签名可以用来提供一份来自“白宫高阶官员”的匿名签名，所有人都不知道具体是哪个官员签署的消息。这是环形签名的适用案例，因为环形签名的匿名性无法突破，且可以随意组队（无需事前的设置）。

环形签名使用三角分布方法从区块链中抽取一定数量的公钥（输出）来与你的密钥进行配合。随着时间的流逝，历史输出可以作为环签名成员被重复利用数次。在环签名中，所有环成员都是相同的、等效的。在门罗中，环形签名通过引入数个“可能的”输入，以隐藏发送者。

Ring Size | **环值**

环值指的是一个环签名中总参与者的数量。一笔交易的环值若为11，说明这其中除了你的“真实”输出之外，还有10个诱饵输出。

Stealth Address | **隐蔽地址**

隐蔽地址是门罗内在隐私性的重要组成部分。他们允许并要求汇款方在每一笔交易中，生成一个随机的一次性地址作为收款地址。收款方只需公布一个地址，每一笔发往他/她的交易都将发送至一个唯一的地址中，并且这些地址和地址之间，地址和公布的地址之间无法关联起来。通过使用隐蔽地址，只有汇款方和收款方可以确定交易的去处。

Tail Emission | **尾部发行**

门罗的区块奖励永远不会降至0。区块奖励将平稳下降至尾部发行阶段（2022年5月），届时奖励为0.6 XMR/区块。

Transactions | **交易**

交易经过加密学签名的，包含门罗币交易细节。

交易的参数包括：一个或多个收款地址，以及对应量的资金，参与环形签名的成员数。

参与的成员更多，隐私的隐藏度就更高。但这也是有代价的，成员越多，交易所占大小就越大，交易费就越高。

交易也可以在线下发起交易，以增加隐私性。

交易ID可以作为交易的唯一识别码，它通常显示为32字节的字符串（十六进制下64位）。

Wallet | **钱包**

门罗账户，或钱包，存储着发送和接收门罗币所需的信息。除了发送和接收门罗币外，门罗钱包还保存着一份你的私密历史交易记录，允许你对消息进行加密学签名。它同时还包含了门罗挖矿软件和地址簿。

加入门罗社区.
新世界大门已经打开, 燎原星火熊熊
点燃, 欢迎你们的到来.