

Bitcoin and Cryptocurrency Technologies

**Arvind Narayanan, Joseph Bonneau, Edward Felten,
Andrew Miller, Steven Goldfeder**

Draft — Oct 6, 2015

Feedback welcome! Email bitcoinbook@lists.cs.princeton.edu

Chapter 6: Bitcoin and Anonymity

“Bitcoin is a secure and anonymous digital currency”

— WikiLeaks donations page

“Bitcoin won't hide you from the NSA's prying eyes”

— Wired UK

One of the most controversial things about Bitcoin is its supposed anonymity. First, is Bitcoin anonymous? As you can see from the mutually contradictory quotes above, there's some confusion about this. Second, do we *want* Bitcoin to be anonymous? There are pros and cons of anonymity, and this leads to some basic questions: is having an anonymous cryptocurrency beneficial for the stakeholders? Is it good for society? Is there a way to isolate the positive aspects of anonymity while doing away with the negative parts?

These questions are hard because they depend in part on one's ethical values. We won't answer them in this chapter, but we'll examine arguments for and against anonymity. But mostly we'll stick to the technology. We'll study various technologies — some already present in Bitcoin and others that have been proposed to be added to it — that aim to increase Bitcoin's anonymity. We'll also look at proposals for alternative cryptocurrencies that have different anonymity properties from Bitcoin. These technologies raise new questions: How well do they work? How difficult would they be to adopt? What are the tradeoffs to be made in adopting them?

6.1: Anonymity Basics

Defining anonymity. Before we can properly discuss whether (or to what extent) Bitcoin is anonymous, we need define anonymity. We must understand what exactly we mean by anonymity, and the relationship between anonymity and similar terms, such as privacy.

At a literal level, anonymous means “without a name.” When we try to apply this definition to Bitcoin, there are two possible interpretations: interacting without using your real name, or interacting without using any name at all. These two interpretations lead to very different conclusions as to whether Bitcoin is anonymous. Bitcoin addresses are hashes of public keys. You don't need to use your real name in order to interact with the system, but you do use your public key hash as your identity. Thus, by the first interpretation, Bitcoin is anonymous as you do not use your real name. However, by the second interpretation, it is not; the address that you use is a pseudo-identity. In the language of computer science, this middle ground of using an identity that is not your real name is called **pseudonymity**.

Recall that you are free to create as many Bitcoin addresses as you like. With this in mind, you might be wondering whether Bitcoin addresses really are pseudo-identities considering that you can create as many of these pseudonyms as you like. As we'll see, this still does not make Bitcoin anonymous.

In computer science, anonymity refers to pseudonymity together with **unlinkability**. Unlinkability is a property that's defined with respect to the capabilities of a specific adversary. Intuitively, unlinkability means that if a user interacts with the system repeatedly, these different interactions should not be able to be tied to each other from the point of view of the adversary in consideration.

Sidebar. The distinction between anonymity and mere pseudonymity is something that you might be familiar with from a variety of other contexts. One good example is online forums. On a forum like Reddit, you pick a long-term pseudonym and interact over a period of time with that pseudonym. You could create multiple pseudonyms, or even a new one for every comment, but that would be tedious and annoying, and most people don't do it. So interacting on Reddit is pseudonymous but not quite anonymous. 4Chan, by contrast, is another online forum that does allow users to post anonymously — with no attribution at all.

Bitcoin is pseudonymous, but pseudonymity is not enough if your goal is to achieve privacy. Recall that the block chain is public and anyone can look up all Bitcoin transactions that involved a given address. If anyone is ever able to link your Bitcoin address to your real world identity, then all of your transactions — past, present, and future — have been linked back to your identity.

To make things worse, linking a Bitcoin address to a real-world identity is often easy. If you interact with a Bitcoin business — be it an online wallet services, exchange, or other merchant — they are going to want your real life identity in order to let you transact with them. An exchange might require you to link your bank account, and a merchant will need your shipping address.

Or you might go to a coffee shop and pay for your coffee with bitcoins. Since you're physically present in the store, the barista knows a lot about your identity even if they don't ask for your real name. Your physical identity thus gets tied to one of your Bitcoin transactions, making all the other transactions that involved that address linkable to you. This is clearly not anonymous.

Side channels. Even if a direct linkage doesn't happen, your pseudonymous profile can be **deanonymized** due to side channels, or indirect leakages of information. For example, someone may look at a profile of pseudonymous Bitcoin transactions and note at what times of day that user is active. They can correlate this information with other publicly available information. Perhaps they'll notice that some Twitter user is active during roughly same time intervals, creating a link between the pseudonymous Bitcoin profile and a real-world identity (or at least a Twitter identity). Clearly pseudonymity does not guarantee privacy or anonymity, and to achieve those, we require the stronger property of unlinkability as well.

Unlinkability. To understand unlinkability in the Bitcoin context more concretely, let's enumerate some key properties that are required for Bitcoin activity to be unlinkable:

1. It should be hard to link together different addresses of the same user.
2. It should be hard to link together different transactions made by the same user.
3. It should be hard to link the sender of a payment to its recipient.

The first two properties are intuitive, but the third one is a bit tricky. If you interpret "a payment" as a Bitcoin transaction, then the third property is clearly false. Every transaction has inputs and outputs, and these inputs and outputs are inevitably going to be in the block chain and publicly linked together. However, what we mean by a payment is not a single Bitcoin transaction, but rather anything that has the effect of transferring bitcoins from the sender to the recipient. It might involve a roundabout route of transactions. What we want to ensure is that it's not feasible to link the sender and the ultimate recipient of the payment by looking at the block chain.

Anonymity set. Even under our broader definition of a payment, the third property seems hard to achieve. Say you pay for a product that cost a certain number of bitcoins and you send that payment through a circuitous route of transactions. Somebody looking at the block chain will still be able to infer something from the fact that a certain number of bitcoins left one address and roughly the same number of bitcoins (minus transaction fees, perhaps) ended up at some other address. Moreover, despite the circuitous route, the initial sending and the ultimate receiving will happen in roughly the same time period because the merchant will want to receive payment without too much of a delay.

Because of this difficulty, we usually don't try to achieve complete unlinkability among all possible transactions or addresses in the system, but rather something more limited. Given a particular adversary, the *anonymity set* of your transaction is the set of transactions which the adversary cannot distinguish from your transaction. Even if the adversary knows you made a transaction, he can only tell that it's one of the transactions in the set, but not which one it is. We try to maximize the size of the anonymity set — the set of other addresses or transactions amongst which we can hide.

Calculating the anonymity set is tricky. Since the anonymity set is defined with respect to a certain adversary or set of adversaries, you must first concretely define what your adversary model is. You have to reason carefully about what that adversary knows, what he doesn't know, and what is it that we are trying to hide from the adversary — that is, what the adversary *cannot* know for the transaction to be considered anonymous. There's no general formula for doing this. It requires carefully analyzing each protocol and system on a case-by-case basis.

In the Bitcoin community, people often carry out intuitive analyses of anonymity services without rigorous definitions. *Taint analysis* is particularly popular: it's a way of calculating how "related" two addresses are. If bitcoins sent by an address S always end up at another address R, whether directly or after passing through some intermediate addresses, then S and R will have a high taint score. The formula accounts for transactions with multiple inputs and/or outputs and specifies how to allocate taint.

Unfortunately, taint analysis is not a good measure of Bitcoin anonymity. It implicitly assumes that the adversary is using the same mechanical calculation to link pairs of addresses. A slightly cleverer adversary may use other techniques such as looking at the timing of transactions or even exploit idiosyncrasies of wallet software as we'll see later in this chapter. So taint analysis might suggest that you have a high degree of anonymity in a certain situation, but in fact you might not.

Why we need anonymity. Having seen what anonymity means, let's answer some meta-questions about anonymity before we go further: Why do people want anonymity? What are the ethical implications of having an anonymous currency?

In block chain-based currencies, all transactions are recorded on the ledger, which means that they are publicly and permanently traceable to the associated addresses. So the privacy of your Bitcoin transactions can potentially be far worse than with traditional banking. If your real-world identity ever gets linked to a Bitcoin address, then you have totally lost privacy for all transactions — past, present, and future — associated with that address. Since the block chain is publicly available, literally anyone might be able to carry out this type of deanonymization without you even realizing that you've been identified.

With this in mind, we can identify two different motivations for having anonymous cryptocurrencies. The first is simply to achieve the level of privacy that we are already used to from traditional banking, and mitigate the deanonymization risk that the public block chain brings. The second is to go above and beyond the privacy level of traditional banking and develop currencies that make it technologically infeasible for anyone to track the participants.

Ethics of anonymity. One of the major concerns with anonymous cryptocurrencies is that they can be used for money laundering or other illegal transactions. The good news is that while cryptocurrency transactions themselves may be pseudonymous or anonymous, the interface between digital cash and fiat currencies is not. In fact, these flows are highly regulated, as we'll see in the next chapter. So cryptocurrencies are no panacea for money laundering or other financial crimes.

Nevertheless one may ask: can't we design the technology in such a way that only the good uses of anonymity are allowed and the bad uses are somehow prohibited? This is in fact a recurring plea to computer security and privacy researchers. Unfortunately, it never turns out to be possible. The reason is that use cases that we classify as good or bad from a moral viewpoint turn out to be technologically identical. In Bitcoin, it's not clear how we could task miners with making moral decisions about which transactions to include.

Our view is that the potential good that's enabled by having anonymous cryptocurrencies warrant their existence, and that we should separate the technical anonymity properties of the system from the legal principles we apply when it comes to using the currency. It's not a completely satisfactory solution, but it's perhaps the best way to achieve a favorable trade-off.

Sidebar: Tor. The moral dilemma of how to deal with a technology that has both good and bad uses is by no means unique to Bitcoin. Another system whose anonymity is controversial is Tor, an anonymous communication network.

On the one hand, Tor is used by normal people who want to protect themselves from being tracked online. It's used by journalists, activists, and dissidents to speak freely online without fear of repercussion from oppressive regimes. It's also used by law enforcement agents who want to monitor suspects online revealing their IP address (after all, ranges or blocks of IP addresses assigned to different organizations, including law enforcement agencies, tend to be well known). Clearly, Tor has many applications that we might morally approve of. However, it also has clearly bad uses: it's used by operators of botnets to issue commands to the infected machines under their control, and it's used to distribute child pornography.

Distinguishing between these uses at a technical level is essentially impossible. The Tor developers and the Tor community have grappled with this conundrum, society at large has grappled with it to some degree as well. We seem to have concluded that overall, it's better for the world that the technology exists. In fact one of the main funding sources of the Tor project is the U.S. State Department. They're interested in Tor because it enables free speech online for dissidents in oppressive regimes. Meanwhile, law enforcement agencies seem to have grudgingly accepted Tor's existence, and have developed ways to work around it. The FBI has regularly managed to bust websites on the "dark net" that distributed child pornography, even though these sites hid behind Tor. Often this is because the operators tripped up. We must remember that technology is only a tool and that perpetrators of crimes live in the real world, where they may leave physical evidence or commit all-too-human errors when interacting with the technology.

Anonymization vs. decentralization. We'll see a recurring theme throughout this chapter that the design criteria anonymization and decentralization are often in conflict with one another. If you recall Chaum's ecash from the preface, it achieved perfect anonymity in a sense, but through an interactive blind-signature protocol with a central authority, a bank. As you can imagine, such protocols are very difficult to decentralize. Secondly, if we decentralize, then we must keep some sort of mechanism to trace transactions and prevent double spending, and this public traceability of transactions is a threat to anonymity.

Later in this chapter, we'll see Zerocoin and Zerocash, anonymous decentralized cryptocurrencies that have some similarities to Chaum's ecash, but they have to tackle thorny cryptographic challenges because of these two limitations.

6.2: How to De-anonymize Bitcoin

We've said several times that Bitcoin is only pseudonymous, so all of your transactions or addresses could get linked together. But let's take a closer look at how that might actually happen.

Figure 6.1 shows a snippet of the Wikileaks donation page (including the quote at the beginning of the chapter). Notice the refresh button next to the donation address. As you might expect, clicking the button will replace the donation address with an entirely new, freshly generated address. Similarly, if you refresh the page or close it and visit it later, it will have another address, never previously seen. That's because Wikileaks wants to make sure that each donation they receive goes to a new public key that they create just for that purpose. Wikileaks is taking maximal advantage of the ability to create new pseudonyms. This is in fact the Bitcoin best practice for anonymity.

Bitcoin is a secure and anonymous digital currency. Bitcoins cannot be easily tracked back to you, and are safer and faster alternative to other donation methods. You can send BTC to the following address:


13DFamCvSxG8EG16VyXzdpfqxyooifswYx 

Figure 6.1: Snippet from Wikileaks donation page. Notice the refresh icon next to the Bitcoin address. Wikileaks follows the Bitcoin best practice of generating a new receiving address for every donation.

At first you might think that these different addresses must be unlinkable. Wikileaks receives each donation separately, and presumably they can also spend each of those donations separately. But things quickly break down.

Suppose Alice wants to buy a teapot that costs 8 bitcoins (more likely 8 centi-bitcoins, at 2015 exchange rates). Suppose, further, that her bitcoins are in three separate unspent outputs at different addresses whose amounts are 3, 5, and 6 bitcoins respectively. Alice doesn't actually have an address with 8 bitcoins sitting in it, so she must combine two of her outputs as inputs into a single transaction that she pays to the store.

But this reveals something. The transaction gets recorded permanently in the block chain, and anyone who sees it can infer that the two inputs to the transaction are under the control of the same user. In other words, **shared spending is evidence of joint control** of the different input addresses.

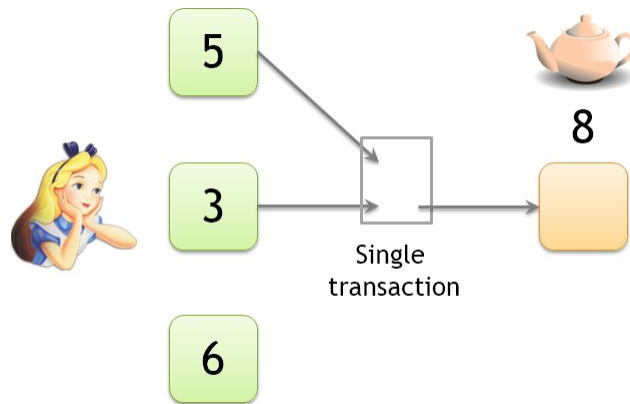


Figure 6.2 : To pay for the teapot, Alice has to create a single transaction having inputs that are at two different address. In doing so, Alice reveals that these two addresses are controlled by a single entity.

But it doesn't stop there. The adversary can repeat this process and **transitively** link an entire cluster of transactions as belonging to a single entity. If another address is linked to **either** one of Alice's addresses in this manner, then we know that all three addresses belong to the same entity, and we can use this observation to cluster addresses. In general, if an output at a new address is spent together with one from any of the addresses in the cluster, then this new address can also be added to the cluster.

Later in this chapter we'll study an anonymity technique called CoinJoin that works by violating this assumption. But for now, if you assume that people are using regular Bitcoin wallet software without any special anonymity techniques, then this clustering tends to be pretty robust. We haven't yet seen how to link these clusters to real-world identities, but we'll get to that shortly.

Going back to our example, suppose the price of the teapot has gone up from 8 bitcoins to 8.5 bitcoins. Alice no longer find a set of unspent outputs that she can combine to produce the exact change needed for the teapot. Instead, Alice exploits the fact that transactions can have multiple outputs, as shown in Figure 6.3. One of the outputs is the store's payment address and the other is a "change" address owned by herself.

Now consider this transaction from the viewpoint of an adversary. He can deduce that the two input addresses belong to the same user. He might further suspect that one of the output addresses also belongs to that same user, but has no way to know for sure which one that is. The fact that the 0.5 output is smaller doesn't mean that it's the change address. Alice might have 10,000 bitcoins sitting in a transaction, and she might spend 8.5 bitcoins on the teapot and send the remaining 9,991.5 bitcoins back to herself. In that scenario the bigger output is in fact the change address.

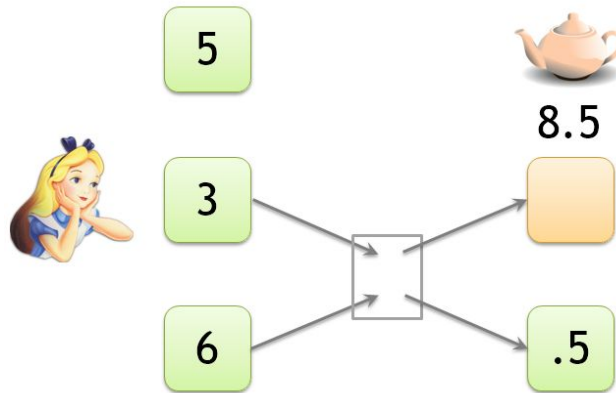


Figure 6.3: Change address. To pay for the teapot, Alice has to create a transaction with one output that goes to the merchant and another output that sends change back to herself.

A somewhat better guess is that if the teapot had cost only 0.5 bitcoins, then Alice wouldn't have had to create a transaction with two different inputs, since either the 3 bitcoin input or the 6 bitcoin input would have been sufficient by itself. But the effectiveness of this type of heuristic depends entirely on the implementation details of commonly used wallet software. There's nothing preventing wallets (or users) from combining transactions even when not strictly necessary.

Idioms of use. Implementation details of this sort are called "idioms of use". In 2013, a group of researchers found an idiom of use that was true of most wallet software, and led to a powerful heuristic for identifying change addresses. Specifically, they found that wallets typically generate a fresh address whenever a change address is required. Because of this idiom of use, change addresses are generally addresses that have never before appeared in the block chain. Non-change outputs, on the other hand, are often not new addresses and may have appeared previously in the block chain. An adversary can use this knowledge to distinguish change addresses and link them with the input addresses.

Exploiting idioms of use can be error prone. The fact that change addresses are fresh addresses just happens to be a feature of wallet software. It was true in 2013 when the researchers tested it. Maybe it's still true, but maybe it's not. Users may choose to override this default behavior. Most importantly, an adversary who is aware of this technique can easily evade it. Even in 2013, the researchers found that it produced a lot of false positives, in which they ended up clustering together addresses that didn't actually belong to the same entity. They reported that they needed significant manual oversight and intervention to prune the false positives.

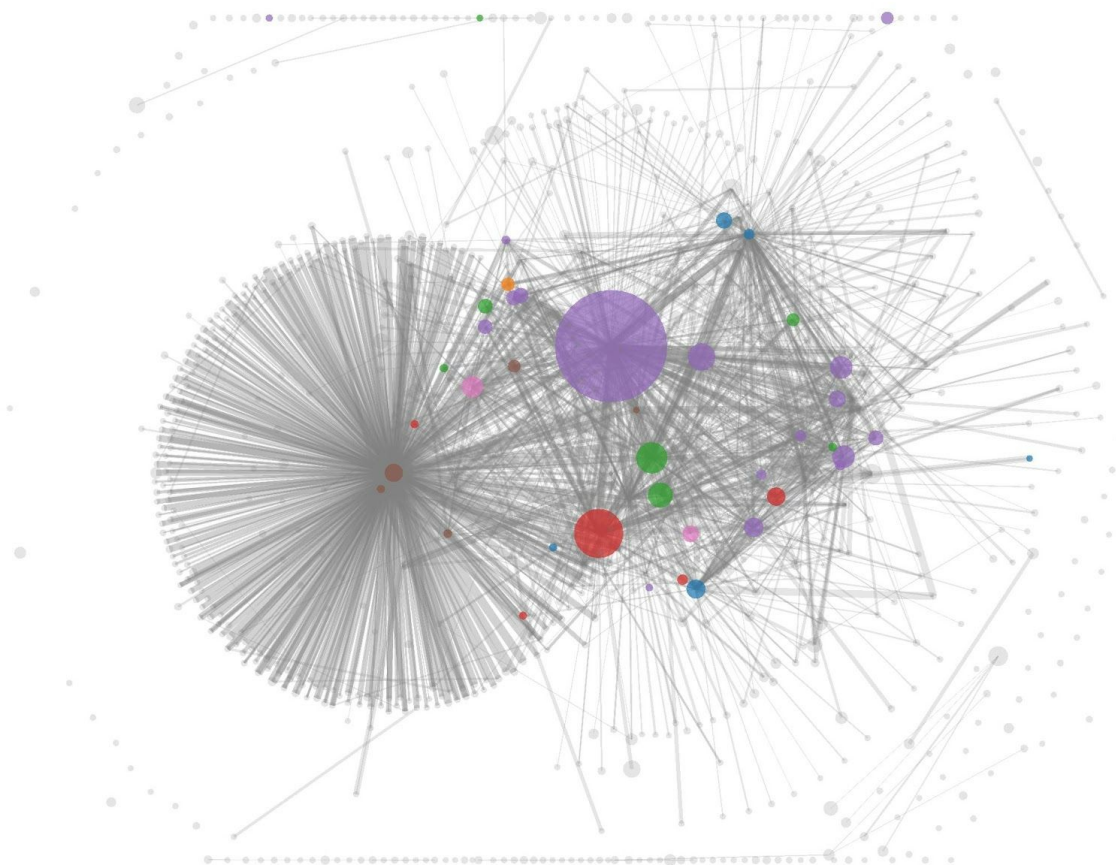


Figure 6.4: Clustering of addresses. In the 2013 paper *A Fistful of Bitcoins: Characterizing Payments Among Men with No Names*, researchers combined the shared-spending heuristic and the fresh-change-address heuristic to cluster Bitcoin addresses. The sizes of these circles represent the quantity of money flowing into those clusters, and each edge represents a transaction.

Attaching real-world identities to clusters. In Figure 6.4, we see how Meiklejohn et al. clustered Bitcoin addresses. But the graph is not labeled — we haven't yet attached identities to the clusters.

We might be able to make some educated guesses based on what we know about the Bitcoin economy. Back in 2013, Mt. Gox was the largest Bitcoin exchange, so we might guess that the big purple circle represents addresses controlled by them. We might also notice that the brown cluster on the left has a tiny volume in Bitcoins despite having the largest number of transactions. This fits the pattern of the gambling service Satoshi Dice, which is basically a (fairly popular) game you send a tiny amount of bitcoins as a wager.

Overall, this isn't a great way to identify clusters. It requires knowledge and guesswork and will only work for the most prominent services. We need something more reliable.

Tagging by transacting. What about just visiting the website for each exchange or merchant and looking up the address they advertise for receiving bitcoins? That doesn't quite work, however,

because most services will advertise a new address for every transaction, and the address shown to you is not yet in the block chain. There's no point in waiting, either, because that address will never be shown to anyone else.

The only way to reliably infer addresses is to actually transact with that service provider — depositing bitcoins, purchasing an item, and so on. When you send bitcoins to or receive bitcoins from the service provider, you will now know one of their addresses, which will soon end up in the block chain (and in one of the clusters). You can then tag that entire cluster with the service provider's identity.

This is exactly what the *Fistful of Bitcoins* researchers did. They bought a variety of things, joined mining pools, used Bitcoin exchanges, wallet services, and gambling sites, and interacted in a variety of other ways with service providers, compromising 344 transactions in all.

In Figure 6.5, we again show the clusters of Figure 6.4, but this times with the labels attached. While our guesses about Mt. gox and Satoshi Dice were correct, the researchers were able to identify numerous other service providers that would have been hard to identify without transacting with them.

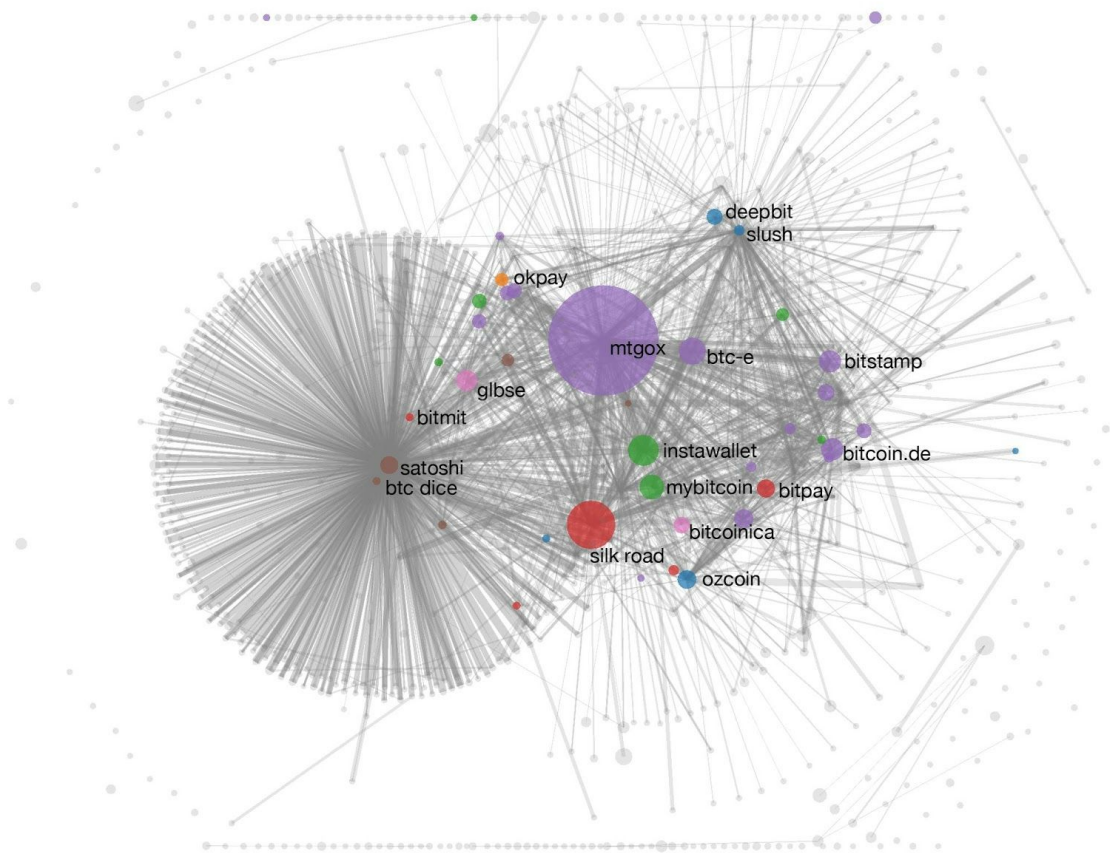


Figure 6.5. Labeled clusters. By transacting with various Bitcoin service providers, Meiklejohn et al. were able to attach real world identities to their clusters.

Identifying individuals. The next question is: can we do the same thing for individuals? That is, can we connect little clusters corresponding to individuals to their real-life identities?

Directly transacting. Anyone who transacts with an individual — an online or offline merchant, an exchange, or a friend who splits a dinner bill using Bitcoin — knows at least one address belonging to her.

Via service providers. In the course of using Bitcoin over a few months or years, most users will end up interacting with an exchange or another centralized service provider. These service typically providers ask users for their identities — often they're legally required to, as we'll see in the next chapter. If law enforcement wants to identify a user, they can turn to these service providers.

Carelessness. People often post their Bitcoin addresses in public forums. A common reason is to request donations. When someone does this it creates a link between his identity and one of his addresses. If they don't use the anonymity sections that we'll look at in the following sections, they risk having all their transactions de-anonymized.

Things get worse over time. The history of deanonymization algorithms shows when the data is publicly available, the adversary's techniques only get more powerful with time. Besides, more auxiliary information becomes available that attackers can use to attach identities to clusters. This is something to worry about if you care about privacy.

The deanonymization techniques we've examined so far are all based on analyzing the graphs of transactions in the block chain. They are collectively known as **transaction graph analysis**.

Network-layer deanonymization. We'll now look at a completely different way in which users can get deanonymized that does not rely on the transaction graph. Recall that in addition to the block chain, Bitcoin also has a peer-to-peer network where messages are sent around that don't necessarily get permanently recorded in the block chain.

In networking terminology, the block chain is called the **application layer** and the peer-to-peer network is the **network layer**. Network-layer deanonymization was first pointed out by Dan Kaminsky in 2011 in a talk at the Black Hat conference. He noticed that when a node creates a transaction, it connects to many nodes at once and broadcasts the transaction. If sufficiently many nodes on the network collude with each other (or are run by the same adversary), they could figure out the first node to broadcast any transaction. Presumably, that would be a node that's run by the user who created the transaction. The adversary could then link the transaction to the node's IP address. An IP address is close to a real-world identity; there are many ways to try to unmask the person behind an IP address. Thus, network-layer de-anonymization is a serious problem for privacy.



Figure 6.6. Network level deanonymization. As Dan Kaminsky pointed out in his 2011 Black Hat talk, “the first node to inform you of a transaction is probably the source of it.” This heuristic is amplified when multiple nodes cooperate and identify the same source.

Luckily, this is a problem of communications anonymity, and that problem has received a ton of attention in the research community. As we saw earlier in Section 6.1, there’s a widely deployed system called Tor that you can use for communicating anonymously.

There are a couple of caveats to using Tor as a network-layer anonymity solution for Bitcoin. First, there can be subtle interactions between the Tor protocol and any protocol that’s overlaid on top of it, resulting in new ways to breach anonymity. Indeed researchers have found potential problems with the anonymity of Bitcoin-over-Tor, and some mitigations have been proposed. Second, there might be other anonymous communication technologies better suited to use in Bitcoin. Tor is intended for “low-latency” activities such as web browsing where you don’t want to sit around waiting for too long. It makes some compromises to achieve anonymity with low latency. Bitcoin, on the other hand, is a high-latency system from the point of view of regular because it takes a while for transactions to get confirmed in the block chain. In theory, at least, you might want to use an alternative anonymity technique such as a **mix net**, but for the moment, Tor has the advantage of being an actual system that has a large user base and whose security has been intensely studied.

So far, we’ve seen how different addresses could get linked together by transaction graph analysis of the block chain, and they could also get linked to a real-world identity. We’ve also seen that a transaction or address could get linked to an IP address based on the peer-to-peer network. The latter problem is relatively easy to solve, even if it can’t be considered completely solved yet. The former problem is much trickier, and we’re going to spend the rest of this chapter talking about ways to solve it.

6.3: Mixing

There are several mechanisms that can make transaction graph analysis less effective. One such technique is mixing, and the intuition behind it is very simple: if you want anonymity, use an intermediary. This principle is not specific to Bitcoin and is useful in many situations where anonymity is a goal. In Bitcoin we call this mixing, and it's illustrated in Figure 6.7.

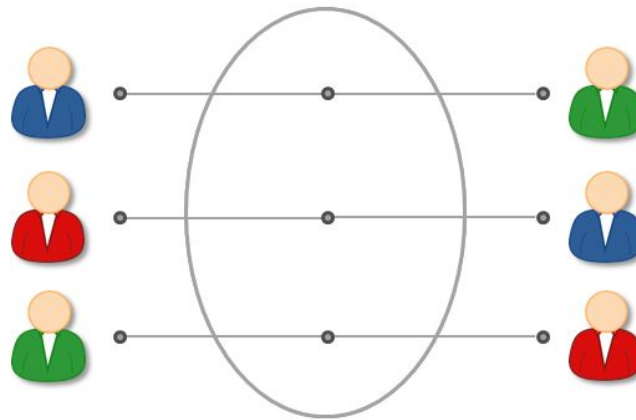


Figure 6.7 : Mixing. Users send coins to an intermediary, and get back coins that were deposited by other users. This makes it hard to trace a user's coins on the block chain.

Online wallets as mixes. If you recall our discussion of online wallets, they may seem to be suitable as intermediaries. Online wallets are services where you can store your bitcoins online and withdraw them at some later date. Typically the coins that you withdraw won't be the same as the coins you deposited. Do online wallets provide effective mixing, then?

Online wallets do provide a measure of unlinkability which can foil attempts at transaction graph analysis — in one case, researchers had to retract a claim which had received a lot of publicity because they link they thought they'd found was a spurious one caused by an online wallet.

On the other hand, there are several important limits to using online wallets for mixing. First, most online wallets don't actually promise to mix users' funds; instead, they do it because it simplifies the engineering. You have no guarantee that they won't change their behavior. Second, even if they do mix funds, they will almost certainly maintain records internally that will allow them to link your deposit to your withdrawal. This is a prudent choice for wallet services for reasons of both security and legal compliance. So if your threat model includes the possibility of the service provider itself tracking you, or getting hacked, or being compelled to hand over their records, you're back to square one. Third, in addition to keeping logs internally, reputable and regulated services will also require and record your identity (we'll discuss regulation in more detail in the next chapter). You won't be able to simply create an account with a username and password. So in one sense it leaves you worse

off than not using the wallet service. That's why we called out these centralized service providers as a vehicle for *de-anonymization* in the previous section.

The anonymity provided by online wallets is similar to that provided by the traditional banking system. There are centralized intermediaries that know a lot about our transactions, but from the point of view of a stranger with no privileged information we have a reasonable degree of privacy. But as we discussed, the public nature of the block chain means that if something goes wrong (say, a wallet or exchange service gets hacked and records are exposed), the privacy risk is worse than with the traditional system. Besides, most people who turn to Bitcoin for anonymity tend to do so because they are unhappy with anonymity properties of the traditional system and want a better (or a different kind of) anonymity guarantee. These are the motivations behind dedicated mixing services.

Dedicated mixing services. In contrast to online wallets, dedicated mixes promise not to keep records, nor do they require your identity. You don't even need a username or other pseudonym to interact with the mix. You send your bitcoins to an address provided by the mix, and you tell the mix a destination address to send bitcoins to. Hopefully the mix will soon send you (other) bitcoins at address you specified. It's essentially a swap.

While it's good that dedicated mixes promise not to keep records, you still have to trust them to keep that promise. And you have to trust that they'll send you back your coins at all. Since mixes aren't a place where you store your bitcoins, unlike wallets, you'll want your coins back relatively quickly, which means that the pool of other coins that your deposit will be mixed with is much smaller — those that were deposited at roughly the same time.

Sidebar: Terminology. In this book, we'll use the term ***mix*** to refer to a dedicated mixing service. An equivalent term that some people prefer is ***mixer***.

You might also encounter the term ***laundry***. We don't like this word, because it needlessly attaches moral meaning to something that's a purely technical concept. As we've seen, there are very good reasons why you might want to protect your privacy in Bitcoin and use mixes for everyday privacy. Of course, we must also acknowledge the bad uses, but using the term laundry promotes the negative connotation, as it implies that your coins are 'dirty' and you need to clean them.

Several researchers — including four of the five authors of this textbook — studied mixes and proposed a set of principles for improving the way that mixes operate, both in terms of increasing anonymity and in terms of the security of entrusting your coins to the mix. We will go through each of these guidelines.

Use a series of mixes. The first principle is to use a series of mixes, one after the other, instead of just a single mix. This is a well-known and well-established principle — for example, Tor, as we'll see in a bit, using a series of routers for anonymous communication. This reduces your reliance on the trustworthiness of any single mix. As long as any one of the mixes in the series keeps its promise and

deletes its records, you have reason to expect that no one will be able to link your first input to the ultimate output that you receive.

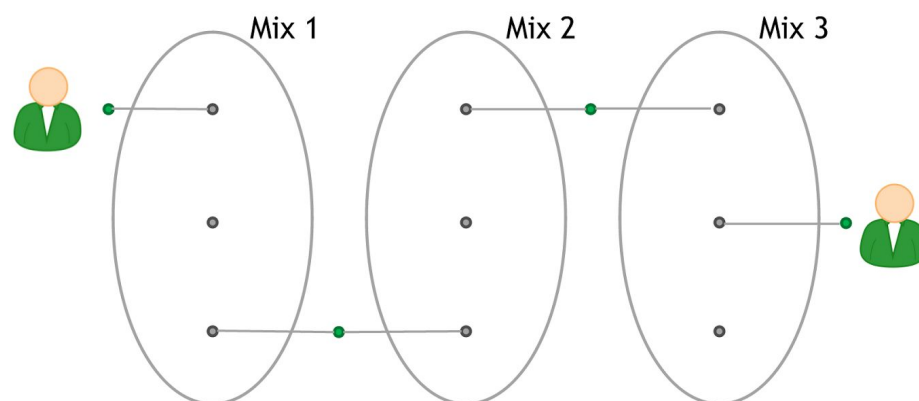


Figure 6.8. Series of mixes. We begin with a user who has a coin or input address that we assume the adversary has managed to link to her. The user sends the coin through various mixes, each time providing a freshly generated output address to the mix. Provided that at least one of these mixes destroys its records of the input to output address mapping, and there are no side-channel leaks of information, an adversary won't be able to link the user's original coin to her final one.

Uniform transactions. If mix transactions by different users had different quantities of bitcoins, then mixing wouldn't be very effective. Since the value going into the mix and coming out of a mix would have to be preserved, it will enable linking a user's coins as they flow through the mix, or at least greatly diminish the size of the anonymity set.

Instead, we want mix transactions to be uniform in value so that linkability is minimized. All mixes should agree on a standard **chunk size**, a fixed value that incoming mix transactions must have. This would increase the anonymity set as all transactions going through *any* mix would look the same and would not be distinguishable based on their value. Moreover, having a uniform size across all mixes would make it easy to use a series of mixes without having to split or merge transactions.

Client side must be automated. In addition to trying to link coins based on transaction values, a clever adversary can attempt various other ways to de-anonymize, for example, by observing the timing of transactions. These attacks can be avoided, but the precautions necessary are too complex and cumbersome for human users. Instead, the client-side functionality for interacting with mixes must be automated and built into privacy-friendly wallet software.

Fees must be all-or-nothing. Mixes are businesses and expect to get paid. One way for a mix to charge fees is to take a cut of each transaction that users send in. But this is problematic for anonymity, because mix transactions can no longer be in standard chunk sizes. (If users try to split and merge their slightly-smaller chunks back to the original chunk size, it introduces serious and hard-to-analyze anonymity risks because of the new linkages between coins that are introduced.)

Don't confuse mixing fees with transaction fees, which are collected by miners. Mixing fees are separate from and in addition to such fees.

To avoid this problem, mixing fees should be all-or-nothing, and applied probabilistically. In other words, the mix should swallow the whole chunk with a small probability or return it in its entirety. For example, if the mix wants to charge a 0.1% mixing fee, then one out every 1,000 times the mix should swallow the entire chunk, whereas 999 times out of 1,000 the mix should return the entire chunk without taking any mixing fee.

This is a tricky to accomplish. The mix must make a probabilistic decision and convince the user that it didn't cheat: that it didn't bias its random number generator so that it has (say) a 1% probability of retaining a chunk as a fee, instead of 0.1%. Cryptography provides a way to do this, and we'll refer you to the *Mixcoin* paper in the Further Reading section for the details. The paper also talks about various ways in which mixes can improve their trustworthiness.

Mixing in practice. As of 2015, there isn't a functioning mix ecosystem. There are many mix services out there, but they have low volumes and therefore small anonymity sets. Worse, many mixes have been reported to steal bitcoins. Perhaps the difficulty of "bootstrapping" such an ecosystem is one reason why it has never gotten going. Given the dodgy reputation of mixes, not many people will want to use them, resulting in low transaction volumes. In the absence of much money to be made from providing the advertised services, mix operators might be tempted to steal funds instead, perpetuating the cycle.

Today's mixes don't follow any of the principles we laid out. Each mix operates independently and typically provides a web interface, with which the user interacts with manually to specify the receiving address and any other necessary parameters. The user gets to choose the amount that she would like to mix. The mix will take a cut of every transaction as a mixing fee and send the rest to the destination address.

We think it's necessary for mixes (and wallet software) to move to the model we presented in order to achieve strong anonymity, resist clever attacks, provide a usable interface, and attract high volumes. But it remains to be seen if a robust mix ecosystem will ever evolve.

6.4: Decentralized Mixing

Decentralized mixing is the idea of getting rid of mixing services and replacing them with a peer-to-peer protocol by which a group of users can mix their coins. As you can imagine, this approach is better philosophically aligned with Bitcoin.

Decentralization also has more practical advantages. First, it doesn't have the bootstrapping problem: users don't have to wait for reputable centralized mixes to come into existence. Second, theft is impossible in decentralized mixing; the protocol ensures that when you put in bitcoins to be mixed, you'll get bitcoins back of equal value. Because of this, even though some central coordination turns out to be helpful in decentralized mixing, it's easier for someone to set up such a service because they don't have to convince users that they're trustworthy. Finally, in some ways decentralized mixing can provide better anonymity.

Coinjoin. The main proposal for decentralized mixing is called Coinjoin. In this protocol, different users jointly create a single Bitcoin transaction that combines all of their inputs. The key technical principle that enables Coinjoin to work is this: when a transaction has multiple inputs coming from different addresses, the signatures corresponding to each input are separate from and independent of each other. So these different addresses could be controlled by different people. You don't need one party to collect all of the private keys.

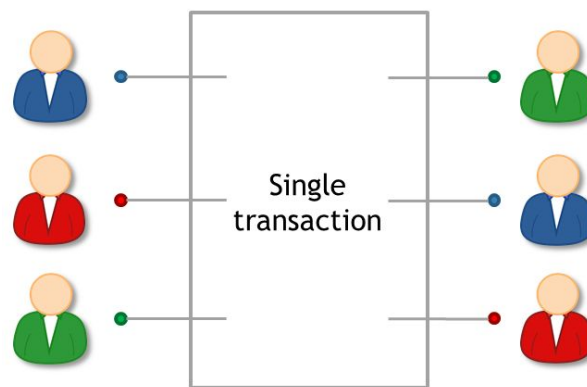


Figure 6.9. A Coinjoin transaction.

This allows a group of users to mix their coins with a single transaction. Each user supplies an input and output address and form a transaction with these addresses. The order of the input and output addresses is randomized so an outsider will be unable to determine the mapping between inputs and outputs. Participants check that their output address is included in the transaction and that it receives the same amount of Bitcoin that they are inputting (minus any transaction fees). Once they have confirmed this, they sign the transaction.

Somebody looking at this transaction on the blockchain — even if they know that it is a Coinjoin transaction — will be unable to determine the mapping between the inputs and outputs. From an outsider's perspective the coins have been mixed, and this is the essence of Coinjoin.

What we've described so far is just one round of mixing. But the principles that we discussed before still apply. You'd want to repeat this process with (presumably) different groups of users. You'd also

want to make sure that the chunk sizes are standardized so that you don't introduce any side channels.

Let's now delve into the details of Coinjoin.

Coinjoin Algorithm:

1. Find peers who want to mix
2. Exchange input/output addresses
3. Construct transaction
4. Send it around, collect signatures
Before signing, each peer checks if her output is present
5. Broadcast the transaction

Finding peers. First, a group of peers who all want to mix need to find each other. This can be facilitated by a servers acting as watering-holes, allowing users to connect and grouping them. Unlike centralized mixes, these servers are not in a position to steal users' funds or compromise anonymity.

Exchanging addresses. Once a peer group has formed, the peers must exchange their input and output addresses with each other. It's important for participants to exchange these addresses in such a way that even the other members of the peer group do not know the mapping between input and output addresses. After all, even if you execute a series of coinjoin transactions with supposedly random sets of peers, an adversary might be able to weasel his way into every one of these peer groups by running lots of nodes. To swap addresses in an unlinkable way, we need an anonymous communication protocol. We could use the Tor network, which we looked at earlier, or a special-purpose anonymous routing protocol called a decryption mix-net.

Collecting signatures and denial of service. Once the inputs and outputs have been communicated, one of these users — it doesn't matter who — will then construct the transaction corresponding to these inputs and outputs. The unsigned transaction will then be passed around; each peer will verify that its input and output address are included correctly, and sign.

If all peers follow the protocol, this system works well. Any peer can assemble the transaction and any peer can broadcast the transaction to the network. Two of them could even broadcast it independently; it will be published only once to the block chain, of course. But if one or more of the peers wants to be disruptive, it's easy for them to launch a denial of service attack, preventing the protocol from completing.

In particular, a peer could participate in the first phase of the protocol, providing its input and output addresses, but then refuse to sign in the second phase. Alternately, after signing the transaction, a disruptive peer can try to take the input that it provided to its peers and spend it in some other transaction instead. If the alternate transaction wins the race on the network, it will be confirmed first and the Coinjoin transaction will be rejected as a double spend.

There have been several proposals to prevent denial of service in Coinjoin. One is to impose a cost to participate in the protocol, either via a proof of work (analogous to mining), or by a proof of burn, a technique to provably destroy a small quantity of bitcoins that you own, which we studied in Chapter 3. Alternatively, there are cryptographic ways to identify a non-compliant participant and kick them out of the group. For details, see the Further Reading section.

High-level flows. We mentioned side channels earlier, and we'll now take a closer look at how tricky side channels can be. Let's say Alice receives a very specific amount of bitcoins, say 43.12312 BTC, at a particular address on a weekly basis, perhaps as income. Suppose further that she has a habit of automatically and immediately transferring 5% of that amount to her retirement account, which is another Bitcoin address. We call this transfer pattern a high-level flow. No mixing strategy can effectively hide the fact that there's a relationship between the two addresses in this scenario. Think about the patterns that will be visible on the block chain: the specific amounts and timing are extraordinarily unlikely to occur by chance.

One technique that can help regain unlinkability in the presence of high-level flows is called **merge avoidance**, proposed by Bitcoin developer Mike Hearn. Generally, to make a payment, a user creates a single transaction that combines as many coins as necessary in order to pay the entire amount to a single address. What if she could avoid the need to merge and consequently link all of her inputs? The merge avoidance protocol enables this allowing the receiver of a payment to provide multiple output addresses — as many as necessary. The sender and receiver agree upon a set of denominations to break up the payment into, and carry it out using multiple transactions. The various transactions as well as the associated addresses will be unlinkable to one another.

Merge avoidance mitigates the problem of high-level flows: an adversary might not be able to discern a flow if it is broken up into many smaller flows that aren't linked to each other. It also defeats address clustering techniques that rely on coins being spent jointly in a single transaction.

6.5: Zerocoin and Zerocash

No cryptocurrency anonymity solution has caused as much excitement as Zerocoin and its successor Zerocash. That's both because of the ingenious cryptography that they employ and because of the powerful anonymity that they promise. Whereas all of the anonymity-enhancing technologies that we have seen so far add anonymity on top of the core protocol, **Zerocoin** and **Zerocash** incorporate anonymity at the protocol level. We'll present a high-level view of the protocol here and simplify some details, but you can find references to the original papers in the Further Reading section.

Compatibility. As we'll see, the strong anonymity guarantees of Zerocoin and Zerocash come at a cost: unlike centralized mixing and Coinjoin, these protocols are not compatible with Bitcoin as it stands today. It is technically possible to deploy Zerocoin with a soft fork to Bitcoin, but the practical difficulties are serious enough to make this infeasible.

Cryptographic guarantees. Zerocoin and Zerocash incorporate protocol-level mixing, and the anonymity properties come with cryptographic guarantees. These guarantees are qualitatively better than those of the other mixing technologies that we have discussed. You don't need to trust anybody — mixes, peers, or intermediaries of any kind, or even miners and the consensus protocol — to ensure your privacy. The promise of anonymity relies only on the adversary's computational limits, as with essentially all cryptographic guarantees.

Zerocoin. To explain Zerocoin, we'll first introduce the concept of Basecoin. Basecoin is a Bitcoin-like altcoin, and Zerocoin is an extension of this altcoin. The key feature that provides anonymity is that you can convert basecoins into zerocoins and back again, and when you do that, it breaks the link between the original basecoin and the new basecoin. In this system, Basecoin is the currency that you transact in, and Zerocoin just provides a mechanism to trade your basecoins in for new ones that are unlinkable to the old ones.

You can view each zerocoin you own as a token which you can use to prove that you owned a basecoin and made it unspendable. The proof does not reveal which basecoin you owned, merely that you did own a basecoin. Similar to the way poker chips work, you can then go ahead and redeem your basecoin by presenting this proof to the miners. When we implement these proofs cryptographically, we need to make sure that each proof can be used only once to redeem a basecoin. Otherwise you'd be able to earn basecoins for free by turning a basecoin into a zerocoin and then redeeming it more than once.

Zero-knowledge proofs. The key cryptographic tool we'll use is a zero-knowledge proof, which is a way for somebody to prove a (mathematical) statement without revealing any other information that leads to that statement being true. For example, suppose you've done a lot of work to solve a hash puzzle, and you want to convince someone of this. In other words, you want to prove the statement

$$\text{I know } x \text{ such that } H(x \parallel \langle \text{other known inputs} \rangle) < \langle \text{target} \rangle.$$

You could, of course, do this by revealing x . But a zero-knowledge proof allows you to do this in such a way that the other person is no wiser about the value of x after seeing the proof than they were before.

You can also prove a statement such as "I know x such that $H(x)$ belongs to the following set: $\{...\}$ ". The proof would reveal nothing about x , nor about which element of the set equals $H(x)$. Zerocoin crucially relies on zero-knowledge proofs, and in fact the statements proved in Zerocoin are very similar to this latter example. In this book, we'll treat zero-knowledge proofs as black-boxes. We'll present the properties achieved by zero-knowledge proofs and show where they are necessary in the protocol, but we will not delve into the technical details of how these proofs are implemented. Zero knowledge proofs are a cornerstone of modern cryptography, and form the basis of many protocols. Once again, we refer the motivated reader to the Further Reading section for more detailed treatment.

Minting Zerocoins. Zerocoins come into existence by minting, and anybody can mint a zerocoin. They come in standard denominations. For simplicity, we'll assume that there is only one denomination worth 1.0 zerocoins, and that each zerocoin is worth one basecoin. While anyone can mint a Zerocoin, just minting one doesn't automatically give it any value — you can't get free money. It acquires value only when you put it onto the block chain, and doing that will require giving up one basecoin.

To mint a Zerocoin, you use a cryptographic **commitment**. Recall from chapter 1 that a commitment scheme is the cryptographic analog of sealing a value in an envelope and putting it on a table in everyone's view.



Figure 6.10: Committing to a serial number.

The real world analog of a cryptographic commitment is sealing a value inside an envelope.

Minting a zerocoin is done in three steps:

1. Generate serial number S and a random secret r
2. Compute $H(S, r)$, the commitment to the serial number
3. Publish commitment onto the block chain as shown in Figure 6.11. This burns a basecoin, making it unspendable, and creates a Zerocoin. Keep S and r secret for now.

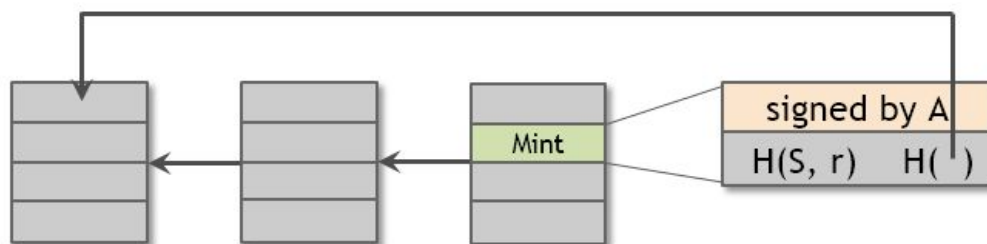


Figure 6.11: Putting a zerocoin on the block chain. To put a zerocoin on the blockchain, you create a special 'mint' transaction whose output 'address' is the cryptographic commitment of the zerocoin's serial number. The input of the mint transaction is a basecoin, which has now been spent in creating the zerocoin. The transaction does *not* reveal the serial number.

To spend a zerocoin and redeem a new basecoin, you need to prove that you previously minted a zerocoin. You could do this by opening your previous commitment, that is, revealing S and r . But this makes the link between your old basecoin and your new basecoin apparent. How can we break the link?

This is where the zero-knowledge proof comes in. At any point, there will be many hash commitments on the block chain — let's call them h_1, h_2, \dots, h_n .

Here are the steps that go into spending a zerocoin with serial number S to redeem a new basecoin:

- Create a special “spend” transaction that contains S , along with a zero-knowledge proof of the statement:
“I know r such that $H(S, r)$ is in the set $\{h_1, h_2, \dots, h_n\}$ ”.
- Miners will verify your zero-knowledge proof which establishes your *ability* to open one of the zerocoin commitments on the block chain, without actually opening it.
- Miners will also check that the serial number S has never been used in any previous spend transaction (since that would be a double-spend).
- The output of your spend transaction will now act as a new basecoin. For the output address, you should use an address that you own.

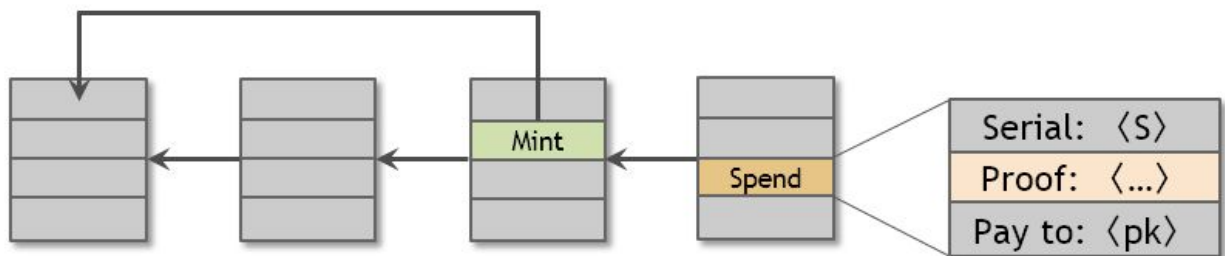


Figure 6.12: Spending a zerocoin. The spend transaction reveals the serial number S committed by the earlier mint transaction, along with a zero-knowledge proof that S corresponds to *some* earlier mint transaction. Unlike a mint transaction (or a normal Bitcoin/basecoin transaction), the spend transaction has no inputs, and hence no signature. Instead the zero-knowledge proof serves to establish its validity.

Once you spend a zerocoin, the serial number becomes public, and you will never be able to redeem this serial number again. And since there is only one serial number for each zerocoin, it means that each zerocoin can only be spent once.

Anonymity. Observe that r is kept secret throughout; neither the mint nor the spend transaction reveals it. That means nobody knows which serial number corresponds to which zerocoin. This is the key concept behind Zerocoin’s anonymity. There is no link on the block chain between the mint transaction that committed a serial number S and the spend transaction that later revealed S to

redeem a basecoin. This is magical-sounding property that is possible through cryptography but we wouldn't get in a physical, envelope-based system. It's as if there are a bunch of sealed envelopes on a table with different serial numbers, and you can prove that a particular serial number is one of them, without having to reveal which one and without having to open any envelopes.

Efficiency. Recall the statement that's proved in a spend transaction:

"I know r such that $H(S, r)$ is in the set $\{h_1, h_2, \dots, h_n\}$ ".

This sounds like it would be horribly inefficient to implement, because the size of the zero-knowledge proofs would grow linearly as n increases, which is the number of zerocoins that have *ever* been minted. Remarkably, Zerocoin manages to make the size of these proofs only logarithmic in n . Note that even though the *statement* to be proved has a linear length, it doesn't need to be included along with the proof. The statement is implicit; it can be inferred by the miners since they know the set of all zerocoins on the block chain. The proof itself can be much shorter. Nevertheless, compared to Bitcoin, Zerocoin still adds quite a sizable overhead.

Zerocash. Zerocash is a different anonymous cryptocurrency that builds on Zerocoin and takes the cryptography to the next level. It uses a cryptographic technique called SNARKs which are a way of making zero-knowledge proofs much more compact and efficient to verify. The upshot is that the efficiency of the system overall gets to a point where it becomes possible to run the whole network without needing a basecoin. All transactions can be done in a zero-knowledge manner. As we saw, Zerocoin supports regular transactions for when you don't need unlinkability, augmented with computationally expensive transactions that are used only for mixing. The mix transactions are of fixed denominations and splitting and merging of values can happen only in Basecoin. In Zerocash, that distinction is gone. The transaction amounts are now inside the commitments and no longer visible on the block chain. The cryptographic proofs ensure that the splitting and merging happens correctly and that users can't create zerocash out of thin air.

The only thing that the ledger records publicly is the existence of these transactions, along with proofs that allow the miners to verify all the properties needed for the correct functioning of the system. Neither addresses nor values are revealed on the block chain at any point. The only users who need to know the amount of a transaction are the sender and the receiver of that particular transaction. The miners don't need to know transaction amounts. Of course, if there is a transaction fee, the miners need to know that fee, but that doesn't really compromise your anonymity.

The ability to run as an entirely untraceable system of transactions puts zerocash in its own category when it comes to anonymity and privacy. Zerocash is immune to the side-channel attacks against mixing because the public ledger no longer contains transaction amounts.

Setting up Zerocash. In terms of its technical properties, Zerocash might sound too good to be true. There is indeed one catch. Zerocash requires "public parameters" to set up the overall system and to kick-start the crypto. Think of these as public keys, except giant ones — over a gigabyte long. That's a problem, but not the most serious one. To generate these public parameters, Zerocash requires *random and secret inputs*. If *anyone* knows these secret inputs, it compromises the security of the

system, because they'd be able to create new zerocoins for themselves without being caught. So these secret inputs must be used once in generating the public parameters and then securely destroyed.

There's an interesting sociological problem here. It's not clear how an entity could set up the system and convince everybody that they have securely destroyed the parameters that were used during the setup. There's been various proposals for how to achieve this, but at the moment we don't have a very clear idea of how to go forward on this.

Putting it all together. Let's look back now and compare the different anonymity solutions that we have seen. We will compare them both in terms of the anonymity properties that they provide, as well as on how deployable they are in practice.

System	Type	Anonymity attacks	Deployability
Bitcoin	Pseudonymous	Transaction graph analysis	Default
Manual mixing	Mix	Transaction graph analysis, bad mixes/peers	Usable today
Chain of mixes or coinjoins	Mix	Side channels, bad mixes/peers	Bitcoin-compatible
Zerocoin	Cryptographic mix	Side channels (possibly)	Altcoin
Zerocash	Untraceable	None	Altcoin, tricky setup

Table 6.13: A comparison of the anonymity technologies presented in this chapter

We start with Bitcoin itself, which is already deployed and is the 'default' system. But it's only pseudonymous and doesn't even aspire to be truly anonymous, and we've seen that powerful transaction graph analysis is possible. We looked at ways to cluster large groups of addresses, and how to sometimes attach real-world identities to those clusters.

The next level of anonymity is using a single mix in a manual way, or doing a Coinjoin by finding peers manually. This obscures the link between input and output but leaves too many potential clues in the transaction graph. Besides, mixes and peers could be malicious, hacked, or coerced into revealing their records. While far from perfect in terms of anonymity, mixing services exist and so this option is usable today.

The third level we looked at is a chain of mixes or Coinjoins. The anonymity improvement comes from the fact that there's less reliance on any single mix or group of peers, and features like standardized chunk sizes and client-side automation that minimize information leaks. Some side channels are still present, and there's also the danger of an adversary who controls or colludes with multiple mixes or peers. Wallets and services that implement a chain of mixes could be deployed and adopted today, to our knowledge a secure mix-chain solution isn't yet readily available.

Next, we saw that Zerocoin bakes cryptography directly into the protocol and brings a mathematical guarantee of anonymity. We think some side channels are still possible, but it's certainly superior to the other mixing-based solutions. However, Zerocoin would have to be launched as an altcoin.

Finally, we looked at Zerocash. Due to its improved efficiency, Zerocash can be run as a fully untraceable — and not just anonymous — cryptocurrency. However, like Zerocoin, Zerocash is not Bitcoin compatible. Worse, it requires a complex setup process which the community is still figuring out how best to accomplish.

We've covered a lot of technology in this chapter. Now let's take a step back. Bitcoin's anonymity (and potential for anonymity) is powerful, and gains in power when combined with other technologies, particularly anonymous communication. As we'll see in the next chapter, this is the potent combination behind the Silk Road and other anonymous online marketplaces.

Despite its power, anonymity is fragile. One mistake can create an unwanted, irreversible link. But anonymity is worth protecting, since it has many good uses in addition to the obvious bad ones. While these moral distinctions are important, we find ourselves unable to express them at a technical level. Anonymity technologies seem to be deeply and inherently morally ambiguous, and as a society we must learn to live with this fact.

Bitcoin anonymity is an active area of technical innovation as well as ethical debate. We still do not know which anonymity system for Bitcoin, if any, is going to become prominent or mainstream. That's a great opportunity for you — whether a developer, a policy maker, or a user — to get involved and make a contribution. Hopefully what you've learned in this chapter has given you the right background to do that.

Further reading

Even more than the topics discussed in previous chapters, anonymity technologies are constantly developing and are an active area of cryptocurrency research. The best way to keep up with the latest in this field is to begin with the papers listed here, and to look for papers that cite them.

The “Fistful of bitcoins” paper on transaction graph analysis:

Meiklejohn, Sarah, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. [A fistful of bitcoins: characterizing payments among men with no names](#). In Proceedings of the 2013 conference on Internet measurement conference, 2013.

A study of mixing technologies and the source of the principles for effective mixing that we discussed:

Bonneau, Joseph, Arvind Narayanan, Andrew Miller, Jeremy Clark, Joshua A. Kroll, and Edward W. Felten. [Mixcoin: Anonymity for Bitcoin with accountable mixes](#). Financial Cryptography 2014.

Coinjoin was presented on the Bitcoin forums by Bitcoin Core developer Greg Maxwell:

Maxwell, Gregory. [CoinJoin: Bitcoin privacy for the real world](#). Bitcoin Forum, 2013.

Zerocoin was developed by cryptographers from Johns Hopkins University. Keep in mind Zerocoin and Zerocash have the most complex cryptography of any scheme we’ve discussed in this book.

Miers, Ian, Christina Garman, Matthew Green, and Aviel D. Rubin. [Zerocoin: Anonymous distributed E-Cash from Bitcoin](#). Proceedings of 2013 IEEE Symposium on Security and Privacy, 2013.

The Zerocoin authors teamed up with other researchers who had developed the SNARK technique. This collaboration resulted in Zerocash:

Ben Sasson, Eli, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. [Zerocash: Decentralized anonymous payments from Bitcoin](#). Proceedings of 2013 IEEE Symposium on Security and Privacy, 2014.

This classic book on cryptography includes a chapter on zero-knowledge proofs:

Goldreich, Oded. Foundations of Cryptography: Volume 1. Cambridge university press, 2007.

The paper that describes the technical design of the anonymous communication network Tor:

Dingledine, Roger, Nick Mathewson, and Paul Syverson. [Tor: The second-generation onion router](#). Naval Research Lab Washington DC, 2004.