Preface

0.1 Writing the Mastering Monero Book

I am Nico ("SerHack"), an Italian security researcher, a Monero contributor, and the publisher of this book. Finding good resources and learning about cryptocurrencies can be a daunting task. For new users, it can be especially challenging to track down documentation written at an understandable technical level. When I first started learning about Monero, I had to spend a great deal of time seeking out and evaluating many different resources on the topic.

I decided to write Mastering Monero to guide you along this journey, whether you're setting up your first wallet or curious about the 'under the hood' technical details. The first few chapters are written for anybody curious about why and how to use Monero; they contain easy-to-understand explanations and examples, alongside instructions for practical use. Later chapters progress into more advanced topics, compiling information for developers who wish to build and contribute to the Monero project.

My adventure into the world of cryptocurrencies began when I learned about Bitcoin in January 2016. From the beginning, I have been concerned about the ramifications of its transparent public ledger. Since Bitcoin and most other cryptocurrencies are built around openly-linked addresses and coins with clear histories, transactions often inadvertently expose users' personal financial details. Every address balance is public information, which allows anybody to research your income, spending habits, and amount of cryptocurrency wealth. This can lead to undesirable consequences, such as price manipulation based on wallet balance.

I thought that Bitcoin was the only cryptocurrency until a friend introduced me to Monero in May 2017. I was blown away by its beautiful new paradigm: a world where vulnerable details such as account balances and transaction amounts are kept confidential to protect both the sender and the receiver. With privacy features implemented by default and always required, the entire Monero blockchain is veiled and users do not even have the option to accidentally send revealing transactions.

Recognizing the importance of this project, I began looking for ways to contribute to the community. I quickly saw an opportunity to support mass adoption by building payment gateways for online businesses, so I spearheaded the Monero Integrations project. This open-source codebase is designed around Monero's privacy-centric mentality: no signup or third-party service is required, since funds are routed directly to the recipient's wallet. The Monero community was very supportive throughout this endeavor, and the entire project was crowdfunded by donations through the Monero Forum Funding System (FFS).

While working on the Monero Integrations project, I learned that the lack of a comprehensive guide to Monero was an obstacle both for end users and prospective contributors. This need for a thorough guide inspired me to write Mastering Monero as a universal resource for our global community. I am grateful for the generous FFS support that has made it possible to publish this document as a free eBook (and physical book!) for the general public. Whether you read Mastering Monero cover-to-cover or jump through sections pertinent to your questions, I hope you enjoy learning about Monero and the exciting projects within the community.

0.2 How this resource is organized

This book is organized roughly in line with the dependencies between the different topics covered.

The first chapter will show how the blockchain resolves several problems with our mainstream economic system and banking system in particular. However, the blockchain is not a full solution to our cryptocurrency needs. One problem still remains: Assuring the privacy of online transactions. With Monero, we will realize what privacy is and why we care about it. This will be helpful and useful for newbies and for users that sometimes forgot the Monero principles.

The second chapter starts to explore the "practical" applications of Monero: we will understand the different types of existing wallets, learn how to create one with Monero GUI, and finally find out how to get our first Monero.

We will explore how Monero actually works in the third chapter. Here we will discover how the Monero Team developed a cryptocurrency that hides transactions from third parties by default.

The fourth chapter overviews a "hot"-topic argument: the mining process which is confirming transactions for other users. We'll see what it's and why it's important in order to guarantee a solid and trustable system for our transactions.

After fourth chapters of introduction based on getting started, mining and learning how Monero works we will dive into the internals Monero more deeply. This chapter includes a lot of technical details about Monero, especially Cryptography details. Stealth addresses, Ring Transactions, and Ring Confidential Transactions won't be buzzwords anymore since *you'll* be informed on how they properly work from the ground up. For developers and experts only, the fifth chapter will be the technical explanation of the CryptoNote Protocol. If you can't understand any of the explanations included in this chapter, don't say we didn't warn you: this book is called Mastering Monero for a reason.

The sixth chapter is for any users that want to help the community in order to translate, build or improve Monero in any way. No worries about how we can help, the important part is doing that! We will join the Monero community!

When we have finished reading the first seven chapters of the book, we'll begin to think how to incorporate Monero into our finances. If Monero is such a revolutionary cryptocurrency, how I can integrate Monero to my business? The seventh chapter will provide you all the answers to these questions. From the JSON RPC to the OpenAlias sub project, everyone can start accepting Monero.

Contact the author and the Monero Community author email: support@masteringmonero.com Monero community: mattermost.getmonero.org

[Links]

Chapter 1

Introduction to cryptocurrencies & Monero

Maria is purchasing a car from George, and in this chapter we'll consider three different ways that she could pay him: traditional banks, transparent cryptocurrencies (e.g. Bitcoin), and Monero.

1.1 Payment through banks

If Maria sends the money to George through the traditional banking system, they trust two intermediate parties (their respective banks) to symbolically move the funds for them.

There is no actual movement of physical bills or assets; both banks simply edit their respective databases to show that the funds have been transferred. When Maria submits the transaction to her bank (whether by wire transfer, her bank's website, or an app), her bank subtracts $2,500 from her account balance on their ledger, then contacts George's bank and requests that they add $2,500 to George's balance.

There are a few drawbacks and risks to this system, and it requires total trust in the banks. Maria, George, and the banks must act on faith that transactions are legitimate and that the ledgers are kept honestly. This trust in the intermediate third parties poses a risk, since a nefarious actor or the banks could “create” money by fraudulently editing the ledger balances or transaction database.

Furthermore, Maria does not actually have possession of $3900, only an IOU from her bank that she must trust is redeemable. She has no way to audit her bank to verify whether they actually have $3900.

In fact, they may not hold that much, since most banks legally operate on *fractional reserve* - meaning that their actual assets are allowed to be significantly less than the total balance promised to account owners.

Depending on how the funds were sent, it could take anywhere from minutes to days before the $2,500 shows up in George's bank account. Since George is not privy to the banks' ledgers or communications, the entire process is opaque and cannot be monitored.

Many people that have not personally experienced economic disruption take functioning banks and the validity of their IOUs for granted. Few individuals consider the unsettling ramifications of handing their lifelong savings to opaque corporations, often putting all their eggs in a single institutional basket. Losses can occur due to:

- negligence (the bank makes a mistake)
- financial issues (the bank overextends their assets or goes out of business)
- malice and corruption (the bank or a rogue employee steals your money)
- hostile third parties (the bank is robbed or a hacker thieves electronic funds)

Thankfully, an emerging new blockchain technology is capable of mitigating all of the above risks by creating a distributed ledger that all parties can equally use, view, and verify. This remarkable capability for strangers to agree on a shared document, which is called decentralized consensus, has been revolutionized in the last decade.

It's easy to be confused about the terminology at first, especially since most people are simultaneously introduced to several jargony concepts. You can think about "blockchains" as a technology that allows networks to establish "decentralized consensus" agreements. By enabling strangers to safely share a ledger, it becomes possible to build "cryptocurrencies" that function as digital cash. There are a multitude of regular currencies (euros, dollars, yen, etc); analogously, various teams have built many different cryptocurrencies (Monero, Ethereum, Bitcoin, etc).

1.2 Introduction to blockchains

Anybody can learn all about Monero and how its blockchain works without having to understand the underlying mathematics and cryptography (similar to how anybody can become internet-savvy without first studying DNS servers and the IPv6 protocol). **This chapter focuses on the key concepts and vocabulary without digging into all of the technical details** - you can jump ahead to chapter 4 and chapter 5 if you want to dive into the cryptographic framework.

1.2.1 What is a blockchain?

The term *blockchain* refers to a particular method for securing records in a database that all network users share. It is groundbreaking for being a *trustless* system, where individuals retain full autonomy over their funds, there is no central authority, and each participant can easily verify and audit the system.

Anyone in the world is welcome to act as a network maintainer, and each participant keeps the others honest by verifying the blockchain. When users broadcast information to be placed on the blockchain, network maintainers group these transmissions into blocks and use cryptographic tools to finalize the records and permanently link them onto the blockchain.

Once data is sealed onto the blockchain, it cannot be deleted, moved, or altered in any way. The records are immutable and each participant on the network has matching copy of the blockchain for their own verification. Most cryptocurrency blockchains employ a clever mining model that encourages network participation and keeps all of the records honest and synchronized. These types of decentralized systems are incredibly robust since there is no single server or central database that can be maliciously attacked or manipulated.

These decentralized systems are also trustless since each participant in the network maintains and verifies their own copy of the records, instead of relying on any third party. Given that blockchains provide a system for global tamperproof recordkeeping, they are extremely well-suited for storing financial data. In fact, the first modern distributed blockchain debuted in 2008 as the mechanism underlying the Bitcoin cryptocurrency.

On October 31st 2008, an anonymous individual or group known as Satoshi Nakamoto published a whitepaper describing “Bitcoin: A Peer-to-Peer Electronic Cash System.” This world-changing document laid out the framework for the open-source decentralized cryptocurrency and the revolutionary blockchain technology that makes it possible.

Figure 1.1 in the first section highlighted how sending money through the traditional banking system requires multiple transactions, separate ledgers, and trust in more than one bank. Figure 1.5 (below) shows how Maria could send money to George by transferring 10.5 Bitcoin from her address (1BuUygisXY) to an address controlled by George (1eK5FSywkp). This example references Bitcoin (BTC) for convenience, however nearly all cryptocurrencies use this type of public ledger and thus experience the following benefits and issues.

1.2.2 Blockchain benefits

A few of the blockchain benefits are immediately apparent:

- **Simplicity (& speed)**: Maria's money is broadcast to George in a single step to update a single ledger. Whereas bank and wire transfers can take days or weeks, cryptocurrency ledgers typically update in seconds or minutes (the transaction confirmation time varies for different cryptocurrencies).

- **No third-party risks**: Maria and George rely on their own cryptographically-secured and self-maintained system instead of placing their money and trust in the hands of third parties.

- **Pseudo-anonymity**: Unlike the banks, cryptocurrency ledgers never record real names

> such as “Maria” and “George” with the accounts. In fact, personal information is never necessary for generating an cryptocurrency wallet. George will access the funds pseudonymously, using his key for the 1eK5FSywkp address to which Maria broadcasted the money (from her account, 1BuUygisXY).

Bitcoin and the other cryptocurrencies that followed have initiated a financial revolution that is still unfolding. With these new decentralized networks, anybody can personally store and globally transfer funds at their own discretion. Prior to cryptocurrencies, it was difficult to store large amounts of wealth securely without trusting your savings to banks or credit unions. Likewise, transferring money to another individual or business required reliance on third-party payment processors for checks, wire transfers, or credit/debit cards.

Thanks to cryptocurrencies, for the first time, anybody can exercise their basic financial rights without requiring access to a bank and approval from external institutions! In mere moments, any device (computer, phone, tablet) can be used to initialize a new cryptocurrency wallet that is fully functional for receiving, storing, and sending funds. Setting up a wallet does not require any kind of identification, fees, or authorization, since the system identifies users by addresses that look like random strings of numbers and letters instead of personally identifiable details such as names, street addresses, or phone numbers.

1.2.3 Blockchain drawbacks

Most cryptocurrencies are pseudo-anonymous, since their users are identified by unintelligible strings of letters and numbers rather than personal identifiers. When you receive a cryptocurrency payment, you do not learn the sender's name; instead, you receive the funds from an address such as: 1A1zP1eP5QGefi2DMPTfTL5SLmv7DivfNa.

While this preserves privacy in some ways, it also exposes some sensitive information. Recall, every participant in a decentralized blockchain system can access a complete copy of the entire set of records. In the context of cryptocurrencies, this ledger is used to ascertain the account balance for any (e.g. Bitcoin) address.

On these shared transparent ledgers every account balance and history is public! In fact, several helpful websites allow you to easily search the blockchain for any address or transaction.

Suppose you run a shop, and one of your customers pays for a loaf of bread from the Bitcoin address 3P3QsMVK89JBNqZQv5zMAKG8FK3kJM4rjt. You can instantly check on the blockchain and see that this account has received more than 5,000 Bitcoins! Knowing that your customer handled $50,000,000 recently, you might be inclined to charge more in the future, or simply rob them now. This privacy issue presents a personal security risk.

In addition to knowing your customers' balances, you can also see every transaction that they have received or sent: the amount, the timestamp, and both participants' addresses. Analysis of transaction activity and history can be used to profile your spending patterns, income, savings, and with whom you interact.

A significant amount of your sensitive personal information can be exposed if your pseudo-anonymous blockchain identity is linked to your real-life identity (for example, during an online purchase or while registering for a cryptocurrency exchange). Often the owner of an account can be revealed with a little bit of research; for instance, you might have already searched for the two Bitcoin addresses listed above to learn that they belong to Satoshi Nakomoto and the Pineapple Fund charity, respectively.

Several companies exist solely to track and deanonymize transparent blockchains. For example, Elliptic offers an interactive explorer that shows the flow of funds between Satoshi, payment processors, exchanges, forums, marketplaces, gambling services, charities, known individuals, and other services.

Figure 1.6 shows a screenshot detailing significant Bitcoin transactions in the early 2010s, including connections between mining pools, Mt. Gox, and the Silk Road marketplace.

Take a moment to consider the valuable sensitive information that you generate each day: credit card transactions, every phrase that you search, products you view or purchase, social media sites that you interact with, etc... All of this information is routinely recorded and monetized by your banks, payment processors, giant tech/data industries, and governments.

This mass collection of your data results in centralization of your personal and private information in vast troves of sensitive material that are juicy targets for hackers and blackmarket resale. It is quite probable that your personal details (such as name, address, email, phone number, etc) are already in the public domain without your knowledge, perhaps connected with your demographic and/or marketing dossier.

Consider the recent Equifax, Target, Home Depot, Uber, and Panera data breaches. In many cases, both personal and financial information were compromised, putting individuals and their cards at risk.

Accidental data breaches are not the only concern. Big data and tech companies carefully record your activities online, so that they can profile your preferences in order to provide better services. Often, this is used for targeted marketing and ads; however, this data can also be leveraged for more questionable uses such as manipulating your feelings or your voting behavior.

Anything that a company tracks about you may end up stolen, carelessly resold, or used unethically. You should exercise great caution regarding your digital footprint, since information cannot be “unleaked” after your personal details are exposed.

Right now, privacy is conspicuously absent from mainstream economic and commercial systems. Traditional payment processors, banks, and cryptocurrencies leave very clear trails that are used tostudy, surveil, and profit from you. Once collected, you often have no way to control or track the proliferation of your data, or know of the privacy and personal security risks that arise from its sale to unknown parties.

The only guaranteed way to exercise your right to financial privacy is to avoid revealing personal information in the first place! To stay safe, we need a way to interact securely - where transactions cannot be linked to your identity, your savings, or other transactions. The Monero cryptocurrency is your best tool for taking all of these matters into your own hands!

1.3 Introducing Monero

MONERO (pronounced /mōnĕrō/, plural moneroj) is a leading cryptocurrency with a focus on private and censorship-resistant transactions. The openly verifiable nature of most cryptocurrencies (such as Bitcoin and Ethereum) allows anybody in the world to track your money. Furthermore, links between your financial records and personal identity may jeopardize your safety.

To avoid these dangers, Monero uses powerful cryptographic techniques to create a network that allows parties to interact without revealing the sender, recipient, or transaction amounts. Like other cryptocurrencies, Monero has a decentralized ledger that all participants can download and verify for themselves.

However, a series of mathematical tricks are used to conceal all of the sensitive details and stymie any blockchain tracking. Monero's privacy features allow the network to assess the validity of a transaction and determine whether or not the sender has a sufficient account balance, without the actually knowing the transaction amount or account balances! Nobody can view others' account balances, and transactions do not reveal the source of the funds being transferred.

One of Monero's crucial defining features is its philosophy of enforced privacy by default. Users are specifically prevented from initializing transactions that are accidentally or intentionally insecure. This provides Monero users with peace of mind since the network will not accept a revealing transaction! Monero users reap all the benefits of a decentralized trustless financial system, without risking the security and privacy downsides of a transparent blockchain.

Figure 1.7 shows how Maria pays George for the car, using Monero. The process is functionally the same as the cryptocurrency transaction shown in figure 1.5, however the sensitive information is cryptographically obscured. Information such as account balances and transaction amounts are marked with “***" in the diagram, since no outside observer can ascertain the values. The mechanics behind these unique privacy features are discussed in chapter 3 (conceptual) and chapter 5 (technical).

1.3.1 Principles of Monero

Monero is designed with the following principles in mind:

- **Network decentralization**: The Monero network and ledger are distributed globally. There is no single server or database that can be maliciously hacked, controlled, or censored. If one government were to shut down Monero nodes in their country, or attempt to limit who can send/receive Monero, the effort would be in vain! The rest of the world will maintain the network and continue processing transactions.

- **Financial security**: The Monero network is self-secured by incorruptible cryptographic mechanisms, so there is no need to trust a third party with responsibility over your funds and transactions. Every single Monero participant can verify the validity of the ledger themselves, so you do not even need to trust the node operators! (You can learn more about the cryptographic techniques that secure Monero in chapter 5.)
- **Financial privacy**: Most blockchain systems achieve strong security at the expense of privacy. However, Monero prioritizes providing total privacy with no security concessions. Transaction amounts, sender identity, and recipient identity are all obfuscated on the blockchain, so your Monero storage and spending activities are not trackable.
- **Fugibility**: The term fungibility refers to assets whose units are considered indistinguishable and interchangeable.. For example, imagine that you let your neighbor borrow 1 kilogram of flour for a cake. When they return flour the next week, of course it will be 1 kilogram of flour from a different source (since they used your original flour for baking). This is not a problem, since flour is fungible. However, vehicles are not fungible; if you let your neighbor borrow your car, you probably want the same one back!

In the case of Monero, its fungibility is a feature of its sophisticated privacy practices; the obfuscated transaction record obscures the history of all monero.. If you let your friend borrow 1 Monero, they can return any 1 Monero, since they're indistinguishable. This particular quality may seem like a minor nuance; however, fungibility is crucially necessary for most practical uses of any currency (see examples below). This characteristic is absent from most cryptocurrencies, with transparent ledgers and trackable histories.

1.3.2 Real-life “use cases” for Monero

This section talks about some of the difficulties and risks that arise from using insecure cryptocurrencies. For simplicity, the examples refer to “Bitcoin” as the prototypical transparent-blockchain currency. However, these drawbacks are present in essentially all cryptocurrencies.

- **Price manipulation**: Sofia is the only mechanic in a small town. One of her customers paid for an oil change with Bitcoin. Sofia later looked up his address on the ledger and saw that the customer's wallet contained enough Bitcoin for a new Lamborghini. Next time he needed a repair, she doubled her prices. If the customer had used Monero, Sofia would have been unable view his balance or use such information to manipulate prices.
- **Financial surveillance**: Oleg's parents send him some Bitcoin to pay for textbooks, then continue to snoop on his Bitcoin address and activity. A few months later, Oleg sends some leftover Bitcoin to the public donation address for an organization that does not align with his parents' political views. He does not realize that they are still monitoring his Bitcoin activity until he receives a furious email from his parents, berating him. If Oleg

had used Monero, his family would not have been upset due to prying into his transaction activity.

- **Supply chain privacy**: Kyung-seok owns a small business providing family catering services for local events. A large food company uses blockchain tracing to identify most of his regular clients. The corporation uses this list to contact Kyung-seok's customers, offering similar deals for 5% less. If Kyung-seok's business used Monero instead, its transaction history could not have been exploited by rival businesses seeking to steal his customers.

- **Discrimination**: Ramona finds her dream apartment, conveniently close to her new job in a great neighborhood. Every month, she promptly pays her rent in Bitcoin. However the landlord notices that some of the payments track back to a legal online casino. The landlord personally despises gambling, and unexpectedly chooses to not renew Ramona's lease. If Ramona paid rent with Monero instead, the landlord would not be able to review its history and discriminate based on her legal source of income.

- **Transaction security/privacy**: Sven sells a guitar to a stranger, and gives the buyer a Bitcoin address from his long-term savings wallet. The buyer checks the blockchain, sees the large sum of money that Sven has saved up, and consequently robs him at gunpoint. If Sven had instead given a Monero address for payment, the buyer would not have been able to view Sven's wealth.

- **Tainted coins**: Loki sells some of his artwork online to save up for college. When he pays tuition, he is shocked to receive a “payment INVALID” error from the school. Unbeknownst to Loki, one of his paintings was purchased using some Bitcoin that was stolen during an exchange hack the previous year. Since the school rejects any payment from a blacklist of “tainted” Bitcoins, they refuse to mark the bill “paid.” Loki is in an extremely difficult position: the Bitcoin that he saved has already been transferred out of his account, yet the tuition bill is still unpaid. This entire situation would have been avoided if Loki sold his paintings for Monero instead, since its fungibility precludes tracking or blacklists.

These examples have shown how Monero’s privacy features keep users safe from snooping family, tainted coins, and unethical business practices. All cryptocurrencies are a relatively new technology, and there is no such thing as “perfect privacy.” If keeping a particular payment secret is a matter of life and death, it may be risky to use any cryptocurrency for that transaction.

1.3.3 Monero: open-source decentralized community and software

Monero is an open-source project actively developed by cryptography and distributed systems experts from all over the world. Many of these developers freely donate their time to The Monero Project. Others are funded by the Monero community so that they can focus entirely on the project.

The decentralized nature of Monero's development team brings several benefits over a consolidated corporation or organization. The Monero Project is a living entity greater than any individual or group. Since both the network and development team are spread across the globe, it cannot be shut down by any single country.

The term open-source means that the source code (software blueprints) are made publicly available for anybody to inspect. The alternative is closed-source software, where developers only deliver the final compiled product (binaries such as .exe files) that cannot be opened and studied. If you use closed-source software, you are trusting the developer and distributor. The problem is that even a developer with the best intentions may make a mistake that hackers later discover and exploit. Only use open-source cryptocurrency software that has been audited by independent parties to verify absence of malicious code, accidental mistakes, and implementation weaknesses.

The cryptocurrency community has embraced open-source software from the very start: Bitcoin was released as a public white paper and open-source community-built code, which stood in stark contrast to the opaque and proprietary decision structure endemic to fiat (government-backed) currencies. Of course, the open-source philosophy has been around much longer than cryptocurrencies! Over 25 years, more than 5,000 coders have contributed to the opensource Linux kernel, which is widely considered to be one of the most secure operating systems.

The trust and security benefits of open-source software are of key importance for any cryptocurrency, so The Monero Project is entirely open-source. The developers use GitHub for version control, which allows anybody to easily review every single line of code proposed to be added, removed, or modified. Over 240 developers have contributed to, reviewed, and tested the Monero code, which drastically lowers the likelihood that any errors have been overlooked. Developers can find more information about interacting with Monero's codebase in chapters 6 and 7.

Development team transparency is very important for community trust, especially for cryptocurrencies. Monero development discussion occurs in open IRC channels, and the Monero Project website hosts public archives of meeting logs.

1.3.4 History of Monero

In 2013 Nicolas van Saberhagen published the “CryptoNote” protocol, which became the foundation for many coins, starting with Bytecoin. Like Bitcoin's Satoshi Nakamoto, the creator of Bytecoin remained anonymous and promoted their coin through a Bitcointalk thread.

Some aspects of Bytecoin appeared dubious under close scrutiny. Bitcointalk member “thankful_for_today” investigated the emissions curve and noted that approximately 82% of the coins had already been emitted, so the circulating coin supply was potentially dangerously centralized.

Ultimately, this greedy premine undermined Bytecoin's credibility and practicality. Thankfully, thankful_for_today recognized the value in CryptoNote's features, and incorporated them into a new project centered around a strong, community-driven development team. The Monero cryptocurrency, spearheaded by thankful_for_today, launched in April 2014. The coin was originally named “BitMonero,” however the community quickly elected to shorten it to “Monero,” which is the word for “coin” in the Esperanto language.

1.3.5 Ethical discussion

Monero was carefully engineered to provide characteristics like fungibility and transaction privacy that are necessary for any currency (crypto- or otherwise) to be feasible for general use. As discussed in the section “Real-life 'use cases' for Monero,” there are significant practical issues that arise with financial systems that do not protect users' privacy.

The very features necessary to keep Monero safe for day-to-day users and businesses are unfortunately also appealing to those wishing to conceal illicit activity. Monero is not specifically designed to facilitate illegal activity, which has plagued every currency since the idea of money was conceived thousands of years ago. The scale of illegal transactions conducted using cryptocurrencies is dwarfed by the staggeringly-vast amount of criminal activity that occurs every day denominated in fiat currencies like Euros, Rupees, Yen, or Dollars.

Monero mining is designed to be compatible with computers, phones, tablets, and most web browsers; this allows anybody to easily enter the mining ecosystem with no barriers from equipment costs. Unfortunately, hackers have taken advantage of this accessibility to create exploitative websites and software that secretly mine Monero for the attacker. Nonconsensual mining is tantamount to theft of resources, and the Monero community recently self-organized a team of volunteers to freely assist victims. The Malware Response Workgroup provides education, tools, and live support to combat software that employs Monero for malicious mining and ransomware.

The creators of Mastering Monero are excited about the currency's use for widespread personal, retail, and commercial applications. We hope that our readers use Monero ethically and often! You can discover online stores that accept Monero through Project Coral Reef. There are several websites that make it easy to use your equipment to philanthropically mine Monero to support various non-profits, such as UNICEF Australia, BailBloc, and Change.org.

Chapter 2

Getting started: receiving, storing and sending Monero

The last chapter focused on WHY to use Monero; in this chapter you will learn HOW to use Monero. You can master Monero without needing to learn any of the complex cryptographic or technical network details, so that extra information is saved for later in the book. Chapter 2 will cover all the practical skills you'll need to get started receiving, storing, and spending your moneroj.

The first part of this chapter covers key concepts and terminology for Monero use, as general information that will apply to any wallet or software. Toward the end of the chapter, you'll find handy guides for carrying out these steps using the free official open-source Monero command line interface (CLI) or graphical user interface (GUI) software.

What is a Wallet?

Before you obtain some Moneroj, you must plan ahead for where you will receive and store your funds. You will need a “wallet” to help you store and spend your Moneroj. Just like physical wallets for your bills, there are lots of different types of Monero wallets, and you can always move some of your money from an old wallet to a new one.

Wallets take care of the complicated cryptographic processes for you, so you don’t need to know any fancy mathematics to use Monero. You will only need to manage a seed and your address(es). Other details like “public keys” and “private keys” are managed behind-the-scenes by your wallet, so they are not discussed until chapter VI.

Your Monero seed is a secret string that your wallet uses to locate and spend your Moneroj, though it is converted into a series of 12-25+ words for convenience. This secret is like a treasure map to your money on the blockchain, and anybody who learns your seed can use their wallet to access and spend your Moneroj. For this reason, you must be extremely careful when you generate and store your seed. Do not set up a wallet in a coffee shop, where other patrons or cameras may see your secret. It is dangerous to store your seed electronically (e.g. in a text file or email) since malicious software or services may run off with your Moneroj.

Your seed is used to generate your address(es), which you can share with others. Most wallets will show your address in two different formats - a written string of numbers/letters, and a visual QR code. These are redundant, and you can safely share either form.

If your wallet is physically damaged, you can simply import your seed into a new wallet, and pick up right where you left off! As long as you have a copy of your seed, you can always access your funds. However, if you lose your seed, there is no way to ever recover access to your Moneroj. You may be familiar with passwords, which can usually be reset by contacting an administrator. Seeds are not like passwords - nobody else knows your secret, and the network is unable to shift your Moneroj to a new account if you lose your seed.

Most software will prompt you to write down the seed when you initialize a new wallet, however some apps do this in the background, and you must take the initiative to backup your wallet. Be sure to do this immediately, or else a damaged device will cause you to permanently lose your funds.

2.2 Selecting the best wallets for your needs

There are various storage solutions, and they vary in terms of convenience, privacy, and security. Your individual needs will determine which type(s) of wallets are best for you. Many people use multiple storage solutions: often a convenient “hot wallet” that holds small amounts for day-to-day use, complemented by a more secure “cold wallet” for long-term savings or large amounts. The wallet types described below differ primarily in where the secrets are stored.

2.2.1 Software and Mobile Wallets

Software wallets (on a desktop PC or mobile device) are convenient for storing and using Monero. Many Monero users have a handy “hot wallet” on their phones, to pay for small purchases. A good rule of thumb is to only walk around carrying as much cryptocurrency as you would feel comfortable holding in regular cash. The secret seed is stored on your device, so your Moneroj could be stolen if you catch a virus.

2.2.2 Hardware Wallets

Hardware wallets are physical devices that can carry out sensitive wallet functions, completely isolated from the connected phone or computer. Hardware wallets have their own built-in screens, to show you the seed and transaction details without ever sending them to an external device!

While hardware wallets are less convenient than software wallets, they are extremely secure! Because of how they store and protect your seed, you can safely use a hardware wallet to send transactions from a device that you suspect or know is compromised with malicious software. The Monero community is currently developing “Kastelo” - the first-ever open-source cryptocurrency hardware wallet.

2.2.3 Paper Wallets

Paper wallets provide an inexpensive way to stash Moneroj that you do not plan to move frequently; you simply print out a physical copy of your public and secret information for safe storage. Since the secrets from your Monero seed are saved only on paper - not digitally - you do not have to worry about viruses or data leaks. However, paper wallets are not convenient for frequent use, since you must transfer the secrets to a digital device every time you wish to send Moneroj.

2.2.4 Web Wallets

Web wallets are Monero accounts that you access through a website hosted by some third party. These online wallets are extremely convenient, however this comes at the expense of your security and privacy. There are essentially two types of web wallets - the crucial difference is whether or not you know the seed.

The riskier type of web wallet stores the money in their own accounts and gives you a username and password to log in (this includes your wallets on exchanges). Since you do not have the seed yourself, you do not personally control your funds; you must trust the service to hold your money for you. You should be extremely wary of storing Moneroj in these types of web wallets, which are essentially providing banking services. They might lose your funds at any time, whether through accident or theft. If the website is shut down, your username and password are useless - since you don't have the seed yourself, your funds are gone.

Well-designed web wallets, such as MyMonero, are developed so that the third-party server never learns your secret. You must enter your seed every time you log in, because it is not known to the provider or stored on your device between uses. This type of wallet is relatively safer since the third party is not holding your funds, only providing a software interface for your browser. If the website for this type of wallet becomes inaccessible, you can enter your seed into a different wallet and fully recover your funds.

While web wallets are convenient, neither type is recommended for long-term storage or large amounts. Both types have security downsides (trusting your funds to a third party, or frequently typing your seed into a web browser) and you lose many of the privacy benefits of Monero.

2.2.5 Cold Wallets

Cold wallets refer to any device that is generally kept offline and used only for storing your secrets. They could use any operating system, and you must be very deliberate with strong security (including a firewall, antivirus, and extreme caution regarding accessing only trusted websites/software). The seed is still on the computer, but you keep the device sequestered from the rest of the world as much as possible.

2.2.6 Monero Wallet Links

Regardless of which type(s) of wallet you choose, be careful to only download vetted software from through proper channels. Phishing schemes and scam wallets are numerous, so be sure to double-check that you are installing legitimate software! If you enter your seed into a malicious wallet, your Moneroj will be gone before realize your mistake.

This section contains links to several open-source wallets that are developed and trusted by the Monero community.

- Mac/Windows/Linux: An official Monero wallet is available with command line interface (CLI) and graphical user interface (GUI) versions. https://getmonero.org/downloads
- Android: Monerujo

- iOS : CakeWallet and XWallet
- Web Wallet : Mymonero.com (safer type of web wallet, secrets stay on your device)

2.2.7 Connecting to a remote node (optional)

You can reduce sync time and disk usage by connecting to a “remote node” instead of storing the entire blockchain on your device. Most mobile wallets are automatically configured to connect to a remote node. If you need to manually direct your software to a remote node, you can use the community resources at node.moneroworld.com port 18089.

Nodes are computers that have downloaded the entire blockchain, and assist other users by syncing their wallets and relaying their transactions. Running your own (local) node is best for privacy, and you can choose to share your node publically if you wish to help secure the network. Remote nodes are convenient, and allow a user to quickly begin using Monero without downloading the entire blockchain.

Running a node is not the same as “mining” Monero. Mining is a different resource-intensive process, not discussed until Chapter IV. Once the blockchain is synced, running a local node is not heavy on CPU or network resources.

2.3 Using Monero

This section explains what you need to know for sending and receiving Monero. All of the examples in this book use the following seed:

```
 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
```

You can import this seed in yourself to practice generating addresses, checking transaction history, and verifying payments. You can use this seed to follow along with examples in the book, but do not send your Monero to it! Anybody else reading Mastering Monero will be able to spend it!

2.3.1 Receiving Monero

To receive Monero, all you have to do is find your address through your wallet and share it with the person sending you Moneroj. Most wallets will show your address in two formats: an alphanumeric string that is easy to copy+paste, and a QR code that is handy for scanning with a camera. Here are examples of both formats, from the DEMO seed above:

<RECEIVING FIGURE DRAFT TO ADDRESS “4BKj...feW5”>

This address that you share is not stored on the blockchain (thanks to a Monero feature known as “stealth addresses” that are discussed in Chapter III and Chapter VI). Monero also allows you to generate multiple “subaddresses” from your single secret seed, so you can share many different addresses that all deposit to the same wallet.

Wallets may wait 10 - 20 minutes for “confirmation” before marking funds as received and safe to spend (you can learn why in Chapter IV). This is a common security practice, and wallets usually show the “unconfirmed” transaction during the waiting period. If your wallet is waiting for a 0.06 XMR payment to confirm, you may see something like this:

```
Balance: 0.075 XMR Available balance: 0.015 XMR
```

There is no need for concern when this occurs! Within less than a half hour, the funds will confirm and transfer to your available balance.

2.3.2 Sending Monero

To send Monero, you only need to input the recipient’s address and the amount that you wish to transfer. If you are sending Moneroj to a business, they may also ask you to include a “payment ID” to connect your payment to your order. If you are sending Moneroj to yourself or a friend, you can leave the payment ID field blank.

Your wallet will add small network fees to pay for sending your transaction. The fee associated with a given transaction varies depending on current network load and the location of your funds. Wallets will usually suggest an appropriate fee to send your transaction in a reasonable time.

2.3.3 Proof of Payment

Given Monero’s anonymity, you might wonder how somebody can prove that a payment was sent. Besides optional payment IDs, Monero has a second feature to selectively reveal proof that you sent funds. A legitimate “transaction key” can only be generated by the true sender.

Example

Suppose your friends Khan and Maria each owe you 0.06552376 XMR for a meal that you split. You only receive 1 payment, with the information below:

- **Amount**: 0.06552376 XMR
- **Transaction ID**: 4b540773ddf9e819f0df47708f3d3c9f7f62933150b90edc89103d36d42ca4b7
- **Received to (your) sub-address**: 899Ao1NQtu4ezACgw1QKXK4QBf5s8a3WHHtAjFfPm3Nf71mAkREEgAuKzASXHt8E7vVJFKsQJuvApBfu21WY9WN97Put8M5

This is a real transaction received by the DEMO wallet on 20-Apr 2018. You can see *some* information through a blockchain explorer, however the Monero sender is always unknown. Both Khan and Maria claim that they sent the payment, so you ask each to provide the transaction key.

Khan: `OutProofV1N4Y5pUJEnRACJyB5C3zK1zTqAihdnN8MfVZhEWfD13Z2N7Npt1uxa1EY7N7jnvuJF76tXUwKrakvZSxTj4Zux5SpavFb4X1jRcLAJ2b5hqviQPiS58j2qH53QL44CJEgHtY5`

Maria: `OutProofV1To53Qu2gegZbUevosKCTwrEdqiECgFyUygutXMEdhrHg1EtXMrFNaszWYFjdU4aXFZ2iPF8G8jzoDJzCoW5dsWvb4mVN65abAya3U47cGXs7WABrTzG5aPfV4YBANhwPgwD2`

When you check both of their transaction keys, Maria’s confirms payment to your address and Khan’s key returns “bad signature.” You can practice this yourself using the above address and transaction keys!

2.4 Operational Security

Monero allows you to “be your own bank”, since nobody can control your funds besides you! This grassroots financial empowerment is one of the greatest benefits of cryptocurrencies. However, with great power comes great responsibility! Keeping operational security (“OpSec”) in mind is important for keeping yourself and your funds safe.

2.4.1 Never say how much Monero you own

Sayings like “loose lips sink ships” exist for a reason. When you publicly disclose about how much Monero you have, you may inadvertently make yourself a target for scams or theft. This is especially true in cases of online forums and social media. Attackers prowl the internet looking for people to reveal information about themselves that they might be able to take advantage of.

Most individuals know better than to post about the balance of their bank account or retirement portfolio on social media. It is a security risk, rude, and can make interpersonal relationships awkward when wealth imbalances are involved. However, lots of people naively declare how much Bitcoin or Monero they have bought.

Remember that cryptocurrency prices are volatile, and are known to increase dramatically. A post stating “I just spent $50 on Bitcoin in case it lasts” from 2012 may have seemed modest at the time, however that $50 (~10 BTC in 2012) was worth nearly a million dollars by the end of 2017, less than 5 years later! Messages on the internet can be hard to erase, so the best way to avoid this situation is never posting in the first place.

Given the general interest in cryptocurrency investing, there is lots of conversation about holdings and portfolio composition. You should always talk in percentages rather than absolute amounts. Instead of saying “20 Moneroj and 0.2 Bitcoin” simply explain that your portfolio is comprised of “60% Monero and 40% Bitcoin”.

Advocating for Monero and teaching new users about it is one thing; boasting about how much XMR you own is another. Outreach is valuable, and the former is helpful for everybody; however, you should avoid the latter at all costs.

2.4.2 Keeping your seed safe

Your funds are only as safe as your seed, and there are two types of concerns: loss due to accident, and loss due to theft.

To avoid the loss due to accident, always make sure your seed is backed up somewhere secure. Always ask yourself: “If my phone dies or this website breaks, do I have a way to access my funds?”. You should consider keeping a second written copy of the seed in a second safe location. You don’t want to lose both your device and your backup seed if your house floods or burns.

To avoid loss due to theft, never share your seed or keys with anybody else. Anyone with access to your seed can steal 100% of your funds, and Monero’s privacy features will make it impossible for you to determine where they went.

2.4.3 Transaction precautions

When sending any large transaction to a new person or exchange, you should always first test the address/service by sending a smaller amount first. This test amount should be no higher than you would be comfortable losing, and is just a ‘pilot’ transaction to confirm that everything is set up correctly for the full amount.

With every cryptocurrency transaction, always double-check the send/receive wallet addresses to make sure they are correct. Even if you copy/paste the address, confirm it was pasted correctly and in its entirety. Hackers have created malware that manipulates cryptocurrency addresses in the clipboard (substituting the attacker’s address for the true recipient). If you visually double-check the address, you can catch this malware before you make a “donation” to a hacker.

2.4.4 Exchange safety

Exchanges create their own wallets for you, and generally do not share the seed with you. This is risky, since you have no way to recover your money if the exchange is attacked, shut down, or otherwise disappears. There is a famous saying “Not your keys? Not your Bitcoin!”, referring to wallets and services that retain their control over your funds.

As mentioned in 2.4.2, you should always be asking yourself “if this website disappears, do I have a way to access my funds?”. A good rule of thumb is to only keep your Moneroj on an exchange if you plan to trade it soon. Otherwise, move it to a wallet that you control.

2.5 “Getting started” for businesses

2.5.1 Monero is ideal for merchants

In this section, we covered all of the key skills for general Monero use. This section introduces a few extra tools for helping merchants integrate Monero into their systems and services. You can skip ahead to the next chapter if you are not involved with incoming business payments.

Merchants accepting payment in Monero benefit from fast, private, and irreversible transactions. There are several tools designed to ensure that accepting Monero is a “user-friendly” experience for both online and brick-and-mortar businesses.

Of course, anybody can use the general skills from the last chapter to set up a wallet and begin receiving Monero immediately. However the tools mentioned in this chapter are designed to facilitate use by businesses that wish to automate payment integration and processes like issuing invoices and receipts.

2.5.2 Friendly tools for accepting Monero

The Monero Integrations payment gateway allows any online shop to add a Monero payment option by simply installing any one of the plugins designed for several popular content management systems. The Monero Integrations solution was created (by the author of this book) to be consistent with the Monero ethos: the entire project free, open source, decentralized and, naturally, private. Transactions are routed directly to your wallet, so there are none of the privacy or security concessions that arise when trusting a third party to process payments.

Kasisto was the first point of sale system that accepted Monero, and is an open-source project requiring no third parties. The application is intended for in-store use on a phone or tablet, and can accept payments nearly instantly by detecting transactions before they have even been mined. You can try a demo at the Kasisto GitHub.

Another payment option is GloBee, which allows merchants to accept both cryptocurrency and credit card payments. GloBee is a third-party company, which allows them to provide additional functionality - for example, accepting many cryptocurrencies with instant settlement into Monero, other cryptocurrencies, and even fiat accounts (e.g. euros or dollars). This gives your business the option to accept cryptocurrencies and be paid immediately in your local fiat currency, eliminating exposure to price volatility risk.

Chapter 3

How Monero works

The first two chapters covered everything you need to know about WHY to use Monero (Chapter 1) and HOW to use Monero (Chapter 2). By now, you have learned everything necessary to begin using Monero yourself!

The rest of this book contains extra details, for those wishing to dive deeper into how Monero works “behind the scenes.” Chapters 3 and 4 describe underlying technologies such as Monero’s privacy features, the blockchain, and the mining process - focusing on the understanding concepts, without digressing into the advanced mathematics. Chapters 5 through {?} contain those nitty-gritty details for developers and cryptography geeks.

Transaction and the Ledger

To set the stage for understanding Monero privacy technologies, we will consider how Monero are sent and received on the ledger. For this chapter, we will focus on blockchain functionality - an inherently tamper-proof shared database that keeps a list of Monero transactions. The details about blockchain security (mining, hashes, etc) are another topic, reserved for Chapter 4.

When you set up your wallet, it uses your secret “seed” to mathematically calculate sets of “public keys” and their corresponding “private keys” (technical details in chapter X). This process is carried out on your device, and can be executed offline; nothing is broadcast or recorded by the network during wallet generation.

You share your address (created from your public keys) to receive funds, which you can then access with your corresponding private keys. Until somebody sends you Monero, there will not be any entries on the blockchain that your private keys can unlock.

When somebody (a customer, an exchange, or a friend) sends you Monero, they will transmit a transaction that transfers some of their Menoroj to a new address that you can unlock with your private keys. In technical lingo, the “output” of their transaction is stored on the blockchain, for you to spend with your private keys at your leisure.

This jargon can be a bit confusing, since the cryptocurrency use of the word “output” is different from the typical meaning. All of the Moneroj that you “own” are simply outputs on the blockchain that your private keys unlock.

Each time you receive Monero, you gain another output; each time you spend Monero, you use up one of your outputs and generate a new one for somebody else.

When your wallet is “scanning” or “syncing” this usually means that it is checking all of the outputs on the blockchain to test whether your private keys unlock any for you to spend. Your wallet “balance” is the total sum of these outputs that you can unlock. When you spend your Monero, you use up some of your outputs as “inputs” to a transaction that you broadcast to the network. Conceptually, the blockchain is simply a record of these transactions, each consuming the sender’s outputs as inputs, to generate a new output for the recipient.

The process described above is slightly simplified, to convey the crucial parts (private/public keys, transactions, inputs/outputs). The following sections delve into more details about how Monero’s privacy technologies.

Privacy Technology Overview

The general principles and terminology introduced above are shared by most cryptocurrencies. Monero provides enhanced functionality and privacy through several unique cryptographic technologies that shield the users and their activity from public visibility. Figure X shows which features protect which transaction details:

- RingCT conceals the transaction amount
- Ring signatures obfuscate the sender
- One-time addresses ensure that the recipient address is not recorded on the blockchain
- Kovri breaks the link between your transactions and physical location by obfuscating the broadcast origin and concealing network signs of Monero activity

Ring Confidential Transaction

RingCT is a cryptographic technology that conceals the amount of Monero being sent in any transaction. With most cryptocurrencies, transaction amounts are sent in cleartext, visible to any observer. RingCT keeps this sensitive information private by allowing the sender to prove that they have enough Monero for a transaction, without revealing the value of that amount! This is possible thanks to cryptographic “commitments” and “range proofs.”

When you send Monero, you “commit” the amount in a private way, revealing just enough information for the network to confirm the validity of the transaction, while not publicly disclosing the amount itself. Although the commitments look like encrypted output amounts, the miners are able to confirm that the transaction is not trying to fraudulently create or over-spend Monero. “Range proofs” are another important mechanism in RingCT, that ensure the committed amount is greater than zero, and less than a certain number. This is important, to prevent senders from committing negative or impossibly-high amounts of Monero. This secures the supply of Monero against fraudulent manipulation.

Prior to RingCT, Monero transactions were broken up into specific denominations (for example, 12.5 XMR would be sent as 10 XMR +2 XMR + 0.5 XMR) and the transaction amounts were visible to outside observers. RingCT was activated in January 2017, and rapid widespread adoption immediately followed. Within 1 month of its activation, approximately 98% of new transactions were using the RingCT protocol! Following Monero’s policy of enforced privacy-by-default, after September 2017, RingCT became mandatory for all Monero transactions. To spend any old pre-RingCT outputs, they must first be converted to RingCT outputs with masked amounts.

Stealth (one-time) addresses

Stealth addresses are a Monero feature that protects the privacy of transaction recipients. Suppose you wish to give a few books about coping with a sensitive illness to your friend

André, however you're about to leave town for a trip and André won't be around until next week. Perhaps you could ask your friendly neighbor to temporarily hold onto the books and pass them forward to the recipient.

Since André is a private person with a sensitive condition, you don't want to simply tell your neighbor his name. How could you arrange the exchange while preserving André's privacy? You could simply make up a one-time random code and tell your neighbor to give the books to whomever presents that code (e.g. give these books to the person who knows the phrase "PolarComboTango357"). Your neighbor will be able to keep track of the books and give them to André, without learning anything about their recipient.

Similar to the way you might use that random non-informative code to keep your neighbor from learning about your book recipients, Monero uses a system of one-time codes to prevent the network from learning about Monero recipients! Instead of explicitly recording the recipient's address in the blockchain (analogously, "give the books to André"), funds are always sent to a one-time "stealth address" (analogously, "give these books to the person who knows the phrase "PolarComboTango357"). The cryptographic techniques that secure stealth addresses solely for the recipient are discussed in chapter X, however the salient points are detailed below.

How is are these one-time addresses generated? Your regular Monero wallet address is a 95-character string, which incorporates two public keys (the "public view" and "public spend" keys) mathematically derived from your seed. When somebody sends you funds, they will use the public keys in your address along with some random data to generate a unique one-time public key. These one-time public keys that are recorded on the blockchain with receipt of funds are called "stealth addresses" because it is impossible for the network or an outside observer to connect these random codes back to the originating wallets. Improving privacy by not recording wallet addresses on the blockchain is a clear consequence of stealth addresses. An even bigger implication is that use of these unique one-time keys prevents multiple payments to the same address from being linked together!

Suppose you create some public art or workspace and post an address for cryptocurrency donations. If you use a coin with a transparent blockchain (e.g. bitcoin), then every incoming transaction will record that address in a searchable linkable form. Anybody can use a blockchain explorer to see how many donations you received, their amount, and whether or not you've moved the funds. Every incoming bitcoin transaction is indexed on the ledger by the address that you shared publicly.

If you post a Monero address instead, your donations are not exposed to public scrutiny. Each donor will generate a unique one-time stealth address, and record that to the ledger. The address that you posted will never appear on the blockchain, and the stealth addresses do not provide any information about the recipient. Since each donor mixes in their own random information to create the stealth address for their transaction, one donor would not recognize the stealth address generated by another.

[image]

All Monero transactions use stealth addresses, to enforce privacy for the entire network; your wallet automatically carries out the conversion from public addresses into unique one-time keys.

Ring Signatures

Ring signatures are a Monero feature designed to protect a transaction's sender by not revealing the true source of the Moneroj being spent. Before jumping into ring signatures (a Monero feature), we'll introduce the concept of digital signatures in general.

"Digital signatures" are a cryptographic method for proving the source of a message and verifying that its contents have not been tampered with. Varying implementations of digital signatures are a key component of all cryptocurrencies. To spend one of your outputs, you compose a message to the network describing the transaction, "sign" it with your corresponding private key, then broadcast the result to the network. Before executing the transaction, the network checks the validity of the signature to verify that the message has not been altered and/or forged by a third party without the correct private key.

With transparent cryptocurrencies (e.g. Bitcoin, etc) the messages describing each transaction explicitly declare which outputs are being spent. This is useful for easy bookkeeping, since the network simply keeps track of unspent transaction outputs (UXTOs) that are valid inputs for new transactions. If somebody tries to spend the same bitcoin output twice (called "double-spending") the fraudulent second transaction would be rejected, since the network knows that the owner already spent that output when they signed the first transaction. Unfortunately, this straightforward proof of ownership is highly detrimental to privacy, by definitively indicating the source of funds, and when a given output is spent.

Monero uses a different scheme known as "ring signatures." This group-signing method allows one member to digitally sign the message on behalf of the group, while mixing in the public keys of the other members so that it is unclear who actively signed the message. It is possible cryptographically verify that one of the "ring members" signed the message, but impossible to determine which of the members actively created the signature.

Ring signatures are used in Monero to blend the keys from multiple outputs on the blockchain, to conceal which output is actually being spent. Suppose Maria wants to spend one of her Monero outputs, shown in red below. Her wallet will semi-randomly select several other past outputs on the blockchain (not belonging to Maria) and mix their public keys into the ring signature as "decoys." The network is able to verify that one of the outputs is being spent, but not which output.

Ring signatures protect the sender in all transactions, since the recipient and Monero network are unable to ascertain which output in the ring is the true source of the funds. A significant consequence of ring signatures is that an outside observer is unable to definitively prove that an output has been spent! The fact that an output appears in a ring signature is entirely inconclusive, since it is impossible to distinguish whether it was truly being spent, or simply passively employed as a decoy ring member.

If it is impossible to tell whether a particular output has been spent, what is to stop somebody from trying to spend the same output twice? With one-output-one-signature transparent blockchains (e.g. Bitcoin) this is a trivial task: if some output has been used in a digital signature once, it is considered spent and cannot be used again. However, Monero outputs can show up in ring signatures before and after they have been spent, so double-spending must be prevented through other means.

This is accomplished through “key images” that are generated and recorded with each signature, uniquely derived from the actual output being spent. The network cannot ascertain which output created the key image, but it can check whether or not the key image has been used before! If an unsavory user were to try and spend the same output twice, it would generate the same key image both times, and the network would reject the fraudulent second transaction. Consequently, the network can prevent double-spend attacks, despite not knowing which outputs are spent!

The Monero network did not originally mandate ring signatures, which unfortunately allowed privacy-damaging “zero-mixin” transactions. Starting in 2016, the network began requiring two ring members for each signature, enforcing privacy-by-default for the sender. This was raised to a minimum “ringsize” of four possible signers in 2017, and set to a minimum of seven possible signers in 2018.

Kovri

Kovri is a Monero feature that protects a transaction’s sender by concealing their physical location. Any device connected to the internet is assigned an IP address as an identifier to help route traffic to the correct user. However, this IP address can easily be connected to a user’s physical location and identity.

The cryptographic measures described in the previous section to protect the sender are effectively circumvented if an adversary uses the IP address that broadcast a transaction to identify its source. It is worth considering the unfortunate scenarios that can arise when Monero network activity is connected to physical location and identity.

The Monero network includes thousands of “nodes” scattered throughout the world. Any volunteer can turn their computer into a node and join the Monero network in receiving and relaying transactions. Nodes and adversaries with significant resources may be able to use a transaction’s originating IP address to identify the physical location of the sender. An unfair node might choose to not relay transactions from certain individuals or groups. Even worse, the geographic information revealed by IP addresses might lead adversaries pay a malicious visit to cryptocurrency users’ doorsteps.

The connection between IP addresses and Monero activity is not only a threat for the users broadcasting transactions. The network traffic through nodes is currently visible to internet service providers and other surveilling parties, which could put node owners at risk if their government or ISP does not endorse cryptocurrencies.

Cryptocurrency miners can also experience unfair treatment if their IP addresses are connected to their network activity. Unfair parties might seek attack certain miners by censoring their blocks, perhaps due to some ideological disagreement, or to limit non-government or non-corporate mining.

Clearly, all parties in the Monero ecosystem benefit from the decoupling of their network activity from their IP addresses (and thus their physical location/identity). This type of anonymity is provided by Kovri, which is a routing technology designed to obscure transmissions' sources. It is based on the decentralized Invisible Internet Project (I2P) specifications, which uses encryption and sophisticated routing techniques to create private network distributed across the internet. Messages broadcast on the I2P-based network cannot be connected to their originating IP addresses, which protects the privacy of end users and eliminates the ability to censor cryptocurrency activity based on physical information about its network participants.

The Monero community is developing this lightweight security-focused software with a general open-source implementation and common API that can be used for other application. Since I2P is a peer-to-peer technology, all cryptocurrencies and services using the Kovri network benefit from increased adoption! Kovri will soon be included with Monero releases, and enforced for all transactions as part of Monero's privacy-by-default policy.

Privacy technology summary

Monero uses several unique privacy technologies to protect various elements of the network and both parties in all transactions. RingCT conceals the amount sent in each transaction. Ring signatures protect the sender by concealing the source of the funds, while One-time addresses ensure that the recipient's address is not recorded on the blockchain. Kovri is a transaction-routing technology that breaks the link between your Monero activity, and your physical location/identity.

Together, these features ensure that Monero users remain anonymous, and that funds are not trackable. By cryptographically eliminating all the links used in analyzing transactions on the blockchain, Monero achieves "fungibility," which is a necessary characteristic of practical currencies. Now that you have read this chapter, you can understand how Monero protects the individuals described in the use cases described earlier.

Chapter 4

Blocks and Mining

In this chapter we'll be learning about blockchain technology and how Monero miners keep the ledger secure. We'll begin with what information a "block" contains, the way that blocks are linked in a tamper-proof chain, and how miners use "proof of work" to arrive at "consensus" on an agreed version of the blockchain. On the topic of miners, we'll discuss the source of new Moneroj, and how the coins are minted and distributed into the ecosystem. Near the end of the chapter, we'll dip our toes into a few cryptographic concepts ("hashes" and "nonces") to really understand what the miners are actually working on.

The simplified anatomy of a block

The previous chapter discussed the way transactions are constructed. In summary, your wallet drafts a message with instructions to transfer one of your outputs to a new recipient. Sensitive information in the message (sender, receiver, and amount) is cryptographically obscured, then your wallet authorizes the message by digitally signing it with your private key.

Now we consider how this message is processed by the network, to enact the actual transfer. Your message is received and relayed by "nodes" on the Monero network, which place your pending transaction in the "memory pool." The Monero miners collect pending transactions from the memory pool, and bundle them together into "blocks." A simplified block is shown below:

[image]

As shown above, each block contains a set of pending transactions, a cryptographic link to the previous block (called a "hash"), and a place for the miner to include a special string that completes the block (called a "nonce).

If you want to learn how hashes and nonces work, there is a friendly introduction with examples toward the end of this chapter (you can skip ahead [link] and read it now if you're curious about the cryptography.) There are only two concepts that you need to know in order to understand how these techniques function to secure the blockchain:

1. The "hash" is a security feature that proves that each block is directly linked to an unaltered version of the previous block. If an attacker tries to tamper with any point in the ledger, even the smallest attempt will be blatantly obvious because the hashes will raise a red flag on every subsequent block.

2. The "nonce" is a special string that completes the block and marks it as prepared for the blockchain. It is extremely computationally difficult to find a nonce that satisfies the requirements necessary to finalize and seal a block. Miners spend most of their time and energy searching for valid nonces. It is impossible to plan ahead for calculating the nonces, and the search must start from scratch with each new block. Nonces are not mathematically meaningful; only one-time strings of random characters.

Miners add new blocks to the longest chain

The miners collect pending transactions from the memory pool, and verify their authenticity by checking that the cryptographic proofs and signatures are valid, and that the key image has not been used before (see the chapter 3 section on ring signatures [link], to review why this is important). They collect the valid transactions into that portion of the block, and include the hash for the previous block to include act as a cryptographic link. Lastly, the miner attempts to guess a nonce that can be used to complete the block.

At any given moment, there are thousands of miners all working separately (or in teams, known as “pools”) to be the first to find a nonce that completes the current block of transactions. Whichever miner finds a nonce and finalizes their block first announces their version to the rest of the network. This miner is paid by receiving brand-new Moneroj called the “block reward,” which is discussed more in the section describing Proof of Work. Upon receiving the completed block, the other miners all add it to their copy of the blockchain, increasing the chain’s “height” by one block. Transactions added to the blockchain are removed from the memory pool, and the other miners discard their own (incomplete) work on that block in order to begin preparing the next one.

The global nature of the Monero ecosystem and unpredictable transmission delays due to network latency mean that occasionally two miners will independently complete slightly-different blocks at the same height. For example, a miner in South America might be the first to complete a block, however another miner in Europe finishes work on a similar block (containing the same transactions) before receiving the broadcast from South America. In this case, the western hemisphere may be temporarily using a different blockchain from the eastern hemisphere. For a brief moment, there exist two competing Monero ledgers that may not contain all of the transactions. One might think that this would be a catastrophic occurrence!

On the contrary, this situation is easily resolved by elegantly imposing a simple rule: miners all agree to work on the solving the next block in the longest chain. This is the central idea to “consensus,” which allows the Monero network to nimbly reunite to a single chain after an accidental split. Instead of trying to resolve the discrepant blocks immediately, the miners just continue trying to solve the next block on whichever chain they are synced to. Within the next few minutes, one of the miners will solve a subsequent block and add it to their chain. As soon as this occurs, their version is the longest chain, so all of the miners unanimously adapt that copy and discard the other “orphaned” block. Any transactions that were in the orphaned block will be re-mined in the next chain, so there is no cause for concern. By simply following whichever chain mines a block next, the network completely resolves any splits and returns to consensus on a single universal ledger.

Monero taxis use a hard puzzle to ensure fairness

These validation and block preparation tasks described above are not computationally difficult. The “hard” part for the miner is finding a nonce that allows them to complete the block. This is a puzzle designed to be extremely challenging and solvable only by brute force testing solutions; there is no way to shortcut the process or mathematically narrow down the search for a valid nonce. Miners simply generate random strings and test whether they complete the block, by trial and error.

The presence of this arbitrary obstacle may seem peculiar at first! The miners carry out a computationally-easy crucial role for the network (validating transactions) but are required to carry out a useless difficult task (finding the nonce) in order to submit their answer.

To understand the reasoning behind this, consider the hypothetical Monero Taxicab Network with only a few vehicles, and many taxi drivers that can check out one of the cabs if they submit an approved route. Throughout the day, potential riders call in and request rides all over the city. All of the requests from riders who have not been picked up are collected in a real-time "pool" of pending rides. Instead of a central taxi authority assigning each incoming ride to a car or driver, each taxi driver looks at the pool and puts together their own list of 5 - 10 trips that they could complete in the next 20 minutes. This part of route/ride planning is easy and fast for experienced taxi drivers! Once a driver puts together a list of trips that they can complete in the next "block" of rides, they complete a final task described below, then submit their route to the Monero Taxicab Network. If the driver has proposed a valid route with real pending riders, then the plan is approved! The riders included on the taxi driver's plan are removed from the pending pool, the taxi driver checks out a vehicle, and travelers are soon shuttled to their destinations. The taxi driver is rewarded with a fixed-size commission for carrying out the route, and the driver also collects fare from each of the riders.

So far, this should seem fairly intuitive! Pending trips are listed in a pool; when a taxi driver successfully submits a proposed block of trips that includes a given pending traveler, the rider is removed from the pool and shuttled to their destination. However, the Monero Taxicab Network has a very peculiar rule: in order for a driver to submit their plan for a block of rides to the Monero Network, they must do some difficult useless task. Imagine that the driver must scramble all of the letters in the riders' destination addresses and use some of them to generate five sentences (> 50 words total) that can say anything, but must have correct grammar/spelling in the local language. A driver submitting a planned route must include both the list of rides and the nonsense ("nonce") sentences that match the letters in the destinations, or else their route will be automatically rejected. There are multiple valid nonce sentences that can be constructed from most sets of letters, and the resulting sentences are absolutely meaningless for anything besides submitting that set of rides to the Monero Taxicab Network.

A property of this type of task, which will have parallels in the Monero cryptocurrency, is that it is very hard to find a nonce, but very easy to verify. For this taxi scenario, is quite difficult to rearrange a dozen addresses into 50 words in meaningful sentences by hand. However, it is easy for somebody else to review the nonce result to see whether it contains 5 valid sentences. When a driver submits their nonce in the format shown below, you can very quickly check that "Apple jam is very bad" is a valid sentence, and that letters are uniquely drawn from the rider destinations. This verification is nearly instantaneous, compared to the time it took the driver to rearrange the letters and find several sentences.

[image]

A seasoned taxi driver would be able to plan the driving route from the trip list in less than 60 seconds, however it will probably take them a few minutes to rearrange letters and find nonce sentences by hand. In fact, most of their effort in preparing their proposed route block will be spent on finding this useless nonce.

Imagine this process from a taxi driver's perspective, beginning right after the last route was assigned. You quickly pull down several trips from the list of pending rides and begin working furiously to rearrange letters from the destinations into some kind of nonce sentence. For a few minutes, you and all of the other taxi drivers are working on the same list of pending rides, each trying to craft a long enough nonce with > 50 words. Suddenly, a different driver submits a list of the rides along with a completed nonce. All of the trips that you were working on disappear from the pool! You have to throw away your work on that block (since the riders are already en route) and switch to the next set of pending trips. The process to find a valid nonce for that set of riders begins again from scratch, based on the new set of letters in their destinations.

Why would the Monero Taxicab Network impose such a difficult useless task upon their drivers? It is actually their surefire way to ensure that customers are served fairly! Imagine that a few of the taxi drivers behave unethically in some way, perhaps ignoring pending rides called in from a certain part of town, or selecting only the riders that are going to businesses that have bribed those unfair drivers. Without the nonce requirement, these small groups or individual malicious drivers might dominate the ride selection process for the whole business by constantly submitting their (unfair or exclusionary) routes as soon as a car becomes available. In this way, they could systematically treat some groups of customers poorly, which is absolutely antithetical to the core principles of the Monero Taxicab Network, which is dedicated to serving all riders fairly.

The nonce task competition between many taxi drivers is crucial to Monero's goal of ensuring that cars and rides are provided fairly. Assuming that all of the taxi drivers can rearrange letters at roughly the same speed as each other, it will be somewhat random which driver lucks into a solution and is allowed to submit their block of plans first. It is unlikely that any driver could be the first to submit multiple block plans in a row (i.e. be the first to find nonce sentences for several sequential blocks) since each driver is competing against, and collectively outnumbered by, all of the other drivers.

Most of the multitude of potential drivers will be honest individuals, submitting fair ride block plans to keep the city running smoothly. If there are a few malicious drivers who wish to submit unfair plans, the nonce task prevents them from controlling the entire system. Statistically, they will occasionally be the first to find the nonce, and thus able to submit their exclusionary route plan for that block. However, the rest of the drivers, most of whom are honest, will immediately begin working on their fair blocks to propose for the next set of rides! Due to the random nature of who lucks into finding a nonce first, the next set of rides is most likely to go to an honest driver who will include the overlooked trips.

This system of imposing useless work to randomize which drivers' routes are accepted allows the Monero Taxicab Network to be sure that the majority of honest drivers are gaining access to the cars to carry out business fairly. The Monero Taxicab Network thus has no central authority that is responsible or liable for controlling ride activity and assignment. Instead, this task is "decentralized" to the individual drivers, employing the useless nonce competition to randomly select which route proposal is accepted. This statistically that the cars will usually go to honest drivers, so the Monero Taxicab Network has an excellent reputation for providing fair service to all customers.

By now you're probably wondering how this extended taxicab metaphor is related to the cryptocurrency that this book is ostensibly about! You've probably guessed that the decentralized Monero Taxicab Network is an overt analogy for the Monero cryptocurrency network, which needs to provide fair global service without any central authority (the importance of trustless decentralization is discussed in Chapter 1 [link]). Each trip corresponds to a Monero transaction, pending in the memory pool until it is selected for a spot in the car/block. The taxi drivers represent miners. Both carry out an easy important jobs (taxi drivers plan routes; miners collect and validate transactions), yet are forced to compete against other drivers/miners in a useless difficult nonce task, so that the barrier randomizes who succeeds first - thus statistically distributing most of the cars/blocks to honest drivers/miners. Whichever taxi driver submits their route first was rewarded with a fixed-size commission, and fare from each of the riders who attained a seat. Likewise, miners are rewarded with a commission (called the "coinbase" or "block reward") for solving each block, and also collect fees from the transactions included in the block.

Cryptocurrency "proof of work" concepts

Now we'll step away from the taxi analogy and directly discuss the systems in place to ensure fairness in Monero. This process of coupling important network functions with useless nonce work is referred to as a "Proof of Work" system. Many cryptocurrencies operate on various PoW systems; there are differences between the implementations and their characteristics, however they share a common theme of enforcing decentralization by requiring validation to be submitted with a nonce. Sometimes the nonce itself is referred to as the "proof of work," referring to the piece of data that was hard to find/create and easy to verify, like the taxi drivers' anagrams.

Miners measure how quickly they can work toward mining blocks in "hashes per second", abbreviated H/s. Each miner can measure their "hashrate," which varies depending on the equipment that they are using to mine. The "network hashrate" refers to the total hashrate of all the miners working on preparing blocks.

PoW systems prevent censorship

In PoW systems, the miners compete at finding nonces to randomize which miner's block is accepted first, becoming the latest block on the longest chain. As mentioned in the taxi analogy, the PoW framework prevents effective censorship. Some malicious miner in the Monero network may try to provide preferential treatment or exclude certain transactions, however an honest Monero miner will simply include those transactions in the next block.

PoW systems prevent double-spend attacks

There is another blockchain challenge solved by PoW systems that was not included in the taxi analogy. Specifically, a malicious miner might try to "double spend" a transaction. This refers to an attack where the miner creates blocks to undo their past transactions and steal back the money for themselves. The attack would have to proceed in this manner:

- Malicious miner Martin uses his first transaction to send Moneroj to victim Valerie
- When the "Martin → Valerie" transaction is mined onto the blockchain, Valerie believes she has been paid.
- Martin takes whatever he was buying from Valerie…
- Then mines a different version of the block that contained the "Martin → Valerie" transaction.
- In Martin's alternative version, the transaction to Valerie doesn't exist! Instead, his second version contains a transaction that sends that Moneroj to one of his wallets, instead of Valerie's
- If the Martin can quickly mine enough blocks to make his chain the longest, then the network will accept his alternate reality. In practice, the infeasibility of this step prevents double-spend attacks from actually occurring.
- Since the key image for Martin's output appears on the chain (now associated with the transaction to himself), the network will no longer accept the "Martin → Valerie" transaction as valid since the key image associated with that output is already spent.

At this point, Martin has left with whatever he bought from Valerie and stolen back the Moneroj that he initially used to pay her. Thankfully, PoW systems prevent an attacker from sustaining this type of double-spend attack, by limiting the speed with which they can generate blocks. Recall that miners will always follow the longest blockchain, so the malicious miner would have to change the block with the previous transaction then re-mine every block afterward, fast enough to overtake the length of the main chain. Since the attacker will be working alone to generate the altered blocks, against the hash rate of the entire rest of the network building the real blockchain, they will be unable to catch up. This type of attack could only feasibly succeed if the malicious miner has as much computing power as the entire rest of the network combined. For this reason, the term "51% attack" is often used to indicate that some sort of malicious activity would require majority hash rate.

A cryptocurrency with more miners will be more difficult to attack in this way, since the malicious miner must have more computing power than the rest of the global network. This is why increasing the total hashrate by including more miners helps to secure the network against attacks.

PoW difficulty adjustment controls blocktime

For practical considerations, the Monero network aims to add a new block onto the chain approximately every two minutes. With each mined block, some transactions are moved from the pending memory pool onto the confirmed blockchain. If the average blocktime becomes too long, transactions will be too slow to confirm. If the average blocktime is too short, then the network could get out of sync more often and run into more issues.

The network controls how fast blocks are mined by adjusting the “difficulty” of the nonce puzzle. As more miners join the network over time, their collective guessing power (hashrate) means that they will be able to search for nonces more quickly. This would statistically cause miners to find blocks faster than the target blocktime of 2 minutes. So the difficulty of the puzzle is increased, meaning that it is takes longer to find a nonce that matches the arbitrary requirements. Likewise, the difficulty can be adjusted to be easier if mining power decreases and blocks start being mined too infrequently.

In the taxi network analogy from the previous section, the difficulty of the letter rearrangement task could be similarly arbitrarily adjusted by requiring more or fewer words in the nonce sentences. If 20% of the taxi drivers (miners) did not participate one day, then it would take longer on average for route plan blocks (completed with nonces) to be submitted, so some cars would be idling with no driver. To correct this, the taxi drivers would agree to lower the nonce requirements from 50 words to 40 words. This would put the drivers back in sync with the availability of cars.

Difficulty increases proportionally with total network hashrate to keep the flow of blocks constant.

Monero uses a modified CryptoNight PoW function

Monero uses the CryptoNight PoW algorithm, which is significantly different than the systems used by most other cryptocurrencies. The most notable difference Monero’s deliberate use of a PoW function that is difficult to optimize for specialized mining equipment.

In most contexts, ‘optimization’ is a good thing, so you might be surprised that Monero’s PoW algorithm intentionally stymies accelerating mining speeds. This is because the ability to create overpowered mining equipment can lead to a dangerous centralization of miners. These risks are perfectly illustrated in the history of Bitcoin mining.

The history of Bitcoin mining

When cryptocurrencies entered the scene with Bitcoin’s appearance in 2009, mining occurred exclusively on computer CPUs. Since the network mining difficulty adjusts to the current total hashrate, CPU mining was adequately profitable in the early days. CPU miners have hashing power on the order of 1,000,000 H/s, written as 1 MH/s for convenience.

Soon, graphics cards were repurposed for mining cryptocurrencies. GPUs are able to attack the mining problem orders of magnitude faster, around 100 MH/s. Since the network difficulty adjusted based on the GPU miners, the CPU miners could not compete (i.e. mining rewards were insufficient to pay for the equipment and electricity costs).

Next, application-specific integrated circuits (ASICs) were built for the sole purpose of mining Bitcoin. These special devices are quite expensive, and mine many thousands of times faster than GPUs - more than 1,000,000 MH/s. By now, the network difficulty has increased to accommodate the ASICs, consequently pushing CPU and GPU miners out of business for Bitcoin.

Bitcoin was initially launched with the vision that anybody in the world with a computer could begin mining to secure the network and obtain Bitcoin. Unfortunately, the creation and proliferation of ASICs very effectively ended this dream. If you wish to begin mining Bitcoin now, you will have to obtain an ASIC for hundreds or thousands of Euros.

This ASIC takeover put the vast majority of Bitcoin miners out of business. The network began its existence secured by scores of computer geeks scattered across the globe, all participating on their personal computers and graphics cards. Sadly, this true decentralization of Bitcoin is a bygone era. Now the network is dominated by several large corporations with massive ASIC farms, who have effectively become the Bitcoin backbone.

ASIC centralization is dangerous

Since many of the main cryptocurrencies are dominated by ASIC miners, it is worth giving consideration to the topic and its risks. Centralization occurs in two forms: ASICs are only produced by a few companies (centralization of manufacturing) and subsequent mining tends to be limited to a few large farms (centralization of mining).

Centralization of ASIC manufacturing and mining to a few large corporations allows hackers, attackers, and governments to exert larger control over the network and its regulation. This begins to nullify many of the benefits of decentralization. For example:

- Universal access to mining flourished in the days of CPU and GPU mining, which uses unregulated general-purpose hardware. However, mining now requires specialized hardware, which is at much greater risk of regulation and control. It is not unlikely that some government may impose bans or require licenses to manufacture/own ASICs.

- Censorship resistance is hindered if mining farms that mine most of the blocks are pressured into confirming or censoring certain transactions. It would be difficult to exert this influence over a global collection of amateur miners, and much easier to impose this kind of activity on centralized mining corporations.

- The potential for an ASIC killswitch is a major concern, since a malicious manufacturer (or one that is following government orders) may include a way to remotely control or shut down their miners. This creates a single point of failure, whose activation would instantly kill most of the network hashrate. This would plunge the network into a sudden vulnerable state with dramatically lessened hashrate to secure the currency.

The ASIC takeover of Bitcoin is complete. While there are still some small-time miners with ASICS working in pools, large mining farms dominate the network hashrate. Concerningly, the majority of Bitcoin ASICs are designed, manufactured, and shipped by a single manufacturer - in stark contrast to the Bitcoin's early days, when miners used every brand, model, and flavor of CPU and GPU to mine.

Monero's CryptoNight PoW actively resists ASICs

Due to underlying egalitarian principles, the Monero community does not approve of ASICs and their inevitable centralization of mining power. While the "CPU-hard" hash algorithm (SHA-256) used by Bitcoin is amenable to ASIC optimization, Monero deters ASIC development by using a "memory-hard" algorithm (CryptoNight) that is difficult to accelerate. [link to: https://cryptonote.org/whitepaper.pdf] Consequently, CPU and GPU mining are both feasible for Monero, even in 2018. There are currently billions of existing devices (any modern x86 CPU and many GPUs) that are capable of mining Monero, so the process is accessible to any internet-connected individual. In fact, it is even possible to mine Monero in a web browser from any phone or computer!

In early March 2018, the Monero community was shocked to learn that CryptoNight ASICs (or similar devices known as field-programmable gate arrays) had been secretly produced and were mining Monero! These devices purported to mine Monero 25 times faster than the leading GPUs, and retrospective hashrate analysis suggests that they accounted for nearly half of the Monero network hashrate in late 2017 and early 2018.

Since the CryptoNote algorithm was designed as a memory-hard function specifically to "close the gap between CPU (majority) and GPU/FPGA/ASIC (minority) miners," the existence of these ASICs was an unexpected discovery. While the CryptoNote authors observe that, "It is appropriate that some users can have a certain advantage over others," they propose that "their investments should grow at least linearly with the power." Naturally, a newer computer or a nicer graphics card will mine more efficiently than older equipment, but ASICs create an extremely disproportionate distribution of hashrate.

The Monero community reacted quickly, proactively taking steps to mitigate ASIC mining before the existence of the devices was even fully confirmed. The spring 2018 Monero routine upgrade included a minor tweak to the CryptoNight algorithm, designed to affect ASICs differently than GPU/CPU miners. This slight variation did not change the difficulty or behavior of the algorithm, so the CPU/GPU miners were able to easily adjust to the new variant when they automatically upgraded with the network.

ASICs, on the other hand, are fundamentally incapable of adapting to new (minor or major) variations. One can think of ASICs as workers that are trained to do one task extremely quickly, but cannot learn to do anything else. The algorithm to be executed is physically etched into the ASIC circuits, so they cannot be reprogrammed or repurposed.

When the minor CryptoNight tweak was implemented at block 1546000, the ASICs became instantly incompatible with the network, and approximately half of the total hashrate vanished instantly. Since the ASICs were unable to adjust to process blocks by a modified algorithm, any blocks they produce are now immediately rejected by the Monero network as invalid.

For the time being, it appears that the Monero network has successfully removed the unexpected ASIC threat. The Monero development team has expressed intent to similarly trivially modify the mining algorithm at each network update. Since Monero carries out routine hard forks every 6 months, this should permanently disincentivize attempts to produce Monero ASICs, since each expensive and lengthy redesign would be promptly rendered obsolete.

A note on PoW alternatives

There are alternative systems for maintaining fairness besides proof of work; examples include proof of stake, proof of space, proof of bandwidth, and even hybrids between multiple types. Each system has its own strengths and drawbacks. PoW is currently the most widely used and field-tested consensus mechanism, and the only one system currently employed by Monero.

Miners are paid for their service

Two sources of income

Each time a miner successfully mines a block (i.e. is the first to find a nonce that completes the next block on the longest chain) they are paid two different ways.

First, the miner collects fees that were included with the transactions. Monero users can increase the likelihood that a miner includes their transaction sooner by increasing its associated fee.

Secondly, the miner receives a reward for contributing a solved block of validated transactions. This “block reward” is like the fixed-size commission the taxi network paid the drivers that submitted successful routes. All miners, upon receiving and confirming the solved block, add this freshly-minted “coinbase” to the address of the miner found the valid nonce first.

This is how all new Moneroj are created, as payment to the first network participant find a nonce for the next set of pending transactions.

All Moneroj originate as “block rewards” to miners

It is a common misconception that miners are finding or creating coins. Actually, miners are simply validating transactions, and are paid for their work with new coins. This introduction of new Moneroj is referred to as “coin emission.”

When Monero was launched, block reward started at more than 30 XMR per 2 minutes. This reward smoothly decreases until it will reach 0.6 XMR per 2-minute block in 2022. Monero’s continuous decrease is designed to provide a more stable environment for miners, compared to the dramatic “halving” events that some cryptocurrencies use to periodically sharply cut the block reward. After 2022, Monero’s “tail emission” will stay constant, guaranteeing that mining a block will always reward 0.6 XMR.

Many cryptocurrencies have a fixed cap on coin emission, with a hard limit on maximum supply. This means that at some point there will be no new coins introduced for miners, who will be forced to subsist entirely on fees. For example, in Bitcoin, this change will occur in 2040, when the supply reaches 21 million Bitcoin and ceases to increase any further. This approach is often touted as a benefit to remain “deflationary,” however these arguments are often based on conflating the concept of inflating monetary supply with a different use of the word “inflation” to describe an undesirable decrease in spending power of a currency. Monero implements the tail emission, with < 1% annual supply inflation, to ensure that future miners are always rewarded for their service and financially incentivised to use their mining power to secure the Monero network.

Cryptographic tools for proof of work

Throughout this chapter, we’ve focused on the functionality of hashes and nonces, describing them primarily through analogies. If you want to learn how they really work, the remainder of the chapter introduces the actual cryptographic principles.

Hashes (general concept)

Hash functions are a cryptographic tool that output a unique number for each input. These algorithms are designed so that any alteration to the input, even very minor changes, will result in an entirely different output. The term “hash” can be used to refer to both the function itself, and its output for a particular input.

Adding, removing, or changing even a single character will result in a totally different hash. Consider the message “Please send 50 euros to Jen.”. We can run the string through one of these algorithms to produce its hash “a2d2a9059ed8d323” The below table shows how the hash output changes dramatically with any modification of the input:

Input	Output	Comment
Please send 50 EUR to Jen.	a2d2a9059ed8d323	True message

Input	Output	Comment
Please send 500 EUR to Jen.	05cbdd8dd96718ac	Added an extra ‘0’ to amount
Please send 60 EUR to Jen.	f5087a90b63b1777	Changed ‘5’ to ‘6’
Please send 50 EUR to Jon.	ffd424b7077a3c58	Changed recipient to ‘Jon’
Please send 50 EUR to Jen.	a2d2a9059ed8d323	Same input = same output!

```
 The example output here is the first 16 characters of each input’s SHA-256 has
$ echo {input} | sha256sum | cut -c1-16
```

Hash functions are heavily utilized throughout cryptocurrency security features. Each block starts with the hash of the entire previous block. This is central to the immutability (tamper-proof characteristics) of the blockchain, since any attempts to change data in previous blocks will result in an entirely different hash output. Because the relationship between hash inputs and hash outputs is unpredictable and impossible to reverse-engineer, an attacker could not change the data in a block and create the exact same hash. Since each block includes a reference to the hash of the block before it, any modification attempt will be obvious on all subsequent blocks.

This idea of an expanding database with each group of entries cryptographically secured by hashes to the previous iteration is a key concept behind the blockchain revolution.

Nonces (general concept)

The term “nonce” refers to a valid to a puzzle that is not inherently physically/mathematically meaningful. For example, consider the following “fill-in-the-blank” questions that a teacher might give their students:

Puzzle A) The Esperanto word for "c___" inspired the name “Monero” Acceptable answer: “coin” Puzzle B) 1 kilogram is equal to "***" grams. Acceptable answer: 1000 Puzzle C) This 3-digit prime number "3*** **" does not repeat any digits Acceptable nonces: any of {307, 317, 347, 349, 359, etc…} Puzzle D) This 5-digit prime number "7**_" does not repeat any digits Acceptable nonces: any of {71263, 72169, 73609, 74869, etc…}

Puzzle A and B are both meaningful, and each have one correct answer (A: “coin” B: “1000”) that the student will want to remember for future problems. Thus, these answers are not considered “nonces”

However, puzzles C and D are both “busy work” tasks that are hard to solve, and do not contribute insight to any real problem. There are multiple solutions that all satisfy the nonce requirement; for question C, the answer “359” is equally as valid as “307.”

A student that spends an hour testing various numbers to come up with the answer “359” for puzzle C has to start the search for a valid nonce from scratch upon encountering each variation on the puzzle, e.g. “This 3-digit prime number “6___” does not repeat any digits.”

Hashes and Nonces in PoW

Cryptocurrencies use the hash of each block to be sure that its contents have not been changed, since modifying a single character would be clearly reflected in a radically-different hash (which propagates through subsequent blocks). The hash of a block includes all of its contents: transactions, headers, the hash of the previous block, and a nonce field.

To work on solving a block, the miners must randomly guess at values for the nonce, attempting to find one that causes the hash of the entire block to produce an output below a certain threshold, which is determined by the current network difficulty. Since it is impossible to predict how changing the inputs to a hash function will affect its outputs, the miners simply have to brute force random nonces over and over again until they find one that produces an output hash below the threshold.

The network changes raise or lower this threshold to adjust the mining difficulty, in order to maintain a 2-minute block time independent of changes in the total hashrate.

Chapter 5

Community and Contributing

Principles of openness

While the Monero cryptocurrency itself epitomizes privacy, its community is built on transparency and collaboration! Users, developers, and researchers communicate on IRC channels open to the public, and key meetings are archived for public view on the official website.

This culture of cooperation and openness is a natural consequence of Monero’s origin as a fork from ByteCoin. The developer of ByteCoin operated with unilateral secrecy, making designs and decisions without community feedback. The resulting development mistakes, especially the egregious premine, ruined the viability of the coin.

The Monero community came into existence by forking the shadowy ByteCoin development into the light of a decentralized, collaborative, and diverse community. This has undoubtedly strengthened the project on many fronts, and the Monero community has learned to thrive on cooperation. Cryptography may be the underpinning of Monero, but the community is its real source of power!

Many great minds work on Monero

The Monero project is a massive community effort, assembled by hundreds of people. At the time of writing, more than 500 individuals have contributed code, including 200 in the last year. Monero has adopted an “un-governance” scheme for organizing growth and development. The project is comprised of several different “branches” working together: the community, the Monero Core Team, the Monero Research Lab, and Monero Workgroups.

The Monero Research Lab conducts cutting-edge basic and applied research on cryptocurrency technologies and analyses. MRL includes many academics and researchers, and studies are published openly at https://lab.getmonero.org/

The Monero Core Team manages many of the critical tasks for Monero. The Core team is tasked to:

- Act as primary trusted arbiters of the Forum Funding System on behalf of the community, so as to ensure the completion of all projects to the satisfaction of the community.
- Manage the codebase of the Monero Project, which includes merging code, keeping backups, and ensuring the safety, security, and free access of the code from any party.
- Steward the general donation fund, and spending the Monero there on anything they see fit to further the Monero Project.
- Act as trusted signers and distributors of reference clients for the Monero coin, and other related technologies. Set a direction and vision for the Monero Project, which is deliberated in the community.

The Monero Workgroups are collaborations formed to join people around unique goals. This allows small teams of individuals to connect and tackle specific tasks. For example, the Monero Hardware Workgroup is well underway on its mission to build the first open-source community-driven hardware wallet. Another instance was the Monero Integrations Workgroup that developed free open-source payment gateways. You can join a workgroup to help with translating Monero, crafting kits for Meetups or helping users with software issues.

The community is what makes Monero possible! Anybody is welcome to contribute code, propose projects, fund proposals, help with outreach, or write books about Monero.

Contributing to the Monero Codebase

Make a pull request for the improvements

Anyone is welcome to contribute to Monero’s codebase! If you have a fix or code change, feel free to submit it as a pull request directly to the “master” branch. In cases where the change is relatively small or does not affect other parts of the codebase it may be merged in immediately by any one of the collaborators. On the other hand, if the change is particularly large or complex, it is expected that it will be discussed at length. To do this:

1. Fork the repository
2. Clone the repo to your machine
3. Make a branch, make necessary changes, and commit the files with a clear commit message
4. Push your changes to your fork by running the command git push origin branch-name
5. Go to your fork to see a Compare and pull request button
6. Add necessary details and documentation

When submitting a pull request, make sure your branch is rebased. No merge commits nor stray commits from other people in your submitted branch, please. You may be asked to rebase if there are conflicts (even trivially resolvable ones).

Patch etiquette

Patches are preferably to be sent via a pull request. If that can't be done, patches in "git format-patch" format can be sent (eg, posted to fpaste.org with a long enough timeout and a link posted to #monero-dev on irc.freenode.net).

Patches should be self contained. A good rule of thumb is to create one patch per separate issue, feature, or logical change. Also, do not make other changes, such as random whitespace changes or reindentation. Please follow the code style of the particular chunk of code that you're modifying. Proper squashing should be done (e.g. if you're code a buggy patch followed by a subsequent patch to fix that bug, then both patches should be merged).

General guidelines

Commit messages should be sensible. That means a subject line that describes the patch, with an optional longer body that gives details, documentation, etc. Well-commented code is strongly encouraged, to help others interpret and constructively interact with your code. If you're adding some new functionality, testing results are helpful to include with your pull request.

If you've made random unrelated changes (including those due to an overzealous editor), you can select which changes go into the coming commit using git add -p, steps through each of the changes to verify which should be included. This helps create clean patches without any irrelevant changes. git diff displays changes in your tree, and git diff --cached will show the changes that are currently staged for commit. Hunks that are added with git add -p, will "move" from the git diff output to the git diff --cached output, so you can see clearly what your commit is going to look like.

More specific guidelines regarding common processes are described here: https://github.com/monero-project/monero/blob/master/CONTRIBUTING.md

Introduction to Monero Core code

Many different repositories are hosted at the Monero Project GitHub [link: https://github.com/monero-project/]. Several of them house components that we've already discussed in Mastering Monero, for example:

- monero: the core of Monero network, written in C++ language
- monero-GUI: Graphical User Interface for Monero, built with Qt library
- monero-CLI: the CLI for Monero (included in the Monero-core repository)
- monero-site: the website for Monero: https://getmonero.org
- sekura: the community hardware wallet
- kovri: The Kovri anonymizing router

These projects are well-documented, so that you can become familiar with the code and make improvements! You can see that there are lots of different sub-projects with opportunities for you to contribute to Monero. Please visit one of the repositories, read through some of the open issues, and consider how you can leave your legacy in the Monero codebase.

Introduction to Monero Development

Building the Monero code is a complex process, so some tips and summaries are included here. Linux systems have a built-in shell that helps with building the Monero core, so consider switching to a *nix-based operating system, if possible. Monero is written in C++, with C++11-style referenciation.

Downloading the Monero Source Code

Monero uses Git as the version control system. This allows developers to track changes and modifications to their code, and coordinate work on shared files. To download the Monero code, simply execute

git clone https://github.com/monero-project/monero

Dependencies

To build Monero software from the source code, your path will need to include the dependencies in the table below. A few of the libraries are also included in this repository (marked as "Vendored"). By default, the build uses the library installed on the system, and ignores the vendored sources. However, if no library is found installed on the system, then the vendored source will be built and used. The vendored sources are also used for statically-linked builds because distribution packages often include only shared library binaries (.so) but not static library archives (.a).

Building Instruction

Monero uses the CMake build system and a top-level makefile that invokes cmake commands, as needed. Once you've installed the dependences, change to the root of the source code directory and execute the make command to begin the build. The process may take up to an hour or two. Once the code has finished building, you can find the Monero binaries in the /build/release/bin folder.

Build Troubleshooting

If you run into errors, the output will likely indicate exactly what went wrong. A few of the common bugs to troubleshoot include:

- An outdated boost version (you may have to manually install the current one)
- Outdated gcc/g++
- Missing libzmq3-dev
- Missing libreadline-dev
- OpenGL errors

You can (optionally) type make debug to compile a debugging build. There are many communities with information to help you with troubleshooting. Search engine queries with your build errors are likely to connect you with a solution or people that can help.

Top-level folders

- cmake: contains compiling rules for the cmake build system.
- contrib
- external: external libraries from open source projects
- include: internal libraries
- src: the main source code for Monero
- translations: folder for translations
- tests: unit tests for Monero
- utils: utilities for Monero code (gpg keys, scripts for build translations)

Building Monero Graphical User Interface

The Monero graphical user interface (GUI) is built with C++ and Qt libraries. Both are necessary to successfully build the GUI. With the dependencies in place, you can clone and build the GUI with the commands:

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

Chapter 6

A Deep Dive into Monero & Cryptography

Since long before the birth of computers, mathematics and cryptography have been at the center of communication and information exchange. While simple ciphers have been around since Caesar's time [link to Caesar cipher], modern cryptography was born during the World Wars for encrypting important and confidential messages. Initially, governments and militaries funded classified cryptography research to identify protocols for protecting state secrets.

Now, cryptography is no longer limited to spies and militaries; it forms the backbone of communication and security in the internet era, and is widely studied by academic and industry researchers scattered throughout the globe.

Today, cryptography is a ubiquitous behind-the-scenes tool that enables security, management, communication, and many of the connections that improve our day-to-day lives. For example, consider the invention of Secure Socket Layer (SSL, deprecated in favor of TSL), which is based on cryptographically signing the content. Hospitals, banks, governments, and businesses all protect your data with cryptography.

This chapter discusses how cryptographic tools can be applied to a decentralized financial database to give rise to cryptocurrencies, especially Monero.

Math Fundamentals

Here is a brief introduction/recap of several mathematical principles that are at the core of cryptography.

Euclidean Division (A/B)

Dividing any number "A" by another "B", written as A/B or A÷B returns an answer that can either be written as a quotient with a remainder, or as a decimal alone.

Generally: A/B = q with remainder r

For example: 12/4 = 3 with remainder 0, which can be written 3.0 in decimal form 13/4 = 3 with remainder 1, which can be written 3.25 in decimal form 27/5 = 5 with remainder 2, which can be written 5.4 in decimal form

Prime Numbers

A prime number is any integer (whole number) that is not divisible by any integer besides ‘1’ and itself. For example, 20 is not prime because it could be divided by 2, 4, 5, or 10, resulting in whole numbers e.g. 20 ÷ 4 = 5 - or - 20 ÷ 10 = 2 7 is prime because any integer that you divide it by will not yield a whole number e.g. 7 ÷ 3 = 2.3333 Some example prime numbers include 3, 5, 7, 11, 13, 97, 223, 997, 3413, 4421, 17837, 145601, 428567, 1171967, and even much larger numbers like 207472224677348520782169522210760858748099647472111729275299258991219668475054965831008441673255007 or the twin primes 2,996,863,034,895 × 2^1,290,000 ± 1, which have over 350,000 digits each!

Modular arithmetic (A mod B)

Modular arithmetic describes numbers that wrap around a particular integer. An intuitive example is the 12-hour clock. If you stay up for 5 hours past 11:00 PM, you would not encounter 16:00 PM o’clock! Instead, at midnight, the time wraps around to zero (so 5 hours past 11:00 PM is 4:00 AM the next day).

Given any two positive numbers, A (the dividend) and B (the divisor), A modulo B = the remainder r from A/B.

In the context of clocks, staying up 5 hours past 11:00 PM could be represented as:

(11:00 PM + 5 hours) mod 12 = … = 16:00 mod 12 = 4:00 (AM)

Elliptic Curve (Ed25519)

Elliptic curves are defined as the set 2-dimensional (x, y) points that satisfy an equation:

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

For example, with fixed coefficients a = 2 and b = 4, this equation becomes

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

which is satisfied by many pairs of points such as:

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

Monero uses a particular Twisted Edwards elliptic curve for cryptographic operations, Ed25519, which is birational equivalent of the Montgomery curve Curve25519.

The ed25519 curve can be expressed algebraically as

$-x^2 + y^2 = 1 – (121665/121666) x^2 y^2$

Thinking back to our general elliptic curve equation, this Twisted Edwards is a special case using the parameters:

a = -1 and b = 121665/121666.

It is well known that NIST4 standard algorithms have had recent issues, [LINK] and the Twisted Edwards curve was selected to address many concerns held by the cryptography community. [LINK]

Recently, it has become clear that a NIST-backed PNRG random number generation algorithm is flawed, and contains a potential backdoor. [LINK] Seen from a broader perspective, curves selected by the NIST are also implicitly supported by the NSA. These endorsements are viewed suspiciously by the cryptography and cryptocurrency communities due to previous incidents where the NSA used their authority over NIST to weaken algorithms suggested by the latter.

Twisted Edwards curve Ed25519 is not subject to any patents, and the team behind it has developed and adapted basic cryptographic algorithms with efficiency in mind. This curve is currently believed to be secure.

Cryptography Basics

Monero is the leading secure, private, and untraceable cryptocurrency because of its unique privacy-oriented cryptographic features, which we'll explore more thoroughly in this chapter This is one of the more technical chapters of the book, due to the mathematical nature of cryptography. More complex techniques are built upon simple principles known as "cryptographic primitives"

A cryptographic primitive is an algorithm that serves as the building block for cryptographic protocols. Monero employs a wide variety of cryptographic primitives for various uses, some of which we covered conceptually in chapters 3 and 4. Monero's intentional approaches to privacy and (ASIC-resistant) proof of work necessitate development with more sophisticated cryptographic tools than those used by many other cryptocurrencies, such as Bitcoin.

Hashing

Chapter 4 discussed the concept of "hashing" and how its uses range from confirming data fidelity to distributing rewards in Proof of Work. Example hashes are shown in the cryptography section toward the end of Chapter 4.

Selecting a good hashing algorithm is crucial for creating generating addresses and keys in a secure way. If two different inputs produce the same hash output, this is known as a "collision." Hashes are commonly used as an identifier in blockchain systems, relying on their effective uniqueness. Furthermore, a collision during address generation would lead to multiple individuals with the same keys and seeds; obviously this would be extremely problematic!

Monero uses the CryptoNight PoW system, which uses a special “CryptoNote” hash algorithm (unlike most cryptocurrencies that are built on standard SHA-256), which is built in the “Keccak” hash function. The Keccak algorithm won a NIST competition to be designated SHA3, and is designed by non-NSA engineers. Monero uses the “Keccak-256” hashing function with 32-byte output for both transaction and block hashing.

Monero PseudoRandom Number Generation (PRNG)

When users and computers are creating new keys, it is crucial that they find new keys that others cannot guess. This is actually a very difficult task, since most software is designed to favor reproducibility. If the computer generates randomness in a predictable way, then the output is be ostensibly random but somewhat easier to guess.

- For example consider a PRNG that simply shuffles the digits of the current time to make a 4-digit key. So at “10:34” it might output “0413” or “1403” or “0134” … If you wanted to keep the output key secret, this would be a terrible method for a few reasons:

- An attacker who knows that you made your key when you got to work around 9:30 AM would know that the digits “0” and “9” appear, which narrows the choices down to significantly fewer options.

- There are no HH:MM times of day with three “9”s. In fact, there are no times with any three digits chosen from {6,7,8,9} since 17:89 h, 18:78 h, etc are impossible times. This rule eliminates many 4-digit pins, leaving the attacker to guess from a much smaller pool

The above clock-based random number generator is awful because using the time of day as an “initial seed” is predictable. The initial seed should be much more difficult for an attacker to guess. Good random number generators introduce lots of “entropy” to make their outputs unpredictable. Simply shuffling 4 digits does not introduce much entropy, another reason that our PRNG above would be insecure.

When generating wallets, the user’s operating system provides the initial seed / entropy source. Monero then repeatedly applies the Keccak hashing function, to lead to an unpredictable and non-reproducible output. Each round of hashing produces an output that is used as the input for the next hash.

Encoding

Base58 is a group of binary-to-text encoding schemes used to represent large integers as alphanumeric text. It is similar to another scheme called Base64, however it has been modified to avoid numbers and letters which might look ambiguous when printed. Monero uses this format, strictly for convenience of human users, who often must manually read or transcribe long addresses.

Monero’s Base58 Alphabet: 123456789ABCDEFGHJKLMNPQRSTUVWXYZ abcdefghijkmnopqrstuvwxyz Note: Zero (0) along with the letters I (uppercase i), O (uppercase o), and l (lowercase L) are not present in this Base58 alphabet due to their ambiguity with each other.

Symmetric and asymmetric cryptography

For encrypting data, algorithms can be characterized as “symmetric” or “asymmetric” depending on what type of keys are used.

Symmetric encryption requires the participants to share a secret, for example you encrypt a message by the password “hunter2” and the recipient uses the password “hunter2” to decrypt it. To communicate in this way, both parties must have agreed on the shared (symmetric) secret ahead of time. This practical issue limits the utility of symmetric encryption for many applications.

Asymmetric encryption allows two parties to interact securely without sharing a particular secret. This type of cryptography is woven into the framework of internet security, end-to-end messengers, and cryptocurrencies.

Basics of Elliptic Curve Cryptography

Elliptic curve point addition and scalar multiplication are used at the very basis of asymmetric elliptic curve cryptography schemes so it's important to have a basic understanding of these concepts to understand the cryptography used in Monero.

Elliptic curve point addition

Elliptic curve point addition is different from the typical addition used in every day arithmatic. To add two points together on an elliptic curve you must find the line between those two points and then find the point at which the curve intersects with that line. That point is then reflected over the x-axis to arrive at the final point. When adding a point to itself, known as point doubling, you must find the tangent line to the starting point to get to the point at which that tangent line intersects with the curve. That point is then reflected over the x-axis to arrive at the final point.

Scalar multiplication

Scalar multiplication utilizes both a point on the curve and an integer. To multiply a point, P, by an integer, S, the point is added to itself S times. In asymmetric crypto schemes a common base point on the elliptic curve is used as a generator point to generate public keys from private keys. When the curve generator point is added to itself many times it becomes very difficult to figure out how many times the generator point was added to itself by only looking at the resulting point. This problem is often referred to as the elliptic curve discrete logarithm problem. Because this problem is so difficult to solve, this kind of scalar multiplication is considered a one way function.

Wallet generation

Seed Generation and Encoding

In chapter 2 we talked about how your wallet generates a secret “seed” that is used to derive all of your keys, and access/spend your funds. In that overview, we simply considered the 25-word “seed mnemonic.”

Behind the scenes, a seed is a unique 256-bit integer from which keys and addresses are derived, for example:

```
112699108505435943726051051450940377552177626778909564691673845134467691053980
```

These are often represented as a 64-digit base16 number, for example:

```
f9296f587419f1cdede67de160fca14d1069ecaa4c52f012af031eeA09ee039c
```

(For mnemonic-style keys, this representation of the seed is actually just the private spend key itself!)

Writing down either of the above key styles would be quite difficult, and most people would be prone to make at least one mistake. Conversion to a seed mnemonic phrase is another step included only for human interpretability and usability. The mnemonic phrase essentially converts the above 256-bit number into to a “24-digit” (24-word) Base1626 “number” (since there are 1626 words in the seed dictionary). This representation of the long seed strings is much easier to read:

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

When your wallet presents the 24-word seed, it adds a 25th word that functions as a “checksum.” This allows later detection of typos or mistakes. Monero’s mnemonic method encodes with a minimum 4:3 ratio. In other words, four bytes creates three words, plus one checksum word; eight bytes creates six words, plus one checksum word; and so on.

The private view key is derived by hashing the seed with Keccak-256, producing a second 256-bit integer, which is then sent to the function called “sc_reduce32” to ensure that it is compatible with the elliptic curve. The seeds created by this method will always be valid scalars as they are sent to sc_reduce32 first.

Asymmetric cryptography in cryptocurrencies

Bitcoin uses an asymmetric encryption with two keys:

- private key - for signing transactions and for decrypting data)
- public key - for signature verification and encrypting data

Monero's more complex cryptographic framework requires four keys:

- public view key - used to verify the validity of addresses,
- private view key - used for viewing data such as the balance, fees and transactions amounts. The view key is never used for creating or signing transactions.
- public spend key - another public key for transaction verification.
- private spend key - used for signing transactions, i.e. spending Moneroj.

Your public Monero address is a direct representation of the pair of public keys, whereas Bitcoin (and clones) use a hash of their single public key. EdDSA keys (both private and public) are 256 bits long, or 64 hexadecimal characters. Not every 256-bit integer is a valid EdDSA scalar (private key); it must be less than the "curve order" described with the equation in the Ed25519 function section.

Monero Public Address

Monero standard address is composed of the two public keys (the public spend key + public view key). It also contains a checksum and a "network byte" which identifies both the network and the address type. Their concatenation follows these rules:

Monero standard address is composed of two public keys:

- public spend key
- public view key

It also contains a checksum and a "network byte" which actually identifies both the network and the address type.

Index	Size in bytes	Description
0	1	identifies the network and address type; 18 is for main net and it refers to "4"; 53 is for test chain corresponded to "9".
1	32	public spend key
33	32	public view key
65	4	checksum (hash created with Keccak function of the previous 65 bytes, trimmed to first 4 bytes)

The 69-byte output from this specification is then encoded into the Monero Base58 format. This conversion increases the length to a 95-character string that is easy to read and write. Example standard address:

4AdUndXHHZ6cfufTMvppY6JwXNouMBzSkbLYfpAV5Usx3skxNgYeYTRj5UzqtReoS44qo9mtmXCqY45DJ852K5Jv2684Rge

The pseudo-code below defines the process of generating an address:

```
Checksum = H(Varint(Prefix) || A || B)
SerializedString = Base58(Prefix || A || B || Checksum)
```

Keys and Address Derivation

To add to the confusion, there are presently at least three different methods of private key derivation in existence for Monero, though Bitcoin also has many:

- **Original (non-deterministic) Style** – The Private Spend Key and Private View Key are both independently and randomly chosen to form an account. There is no good way to back up a nondeterministic account other than keeping copies of the files. For these reasons, it is not recommended to use an account of this type.

- **Mnemonic (Electrum or Deterministic) Style** – In this style, the Private View Key is derived from the Private Spend Key, so you only need to remember one thing: the seed, which is actually just a representation of the Private Spend Key itself. The Private View Key is derived by hashing the Private Spend Key with Keccak-256, producing a second 256-bit integer, which is then sent to a function that check if the private key is a valid EdDSA scalar. You can backup accounts of this type by writing down or otherwise saving the 25 word deterministic seed; you can easily restore using both Simplewallet and MyMonero.

- **MyMonero Style** – This is similar to 2., but uses a 13 word seed instead of a 25 word seed. The 13 words convert to a 128-bit integer that is used for both spend and view key derivation, in the following form: the 128-bit integer is hashed with Keccak-256 to produce a 256-bit integer, a. a is sent to a function, which returns the Private Spend Key. a is hashed once more with Keccak-256 to produce a second 256-bit integer, b. b is then sent to the same function, which returns the Private View Key based on b.

You may have noticed a critical difference between this style and the Electrum Style: MyMonero's Private View Key derivation is done by hashing random integer a, while Electrum Style derivation is done by hashing the Private Spend Key. This means that 13 and 25 word seeds are not compatible – it is not possible to create an Electrum Style seed (and account) that matches a MyMonero Style seed (and account) or vice versa; the view keypair will always be different. The rest of the address generation process is the same in all three cases. The Private Spend Key and Private View Key are sent to the ed25519 scalarmult function to create their counterparts, the Public Spend Key and Public View Key.

The Monero Blockchain

Monero has a unique kind of blockchain. We talked about what a blockchain is and why it is important. Basically, a blockchain is a distributed public ledger where each payment is recorded. The blockchain cannot be modified due to its distributed nature. It is based on various cryptography protocols and algorithms in order to avoid any cheating.

Lightning Memory Mapped Database

Monero uses the Lightning Memory Mapped Database (LMDB) system to store its blockchain. LMDB is a software library that provides a high-performance embedded transactional database in the form of a key-value store. This means that it is highly effective, and easy to search.

LMDB is written in C++ with API bindings for several programming languages and is developed by Symas Corporation. Here are a few LMDB features:

- stores arbitrary key/data pairs as byte arrays, meaning
- has a range-based search capability, allowing fast search (allowing...)
- supports multiple data items for a single key, providing (providing....)
- has a special mode for appending records at the end of the database which
- gives a dramatic write performance increase over other similar stores.

The block

Block structure

The CryptoNote standards define the way to store and delineate data within blocks and on the blockchain. The block structure contains 3 main components:

- block header;
- base transaction body
- list of transaction identifiers.

The list starts with the number of transaction identifiers that it contains.

Block Header

Each block starts with header containing key metadata. The “major_version” defines the block header parsing rules, so it can be interpreted correctly. The table below describes version 1 of the block header format. The “minor_version” defines the interpretation details that are not related to the main header parsing.

Even if the minor version is unknown, it is always safe to parse the block header of a particular major version. Parsing the block header with an unknown major version is risky, since the content of the block header may be misinterpreted.

Field	Type	Content
major_version	varint	Major block header version (always 1)
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

Varint means an integer encoded in a variable-length prefix-free representation, such encoding does not contain null bytes.

Base Transaction

Each valid block contains a single “base transaction” that awards the miner with the coinbase block reward. The base transaction must follow the coin emission rules, and include the block height field.

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	Array of outputs

List of Transaction Identifiers

Base transaction is followed by a list of transaction identifiers. A transaction identifier is a transaction body hashed with the Keccak hash function. The list starts with the number of identifiers and is followed by the identifiers themselves if it is not empty.

Calculation of Block Identifier

The identifier of a block is the result of hashing the following data with Keccak-256:

- size of block_header, Merkle root hash, and the number of transactions in bytes (varint)
- block_header,
- Merkle root hash,
- number of transactions (varint).

The goal of the Merkle root hash is to “attach” the transactions referred to in the list to the block header: once the Merkle root hash is fixed, the transactions cannot be modified. This is the reason for which you can’t modify or cheat the blockchain.

Calculating Merkle Root Hashing

Fees

As we introduced fees in the third chapter, it’s important to know how they are calculated and how Monero developers might reduce them with “Bulletproofs”.

Fees are like Taxes for the network. They incentive miners that will mine a lot of transactions then they will receive a reward which was composed by all the fees contained in the transaction.

Before going deep on Fees, we have to discuss about the Block Dynamic Size for Monero and the Block Static Size for Bitcoin. With Bitcoin, the block size was setup on 1 mB. In this moment, Bitcoin Blockchain has a scalability issue since each block can contain only a defined number of transactions.

Then, the block size limit has created a bottleneck in bitcoin, resulting in increasing transaction fees and delayed processing of transactions that cannot be fit into a block. Various proposals have come forth on how to scale bitcoin, and a contentious debate has resulted.

Instead of the Bitcoin method, Monero uses dynamic block size mechanism to control the rate at which the block size can grow. This is called the Penalty Function for oversize blocks, originally purposed by CryptoNote Developers. As part of consensus rules, part of the base block reward is withheld should a miner expand the block size above the median size of the last 100 blocks. This is coupled with a do-not-relay minimum fee.

The fees sent to a miner in a transaction are defined as:

Miner Fees = BaseReward - Penalty

Penalty is being calculated as

Penalty = BaseReward * ((BlockSize / MN) - 1)^2

Where :

- MN is the median of the block size over the last N blocks, with N being 100 in Monero

- BlockSize is the size of the current block
- BaseReward is the reward as per the emission curve or where applicable the tail emission

In this way, the maximum allowed block size is 2 MN which it should stop any spamming attacks to the Monero Blockchain.

Note that the formula of the BaseReward is defined as follows:

BaseReward = 2 * ((S - A) * 2-20 * 10-12)

Where:

- 2 is the adjustment factor for the switch to two minute blocks
- S is the initial number of atomic units is = 264 - 1
- A is the current circulation, which can be found here. In addition, the current circulation (emission) displayed on the block explorer has to be multiplied with 1012 (Monero uses 12 decimal places) to convert it to atomic units.

Observe that the minimum block size limit is 300 kB. Thus, miners are able to construct blocks up to 300 kB without incurring a penalty. In other words, aforementioned penalty function only “kicks in” for blocks bigger than 300 kB.

But let’s imagine a different scenario. What will happen if the median block size (retrieved by the last 100 blocks for Monero cryptocurrency) significantly diverges from the minimum block size?

Well, dynamic fee algorithm comes to play.

The dynamic fee algorithm is calculated by the weight in kB (Kilobyte) of transaction, meaning heavier a transaction is , higher the fee will be. Regarding the Monero situation, we have a low fee per kB. However, due to the high transaction size, the absolute default fee (in economical terms) is quite high.

Note that the transaction size is this big due to Monero’s inherent default privacy, i.e., the range proofs, which mask the amount values, make up ~12 kB of a single transaction. As I like telling to anyone, privacy costs.

Unfortunately that is not the only way to increase fees. Transaction priority also increases fees. A higher priority will increase the fee so that it successfully competes with other transactions to become part of the next block on the blockchain.

It is defined as:

Fee per kB = (R/R0) * (M0/M) * F0 * (60/300) * 4

- R is the base reward
- R0 is the reference base reward (10 XMR)

- M is the block size limit
- M0 is the minimum block size limit (300 kB)
- F0 is 0.002 XMR
- 60/300 is the adjustment factor to account for the increase of the minimum block size limit (60 kB -> 300 kB)
- 4 is the adjustment factor to account for the default fee multiplier. That is, the lowest fee level uses a multiplier of 1, whereas the default fee level uses a multiplier of 4

Basically the inverse of the percentage increase of the median block size (against a base of the minimum block size) translates to the percentage reduction in fees. More specifically, a 600 kB median block size, which is a 100% (or factor 2) increase translates to a 50% (1/2) reduction in fees.

So why did the significant price increase not lead to a significant reduction in absolute fees, i.e., fees in XMR terms? Well, basically, the factor increase in price was significantly higher than the factor increase in usage. Furthermore, the median block size needs to be constantly above 300 kB in order for the dynamic fee algorithm to work properly. Moreover, the algorithm was designed to correlate with price, but, as we can see, price is imperfectly correlated with usage. In sum, whilst usage has grown a lot, it hasn't grown as much as the price and therefore fees (in XMR terms) have not declined yet.

Bulletproofs

As some users reported, Monero fees are high. This is partially wrong since a comparison between others cryptocurrencies and Monero showed Monero has one of the lowest fee per kB.

Originally the problem is our transactions are heavy for the blockchain. Let's talk about the reasons.

Monero needs to have some "tests" in order to prevent any abuse. An example of preventing abuse could be proving the transaction fee is correct, proving the amount is right and no one is trying to commit a double spend.

In the development field, there is a situation called Overflow meaning a variable reaches the max values. Anyone knows electronic devices have a limited value for anything since "infinite" is an abstract of our world.

Privacy Transactions

Stealth Addresses

Chapter 3 described a situation where Leo sent George some Monero and in doing so he used George's public keys to produce a one-time public key, also known as a stealth address, that is unlinkable to the George's real keys. This section will go deeper to explain the cryptography behind that one-time public key.

Sending

The highly technical formula described in the CryptoNote whitepaper to produce this public output is `P = Hs(rA|i)G + B` . This means that when Leo wants to send Monero to George he generates a 256 bit pseudorandom scalar to be used as the transaction private key, r. Leo is the only person that will ever know this key, not even George. Leo then multiplies George's public view key, A, by his pseudorandom scalar and then concatenates the output index, i, to resulting point.

This data is then run through the `Hash to Scalar` function. This function takes the input data, hashes it using the Keccak-256 algorithm, then takes that resulting hash modulo 2^{252} + 27742317777372353535851937790883648493, a prime order of the ed25519 basepoint. The ed25519 basepoint, G, is then multiplied by the scalar that is output from that function. Finally, Leo adds this point with George's public spend key, B, to produce the final output, P.

Receiving

Now, as described in chapter 3, George must scan the blockchain for outputs that belong to him. To do this he must calculate `P' = Hs(aR|i)G + B` . The process is very similar to what Leo had to do to send the Monero. George will get the public transaction key, R, used in the transaction from the blockchain and multiply it by his private view key, a. He then must concatenate the output index to the resulting point and that data through the `Hash to Scalar` function. He then multiplies the ed25519 basepoint, G, by the resulting scalar and finally adds his own public spend key, B, to the resulting point to produce the final point P'. If the output George generated independently, P', matches the output from the blockchain, P, then George knows that he owns that output and can spend the associated Monero.

Ring Signatures

Subaddresses

One way to create off-chain likability is through address reuse. If George uses a Monero address as a deposit address when he withdraws money from an exchange and then uses the same address to receive a payment from Leo, it is possible for the exchange to collude with Leo to find out that the George who used the exchange and the George that accepted Leo's payment are in fact the same person. To prevent this type of attack, Monero has a feature called subaddresses. A subaddress is an address that is derived from a main address and is cryptographically unlinkable from that main address.

Creating subaddresses

So how are subaddresses actually made? Suppose that George has a pair of private keys, the private view key and the private spend key, (a,b) and a corrosponding pair of public keys (A,B) which is equal to (a,b)G where G is the generator point on the elliptic curve. First we must chose a number, `i` , called the index. Typically the first subaddress will be at index 1. The subaddress will also have a pair of private keys and a pair of public keys for each index of `i` , which we can then call (c_i,d_i) and (C_i,D_i). The formula to create a subaddress public spend key, D_i, is `Hs(a|i)G+B` . This means that first the index, `i` , is concatenated to the main address private view key, `a` , and then passed through the `hash_to_scalar` function (note: in practice the reference client wallet also concatenates the string "SubAddr" to the data to be hashed as a common salt). Then the curve generator point, `G` , is multiplied by the resulting scalar and finally the main address public spend key is added to the resulting point through elliptic curve point addition. After this subaddress public spend key is generated, it is recorded for use later. To calculate the subaddress public view key, C_i, D_i is multiplied by the main address private spend key ($C_i = a*D_i$). Now that the subaddress public keys have been generated they can be encoded into a valid Monero address. Subaddresses are base58 encoded with a checksum just like normal addresses but the network byte is different. Mainnet subaddresses use the network byte 0x42 which makes every mainnet subaddress start with `8` .

Sending to a subaddress

When a wallet sees a Monero address with a subaddress network byte it knows that it must construct the transaction slightly differently. Typically with a non-subaddress transaction the transaction public transaction key is calculated by generating 32 random bytes as the private key and then multiplying that private key by the elliptic curve generator point through elliptic curve scalar multiplication. For a subaddress transaction however, the private transaction key is instead multiplied by the receiving subaddress's public spend key. The sender then has the choice to either send the change to his main address or one of his own subaddresses, though the result is typically the same for most users.

Receiving a transaction to a subaddress

As explain in the stealth address section, Monero users must scan incoming transactions with their private view key to check if they own a specific output on the blockchain. Subaddresses don't use the typical system where the receiver calculates `P'= Hs(aR)G+B` then if P' is equal to the output, P on the blockchain then he knows that he owns that output. Instead the formula for subaddresses is expressed as `Di' = P - Hs(aR)G` . This means that the receiver calculates Hs(aR)G as usual (see the stealth addresses section of chapter 6 to learn what this means) and then subtracts the resulting scalar from the output that is found on the blockchain, P. If the result is equal to a subaddress public spend key that was recorded when creating that subaddress, then the receiver is assured that he owns that output.

Chapter 8

Integrating Monero

So far, this book has focused on getting started with Monero for the end user. In this chapter, we instead include information about how you can implement Monero in your services and systems! Merchants that accept Monero can expect fast, private, irreversible transactions. There are several systems for integrating Monero into your business.

For example, the Monero Integrations plugin (else created by SerHack) allows any online shop to include a Monero payment option. Setting it up is as easy as installing one of the plugins for many popular Content Management Systems. In the spirit of the currency that it is named after, the Monero Integrations project is entirely decentralized. There is no third party to trust or register with, and Monero payments from customers are routed directly to your wallet.

Kasisto was the first a Monero Point of Sale payment system, and was designed for accepting Monero in-store, on a phone or tablet. Another option could be Globee, developed by Riccardo Spagni, which is a convenient yet proprietary, centralized service.

Any implementation with features such as checking wallet balances, asking for integrated addresses or a subaddress will require a direct connection with a wallet. The Monero Wallet Remote Protocol Communication (RPC, called monero-wallet-rpc) allows you to manage the wallet with JSON calls.

Using Monero Wallet Remote Protocol Communication

Setting up the Monero Wallet RPC

First, you can initialize the Monero Wallet RPC: ./monero-wallet-rpc --rpc-bind-port 18082 --disable-rpc-login --log-level 2 --wallet-file your-wallet-file --prompt-for-password If you wish to use a remote node you can add the --daemon-address flag followed by the address of the node, for example: --daemon-address node.moneroworld.com:18089 Remember that monero-wallet-rpc doesn't bind your IP address and the port by default, so you must specify --rpc-bind-ip yourip to connect remotely.

JSON RPC Protocol

JSON-RPC is a stateless, lightweight RPC protocol, using the JSON RFC 4672 data format. The specification primarily defines several data structures, and the rules for processing them. The protocol is "transport-agnostic," meaning that its function is independent of the underlying transport mechanism. Thus, the same concepts can be applied within a given process, over sockets, through HTTP connections, or within various messaging environment.

In order to receive any information from the wallet RPC, you must send a message with the POST method. The JSON-RPC API accepts messages with the format: { “jsonrpc” : version , “method” : method, “params”: params, “id”: id },

using fields described by:

Variable	Description
version	Version of the JSON RPC protocol. Monero Wallet RPC supports the 2.0 version.
method	method you would like to call.
params	array; parameters for the method. They can include any additional information.
id	number (int); start from 0, used for tracking the responses.

Monero RPC Calls

Monero RPC API is compatible with methods provided by the Monero developers. Methods include transferring to an address a given amount, requesting subaddresses or integrated addresses. In this section, you will read about the “famous” methods for calling an RPC action.

For example:

- Transfer
- make_uri
- parse_uri
- make_integrated address ...

Specific Instructions for Official Monero GUI

The following instructions show how to carry out the tasks described above through the Monero graphical user interface (GUI). If you are using a different wallet, you can skip this section.

Setting up a wallet

The official Monero GUI can be downloaded through www.getmonero.org. Once you have unpackaged and loaded the application, you will be presented with a language selection screen:

If you don't see your language above, please feel free to submit a translation to help others! Once you select a language, the GUI will ask whether you wish to generate a new seed, or restore an existing wallet.

The Testnet and Stagenet are for developers, so do not check either option. If this is your first Monero wallet, press "create new wallet." The Monero software will generate a new seed for you, and show you the 25-word seed mnemonic. Be sure to write this down and store it in a safe place where nobody else will find it!

Next, you will have the option to start your own node, or connect to a remote node.

You can enter a wallet password, to keep your fund secure if somebody else accesses your computer. The wallet password is a local security feature, like a PIN screen unlock. It does not impact the way or how your Moneroj are stored on the blockchain, so restoring your wallet from the seed will bypass the local passphrase.

Receiving Monero

The "Receive" tab of the Monero GUI contains both the text and QR-code forms of your receiving addresses. The "Create new address" button generates as more "subaddresses," which will all direct to this same wallet (seed). You can indicate an "Amount to receive," which will be encoded into the QR code.

Sending Monero

To send Monero, you simply specify the amount that you wish to send and the recipient's address. The Payment ID field can be left blank, unless your recipient specifies a Payment ID in advance. The Description field is stored locally, so you can leave notes for yourself.

Transaction History

Proof of Payment

Proof of payment verification is available through the GUI under “prove/check.” The below screenshot shows the transaction ID, address, and transaction key from the Maria & Kahn example above.

=======

Problems and Troubleshooting

Problem: I transferred Moneroj to my wallet, but my balance is still 0

Available solutions

1) Check if you copied the correct Monero Address (sometimes malware can try to edit the copied Monero address);

2) Verify whether transaction actually arrived at your wallet / address by these steps:

```
A. Go to the Settings page of the GUI and press on Show seed & keys. Subsequent
copy the private view key.

B. Go to https://xmrchain.net/ blockexplorer

C. Enter your transaction ID / hash.

D. Now search the page for Decode outputs.

E. Enter your private view key. In addition, enter your address in the second b

F. Press on decode outputs.

G. If it shows "output true" it proves you correctly received your XMR
```

3) Verify if you are connected to the blockchain (using either remote nodes or local daemon)

If everything above checks out it's most likely a visibility bug, meaning that your Moneroj are in the correct place, but your wallet has not yet found them. The Monero GUI uses a local cache that can take a few seconds to refresh, especially on Windows. If this problem persists, try to reach our channels for support (see chapter V).

Glossary

Account

Those familiar with Monero's predecessors will be more familiar with the term wallet to describe this. In Monero we call this an account, and it is a private account owned and operated by a Monero user.

Your account contains all of the Monero transactions you have sent and received. Your account balance is a sum of all the Monero you've received, less the Monero you've sent. When using Monero you may notice that your account has two balances, a locked and an unlocked balance. The unlocked balance contains funds that can be spent immediately, and the locked balance contains funds that you can't spend right now. You may receive a transaction that has an unlock time set, or you may have sent some Monero and are waiting for the change to come back to your wallet, both situations that could lead to those funds being locked for a time.