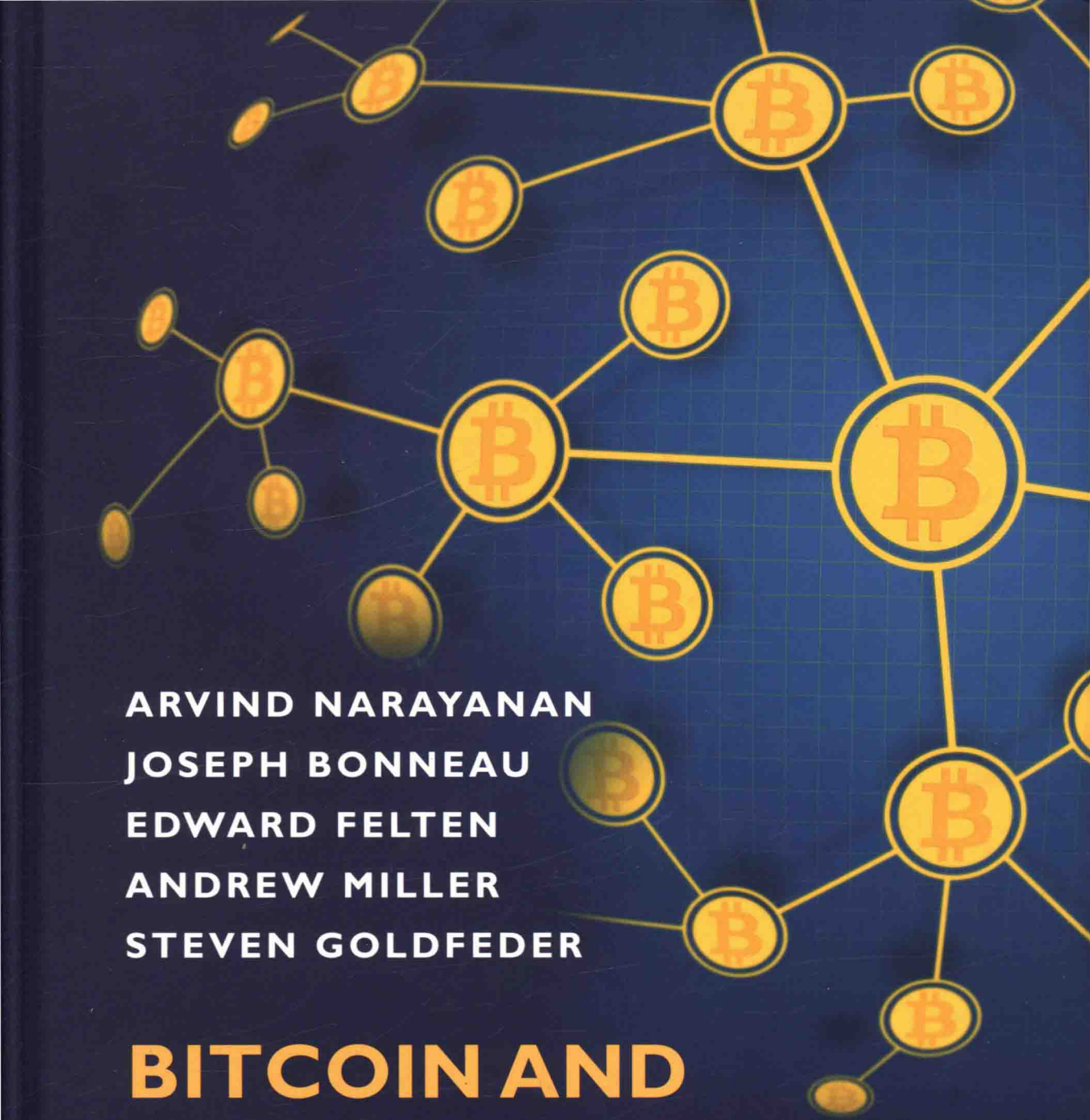

ARVIND NARAYANAN
JOSEPH BONNEAU
EDWARD FELTEN
ANDREW MILLER
STEVEN GOLDFEDER

BITCOIN AND CRYPTOCURRENCY TECHNOLOGIES

A Comprehensive Introduction

COMPUTER SCIENCE

Bitcoin and Cryptocurrency Technologies provides a comprehensive introduction to the revolutionary yet often misunderstood new technologies of digital currency. Whether you are a student, software developer, tech entrepreneur, or researcher in computer science, this authoritative and self-contained book tells you everything you need to know about the new global money for the Internet age.

How do Bitcoin and its block chain actually work? How secure are your bitcoins? How anonymous are their users? Can cryptocurrencies be regulated? These are some of the many questions this book answers. It begins by tracing the history and development of Bitcoin and cryptocurrencies, and then gives the conceptual and practical foundations you need to engineer secure software that interacts with the Bitcoin network as well as to integrate ideas from Bitcoin into your own projects. Topics include decentralization, mining, the politics of Bitcoin, altcoins and the cryptocurrency ecosystem, the future of Bitcoin, and more.

- An essential introduction to the new technologies of digital currency
- Covers the history and mechanics of Bitcoin and the block chain, security, decentralization, anonymity, politics and regulation, altcoins, and much more
- Features an accompanying website that includes instructional videos for each chapter, homework problems, programming assignments, and lecture slides
- Also suitable for use with the authors' Coursera online course
- Electronic solutions manual (available only to professors)

ARVIND NARAYANAN is assistant professor of computer science at Princeton University. JOSEPH BONNEAU is a postdoctoral researcher at the Applied Cryptography Group at Stanford University. EDWARD FELTEN is director of Princeton's Center for Information Technology Policy. ANDREW MILLER is a PhD student in computer science at the University of Maryland. STEVEN GOLDFEDER is a PhD student in computer science at Princeton.

"Block chain technology is set to disrupt many different industries. If you want to get up to speed on this fast-moving technology, this book should be your first stop."
—Campbell R. Harvey, Duke University

"Among this book's many features are lots of nice, concrete examples and pleasant anecdotes, as well as a highly readable and enjoyable history of cryptocurrencies. Strongly recommended."
—Tyler Moore, University of Tulsa

Cover art courtesy of Shutterstock

PRINCETON
press.princeton.edu

ISBN: 978-0-691-17169-2
90000
9 780691 171692

NARAYANAN, BONNEAU
FELTEN, MILLER
AND GOLDFEDER

BITCOIN AND CRYPTOCURRENCY TECHNOLOGIES

PRINCETON

BITCOIN AND CRYPTOCURRENCY TECHNOLOGIES

A Comprehensive Introduction

ARVIND NARAYANAN, JOSEPH BONNEAU, EDWARD FELTEN,
ANDREW MILLER, AND STEVEN GOLDFEDER

PRINCETON UNIVERSITY PRESS
Princeton and Oxford

Published by Princeton University Press, 41 William Street, Princeton, New Jersey 08540
In the United Kingdom: Princeton University Press, 6 Oxford Street, Woodstock, Oxfordshire OX20 1TR

press.princeton.edu
Cover image: Courtesy of Shutterstock

ISBN 978-0-691-17169-2

Library of Congress Cataloging-in-Publication Data

Names: Narayanan, Arvind, author.
Title: Bitcoin and cryptocurrency technologies : a comprehensive introduction / Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder.
Description: Princeton : Princeton University Press, [2016] | Includes bibliographical references and index.
Identifiers: LCCN 2016014802 | ISBN 9780691171692 (hardcover : alk. paper)
Subjects: LCSH: Bitcoin. | Electronic funds transfers. | Cryptography. | Money.
Classification: LCC HG1710 .N35 2016 | DDC 332.1/78—dc23 LC record available at https://lccn.loc.gov/2016014802

British Library Cataloging-in-Publication Data is available

This book has been composed in Charis

Printed on acid-free paper. ∞

Printed in the United States of America

5 7 9 10 8 6 4

Preface

There's a lot of excitement about Bitcoin and cryptocurrencies. Optimists claim that Bitcoin will fundamentally alter payments, economics, and even politics around the world. Pessimists claim Bitcoin is inherently broken and will suffer an inevitable and spectacular collapse.

Underlying these differing views is significant confusion about what Bitcoin is and how it works. We wrote this book to help cut through the hype and get to the core of what makes Bitcoin unique. To really understand what is special about Bitcoin, we need to understand how it works at a technical level. Bitcoin truly is a new technology, and we can only get so far by explaining it through simple analogies to past technologies.

We assume that you have a basic understanding of computer science—how computers work, data structures and algorithms, and some programming experience. If you're an undergraduate or graduate student of computer science, a software developer, an entrepreneur, or a technology hobbyist, this textbook is for you.

In this book, we address the important questions about Bitcoin. How does Bitcoin work? What makes it different? How secure are your bitcoins? How anonymous are Bitcoin users? What applications can we build using Bitcoin as a platform? Can cryptocurrencies be regulated? If we were designing a new cryptocurrency today, what would we change? What might the future hold?

After reading this book, you'll know everything you need to be able to separate fact from fiction when reading claims about Bitcoin and other cryptocurrencies. You'll have the conceptual foundations you need to engineer secure software that interacts with the Bitcoin network. And you'll be able to integrate ideas from Bitcoin into your own projects.

The online supplementary materials for this book include a series of homework questions to help you understand each chapter at a deeper level. In addition, there is a series of programming assignments in which you'll implement various components of Bitcoin in simplified models. Most of the material of this book is also available as a series of video lectures on Coursera. (A link to the supplementary materials can be found at http://press.princeton.edu/titles/10908.html.) You should also supplement your learning with information you can find online, including the Bitcoin wiki, forums, and research papers, and by interacting with your peers and the Bitcoin community.

BITCOIN AND CRYPTOCURRENCY TECHNOLOGIES

Foreword
THE LONG ROAD TO BITCOIN

JEREMY CLARK

The path to Bitcoin is littered with the corpses of failed attempts. I've compiled a list of about a hundred cryptographic payment systems, both e-cash- and credit-card-based technologies, that are notable in some way (Table 0.1). Some are academic proposals that have been widely cited, while others are actual systems that were deployed and tested. Of all the names on this list, there's probably only one that you recognize—PayPal. And PayPal survived only because it quickly pivoted away from its original idea of cryptographic payments on handheld devices!

There's a lot to learn from this history. Where do the ideas in Bitcoin come from? Why do some technologies survive while many others die? What does it take for complex technical innovations to be successfully commercialized? If nothing else, this story will give you an appreciation of how remarkable it is that we finally have a real, working payment mechanism that's native to the Internet.

TRADITIONAL FINANCIAL ARRANGEMENTS

If you imagine a world without governments or currency, one system that could still work for acquiring goods is barter. Suppose Alice wants a tool, and Bob wants medicine. If each of them happen to have what the other person needs, then they can swap and both satisfy their needs.

But suppose Alice has food that she's willing to trade for a tool, while Bob, who has a tool, doesn't have any need for food. He wants medicine instead. Alice and Bob can't trade with each other, but if there's a third person, Carol, who has medicine that she's willing to trade for food, then it becomes possible to arrange a three-way swap where everyone gets what they need.

The drawback, of course, is coordination—arranging a group of people, whose needs and wants align, in the same place at the same time. Two systems emerged to solve coordination: credit and cash. Historians, anthropologists, and economists debate which of the two developed first, but that's immaterial for our purposes.

TABLE 0.1. NOTABLE ELECTRONIC PAYMENT SYSTEMS AND PROPOSALS

ACC	CyberCents	IKP	MPTP	Proton
Agora	CyberCoin	IMB-MP	Net900	Redi-Charge
AIMP	CyberGold	InterCoin	NetBill	S/PAY
Allopass	DigiGold	Ipin	NetCard	Sandia Lab E-Cash
b-money	Digital Silk Road	Javien	NetCash	Secure Courier
BankNet	e-Comm	Karma	NetCheque	Semopo
Bitbit	E-Gold	LotteryTickets	NetFare	SET
Bitgold	Ecash	Lucre	No3rd	SET2Go
Bitpass	eCharge	MagicMoney	One Click Charge	SubScrip
C-SET	eCoin	Mandate	PayMe	Trivnet
CAFÉ	Edd	MicroMint	PayNet	TUB
Checkfree	eVend	Micromoney	PayPal	Twitpay
ClickandBuy	First Virtual	MilliCent	PaySafeCard	VeriFone
ClickShare	FSTC Electronic Check	Mini-Pay	PayTrust	VisaCash
CommerceNet	Geldkarte	Minitix	PayWord	Wallie
CommercePOINT	Globe Left	MobileMoney	Peppercoin	Way2Pay
CommerceSTAGE	Hashcash	Mojo	PhoneTicks	WorldPay
Cybank	HINDE	Mollie	Playspan	X-Pay
CyberCash	iBill	Mondex	Polling	

In a credit-based system, Alice and Bob would be able to trade with each other in the example above. Bob would give Alice the tool, and Bob gets a favor that's owed to him. In other words, Alice has a debt that she needs to settle with Bob some time in the future. Alice's material needs are now satisfied, but she has a debt that she'd like to cancel, so that's her new "want." If Alice encounters Carol in the future, Alice can trade her food for Carol's medicine, then go back to Bob with the medicine and cancel the debt.

In contrast, in a cash-based system, Alice would buy the tool from Bob. Later, she might sell her food to Carol, and Carol can sell her medicine to Bob, completing the cycle. These trades can happen in any order, provided that the buyer in each transaction has cash on hand. In the end, of course, it's as if no money ever changed hands.

Neither system is clearly superior. A cash-based system needs to be bootstrapped with some initial allocation of cash, without which no trades can occur. A credit-based system doesn't need bootstrapping, but the drawback is that anyone who's owed a debt is taking on some risk. There's a chance that the other person never settles the debt.

Cash also allows us to be precise about how much something is worth. If you're bartering, it's hard to say whether a tool is worth more than medicine or medicine is worth more than food. Cash lets us use numbers to talk about value. That's why we use a blended system today—even when we're using credit, we measure debt in the amount of cash it would take to settle it.

These ideas come up in many contexts, especially in online systems, where users trade virtual goods of some kind. For example, peer-to-peer file-sharing networks must deal with the problem of freeloaders, that is, users who download files without sharing in turn. While swapping files might work, there is also the issue of coordination: finding the perfect person who has exactly the file you want and wants exactly the file you have. In projects like MojoNation and academic proposals like Karma, users are given some initial allocation of virtual cash that they must spend to receive a file and earn when they send a copy of a file to another user. A network of nodes (centralized for MojoNation and decentralized for Karma) keeps track of users' balances, and MojoNation explored implementing an exchange service between their internal currency and traditional currency. While MojoNation did not survive long enough to implement such an exchange, it became the intellectual ancestor of some protocols used today: BitTorrent and Tahoe-LAFS.

THE TROUBLE WITH CREDIT CARDS ONLINE

Credit and cash are fundamental ideas, to the point that we can sort the multitude of electronic payment methods into two piles. Bitcoin is obviously in the "cash" pile, but let's look at the other one first.

Credit card transactions are the dominant payment method used on the web today. If you've ever bought something from an online seller such as Amazon, you know how the arrangement goes. You type in your credit card details, you send it to Amazon, and then Amazon takes these credit card details and talks to a financial system involving processors, banks, credit card companies, and other intermediaries.

In contrast, if you use something like PayPal, what you see is an intermediary architecture. A company sits between you and the seller, so you send your credit card details to this intermediary, which approves the transaction and notifies the seller. The intermediary will settle its balance with the seller at the end of each day.

What you gain from this architecture is that you don't have to give the seller your credit card details, which can be a security risk. You might not even have to give the seller your identity, which would improve your privacy as well. The downside is that you lose the simplicity of interacting directly with the seller. Both you and the seller might have to have an account with the same intermediary.

Today most of us are comfortable with giving out our credit card information when shopping online, or at least we've grudgingly accepted it. We're also used to companies collecting data about our online shopping and browsing activities. But in the 1990s, the web was new, standards for protocol-level encryption were just emerging, and these concerns made consumers deeply uncertain and hesitant. In particular, it was considered crazy to hand over your credit card details to online vendors of unknown repute over an insecure channel. This environment generated a lot of interest in the intermediary architecture.

A company called FirstVirtual was an early payment intermediary, founded in 1994. Incidentally, they were one of the first companies to set up a purely virtual office with employees spread across the country and communicating over the Internet—hence the name.

FirstVirtual's proposed system was a little like PayPal's current system but preceded it by many years. As a user, you'd enroll with them and provide your credit card details. If you wanted to buy something from a seller, the seller would contact FirstVirtual with the details of the requested payment, FirstVirtual would confirm these details with you, and if you approved, your credit card would be billed. But two details are interesting. First, all of this communication happened over email; web browsers back in the day were just beginning to universally support encryption protocols like HTTPS, and the multiparty nature of payment protocol added other complexities. (Other intermediaries took the approach of encoding information into URLs or using a custom encryption protocol on top of HTTP.) Second, the customer would have 90 days to dispute the charge, and the merchant would receive the money only after those 3 months! With today's systems, the merchant does get paid immediately, but there still is the risk that the customer will file a chargeback or dispute the credit card statement. If that happens, the merchant will have to return the payment to the credit card company.

In the mid-1990s, a competing approach to the intermediary architecture was developed, which we'll call the SET architecture. SET also avoids the need for customers to send credit card information to merchants, but it additionally avoids the user having to enroll with the intermediary. In SET, when you are ready to make a purchase, your browser passes your view of the transaction details to a shopping application on your computer. The application encrypts it together with your credit card details in such a way that only the intermediary can decrypt it, and no one else can (including the seller). Having encrypted your data in this way, you can send it to the seller knowing that it's secure. The seller blindly forwards the encrypted data to the intermediary—along with their own view of the transaction details. The intermediary decrypts your data and approves the transaction only if your view matches the seller's view.

SET was a standard developed by Visa and MasterCard, together with many technology heavyweights of the day: Netscape, IBM, Microsoft, Verisign, and RSA. It was an umbrella specification that unified several existing proposals.

One company that implemented SET was CyberCash. It was an interesting company in many ways. In addition to credit card payment processing, they had a digital cash product called CyberCoin. This was a micropayment system—intended for small payments, such as paying a few cents to read an online newspaper article. That meant you'd probably never have more than $10 in your CyberCoin account at any time. Yet, amusingly, they were able to get U.S. government (FDIC) insurance for each account for up to $100,000.

There's more. Back when CyberCash operated, there was a misguided—and now abandoned—U.S. government restriction on the export of cryptography, which was

considered a weapon. That meant software that incorporated meaningful encryption couldn't be offered for download to users in other countries. However, CyberCash was able to get a special exemption for their software from the Department of State. The government's argument was that extracting the encryption technology out of CyberCash's software would be harder than writing the crypto from scratch.

Finally, CyberCash has the dubious distinction of being one of the few companies affected by the Y2K bug—it caused their payment processing software to double-bill some customers. They later went bankrupt in 2001. Their intellectual property was acquired by Verisign, which then turned around and sold it to PayPal, where it lives today.

Why didn't SET work? The fundamental problem has to do with certificates. A certificate is a way to securely associate a cryptographic identity, that is, a public key, with a real-life identity. It's what a website needs to obtain—from companies like Verisign, which are called "certification authorities"—to be identified as secure in your browser (typically indicated by a lock icon). Putting security before usability, CyberCash and SET decided that not only would processors and merchants in their system have to get certificates, but all users also would have to get one as well. Obtaining a certificate is about as pleasant as doing your taxes, so the system was a disaster. Over the decades, mainstream users have given a firm and collective "no" to any system that requires end-user certificates, and such proposals have now been relegated to academic papers. Bitcoin deftly sidesteps this hairy problem by avoiding real-life identities altogether. In Bitcoin, public keys themselves are the identities by which users are known, as discussed in Chapter 1.

In the mid-1990s, when SET was being standardized, the World Wide Web Consortium was also looking at standardizing financial payments. They wanted to do it by extending the HTTP protocol instead, so that users wouldn't need extra software for transactions—they could just use their browsers. In fact, the Consortium had a very general proposal for how you might extend the protocol, and one of the use cases that they had was handling payments. This never happened—the whole extension framework was never deployed in any browsers. In 2015, almost two decades later, the Consortium announced that it wanted to take another crack at it, and that Bitcoin would be part of that standardization this time around. Given all the past failures, however, I won't be holding my breath.

FROM CREDIT TO (CRYPTO) CASH

Now let's turn to cash. I compared cash and credit earlier, and noted that a cash system needs to be bootstrapped, but the benefit is that it avoids the possibility of a buyer defaulting on her debt. Cash offers two additional advantages. The first is better anonymity. Since your credit card is issued in your name, the bank can track all your spending. But when you pay in cash, the bank doesn't come into the picture, and the other party

doesn't need to know who you are. Second, cash can enable offline transactions where there's no need to phone home to a third party to get the transaction approved. Maybe the seller later uses a third party like a bank to deposit the cash, but that's much less of a hassle.

Bitcoin doesn't quite offer these two properties, but it comes close enough to be useful. Bitcoin is not anonymous to the same level as cash is. You don't need to use your real identity to pay in Bitcoin, but it's possible that your transactions can be tied together using clever algorithms based on the public ledger of transactions and then further linked to your identity if you're not careful. Chapter 6 gets into the messy but fascinating details behind Bitcoin anonymity.

Bitcoin doesn't work in a fully offline way either. The good news is it doesn't require a central server, instead relying on a peer-to-peer network, which is resilient in the way that the Internet itself is. Chapter 3 looks at tricks like "green addresses" and micropayments, which allow offline payments in certain situations or under certain assumptions.

The earliest ideas about applying cryptography to cash came from David Chaum in 1983. Consider this concept by means of a physical analogy. Let's say I start giving out pieces of paper that say: "The bearer of this note may redeem it for one dollar by presenting it to me" with my signature attached. If people trust that I'll keep my promise and consider my signature unforgeable, they can pass around these pieces of paper just like banknotes. In fact, banknotes themselves got their start as promissory notes issued by commercial banks. It's only in fairly recent history that governments stepped in to centralize the money supply and legally require banks to redeem notes.

I can do the same thing electronically with digital signatures, but that runs into the annoying "double-spending" problem—if you receive a piece of data representing a unit of virtual cash, you can make two (or more) copies of it and pass it on to different people. To stick with this analogy, let's stretch it a little bit and assume that people can make perfect copies and we have no way to tell copies from the original. Can we solve double spending in this world?

Here's a possible solution: I put unique serial numbers on each note I give out. When you receive such a note from someone, you check my signature, but you also call me on the phone to ask whether a note with that serial number has already been spent. Hopefully I'll say no, in which case you accept the note. I'll record the serial number as spent in my ledger, and if you try to spend that note, it won't work, because the recipient will call me and I'll tell them the note has already been spent. What you'll need to do instead is to periodically bring me all the notes you've received, and I'll issue you the same number of new notes with fresh serial numbers.

This works. It's cumbersome in real life, but straightforward digitally, provided I've set up a server to do the signing and recordkeeping of serial numbers. The only problem is that this isn't really cash anymore, because it's not anonymous—when I issue a note to you, I can record the serial number along with your identity, and I can do the same

when someone else later redeems it. That means I can keep track of all the places where you're spending your money.

Here is where Chaum's innovation comes in. He figured out how to both keep the system anonymous and prevent double spending by inventing the digital equivalent of the following procedure: when I issue a new note to you, *you* pick the serial number. You write it down on the piece of paper, but cover it so that I can't see it. Then I'll sign it, still unable to see the serial number. This is called a "blind signature" in cryptography. It'll be in your interest to pick a long, random serial number to ensure that it will most likely be unique. I don't have to worry that you'll pick a serial number that's already been picked—you only shoot yourself in the foot by doing so and end up with a note that can't be spent.

This was the first serious digital cash proposal. It works, but it still requires a server run by a central authority, such as a bank, and for everyone to trust that entity. Moreover, every transaction needs the participation of this server to be completed. If the server goes down temporarily, payments grind to a halt. A few years later, in 1988, Chaum in collaboration with two other cryptographers, Amos Fiat and Moni Naor, proposed *offline* electronic cash. At first sight, this might seem impossible: if you try to spend the same digital note or coin at two different shops, how can they possibly stop this double spend unless they're both connected to the same payment network or central entity?

The clever idea is to stop worrying about preventing double spending and focus on detecting it, after the fact, when the merchant reconnects to the bank server. After all, this approach is why you're able to use your credit card on an airplane even if there is no network connection up in the skies. The transaction processing happens later, when the airline is able to reconnect to the network. If your card is denied, you'll owe the airline (or your bank) money. If you think about it, quite a bit of traditional finance is based on the idea of detecting an error or loss, followed by attempting to recover the money or punish the perpetrator. If you write someone a personal check, they have no guarantee that the money is actually in your account, but they can come after you if the check bounces. Conceivably, if an offline electronic cash system were widely adopted, the legal system would come to recognize double spending as a crime.

Chaum, Fiat, and Naor's idea for detecting double spending was an intricate cryptographic dance. At a high level, what it achieved was this: every digital coin issued to you encodes your identity, but in such a way that no one except you—not even the bank—can decode it. Every time you spend your coin, the recipient will require you to decode a random subset of the encoding, and they'll keep a record of this. This decoding isn't enough to allow them to determine your identity. But if you ever double spend a coin, eventually both recipients will go to the bank to redeem their notes, and when they do this, the bank can put the two pieces of information together to decode your identity completely, with an overwhelmingly high probability.

You might wonder whether someone can frame you as a double spender in this sys-

tem. Suppose you spend a coin with me, and then I turn around and try to double spend it (without redeeming it with the bank and getting a new coin with my identity encoded). This won't work—the new recipient will ask me to decode a random subset, which will almost certainly not be the same as the subset you decoded for me, so I won't be able to comply with their decoding request.

Over the years, many cryptographers have looked at this construction and improved it in various ways. In the Chaum-Fiat-Naor scheme, if a coin is worth $100, and you wanted to buy something that cost only $75, say, there's no way to split that coin into $75 and $25 coins. All you could do is go back to the bank, cash in the $100 coin, and ask for a $75 coin and a $25 coin. But a 1991 paper by Tatsuaki Okamoto and Kazuo Ohta uses Merkle trees to create a system that does allow you to subdivide your coins. Merkle trees would show up in Bitcoin as well, and we'll meet them in Chapter 1. The Chaum-Fiat-Naor scheme also leaves a lot of room for improvements in efficiency. In particular, the application of something called "zero-knowledge proofs" to this scheme (most notably by Stefan Brands in the 1990s, and Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya in 2005) was very fruitful—zero-knowledge proofs have also been applied to Bitcoin, as discussed in Chapter 6.

But back to Chaum: he took his ideas and commercialized them. He formed a company in 1989 called "DigiCash," probably the earliest company that tried to solve the problem of online payments. They had about a 5-year head start on other companies like FirstVirtual and CyberCash, just discussed. The actual cash in DigiCash's system was called "ecash," and they had another system called "cyberbucks." Some banks actually implemented it—a few in the United States and at least one in Finland. This was in the 1990s, long before Bitcoin, which might come as a surprise to some Bitcoin enthusiasts who view banks as tech-phobic, anti-innovative behemoths.

Ecash is based on Chaum's protocols. Clients are anonymous, so banks can't trace how the former are spending their money. But merchants in ecash aren't anonymous. They have to return coins as soon as they receive them, so the bank knows how much they're making, at what times, and so on.

When you want to send money, you'd click on a link provided by the recipient that takes you to the DigiCash website. That would then open a reverse web connection back to your computer. That means your computer had to have the ability to accept incoming connections and act as a server. You'd have to have your own IP address, and your Internet service provider would have to allow incoming connections. If the connection was successful, then the ecash software would launch on your computer, and you'd be able to approve the transaction and send the money.

Chaum took out several patents on DigiCash technology, in particular on the blind-signature scheme that it used. His action was controversial, and it stopped other people from developing ecash systems that used the same protocol. But a group of cryptographers who hung out on what was called the "cypherpunks" mailing list wanted an alternative. Cypherpunks was the predecessor to the mailing list where Satoshi Nakamoto