Businessweek

Oct 25, 2022, 05:00 AM

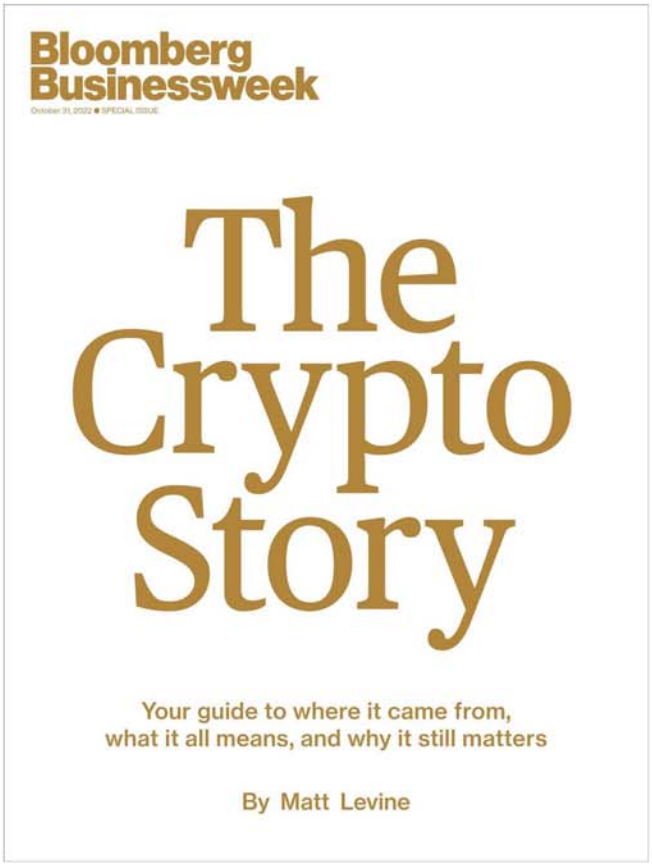
The Crypto Story

Where it came from, what it all means, and why it still matters.

By Matt Levine

Share this article:

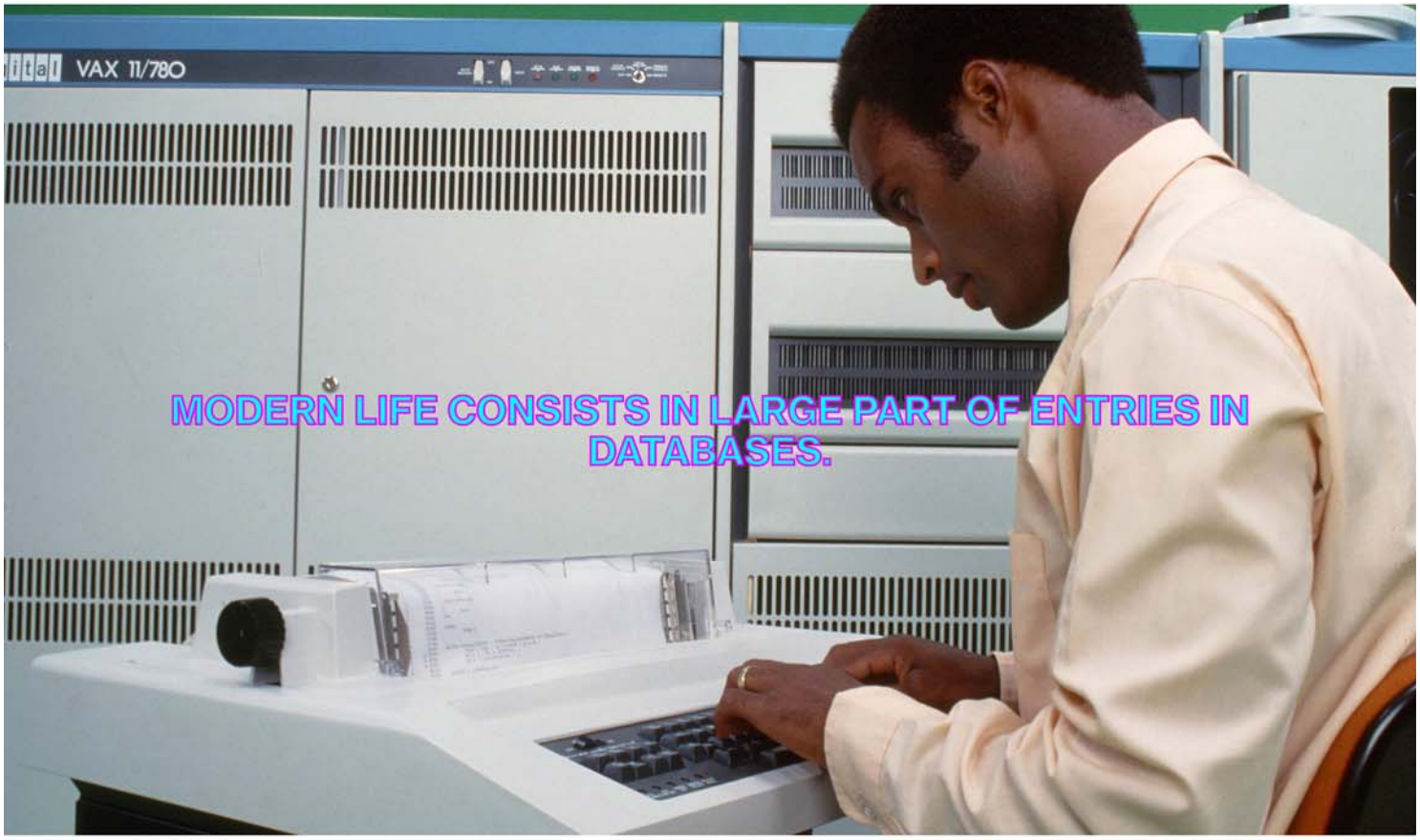
There was a moment not so long ago when I thought, "What if I've had this crypto thing all wrong?" I'm a doubting normie who, if I'm being honest, hasn't always understood this alternate universe that's been percolating and expanding for more than a decade now. If you're a disciple, this new dimension is the future. If you're a skeptic, this upside-down world is just a modern Ponzi scheme that's going to end badly—and the recent "crypto winter" is evidence of its long-overdue ending. But crypto has dug itself into finance, into technology, and into our heads. And if crypto isn't going away, we'd better attempt to understand it. Which is why we asked the finest finance writer around, Matt Levine of Bloomberg Opinion, to write a cover-to-cover issue of *Bloomberg Businessweek*, something a single author has done only one other time in the magazine's 93-year history ("What *Is* Code?," by Paul Ford). What follows is his brilliant explanation of what this maddening, often absurd, and always fascinating technology means, and where it might go. —*Joel Weber, Editor,* Bloomberg Businessweek



Featured in *Bloomberg Businessweek*, Oct. 31, 2022. Subscribe now.

I
Ledgers, Bitcoin, Blockchains

A.
Life in Databases



If you have money, what you have is an entry in your bank's database saying how much money you have. If you have a share of stock, what you have is generally an entry on a list—kept by the company or, more likely, some central intermediary[1]—of who owns stock.

1 These intermediaries include the Depository Trust & Clearing Corp., which owns most of the shares of most US companies on behalf of everyone else. If you own stock, what you have is an entry on DTCC's list entitling you to some of the shares DTCC holds, and it has an entry on a company's list of how many shares it owns.

If you own a house, things are slightly different. There's a house involved. But your ownership of that house is probably written down in some database; in the US this often means there's a record of you buying the house—your title—in a filing cabinet in the basement of some county clerk's office. (It's not a very *good* database.) In many ways the important thing here is the house: You have a key to the front door; your stuff is there; your neighbors will be unsurprised to see you leaving the house in the morning and would be surprised to see someone else coming back in. But in many other ways the important thing is the entry in the database. A bank will want to make sure you have the title before giving you a mortgage; a buyer will want to do the proper procedures to that record before paying you for the house. The key will not suffice.

Lots of other stuff. Much of modern life occurs online. It's not quite true that your social life and your career and your reputation consist of entries in the databases of Meta Platforms and Google and Microsoft, but it's not quite false, either.

Some of this stuff has to do with computers. It's far more convenient for the money to be computer entries than sacks of gold or even paper bills.

Some of it is deeper than that, though. What *could* it mean to own a house? One possibility is the state of nature: Owning a house means 1) you're in the house, and 2) if someone else tries to move in, you're bigger than them, so you can kick them out. But if they're bigger than you, now they own the house.

Another possibility is what you might think of as a village. Owning a house means you live there and your neighbors all know you live there, and if someone else tries to move in, then you and your neighbors combined are bigger than them. Homeownership is mediated socially by a high-trust network of peers.

Neighborhoods, where everybody knows your name.

A third possibility is what you might think of as a government. Owning a house means the government thinks you own the house, and if someone else tries to move in, then the government will kick them out.[2] Homeownership is mediated socially by a government. The database is a way for the government to keep track. You don't have to trust any particular person; you have to trust the rule of law.

2 You don't need to live there, because the government's knowledge is sufficient. You can rent out the house: Someone else can move in with your permission. If you revoke the permission, you can go to the government, and it will—subject to landlord-tenant law, etc.—kick the person out.

Money is a bit like that, too. Sacks of gold are a fairly straightforward form of it, but they're heavy. A system in which your trusted banker holds on to your sacks for you and writes you letters of credit, and you can draw on those letters at branches of the bank run by your banker's cousin—that's pretty good, though it relies on trust between you and the banker, as well as the banker and the banker's cousin. A system of impersonal banking in which the tellers are strangers and you probably use an ATM anyway requires trust in *the system*, trust that the banks are constrained by government regulation or reputation or market forces and so will behave properly.

Saying that modern life is lived in databases means, most of all, that modern life involves a lot of trust.

Sometimes this is because we know them and consider them to be trustworthy. More often it means we have an abstract sense of trust in the broader system, the system of laws and databases and trust itself. We assume that we can trust the systems we use, because doing so makes life much easier than *not* trusting them and because that assumption mostly works out. It's a towering and underappreciated achievement of modernity that we mostly do trust the database-keepers,




and that they mostly are trustworthy.

B.
What If You Don't Like It?

i. Distrust

But we don't always trust them, and they're not always trustworthy.

Sometimes they just aren't. There are banks you can't trust to hold your money for you and places where you can't trust the rule of law to regulate them. There are governments you can't trust not to seize your money from the banks, or falsify election results, or change the property registry and take your house. There are social media companies you can't trust not to freeze your account arbitrarily. Most people in the US, most days, live in a high-trust world, where it's easy and reasonable to trust that the intermediaries who run the databases that shape our lives will behave properly. But not everyone everywhere lives like that.

Even in the US, trust can be fragile. The 2008 financial crisis caused huge and lasting damage to a lot of people's trust in the banking system. People trusted banks to do nice, safe, socially productive things, and it turned out they were doing wild, risky things that caused an economic crisis. After that it became harder for many people to trust banks to hold their savings.

Also, though, you might have a philosophical objection to trust. Even if your bank has an absolutely unblemished record of keeping track of your money, that might not be good enough for you. Your bank is, to you, a black box. "How do I know you'll give me my money back?" you could ask the bank. And the bank will say things like "Here are our audited financial statements" and "We are regulated by the Federal Reserve and insured by the Federal Deposit Insurance Corp." and "We have never *not* given back anyone's money." And you'll say, "Yes, yes, that's all fine, but how do I *know*?" You don't. Trust is built into the system, a prerequisite. You might want proof.[3]

3 This is probably a modern desire, or at least a desire that's more intense and easier to satisfy in modern times. In a world without the internet, without Wikipedia, without links, without open-source software, etc., you had to take a million facts on faith every day; what were you going to do, look them all up?

Can you name this bank? Doesn't matter, it's still a black box!

ii. Composability

Even if you're generally cool with trusting the keepers of modern databases, you might have a more technical objection. These databases aren't always very good. Lots of the banking system is written in a very old computer language called Cobol; in the US people still frequently make payments—electronic transfers between electronic databases of money—by writing paper checks and putting them in the mail. US stock trades take two business days to settle: If I buy stock from you on a Monday, you deliver the stock (and I pay you) on Wednesday. This isn't because your broker has to put stock certificates in a sack and bring them over to my broker's office, while my broker puts dollar bills in a sack and brings them over to your broker's office, but because the actual process is a descendant of that. It's slow and manual and sometimes gets messed up; lots of stock trades "fail."

Subscribe and listen to "The Crypto Story" in parts, delivered every Sunday.

Don't even get me started on the property registry. If you buy a house, you have to go to a ceremony—a "closing"—where a bunch of people with jobs like "title company lawyer" mutter incantations that let you own the house. It can take hours.

If your model of how a database should work comes from modern computers, the hours of incantations seem insane. "There should be an API," you might think: There should be an application programming interface allowing each of these databases to interact with the others. If your bank is thinking about giving you a mortgage, it should be able to query the property database automatically and find out that you own your house, rather than send a lawyer to the county clerk's office. And it should be able to query the Department of Motor Vehicles registry automatically

and get your driver's license for identification purposes, and query your brokerage account automatically and examine your assets.

Modern life consists of entries in databases:

What if we updated them?

What if we rewrote all the databases from scratch, in modern computer languages using modern software engineering principles, with the goal of making them interact with one another seamlessly?

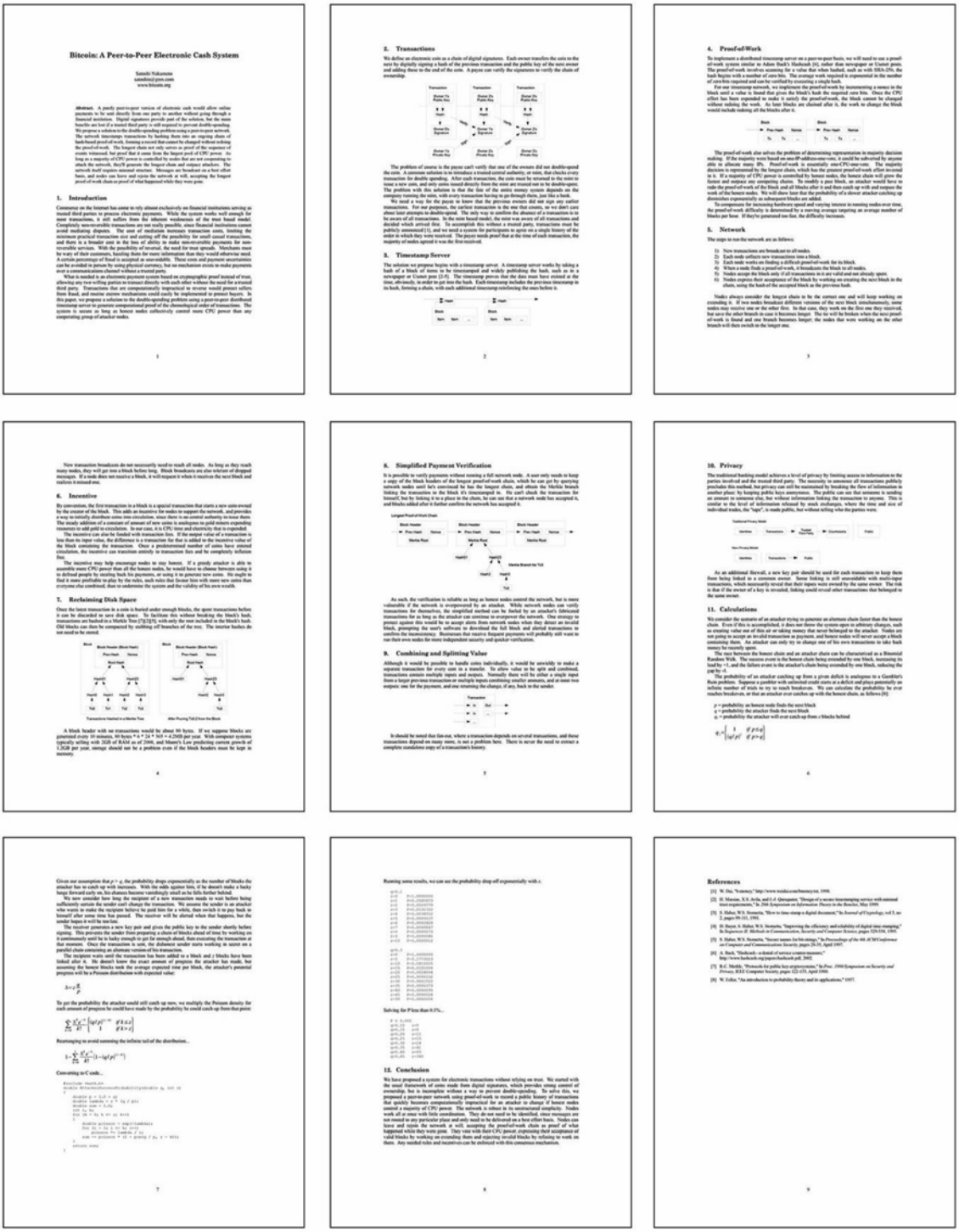
If you did that, it would be almost like having one database, the database of life: I could send you money in exchange for your house, or you could send me social reputation in exchange for my participation in an online class, or whatever, all in the same computer system.

That would be convenient and powerful, but it would also be scary. It would put even *more* pressure on trust. Whoever runs that one database would, in a sense, run the world. Whom could you trust to do that?

C.
Digital Cash

In 2008, Satoshi Nakamoto published a method for everyone to run a database, thus inventing "crypto."

Well, I'm not sure that's what Satoshi thought he was doing. Most immediately he was inventing *Bitcoin: A Peer-to-Peer Electronic Cash System*, which is the title of his famous white paper.



Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi's white paper.

What Satoshi said he'd invented was a sort of cash for internet transactions, "an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party." If I want to buy something from you for digital cash—Bitcoin—I just send you the Bitcoin and you send me the thing; no "trusted third party" such as a bank is involved.

When I put it like that, it sounds as if Satoshi invented a system in which I can send you Bitcoin and *nobody* else is involved. What he actually invented was a system in which *lots* of other people are involved.

i. Digression: What are you even reading? Why are you reading it? Why am I writing it?

Hi! I'm Matt. I'm a former lawyer and investment banker. Now I'm a columnist at Bloomberg Opinion. In my day job, I write about finance. I like finance. It's fun to write about. It's a peculiar way of looking at the world, a series of puzzles, a set of structures that people have imposed on economic reality. Often those structures are arcane and off-putting, and it's satisfying to understand what they're up to. Everything in finance is accreted on top of a lot of other things in finance. Everything is weird and counterintuitive, and you often have to have a sense of financial history and market practice to understand why anyone is doing any of the things they're doing.

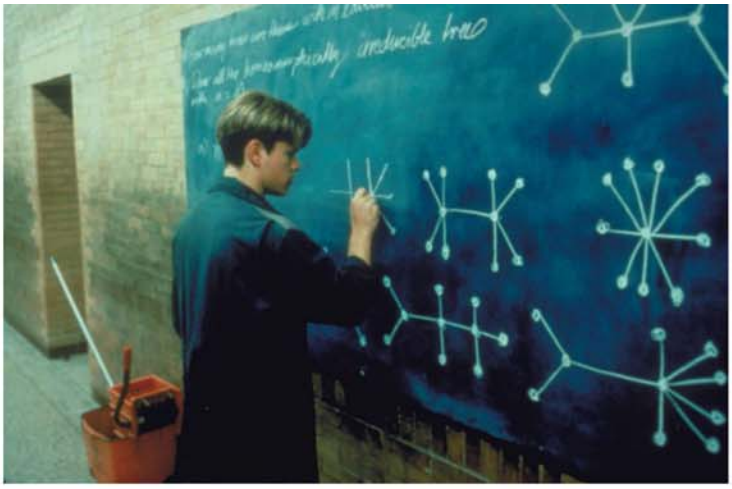
For the past few years the most polarizing thing in finance has been crypto. Crypto is a set of ideas and products and technologies that grew out of the Bitcoin white paper. But it's also, let's be clear, a set of lines on charts that went up. When Satoshi invented Bitcoin, one Bitcoin was worth zero dollars: It was just an idea he made up. At its peak last November, one Bitcoin was worth more than $67,000, and the total value of all the crypto in circulation was something like $3 trillion. Many people who got into crypto early got very rich very fast and were very annoying about it. They bought Lamborghinis and islands. They were pleased with themselves: They thought crypto was the future, and they were building the future and being properly and amply rewarded for it. They said things like "Have fun staying poor" and "NGMI" ("not gonna make it") to people who didn't own crypto. They were right and rich and wanted you to know it.

Many other people weren't into crypto. They got the not-entirely unjustified impression that it was mostly useful for crime or for Ponzi schemes. They asked questions like "What is this for?" or "Where did all this money come from?" or "If you're building the future, what is the actual *work* you're doing?" or "If you're building the future, why does it seem so grim and awful?" And the crypto people, often, replied: "Have fun staying poor."

And then, this year, those lines on charts went down. The price of one Bitcoin fell below $20,000; the total value of crypto fell from $3 trillion to $1 trillion; some big crypto companies failed. If you're a crypto skeptic, this was very satisfying, not just as a matter of schadenfreude but also because maybe now everyone will shut up about crypto and you can go back to not paying attention to it. For crypto enthusiasts, this was just a reason to double down on grinding: The crash would shake out the casual fans and leave the true believers to build the future together.

In a sense it's a dumb time to be talking about crypto, because the lines went down. But really it's a good time to be talking about crypto. There's a pause; there's some repose. Whatever is left in crypto is not *just* speculation and get-rich-quick schemes. We can think about what crypto means—divorced, a little bit, from the lines going up.

I don't have strong feelings either way about the value of crypto. I like finance. I think it's interesting. And if you like finance—if you like understanding the structures that people build to organize economic reality—crypto is amazing. It's a laboratory for financial intuitions. In the past 14 years, crypto has built a whole financial system from scratch. Crypto constantly reinvented or rediscovered things that finance had been doing for centuries. Sometimes it found new and better ways to do things.



Often it found worse ways, heading down dead ends that traditional finance tried decades ago, with hilarious results.

Often it hit on more or less the same solutions that traditional finance figured out, but with new names and new explanations. You can look at some crypto thing and figure out which traditional finance thing it replicates. If you do that, you can learn something about the crypto financial system—you can, for instance, make an informed guess about how the crypto thing might go wrong—but you can also learn something about the traditional financial system: The crypto replication gives you a new insight into the financial original.

Also, I have to say, as someone who writes about finance, I have a soft spot for stories of fraud and market manipulation and smart people putting one over on slightly less smart people. Often those stories are interesting and illuminating and, especially, funny. Crypto has a very high density of stories like that.

And so, now, I write a lot about crypto. Including quite a lot right here.

I need to give you some warnings. First, I don't write about crypto as a deeply embedded crypto expert. I'm not a true believer. I didn't own any crypto until I started working on this article; now I own roughly $100 worth. I write about crypto as a person who enjoys human ingenuity and human folly and who finds a lot of both in crypto.

Conversely, I didn't sit down and write 40,000 words to tell you that crypto is dumb and worthless and will now vanish without a trace. That would be an odd use of time. My goal here is not to convince you that crypto is building the future and that if you don't get on board you'll stay poor. My goal is to convince you that crypto is interesting, that it has found

some new things to say about some old problems, and that even when those things are wrong, they're wrong in illuminating ways.

Also, I'm a finance person. It seems to me that, 14 years on, crypto has a pretty well-developed financial system, and I'm going to talk about it a fair bit, because it's pretty well-developed and because I like finance.

Riveted: A crowd at a Bitcoin conference in Miami, April 2022.

A financial system is, well, a series of databases. It's a way to shuffle around claims on tangible stuff; it's an adjunct to the real world. A financial system is good if it makes it easier for farmers to grow food and families to own houses and businesses to make awesome computer games, if it helps to create and distribute abundance in real life. A financial system is bad if it trades abstract claims in ways that enrich the people doing the trading but don't help anyone else.

I ... ehhh ... uh. A salient question in crypto, for the past 14 years, has been: What is it good for? If you ask for an example of a business that actually uses crypto, the answers you'll get are mostly financial businesses: "Well, we built a really great exchange for trading crypto." Cool, OK. Sometimes these answers are plausibly about creating or distributing abundance: "Crypto lets emigrants send remittances cheaply and quickly." That's good. Often they're about efficient gambling. Gambling is fun, nothing against it. But a financial system that was purely about gambling would be kind of limited.

Meanwhile, crypto's most ardent boosters say crypto is about building real, useful things. Crypto will redefine social relationships, and gaming, and computers. It will build the metaverse. Crypto is the vital component of the next leap in the internet; crypto will build "web3" to replace our current "web2." Maybe? If you ask for an example of a business that actually uses crypto, you'll get a ton of real, lucrative financial businesses, then some vague theoretical musings like "Well, maybe we could build a social media network on web3?"

It's still early. Maybe someone will build a really good social media network on web3. Maybe in 10 years, crypto and blockchains and tokens will be central to everything that's done on the internet, and the internet will be (even more than it is now) central to everything that's done in human life, and the crypto early adopters will all be right and rich while the rest of us will have fun staying poor, and schoolchildren will say, "I can't believe anyone ever doubted the importance of Dogecoin."

I don't want to discount that possibility, and I do want to speculate about it a little bit, maybe sketch a picture of what that might mean. I'm not going

to give you a road map for how we'll get there. I'm not a tech person, and I'm not a true believer. But it is worth trying to understand what crypto could mean for the future of the internet, because the implications are sometimes utopian and sometimes dystopian and sometimes just a modestly more efficient base layer for stuff you do anyway. Plus the finance is cool, and it's cool now.

ii. Digression: Names and people

Before we go on, let me say some things about some names. First, "crypto." This thing I'm writing about here: There's not a great *name* for it. The standard name, which I'll use a lot, is crypto, which I guess is short for "cryptocurrency." This is not a great name, because 1) it emphasizes *currency*, and a lot of crypto is not particularly about currency, and 2) it emphasizes *cryptography*, and while crypto *is* in some deep sense about cryptography, most people in crypto are not doing a ton of cryptography. You can be a crypto expert or a crypto billionaire or a leading figure in crypto without knowing much about cryptography, and people who *are* cryptography experts sometimes get a bit snippy about the crypto people stealing their prefix.

There are other names for various topics in crypto–

"blockchain"

"tokens"

"DeFi"

"web3"

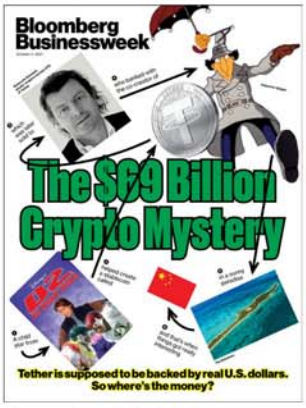
"the metaverse"

–and they're sometimes used broadly to refer to a lot of what's going on in crypto, but it's not like they're great either. So I'll mostly stick with "crypto" as the general term.

Second, "Satoshi Nakamoto." That's a pseudonym, and whoever wrote his white paper has done a reasonably good job of keeping himself, herself, or themselves pseudonymous ever since. (There's a lot of speculation about who the author might be. Some of the funnier suggestions include Elon Musk and a random computer engineer named, uh, Satoshi Nakamoto. I'm going to call Satoshi Nakamoto "Satoshi" and use he/him pronouns, because most people do.)

A related point. Other than (maybe?) Satoshi, basically everyone involved in cryptocurrency is a hilariously outsize personality. It's a good bet that if you read an article about crypto, it will feature wild characters. (One story in *Bloomberg Businessweek* last year mentioned "sending billions of

perfectly good US dollars to the *Inspector Gadget* co-creator's Bahamian bank in exchange for digital tokens conjured by the *Mighty Ducks* guy and run by executives who are targets of a US criminal investigation.") Except this one! There won't be a single exciting person in this whole story. My goal here is to explain crypto, so that when you read about a duck guy doing crypto you can understand what it is that he's doing.



"Anyone Seen Tether's Billions?" by Zeke Faux.

iii. Digression: The "crypto" in crypto

Cryptography is the study of secret messages, of coding and decoding. Most of what I talk about in this article won't be about cryptography; it will be about, you know, Ponzis. But the base layer of crypto really is about cryptography, so it will be helpful to know a bit about it.

The basic thing that happens in cryptography is that you have an input (a number, a word, a string of text), and you run some function on it, and it produces a different number or word or whatever as an output. The function might be the Caesar cipher (shift each letter of a word by one or more spots in the alphabet, so "Caesar" becomes "Dbftbs"), or pig Latin (shift the first consonants of the word to the end and add "-ay," so "Caesar" becomes "Aesar-say"), or something more complicated.

tbuptij atoshi-Say
oblbnpup akamoto-Nay

A useful property in a cryptographic function is that it be "one-way."[4] This means it's easy to turn the input string into the output string, but hard to do it in reverse; it's easy to compute the function in one direction but impossible in the other. (The classic example is that multiplying two large prime numbers is quite straightforward; factoring an enormous number into two large primes is hard.) The Caesar cipher is easy to apply and easy to reverse, but some forms of encoding are easy to apply and much more difficult to reverse. That makes them better for secret codes.

4 This has a more technical meaning than what I'm using here. What I call a one-way function in the text is, more strictly, a function that we hope is one-way, based on current understanding of computer technology and math and cryptography.

One example of this is a "hashing" function, which takes some input text and turns it into a long number of a fixed size. So I could run a hashing function on this article—a popular one is called SHA-256, which was invented by the National Security Agency[5]—and generate a long, incomprehensible number from it. (To make it more incomprehensible, it's customary to write this number in hexadecimal, so that it will have the digits zero through 9 but also "a" through "f.") I could send you the number and say, "I wrote an article and ran it through a SHA-256 hashing algorithm, and this number was the result." You'd have the number, but you wouldn't be able to make heads or tails of it. In particular, you couldn't plop it into a computer program and *decode* it, turning the hash back into this article.

5 If you want to try it for yourself, there are various SHA-256 calculators online; one is at Xorbin.com. Or if you want to program it yourself, or do some hashing with pencil and paper, there is a US government publication (FIPS PUB 180-4) that spells out the algorithm. (Or it's on Wikipedia.)

The hashing function is one-way; the hash tells you nothing about the article, even if you know the hashing function. The hashing function basically shuffles the data in the article: It takes each letter of the article, represented as a binary number (a series of bits, 0s and 1s), and then shuffles around the 0s and 1s lots of times, mashing them together until they are all jumbled up and unrecognizable. The hashing function gives clear step-by-step instructions for how to shuffle the bits together, but they don't work in reverse.[6] It's like stirring cream into coffee: easy to do, hard to undo.

6 A simple example: One way to mash data together is called an XOR function, for "exclusive or." If you apply XOR to two bits (1 or 0), it returns 1 if one of those bits is a 1, and 0 if both or neither of them are. Say you ran XOR on the numbers 1100 and 0101, applying it first to the digits in the first position of each number (1 and 0), then to the digits in second position (1 and 1), and so on. It would return 1001. Knowing the inputs, it is easy to compute the output. But if you know the output is 1001, you don't know the inputs: They could be 1100 and 0101, or 0011 and 1010, or 1001 and 0000, or 1111 and 0110, etc. If you take half of this article and XOR it with the other half, you'll have some mess that's hard to turn back into the article. If you do that dozens of times, you have cryptography.

Applying a SHA-256 algorithm will create a 64-digit number for data of any size you can imagine. Here's a hash of the entire text of James Joyce's 730-page novel *Ulysses*:

3f120ea0d42bb6af2c3b858a08be9f737dd422f5e92c04f82cb9c40f06865d0e

It fits in the same space as the hash of "Hi! I'm Matt":

86d5e02e7e3d0a012df389f727373b1f0b1828e07eb757a2269fe73870bbd044

But what if I wrote "Hi, I'm Matt" with a comma? Then:

9f53386fc98a51b78135ff88d19f1ced2aa153846aa492851db84dc6946f558b

There's no apparent relationship between the numbers for "Hi! I'm Matt" and "Hi, I'm Matt." The two original inputs were almost exactly identical; the hash outputs are wildly different. This is a critical part of the hashing function being one-way: If similar inputs mapped to similar outputs, then it would be too easy to reverse the function and decipher messages. But for practical purposes, each input maps to a *random* output.[7]

7 Since hashes spit out a fixed number of digits, it's possible that two different inputs could map to the same hash. This is called a collision. But a 64-digit hexadecimal number allows for a lot of different hashes—16 ^ 64, or about 10 ^ 77 of them, or many billion times more than the number of atoms on Earth.

What's the point of a secret code that can't be decoded? For one thing, it's a way to verify. If I sent you a hash of this article, it wouldn't give you the information you need to re-create the article.[8] But if I then sent you the *article*, you could plop *that* into a computer program (the SHA-256 algorithm) and generate a hash. And the hash you generate will exactly match the number I sent you. And you'll say, "Aha, yes, you hashed that article all right." It's impossible for you to *decode* the hash, but it's easy for you to *check* that I had *encoded* it correctly.

8 Exercise for the reader: I have included some hashes of some texts in this article, and I have talked about the hash of this article, but I haven't included the hash of this article in the article. Why not? (Believe me, I wanted to.)

This would be dumb to do with this article, but the principle has uses. A simple, everyday one is passwords. If I have a computer system and you have a password to log in to the system, I need to be able to check that your password is correct. One way to do this is for my system to store your password and check what you type against what I've stored: I have a little text file with all the passwords, and it has "Password123" written next to your username, and you type "Password123" on the login screen, and my system checks what you type against the file and sees that they match and lets you log in. But this is a dangerous system: If someone steals the file, they would have everyone's password. It's better practice for me to *hash* the passwords. You type "Password123" as your password when setting up the account, and I run it through a hash function and get back

008c70392e3abfbd0fa47bbc2ed96aa99bd49e159727fcba0f2e6abeb3a9d601

and I store *that* on my list. When you try to log in, you type your password, and I hash it again, and if it matches the hash on my list, I let you in. If someone steals the list, they can't decode your password from the hash, so they can't log in to the system.[9]

9 It's beyond the scope here, but there's a lot more cryptographic fun—"rainbow tables," "salt," etc.—involved in defeating or strengthening this sort of security.

There are other, more crypto-nerdy uses for hashing. One is a sort of time stamping. Let's say you predict some future event, and you want to get credit when it does happen. But you don't want to just go on Twitter now and say, "I predict that the Jets will win the Super Bowl in 2024," to avoid being embarrassed or influencing the outcome or whatever. One thing you could do is write "the Jets will win the Super Bowl in 2024" on a piece of paper, put it in an envelope, seal the envelope, and ask me to keep it until the 2024 Super Bowl, after which you'll tell me either to open the envelope or burn it. But this requires you and everyone else to trust me.

Another, trustless thing you could do is type "the Jets will win the Super Bowl in 2024" into a cryptographic hash generator, and it will spit out:

64b70b0494580b278d7f1f551d482a3fb952a4b018b43090ffeb87b662d34847

and then you can tweet,

Not a real tweet! But you can follow me on Twitter at @matt_levine.

Everyone will say, "Well, aren't you annoying," but they won't be able to decode your prediction. And then in a while, when the Jets win the Super Bowl, you can say, "See, I called it!" You retweet the hashed tweet and the plain text of your prediction. If anyone is so inclined, they can go to a hash calculator and check that the hash really matches your prediction. Then all the glory will accrue to you.

Aside from hashing, another important one-way function is public-key encryption. I have two numbers, called a "public key" and a "private key." These numbers are long and random-looking, but they're related to each other: Using a publicly available algorithm, one number can be used to lock a message, and the other can unlock it. The two-key system solves a classic problem with codes: If the key I use to encrypt a message is the same one you'll need to decode it, at some point I'll have to have sent you that key. Anyone who steals the key in transit can read our messages.

With public-key encryption, no one needs to share the secret key. The public key is public: I can send it to everyone, post it on my Twitter feed, whatever. The private key is private, and I don't give it to anyone. You want to send me a secret message. You write the message and run it through the encryption algorithm, which uses 1) the message and 2) my public key (which you have) to generate an encrypted message that you send to me. Then I run the message through a decryption program that uses 1) the encrypted message and 2) my *private* key (which only I have) to generate the original message, which I can read. You can encrypt the message using my public key, but nobody can *decrypt* it using the public key. Only I can decrypt it using my private key. (The function is one-way as far as you're concerned, but I can reverse it with my private key.)

A related idea is a "digital signature." Again, I have a public key and a private key. My public key is posted in my Twitter bio. I want to send you a message, and I want you to know that I wrote it. I run the message through an encryption program that uses 1) the message and 2) my private key. Then I send you 1) the original message and 2) the encrypted message.

You use a decryption program that uses 1) the encrypted message and 2) my public key to decrypt the message. The decrypted message matches the

original message. This proves to you that I encrypted the message. So you know that I wrote it. I could've just sent you a Twitter DM instead, but this is more cryptographic.

Imagine a simple banking system in which bank accounts are public: There's a public list of accounts, and each one has a (public) balance and public key. I say to you: "I control account No. 00123456789, which has $250 in it, and I'm going to send you $50." I send you a digitally signed message saying "here's $50," and you decode that message using the public key for the account, and then you know that I do in fact control that account and everything checks out. That's the basic idea at the heart of Bitcoin, though there are also more complicated ideas.

iv. How Bitcoin works

The simple form of Bitcoin goes like this. There's a big public list of addresses, each with a unique label that looks like random numbers and letters, and some balance of Bitcoin in it. An address might have the label "1A1zP1eP5QGefi2DMPTfTL5SLmv7DivfNa"[10] and a balance of 68.6 Bitcoin. The address acts as a public key.[11] If I "own" those Bitcoin, what that means is I possess the private key corresponding to that address, effectively the password accessing the account.

10 This is famous in crypto lore, the address that received the first Bitcoin. Presumably it belongs to Satoshi Nakamoto.

11 The address is actually a hash of the public key. But "it is in fact perfectly legitimate cryptographic terminology to refer to the pubkey hash as a public key itself," wrote Vitalik Buterin, creator of Ethereum, another major blockchain, in his 2014 white paper explaining that project. Good enough for Vitalik, good enough for me.

Because I have the private key, I can send you a Bitcoin by signing a message to you with my private key. You can check that signature against my public key and against the public list of addresses and Bitcoin balances. That information is enough for you to confirm that I control the Bitcoin that I'm sending you, but not enough for you to figure out my private key and steal the rest of my Bitcoin.

That kind of means I can send you a Bitcoin without you trusting me, or me trusting you, or either of us trusting a bank to verify that I have the money. "We define an electronic coin as a chain of digital signatures," Satoshi wrote. The combination of public address and private key is enough to define a coin. Cryptocurrency is called cryptocurrency because it's a currency derived from cryptography.

Satoshi said a Bitcoin is essentially a chain of signatures.

You'll notice that all we've done here is exchange a message, and somehow called the result of that a currency. The traditional financial system isn't so different: Banks don't move around sacks of gold or even very many paper bills. They're keepers of databases. What happens, roughly, when I make a $100 payment to you is my bank sends a message to your bank telling it to update its ledger.

Similarly, in Bitcoin the messages change a (public) ledger of who holds what. But who maintains *that*? The rough answer is that the Bitcoin network–thousands of people who use Bitcoin and run its software on their computers–keeps the ledger, collaboratively and redundantly. There are thousands of copies of the ledger; every node on the network has its own list of how many Bitcoin are in each address.

Then, when we do a transaction–when I send you a Bitcoin–we don't just do it privately; we broadcast it to the entire network so everyone can update their lists. If I send you a Bitcoin from my address, and my signature on the transaction is valid, everyone will update their ledgers to add one Bitcoin to your address and subtract one from mine.

The ledger is not really just a list of addresses and their balances; it's actually a record of *every single transaction*.[12] The ledger is maintained by everyone on the network keeping track of every transaction for themselves.[13]

12 Actually it's *only* that, not a list of addresses and their balances at all. I describe it that way in the text for convenience, and you can reconstruct the list of addresses and balances from the record of all transactions, and people do, but that's not technically what a Bitcoin's ledger is.

13 There's a section in the Bitcoin white paper titled "Reclaiming Disk Space," about how the network can in effect compress some of the data it keeps about old transactions using Merkle trees, all of which is beyond the scope of this piece, but people in crypto say "Merkle trees" a lot, so there you go.

That's nice! But now, instead of trusting a bank to keep the ledger of your money, you're trusting thousands of anonymous strangers.

What have we accomplished?

Well it's not quite as bad as that. Each transaction is provably correct: If I send a Bitcoin from my address to yours and sign it with my private key, the network will include the transaction; if I try to send a Bitcoin from someone else's address to yours and don't have the private key, everyone on the network can see that it's fake and won't include the transaction. Everyone runs open-source software to update the ledger for transactions that are verifiable. Everyone keeps the ledger, but you can prove that every transaction in the ledger is valid, so you don't have to trust them *too* much.

Incidentally, I am saying that "everyone" keeps the ledger, and that was probably roughly true early in Bitcoin's life, but no longer. There are thousands of people running "full nodes," which download and maintain and verify the entire Bitcoin ledger themselves, using open-source official Bitcoin software. But there are millions more not doing that, just having some Bitcoin and trusting that everybody else will maintain the system correctly. Their basis for this trust, though, is slightly different from the basis for your trust in your bank. They could, in principle, verify that everyone verifying the transactions is verifying them correctly.

Notice, too, that there's a financial incentive for everyone to be honest: If everyone is honest, then this is a working payment system that might be valuable. If lots of people are dishonest and put fake transactions in their ledgers, then no one will trust Bitcoin and it will be worthless. What's the point of stealing Bitcoin if the value of Bitcoin is zero?

This is a standard approach in crypto: Crypto systems try to use economic incentives to make people act honestly, rather than *trusting* them to act honestly.

That's most of the story, but it leaves some small problems. Where did all the Bitcoin come from? It's fine to say that everyone on the network keeps a ledger of every Bitcoin transaction that ever happened, and your Bitcoin

can be traced back through a series of previous transactions. But traced back to what? How do you start the ledger?

Another problem is that the *order* of transactions matters: If I have one Bitcoin in my account and I send it to you, and then I send it to someone else, who actually has the Bitcoin? This seems almost trivial, but it's tricky. Bitcoin is a decentralized network that works by broadcasting transactions to thousands of nodes, and there's no guarantee they'll all arrive in the same order everywhere. And if everyone doesn't agree on the order, bad things—"double spending," or people sending the same Bitcoin to two different places—can happen. "Transactions must be publicly announced," wrote Satoshi, "and we need a system for participants to agree on a single history of the order in which they were received."

That system, I'm sorry to say, is the blockchain.

v. Oh, the blockchain

Every Bitcoin transaction is broadcast to the network. Some computers on the network—they're called "miners"—compile the transactions as they arrive into a group called a "block." At some point, a version of a block becomes, as it were, official: The list of transactions in that block, in the order in which they're listed, becomes canonical, part of the official Bitcoin record. We say that the block has been "mined."[14] In Bitcoin, a new block is mined roughly every 10 minutes.[15]

14 Actually, a block becomes *really* canonical when it has "five confirmations": when it has been mined, and then another block has been mined that refers back to it, and then another block has been mined referring to *that* block, etc., five times, so that the chain has continued five blocks after the block in question.

15 You can see a finished block online on any "block explorer" site. For example, block 755965, mined on Sept. 27, is basically a list of 2,466 transactions between different addresses. An address starting bc1qns sent 0.0052 Bitcoin to an address starting 16qZC7; 39VgGL split 0.012 Bitcoin between 14NrDK and 37o1E3; and so on.

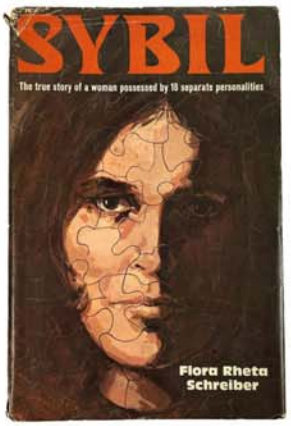
The miners then start compiling a new block, which will also eventually be mined and become official. Here's where hashing becomes important. That new block will refer to the block before it by containing a hash of that block —this confirms that the block before it 1) is correct and accepted by the network and 2) came before it in time. Each block will refer to the previous block in a chain—oh, yes, a blockchain. The blockchain creates an official record of what transactions the network has agreed on and in what order. The hashes are time stamps; they create an agreed order of transactions.

You could imagine a simple system for doing this. Every 10 minutes a miner proposes a list of transactions, and all the computers on the Bitcoin network vote on it. If it gets a majority, it becomes official and is entered into the blockchain.

Unfortunately this is a bit too simple. There are no *rules* about who can join the Bitcoin network: Anyone who hooks up a computer and runs the open-source Bitcoin software can do it. You don't have to prove you're a good person, or even a person. You can hook up a thousand computers if you want.

What mining looks like, in Nadvoitsy, Russia.

This creates a risk of what's sometimes called a "Sybil attack," named not after the ancient Greek prophetesses but, rather, after the 1973 book about a woman who claimed to have multiple personalities. The idea of a Sybil attack is that, in a system where the ledger is collectively maintained by the group and anyone can join the group without permission, you can spin up a bunch of computer nodes so that you look like thousands of people. Then you verify bad transactions to yourself, and everyone is like, "Ah, well, look at all of these people verifying the transactions," and they accept your transactions as the majority consensus, and either you manage to steal some money or you at least throw the whole system into chaos.



Sybil (1973).

The solution to this is to make it *expensive* to verify transactions.

To mine a block, Bitcoin miners do an absurd and costly thing. Again, it involves hashing. Each miner takes a summary of the list of transactions in the block, along with a hash of the previous block. Then the miner sticks another arbitrary number—called a "nonce"—on the end of the list. The miner runs the whole thing (list plus nonce) through a SHA-256 hashing algorithm. This generates a 64-digit hexadecimal number. If that number is small enough, then the miner has mined the block. If not, the miner tries again with a different nonce.

What "small enough" means is set by the Bitcoin software and can be adjusted to make it easier or harder to mine a block. (The goal is an average of one block every 10 minutes; the more miners there are and the faster their computers are, the harder it gets.) Right now, "small enough" means that the hash has to start with 19 zeros. A recent successful one looked like this:

000000000000000000006c9f1194ce7ff75c5f265d5520878e9e9392c3c8ff203

It's like a game of 20 questions where you're constantly guessing a number that will work. Except you get no clues, and it's many, many, many times more than 20 guesses. It is vanishingly, vanishingly unlikely that any particular input—any list of transactions plus a nonce—will hash to a number that starts with 19 zeros. The odds are roughly 75 sextillion-to-1 against. So the miners run the hash algorithm over and over again, trillions of times, guessing a different nonce each time, until they get a hash with the right number of zeros.[16] The total hash rate of the Bitcoin network is something north of 200 million terahashes per second—that is, 200 quintillion hash calculations per second, which is 1) a lot but 2) a lot fewer than 75 sextillion. It takes many seconds—600 on average—at 200 quintillion hashes per second to guess the right nonce and mine a block.

16 Vitalik again: "Because SHA256 is designed to be a completely unpredictable pseudorandom function, the only way to create a valid block is simply trial and error, repeatedly incrementing the nonce and seeing if the new hash matches."

This is a race. Only one miner gets to mine a block, and that miner gets rewarded with Bitcoin. To mine a block is also to "mine" new coins—to pry them out of the system after much computational work, like finding a seam of gold after picking through rock. Hence the metaphor.

An old-fashioned prospector, circa 1860.

When miners find the right number of zeros, they publish the block and its hash to the Bitcoin network. Everyone else reviews the block and decides if it's valid. ("Valid" means all the transactions on the list are valid, the hash is correct, it has the right number of zeros, etc.) If they do, then they start work on the next block: They take the hash of the previous block, plus the transactions that have come in since then, plus a new nonce, and try to find a new hash. Each block builds on the one before.

vi. Mining

All of this is incredibly costly: Miners need special hardware to do all of these hashing calculations over and over again, and these days run huge farms of always-on computers. Mining Bitcoin uses as much electricity as various medium-size countries. This is not great for the environment. The most famous description of Bitcoin, attributed to a Twitter poster, might be:

"Imagine if keeping your car idling 24/7 produced solved Sudokus you could trade for heroin."

And it is in some sense purely wasteful. People sometimes say Bitcoin miners are, like, solving difficult math problems to do their mining, but they aren't, really. They're brute-force guessing quintillions of numbers per second to try to get the right hash. No math problems are being solved, and nothing is added to the world's knowledge, by those quintillions of guesses.

But the miners *are* solving an important problem for Bitcoin, which is the problem of keeping its network and its ledger of transactions secure. It's *demonstrably costly* to confirm Bitcoin transactions, so it's hard to fake, hard to run a Sybil attack. That's why Satoshi, and everyone else, calls this method of confirming transactions "proof of work." If you produce the right hash for a block, it proves you did a lot of costly computer work. You wouldn't do that lightly.

Proof-of-work mining is a mechanism for *creating consensus among people with an economic stake in a system*, without knowing anything else about them. You'd never mine Bitcoin if you didn't want Bitcoin to be valuable. If you're a Bitcoin miner, you're invested in Bitcoin in some way; you've bought computers and paid for electricity and made an expensive, exhausting bet on Bitcoin. You have proven that you care, so you get a say in verifying the Bitcoin ledger. And you get paid. You get paid Bitcoin, which gives you even more of a stake in the system.

These Bitcoin come out of nowhere; they're generated by this mining, by the core Bitcoin software. In fact, all Bitcoin are generated by mining; there was never an initial allocation of Bitcoin to Satoshi Nakamoto or to early investors or anyone else. This is the answer to the question of where Bitcoin come from: They were all mined.

Originally the mining reward, which is set by the software, was 50 Bitcoin per block; currently it's 6.25 Bitcoin. One important point about these mining rewards is that they cost Bitcoin users money. Every block—roughly every 10 minutes—6.25 new Bitcoin are produced out of nowhere and paid to miners for providing security to the network. That works out to more than $6 billion per year.[17] This cost is indirect: It is a form of inflation, and as the supply of Bitcoin grows,[18] each coin in theory becomes worth a little less, all else being equal. Right now, the Bitcoin network is paying around 1.5% of its value per year to miners.

17 That is, 6.25 Bitcoin every 10 minutes, or 37.5 per hour, or 900 per day, multiplied by 365 (days a year), multiplied by the price of Bitcoin.

18 Famously, though, there will only ever be 21 million Bitcoin. It's written into the code. So what happens when that limit is reached? What incentive could miners have to keep the Bitcoin network running? Transaction fees. The Bitcoin code also lets miners collect a slice of each transaction, and this will become the only method for rewarding them once the last coin is mined. (Current estimates are that this won't happen until 2140.)

That's lower than the inflation rate of the US dollar. Still, it's worth noting. Every year, the miners who keep the Bitcoin system secure capture a small but meaningful chunk of the total value of Bitcoin. Bitcoin users get something for that $6 billion:[19]

If you can make a lot of money mining Bitcoin, a lot of people will want to mine Bitcoin. This will make it harder for one person to accumulate most of the mining power in Bitcoin. If one person or group got a majority of the mining power, they could do bad things: They could mine a bad block–double-spending coins, reversing recent transactions, etc. (This is called a "51% attack.") When there are billions of dollars up for grabs for miners, people will invest a lot of money in mining, and it will be expensive to compete with them. And if you invested billions of dollars to accumulate a majority of the mining power in Bitcoin, you would probably care a lot about maintaining the value of Bitcoin, and so you'd be unlikely to use your powers for evil.

19 In 2021, Vitalik wrote a post about this, which began: "The Bitcoin and Ethereum blockchain ecosystems both spend far more on network security—the goal of proof of work mining—than they do on everything else combined. The Bitcoin blockchain has paid an average of about $38 million per day in block rewards to miners since the start of the year, plus about $5m/day in transaction fees. The Ethereum blockchain comes in second, at $19.5m/day in block rewards plus $18m/day in tx fees."

II
What Does It Mean?

So, huh, that's neat. OK, then. I've described in some detail the workings of the thing, Bitcoin, that Satoshi Nakamoto invented. But let's take a step back: What exactly is it that he invented?

The simplest answer is that he invented Bitcoin.

At its peak, the total value of Bitcoin in the world was more than $1 trillion. There are thousands of articles about it; it has lots of investors and fans and believers. Some of these people are called "Bitcoin maximalists"; they believe that the only really interesting and valuable thing in the world of crypto is Bitcoin. Those people could stop here, I guess. There it is, Bitcoin.

Here, though, I want to keep going. I want to talk about different ways that you might *generalize* Satoshi's invention. There are different ways to interpret what Satoshi was up to and what he accomplished, and each interpretation points you to a different direction for crypto.

A.
A Store of Value

A minimal generalization of Bitcoin is something like: Satoshi invented a technology for people to send numbers to one another. That's not nothing. Before Satoshi, I could've written you an email that said "132.51," but you'd have no way of knowing whether I had the 132.51 on my computer or whether I'd already sent the 132.51 to someone else, and you'd have no way of proving to other people that you now had the 132.51 on your computer and could send it to them.

I realize that paragraph sounds very stupid, because it is. You definitely have 132.51 on your computer, as well as every other conceivable number; computers can generate numbers arbitrarily and more or less for free. Open a spreadsheet, type "132.51," and there you go. In a sense, the

technological accomplishment of Bitcoin is that it invented a decentralized way to *create scarcity on computers*. Bitcoin demonstrated a way for me to send you a computer message so that you'd have it and I wouldn't, to move items of computer information between us in a way that limited their *supply* and transferred *possession*.

But the technological accomplishment is not the whole story, arguably not even the most important part. The wild thing about Bitcoin is not that Satoshi invented a particular way for people to send numbers to one another and call them payments. It's that people accepted the numbers as payments.

There's nothing inherent in the technology that would make that happen. People might have read the Bitcoin white paper and said, "Huh, this is a cool way to send payments, but your problem is that you aren't sending *dollars*, you're sending this thing you just made up, and who wants that?" Well, most of them did say that, initially. But lots of people eventually decided that Bitcoin was valuable.

That's weird! Satoshi was like,

And enough people were like,

Illustration: C.W. Moss

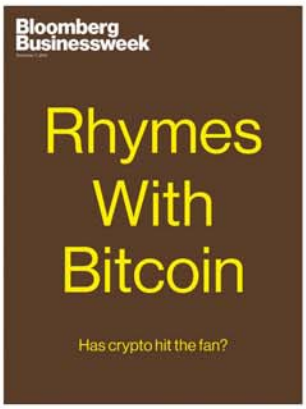
that now crypto is a trillion-dollar business. That social fact, that Bitcoin was *accepted* by many millions of people as having a lot of value, might be the most impressive thing about Bitcoin, much more than the stuff about hashing.

i. Shitcoins

Here's another extremely simple generalization of Bitcoin:

1. You can make up an arbitrary token that trades electronically.
2. If you do that, people might pay a nonzero amount of money for it.
3. Worth a shot, no?

As Bitcoin became more visible and valuable, people just ... did ... that? There was a rash of cryptocurrencies that were sometimes subtle variations on Bitcoin and sometimes just lazy knockoffs. "Shitcoins" is the mean name for them.



See what we did there?

In 2013 two software engineers threw together a cryptocurrency and gave it a logo of Doge, the talking shiba inu meme. They called it Dogecoin, and it was a parody of the coin boom. It's worth about $8 billion today. I'm not going to explain that to you. Nobody is going to explain that to you. Certainly the guys who invented Dogecoin don't understand it; one of them has taken to Twitter to say he hates it. It's just, like, if you're making up an arbitrary token that trades electronically, and you hope people will buy it for no particular reason, you might as well make it fun. Slap a talking dog on it; give people stuff to make jokes about online.

Incidentally, here's a fun argument that was made against Bitcoin early in its life:

1. There's a limited supply of bitcoin.
2. But the Bitcoin software is open-source and can be cloned trivially.
3. So if the price of Bitcoin gets above, you know, $100, someone will just invent Blitcoin, which will be an exact copy of Bitcoin.
4. Bitcoin is arbitrary, and Blitcoin is arbitrary, so there's no reason that Blitcoin should trade at much of a discount to bitcoin.

5. This will dilute the value of bitcoin: any sensible person would rather pay $90 for Blitcoin than $105 for Bitcoin, since they're the same thing but one is cheaper.
6. Therefore, there's an infinite supply of Bitcoin or things that are exactly like it, so the value of Bitcoin cannot get too high.

This argument turned out to be mostly wrong. Socially, cryptocurrency is a coordination game; people want to have the coin that other people want to have, and some sort of abstract technical equivalence doesn't make one cryptocurrency a good substitute for another. Social acceptance–legitimacy–is what makes a cryptocurrency valuable, and you can't just copy the code for that.

That's a revealing fact: What makes Bitcoin valuable isn't the elegance of its code, but its social acceptance.[20] A thing that worked exactly like Bitcoin but didn't have Bitcoin's lineage–didn't descend from Satoshi's genesis block and was just made up by some copycat–would have the same technology but none of the value.

20 One advantage that Bitcoin has over a hypothetical copycat: more miners. A big diverse pool of miners keeps a cryptocurrency's blockchain more secure than a small concentrated pool. But that security is just an instantiation of its broader social acceptance. All the Bitcoin miners *could* quit and become Blitcoin miners. They just don't.

ii. An uncorrelated asset

Here's another generalization of Bitcoin:

1. Satoshi made up an arbitrary token that trades electronically for some price.
2. The price turns out to be high and volatile.
3. The price of an arbitrary token is ... arbitrary?

This may not sound that great to you. But it's very interesting as a matter of finance theory. Modern portfolio theory demonstrates that adding an uncorrelated asset to a portfolio can improve returns and reduce risk. Big institutions will invest in timberland or highway tolls or hurricane insurance, because they think that those things *won't* act just like stocks or bonds, that they'll diversify their portfolios, that they'll hold up even in a world where stocks go down.

To the extent that the price of Bitcoin 1) mostly goes up, though with lots of ups and downs along the way, and 2) goes up and down for reasons that are arbitrary and mysterious and *not* tied to, like, corporate earnings or the global economy, then Bitcoin is interesting to institutional investors.

There are variations. For instance:

1. Bitcoin is not just *uncorrelated* to regular financial stuff–it's a *hedge to inflation*. If the Federal Reserve is printing money recklessly, the dollar will lose value, but Bitcoin is in limited supply and will maintain its value even as the dollar is inflated away.
2. Bitcoin is like gold but more convenient. The value of gold is also somewhat arbitrary and mysterious, but it's a store of value that's not tied to corporate earnings and central bank policy. Investors who like gold should buy Bitcoin.

Well, those are some things that people said. In practice, it turns out that the price of Bitcoin is pretty correlated with the stock market, especially tech stocks. Bitcoin *hasn't* been a particularly effective inflation hedge: Its price rose during years when US inflation was low, and it's fallen this year as inflation has increased. The right model of crypto prices might be that they go up during broad speculative bubbles when stock prices go up, and then they go down when those bubbles pop. That's not a particularly appealing story for investors looking to diversify. You want stuff that goes up when the broad bubbles pop!

iii. GameStop

The simple story of GameStop is that some people on the internet liked the stock.

I'm not going to dwell on the meme-stock phenomenon here–I dwelt on it in this publication last December. But one important possibility is that the first generalization of Bitcoin, that an arbitrary tradeable electronic token can become valuable just because people want it to, permanently broke everyone's brains about all of finance.

Before the rise of Bitcoin, the conventional thing to say about a share of stock was that its price represented the market's expectation of the present value of the future cash flows of the business. But Bitcoin has no cash flows; its price represents what people are willing to pay for it. Still, it has a high and fluctuating market price; people have gotten rich buying Bitcoin. So people copied that model, and the creation of and speculation on pure, abstract, scarce electronic tokens became a big business.

A share of stock is a scarce electronic token. It's also something else! A claim on cash flows or whatever. But one thing that it is is an electronic token that's in more or less limited supply. If you and your friends online want to make jokes and invest based on those jokes, then, depending on your sense of humor and which online chat group you're in, you might buy either Dogecoin or GameStop Corp. stock, and for *your* purposes those things are not *that* different.

B.
A Distributed Computer

Here's another, very different generalization of Bitcoin. In its sharpest form, it's mostly attributed to programmer Vitalik Buterin, another colorful

character whom we won't discuss.[21] It goes like this:

21 There are two books by former Bloomberg journalists about the early days of Ethereum, in which Vitalik is the star but by no means the only important player: Camila Russo's *The Infinite Machine* and Matthew Leising's *Out of the Ether*.

1. Look, this thing you made is a big, sprawling *computer*. The blockchain is doing the functions of a computer. Specifically, it's keeping a database of Bitcoin transactions.
2. This computer has some fascinating properties. It's *distributed*: The computer's data aren't kept on any one particular machine but spread out among lots of nodes. The blockchain creates a mechanism to make sure they all agree on what the database says. It's *decentralized*: Different people run the database on their own separate machines. It's *secure* and *final*: Because of how transactions are encoded into blocks, it's more or less impossible for someone to reach back into the database and change a transaction from last week. And it's *trustless* and *permissionless*: Anyone who wants to can download the blockchain or mine Bitcoin. The mining mechanism gives people incentives to collaborate and compete with one another to keep the database secure and up to date.
3. But it's not a very *good* computer. Mostly it just keeps a list of payments.

Vitalik at ETHDenver in February. Told you he was colorful!

i. Ethereum

The computer that Vitalik[22] invented is generally called Ethereum, or the Ethereum Virtual Machine: It's a *virtual* computer, distributed among thousands of redundant nodes. Each node knows the "state" of the computer—what's in its memory—and each transaction on the system updates that state.

22 Similar to Satoshi, Vitalik Buterin is widely referred to in the crypto world by his first name, and I follow that convention here.

Ethereum works a lot like Bitcoin: People create transactions, they broadcast them to the network, the transactions are included in a block, the blocks get chained together, everyone can see every transaction, etc. The currency of the Ethereum blockchain is called ... I dunno, it's common to call it "Ether," though sometimes people say "Ethereum," and often they just write "ETH." (Similarly, Bitcoin is sometimes written "BTC.") In conversation it's mostly shortened to "Eth," pronounced "Eeth."

But whereas Bitcoin transactions are mostly about sending payments,[23] actions on Ethereum are conceived of more generally: Ethereum is a big virtual computer, and you send it instructions to do stuff on the computer. Some of those instructions are "Send 10 Ether from Address X to Address Y": One thing in the computer's memory is a database of Ethereum addresses and how much Ether is in each of them, and you can tell the computer to update the database.

23 This is an exaggeration; there's a scripting language in Bitcoin and some ability to write programs.

But you can also write *programs* to run on the computer to do things automatically. One sort of program might be: Send 10 Ether to Address Y *if* something happens. Alice and Bob might want to bet on a football game, or on a presidential election, or on the price of Ether.[24] They might write a computer program on the Ethereum Virtual Machine to do that. The program would have its own Ethereum account where it could keep Ether, and its programming logic would say something like "if the Jets win on Sunday"–or "if Joe Biden wins the election," or "if Ether trades above $1,500 on November 1"–"then send the money in this account to Alice; otherwise send it to Bob." Alice and Bob might then each send one Ether to the account, and it would whir along for a bit checking the football scores or the election results or the Ether price,[25] and then when it had an answer to its question–who won the game or the election or is Ether above $1,500–it would automatically resolve the bet and send two Ether to the winner.

24 Alice and Bob are stock characters in discussions of crypto. Not interesting characters, though.

25 How does it check? The standard solution in crypto is called an "oracle." It's a program that will periodically query some company or website that tracks the relevant information (election results, football scores, weather, etc.) and post the answer to the Ethereum blockchain. An oracle is essentially a way to bring information from the outside world (the real world, or just the internet) onto the blockchain.

Or you could have a program that says: "If anyone sends one Ether to this program, the program will send them back something nice." "Something

nice" is pretty hazy there, and frankly it's pretty hazy in actual practice, but in concept anything that you can put into a computer program could be the reward here. So "send me one Ether and I will send you back a digital picture of a monkey" would be one possible program, and I guess it sounds like I'm joking, but for a while digital pictures of monkeys were selling for millions of dollars on Ethereum. Or there's a thing called the Ethereum Name Service, or ENS, which allows people to register domain names like "matthewlevine.eth" and use them across various Ethereum functions. You send Ether to the ENS program, and it registers that name to you—you send in money, and it sends you back a domain.

Funny, that doesn't look like a contract.

The standard analogy here is a vending machine: A vending machine is a computer in the real world, where you put in a dollar and you get back something that you want. You don't negotiate with the vending machine, or make small talk about the weather while it rings you up. The vending machine's side of the transaction is entirely automated. Its programming makes it respond deterministically to you putting in money and pressing buttons.

In the crypto world, these programs are called "smart contracts." The name is a bit unfortunate. A smart contract is a computer program that runs on the blockchain. Some smart contracts look like contracts: Alice and Bob's bet on the price of Ethereum looks a lot like a financial derivative, which is definitely a contract. Some smart contracts look like vending machines: They sit around in public waiting for people to put money in, and then they spit out goods. A vending machine is not exactly a normal contract, but it is a *transaction*, and people who are into philosophizing about contracts like thinking about vending machines.

But some smart contracts just look like, you know, computer programs. The concept is more general than the name. In the Ethereum white paper, Vitalik Buterin wrote:

> **Note that "contracts" in Ethereum should not be seen as something that should be "fulfilled" or "complied with"; rather, they are more like "autonomous agents" that live inside of the Ethereum execution environment, always executing a specific piece of code when "poked" by a message or transaction, and having direct control over their own ether balance and their own key/value store to keep track of persistent variables.**

There are limits: Ethereum is a distributed computer, but it doesn't have a keyboard and a monitor. It would be hard to play *Call of Duty* on the Ethereum Virtual Machine. But Ethereum's blockchain and smart contracts can serve as sort of a back end to other types of programs. Developers build

"dapps,"

or decentralized apps, on Ethereum and other blockchains. These are computer programs that mostly run on the web (perhaps on some centralized or cloud server) but keep some of their essential data on the blockchain. You play a computer game, and your character's attributes are stored on the blockchain. A normal program on the game company's servers renders the character's sword on your screen, but the fact that she has the sword is stored on the blockchain.

One other limit is that it's a *slow* computer. The way Ethereum executes programs is that you broadcast the instructions to thousands of nodes on the network, and they each execute the instructions and reach consensus on the results of the instructions. That all takes time. Your program needs to run thousands of times on thousands of computers.

Computers and network connections are pretty fast these days, and the Ethereum computer is fast *enough* for many purposes (such as transferring Ether, or keeping a database of computer game characters). But you wouldn't want to use this sort of computer architecture for extremely time-sensitive, computation-intensive applications. You wouldn't want, like, a self-driving car running on the Ethereum Virtual Machine. You wouldn't want thousands of computers around the world redundantly calculating how far you are from hitting someone before you could brake.

ii. Proof of stake

This distributed computer, the Ethereum Virtual Machine, takes its basic design from Bitcoin. There are blocks, everyone can see them, they are chained together, transactions are signed with private keys, everything is hashed, etc. It's just that, in addition to sending money to people, you can send computer instructions to the blockchain, and the blockchain will execute them.

What that means is that there are thousands of computers each running nodes of the Ethereum network, and all those computers will agree about what happens on that network, who sent money to whom, and what computer instructions executed when. The fact that Ethereum is a distributed, virtual computer means that all those actual computers can come to a *consensus* about what operations executed when. And the reason this was possible is that Bitcoin showed how a decentralized computer network could reach consensus. The stuff with the hashing and the mining and the nonces and the electricity: That is Bitcoin's consensus mechanism, proof of work (or PoW).

Until last month, it was also Ethereum's. There were some technical differences, but the basic mechanics were pretty similar. Miners did a bunch of hashes of block data, and whoever found the right hash first mined the block and got a reward. Because this was expensive and wasted a lot of resources, it demonstrated a commitment to the Ethereum ecosystem. But the waste itself was bad.

And so, on Sept. 15, after years of planning, Ethereum switched to a new consensus mechanism: Ethereum now uses something called proof of stake (or PoS). The basic ideas remain the same. People do transactions and broadcast them to the Ethereum network. A bunch of computers—in PoW they're called "miners," in PoS they're called "validators"—work to compile these transactions into an official ordered list, called the blockchain. Anyone with a computer can be a miner/validator; the protocol is open to everyone. But the miners/validators have to prove their

commitment to the system to mine/validate blocks. In PoW, the way you prove that is by using a lot of electricity to do hashes. In PoS, the way you prove that is by having a lot of Ether.

Oversimplifying a bit, the general mechanics are:

1. ANYONE CAN VOLUNTEER TO BE A VALIDATOR BY "STAKING" SOME OF THE NETWORK'S CURRENCY, DEPOSITING IT INTO A SPECIAL SMART CONTRACT. THE STAKED CURRENCY CAN'T BE WITHDRAWN FOR SOME PERIOD.[26] ON ETHEREUM, YOU NEED TO STAKE 32 ETHER—CURRENTLY $40,000 OR SO—TO BE A VALIDATOR.
2. VALIDATORS GET TRANSACTIONS AS THEY COME IN AND COMPILE THEM INTO BLOCKS.[27]
3. AT FIXED INTERVALS (SAY, EVERY 12 SECONDS), ONE VALIDATOR IS RANDOMLY CHOSEN TO PROPOSE A BLOCK, AND SOME OTHER SET OF VALIDATORS IS CHOSEN TO REVIEW THE PROPOSED BLOCK AND VOTE ON IT.
4. THE RANDOMLY CHOSEN VALIDATORS AGREE ON WHETHER TO ADD THE BLOCK TO THE CHAIN. IF EVERYONE IS DOING THEIR JOB HONESTLY AND CONSCIENTIOUSLY, THEY'LL MOSTLY AGREE, AND THE BLOCK WILL BE ADDED.
5. THE VALIDATORS GET PAID FEES IN ETHER.
6. IF A VALIDATOR ACTS DISHONESTLY OR LAZILY—IF IT PROPOSES WRONG BLOCKS, OR IF IT FAILS TO PROPOSE OR VOTE ON BLOCKS, OR IF SOMEONE TURNS OFF THE COMPUTER RUNNING THE VALIDATOR—IT CAN HAVE SOME OR ALL OF ITS STAKE TAKEN AWAY AS A PENALTY.

26 As of this writing, Ethereum has moved to proof of stake but is still working on other related upgrades. One result is that staked Ether currently can't be withdrawn at all, until some future upgrades are completed. Even then, withdrawals will take time.

27 In fact there is a division of labor in Ethereum where there are specialized companies called "block builders" that compile blocks for validators to validate.

I mean, that's the concept, but when I write it out like that, it sounds more manual than it is. Nobody is sitting around reviewing every transaction and agonizing over whether it's legitimate. The validators are just running the official open-source Ethereum software. It is all pretty automatic, and you can run it on a laptop with good backup power and a solid internet connection. The big outlay may be the $40,000 to buy Ether. It's not hard to contribute to the consensus. It's hard to *override* it. But being an honest validator is pretty easy.

When we discussed proof-of-work mining, I said that crypto systems are designed to operate on consensus among people with an economic stake in the system. PoW systems demonstrate economic stake in a cleverly indirect way: You buy a bunch of computer hardware and pay for a lot of electricity and do a bunch of calculations to prove you really care about Bitcoin. PoS systems demonstrate the economic stake directly: You just invest a lot of money in Ethereum and post it as a bond, which proves you care.

Making coins is a lot of work.

This is more efficient, in two ways. First, it uses less electricity. Burning lots of electricity to do trillions of pointless math calculations a second, in a warming world, seems dumb. Proof of stake uses, to a first approximation, no electricity. You're simply keeping a list of transactions, and you just have to compile the list once, not 200 quintillion times. The transition to PoS cut Ethereum's energy usage by something like 99.95%.

Second, PoS more directly measures your stake in the system. You demonstrate your stake in Ethereum by 1) owning Ether and 2) putting it at risk[28] to validate transactions. To take control of the PoS system and abuse it for your own nefarious purposes, you need to own a *lot* of Ether, and the more you own, the less nefarious you'll want to be. "Proof of stake can buy something like 20 times more security for the same cost," Vitalik has argued.

28 One risk is "slashing": If you do nefarious things, and other validators notice, they can slash your stake and take away your Ether. Conceptually, the bigger risk is that the value of your Ether will fall while you have it locked up: If you do things to undermine confidence in Ethereum, then the value of your stake will drop.

- **Staking**

Here's how a Bitcoin miner makes money:

1. Spend dollars to buy computers and electricity.
2. Use the computers and electricity to generate Bitcoin.
3. Sell the Bitcoin, or hold them and hope they go up.

Here's how an Ethereum validator makes money:

1. Buy Ether.
2. Lock it up.
3. Get paid fees in Ether that are, roughly, a percentage of the Ether you've locked up. Currently the fees are around 4%.

There's still some computer hardware involved–you have to run software to compile and check transactions–but not much of it; again, it can be a laptop. The capital investment isn't in computers but in the relevant cryptocurrency. The transaction is very close to: "Invest a lot of cryptocurrency and then get paid *interest* on that cryptocurrency."

You can make it even easier on yourself. Instead of downloading the software to run a full Ethereum validator node, and depositing 32 Ether, you can hand your Ether over to someone else and let *them* be a validator. It doesn't need to be 32 Ether: If you have 1 Ether, and 31 other people each have 1 Ether, and you all pool your Ether together, then you have enough to stake, validate transactions, and earn fees. And then you each can have a cut of the fees. The work of validating transactions can be completely separated from the actual staking of Ether.

And in fact a lot of Ethereum validation runs through crypto *exchanges* such as Coinbase, Kraken, and Binance, which offer staking as a product to their customers. (The biggest is a thing called Lido Finance, which isn't an exchange but a sort of decentralized staking pool.) The customers keep their Ether with the exchange anyway, so they might as well let the exchange stake it for them and earn some interest.

Yes: interest. If you're putting crypto into a staking pool, what it looks like to you is simply earning *interest* on your crypto: You have 100 tokens, you lock them up for a bit, you get back 103 tokens. The stuff about validating transactions occurs in the background, and you don't really have to worry about it. You just get a percentage return on your money–around 4% now, but maybe less after fees–from locking it up. (Before you compare that to the passive income you might earn on, say, a bond, remember this is paid in volatile Ether.)

Crypto has found a novel way to *create yield*. We'll talk about others later–crypto has a whole business of "yield farming"–but this is one. You can deposit your crypto into an account, and it will pay you interest. It will pay you interest not for the reason banks generally do–because they're lending

your money to some other customer who will make use of it—but because you are, in your small way, helping to maintain the security of the transaction ledger.

iii. Gas

Another difference between Ethereum and Bitcoin is that transaction fees are much more important in Ethereum.

The basic reason is that every transaction in Bitcoin is more or less the same: "X sends Y Bitcoin to Z." In Ethereum, though, there are transactions like "Run this complicated computer program with 10,000 steps." That takes longer. Thousands of nodes on the Ethereum network have to run and validate each computational step of each contract. If a contract requires a lot of steps, then it will use a lot more of validators' time and computer resources. If it requires infinite steps, it would crash the whole thing.

To address this issue, Ethereum has "gas," which is a fee that people and smart contracts pay for computation. Each transaction specifies 1) a maximum gas limit (basically a number of computational steps) and 2) a price per unit of gas. If the transaction uses up all its gas—if it takes more steps to execute than the gas limit—it fails (and still pays the gas fee). This deters people from sending superlong transactions that clog the network, and it absolutely prevents them from clogging the network forever.

In early Ethereum, the gas fees, as well as built-in mining rewards, were paid to the miner who mined a block. Since the move to PoS, the built-in rewards are lower (because it's much less expensive to be a validator than a miner, so you don't need to get paid as much). And now some of the gas fees are "burned" (the Ether just vanishes) instead of being paid to validators. The basic result is that Ether as a whole is paying less for security under PoS than it used to.

There are still gas fees, though, and some of them still go to validators. And generally speaking, the more you offer to pay for gas, the faster your transaction will be executed: If Ethereum is busy, paying more for gas gets you priority for executing your transactions. It is a shared computer where you can pay more to go first.

iv. Tokens

● ERC-20

One thing a smart contract can do in Ethereum is create new cryptocurrencies. These cryptocurrencies are generally called "tokens."

Why would you want to do this? One reason we already talked about:

1. You can make up an arbitrary token that trades electronically.
2. If you do that, people might pay a nonzero amount of money for it.
3. Worth a shot, no?

This is extremely *easy* to do in Ethereum. (The Ethereum white paper includes a four-line code snippet "for implementing a token system" on Ethereum.) And so there's the Shiba token, which calls itself "a decentralized meme token that evolved into a vibrant ecosystem." It's Dogecoin but on Ethereum, easy. It has a "Woof Paper."

But there are lots of other reasons to create cryptocurrencies. If you set up some sort of app that does a thing on the Ethereum system and you want to charge people money for doing that thing, what sort of money should you charge them? Or if you set up a two-sided marketplace that connects people who do a thing with people who want the thing done, what sort of money should the people who want the thing use to pay the people who do the thing?

Dollars are a possible answer, though an oddly hard one: US dollars don't live on the blockchain, but in bank accounts. Ether is the most obvious answer: You've set up an app in Ethereum, so you should take payment in the currency of Ethereum. But a persistently popular answer is: You should take payment in *your own currency*. People who add value to your service should be paid in your own special token; people who make use of the service should pay for it in that token. And then if the service takes off, the token might become more valuable.

An asset class?

We'll discuss this idea in more detail later. For now, I'll just say that Ethereum has a standard for how these sorts of tokens should be implemented, and it's called ERC-20. And when there are decentralized apps on the Ethereum blockchain, there's a good chance that they'll say they have an ERC-20 token.

One essential property of an ERC-20 token is that it's fungible—like dollars, or Bitcoin, or Ether. If I create an ERC-20 token called Mattcoin and mint a billion Mattcoins, each of those billion tokens works exactly the same and is exactly interchangeable. They all trade at the same price, and nobody wants, or gets, any particular identified Mattcoin.

● ERC-721

There's another way to do a token, though. You could have a series of tokens, each with a number. Token No. 1 in the series is different from Token No. 99, in the sense that Token No. 1 has the number 1 and Token No. 99 has the number 99. This is generally referred to as a nonfungible token, or NFT. The most popular Ethereum standard for NFTs is called ERC-721, and you'll see that name sometimes.

Let me quote a bit of the ERC-721 standard:

> **The ERC-721 introduces a standard for NFT, in other words, this type of Token is unique and can have different value than another Token from the same Smart Contract, maybe due to its age, rarity or even something else like its visual. Wait, visual?**
>
> **Yes! All NFTs have a [numerical] variable called tokenId, so for any ERC-721 Contract, the pair contract address, [numerical] tokenId must be globally unique. That said, a dapp can have a "converter" that uses the tokenId as input and outputs an image of something cool, like zombies, weapons, skills or amazing kitties!**

Look how minimal this standard is, despite the zombies and kitties. An NFT consists of a series of numbered tokens, and the thing that makes it an NFT is that it has a different number in its tokenId field from the other tokens in its series.

If you'd like to imagine that this different number makes it something cool, like a zombie, or a kitty, you can! Go right ahead! Or if there's a computer program—or an Ethereum dapp—that looks at your number and says, "Ah, right, *this* number corresponds to a zombie with green hair and a fetching scar on his right cheek," then the computer program is free to say that—and even serve you up a picture of that zombie—and you are free to believe it.

We'll come back to this.

v. An ICO

Here's another important difference between Ethereum and Bitcoin. Bitcoin never raised money; Ethereum did.

You can think of Bitcoin as more or less the open-source passion product of one anonymous guy who really likes cryptography. The cost of building the basic system of Bitcoin was Satoshi's time, which he donated. Then he mined the first Bitcoins and got super rich, probably, but that came later.

Ethereum was a bit more complicated to build. Vitalik Buterin is the intellectual leader of Ethereum, but there were a bunch of co-founders. There were legal entities. There were programmers. They spent a lot of time on it. They had to pay for food deliveries.

You could imagine Vitalik saying: "OK, we are a company, we're Ethereum Inc., we're going to start the Ethereum blockchain and make money from it, and we'll sell shares in Ethereum Inc. to raise the money to build the blockchain. Sell 20% of Ethereum Inc. for cash, get the cash, build the blockchain and, I don't know, collect royalties. Ethereum Inc. collects 0.01% of every Ethereum transaction forever." Ethereum was a well-known and much-hyped project even before it launched, and they could easily have found investors.

They didn't do that, for philosophical and economic reasons. They wanted a decentralized blockchain ecosystem, and having it owned by a corporation would defeat the purpose.[29] So they didn't sell shares. They sold tokens. In July 2014 they sold Ether "at the price of 1000-2000 ether per BTC, a mechanism intended to fund the Ethereum organization and pay for development," according to Ethereum's white paper. In all, they sold about 60 million Ether for about $18.3 million, all before the technology itself went live. Ethereum's genesis block was mined in July 2015. Today there is a total of about 122 million Ether outstanding. Some of that, like Bitcoin, comes from mining or, now, validating. But almost half of it was purchased, before Ethereum was launched, by people who wanted to bet on its success.

29 There is an Ethereum Foundation, which works on Ethereum development and has a certain amount of moral authority, but it has no control over the blockchain itself.

Camila Russo, in her book about Ethereum, wrote:

> **A whole new financing model had been tested. One where a ragtag group of feuding hackers with no business plan and no live product, let alone users or revenue, could raise millions of dollars from thousands of people from all over the world. Before, anyone who wanted to buy stock in big tech firms like Facebook or Google would need a U.S. bank account; things got even more complicated for those who wanted to invest in startups that hadn't gone to the public markets to raise funds. Now anyone could be an investor in one of the most cutting-edge technology companies out there. All they needed was an internet connection and at least 0.01 Bitcoin.**

Which of these ragtag groups' tokens would you choose?

It worked out well for those investors; that 60 million Ether is worth billions of dollars today.

But as a "whole new financing model," it's a mixed bag. Many people, particularly securities regulators, think it's *good* that startups usually can't raise money from the general public without at least a business plan. And there's a sense in which this sale—the Ether "pre-mine," or "initial coin offering" (ICO)—was the original sin of crypto as a financing tool. A lot of other crypto teams copied Ethereum's approach, writing up vague plans to build some project and then raising money by preselling tokens that would be useful if the project ever happened. The analogy I've tweeted is that ICOs are "like if the Wright Brothers sold air miles to finance inventing the airplane."

There was a 2017 ICO boom in which a lot of projects raised a lot of money by selling tokens that never turned out to be useful. When ragtag groups of hackers with no business plan can raise millions of dollars from anyone with an internet connection, they all will. The odds that any particular one of those non-business-plans will succeed are low. The odds that any particular one will be a scam are high.

vi. Other chains

● Layers, trilemmas

The basic ideas of Ethereum—a distributed computer, smart contracts, dapps, new tokens, etc.—caught on broadly within crypto. Ether is now the second-biggest cryptocurrency (well behind Bitcoin, well ahead of everything else).

But it has a lot of competition: If you're building smart contracts, there are a lot of other blockchains where you can run them, including

BNB Chain,
Solana,
Avalanche,
Cardano,
Tezos,
Polkadot,
Algorand,
Tron,
and Terra 2.0.

(Terra 1.0 blew itself up, oops.)

These platforms, like Bitcoin and Ethereum, are called "Layer 1 blockchains," meaning they're entirely separate from one another; each Layer 1 blockchain maintains its own ledger. They compete with one another much like other tech platforms do, arguing that they offer better performance, a better environment for developers, a different programming style, better tools.

A famous problem in crypto is the "blockchain trilemma": Blockchains want to be scalable (they can process a lot of transactions quickly), decentralized (they don't depend on a few trusted parties), and secure (a minority of computers on the network can't successfully attack it). But, says the trilemma, you can only choose two of those three.

Bitcoin and Ethereum chose decentralization and security, which makes them fairly slow computers. Other blockchains are faster but more centralized: If your consensus mechanism is "We trust six computers to verify all transactions," that's going to be faster than Bitcoin's proof-of-work algorithm. But if someone hacks those six computers, look out.

People on the decentralized-and-secure blockchains spend a lot of time thinking about scaling. Often this involves so-called Layer 2 systems, which are built on top of Layer 1 technology such as Bitcoin and Ethereum. For instance, Bitcoin has the Lightning Network, a Layer 2 payment system that basically lets people on the Bitcoin blockchain set up payments to each other without running all of them through the blockchain. This makes the payments faster and cheaper, and they periodically settle on the blockchain for security.

Much of the thought in Ethereum these days is about how to scale it so that it can execute large numbers of transactions quickly and cheaply–a prerequisite for building a universal world computer, or frankly even a payments system that can compete with credit cards. Much of the action here is about Layer 2 systems that specialize in doing some sorts of transactions off the main Ethereum blockchain (where space is somewhat scarce and expensive) and then saving the results to that blockchain (where transactions are secure and immutable).[30]

30 A lot of the interesting *cryptography* in crypto is found here—people love talking about "zero-knowledge proofs" in this context—because in some broad sense these are problems of data compression. You want to be able to do a lot of transactions on a Layer 2, and then write down those transactions in an abridged way on the Ethereum blockchain, while still being confident that the secure and immutable Ethereum blockchain keeps track of all of them.

- **Bridges, wrapping**

Some people in crypto are faithful to a single blockchain: They are Bitcoin maximalists, or Ethereum or Avalanche or Solana loyalists. Many people are generalist dabblers, though. They like buying lots of tokens on lots of blockchains, because they see merit in many blockchain platforms, or because they like it when lines go up.

One result of this is that sometimes you will want to own the token of one blockchain on another blockchain. This comes up a lot in decentralized finance, or DeFi, the system of crypto exchanges and financial products that live on blockchains. If you're writing smart contracts to trade tokens on the blockchain, those smart contracts—computer programs—will have to run on one particular blockchain. But you might not want to limit yourself to the tokens of that blockchain. (You are, after all, building an exchange.) You might want to write programs on the Ethereum blockchain that trade Bitcoin or write programs on Solana that trade Ether. Say you want to write a smart contract on Solana to swap some Ether for SOL (the Solana token). How do you do that?

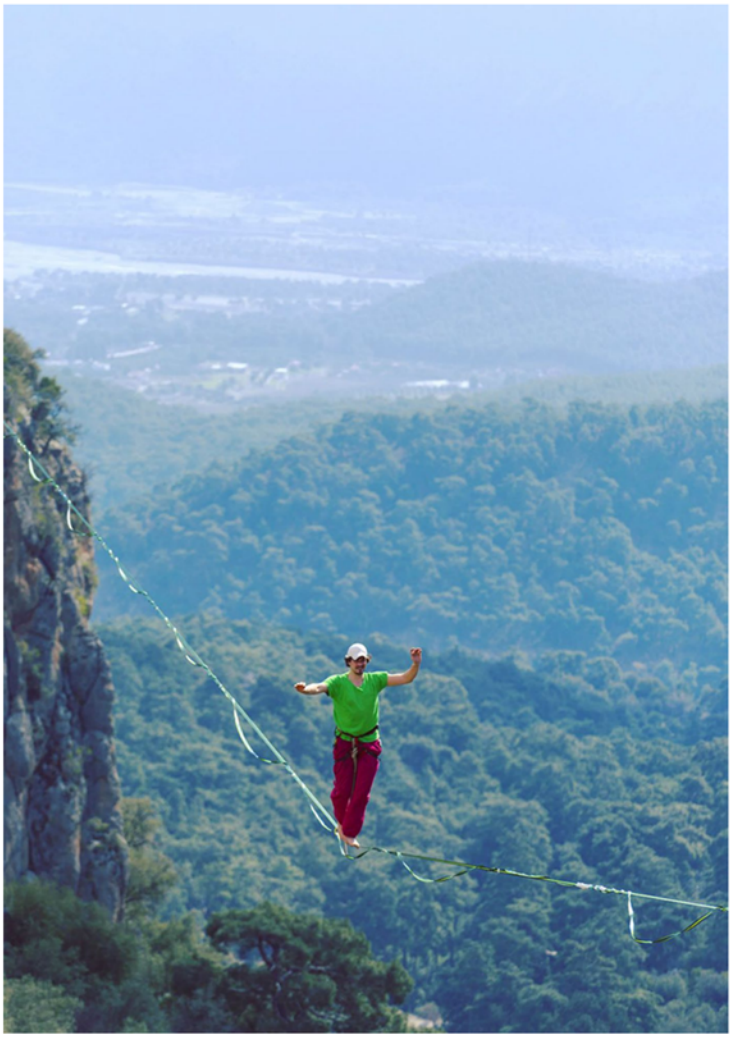
You don't. Your Ether lives on the Ethereum blockchain. "You have Ether" is a fact about the ledger on that blockchain. The Solana blockchain has no record of that. Nor does Ethereum have any record of Solana tokens. It's all incompatible, separate systems. It's separate banks and property registers and DMVs all over again.

That's an annoying answer and can't really be right, so there are workarounds. The principal one is called a "bridge." A bridge is, generally, a smart contract on one blockchain, a smart contract on another blockchain, and some sort of trusted computer program that sits between them and passes messages. If you want to swap some Ether for SOL, you find a bridge. You send your Ether to the bridge's smart contract in Ethereum, which locks it up: As far as the Ethereum blockchain is concerned, your Ether belongs to the bridge now. The bridge's off-chain computer program sees this and alerts its Solana smart contract, which gives you the equivalent of the Ether on the Solana blockchain.

"The equivalent of the Ether" could, I suppose, be some amount of SOL at whatever the current Ether/SOL exchange rate is. (The bridge could also be an exchange.) But the normal approach is to take Ether on Ethereum and give you back "wrapped Ether" (sometimes "wETH") on Solana. Wrapped Ether is a token issued by the bridge's smart contract on the Solana blockchain, representing a claim on the bridge's Ether on the Ethereum blockchain. It is, as it were, Ether on the Solana network.

1+1=3?

Bridges are notorious sites of risk in crypto: A big bridge contract will have a lot of crypto locked up in it, will need to regularly send and receive crypto from strangers, and will have to interoperate between different environments. If you can find a bug in a bridge, you can make a lot of money. People do that pretty regularly. An actual big Solana/Ethereum bridge is called Wormhole; it was hacked this year for about $320 million of wETH.



C.
A Slow Database

Here's a slightly different generalization of Bitcoin:

1. Look, you've built a distributed *database*.
2. This database has some fascinating properties. It's distributed, decentralized, secure, trustless, and permissionless.
3. Your database happens to track the ownership of electronic coins.
4. What if we built a database like that to track the ownership of *other* things?

i. Map and territory

"Modern life consists in large part of entries in databases," I said, and then I listed a few, starting with money. There's a reason I started with money, and why Satoshi did, too. A dollar *is* an entry on a list of dollars. If you have

dollars, what you have is an entry in your bank's database saying how many dollars you have. This entry is the dollars. The bank doesn't have sacks of gold or a big box of paper currency that the database refers to. It just has the database.

This is even more true for Bitcoin. There are no paper Bitcoins at all. If the Bitcoin ledger says you have a Bitcoin, then you have a Bitcoin. That's all a Bitcoin is.

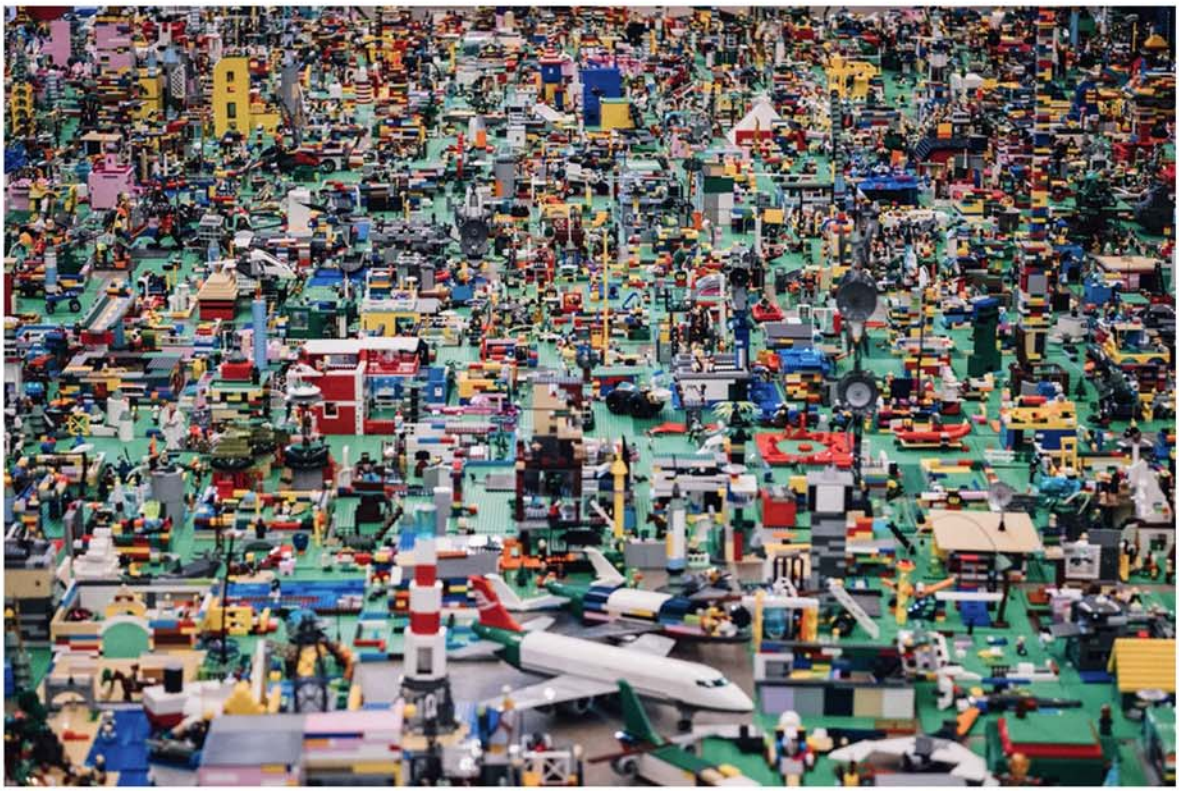
Almost nothing *else* works quite that way, though. If you own a house, in some legal sense the main thing that you own is an entry in a property register. But in some much more important sense, the main thing that you own is *the house*. If your house burned down, you could not sleep in your deed.

Conversely, if someone snuck into the property registry at night and slipped in a new deed saying that *they* owned your house, and then they showed up at your house to kick you out, you might reasonably raise objections like "But all my stuff is here," or "But I have the keys to the locks," or "But all my neighbors know I live here," or "But this is the address on my driver's license," and you might convince everyone—the sheriff, the courts, your mortgage lender—that in fact you own the house and the deed is wrong. The property registry is not *definitive* for house ownership the way the Bitcoin ledger is definitive for Bitcoin ownership.

But the idea of putting important databases of real-world stuff "on the blockchain"—for some reason the definite article is always used with this stuff, and one shouldn't worry too much about *which* blockchain—has a lot of appeal. People are always talking about moving real estate registries or cargo manifests or carbon emissions onto the blockchain.

This is appealing because, as a database, the blockchain has some nice properties. The important public blockchains such as Bitcoin and Ethereum are secure, open, and permissionless. Anyone can prove they own a Bitcoin, and that ownership can't be reversed arbitrarily.

And anyone can *build* on these systems.



If you want to build an exchange for trading Ethereum, you can just do it; the blockchain is public, and the standards for building Ethereum programs are open. Building a new system for buying *houses* is overwhelmingly difficult. You'd need to get lots of banks and county property registries and appraisers on board. But if, somehow, the houses were moved to the blockchain, that would permanently allow for permissionless innovation: Everyone could build programs and exchanges and derivatives and interfaces for house buying, and the best ones could win.

The problem is that houses can't live on the blockchain. They live in the real world. They can burn down and stuff. Connecting the electronic artifact on the blockchain—the house token?—to its real-world referent—the house—is philosophically and practically tricky.

How to make this connection is largely an unsolved problem, and quite possibly an *unsolvable* one, but also an important one. Crypto's *financial* system is well-developed and has some advantages in openness and permissionless innovation over the traditional financial system. If you could ingest the physical world into that financial system, you'd have something cool.

ii. Enterprise blockchain

A less ambitious version of this is: Look, banks are already keeping lots of databases to track lots of things. Dollars in bank accounts but also loans, tradable securities, derivative contracts, trade financing, all sorts of stuff. Some of those databases are slow, some are written in Cobol, and some require an exchange of faxes to settle transactions. It would be nice if those databases were faster, if they could talk to each other efficiently. It would be nice if JPMorgan Chase's database could talk to Goldman Sachs's database, if there weren't a manual and contentious process of reconciling trades between banks.

In 2010, if you were a loan trader at a bank, and you thought,

> **"Ugh, our systems for trading loans are so slow and kludgy,"**

and you walked into the CEO's office and said,

> **"We should spend tens of millions of dollars hiring top-notch programmers to build a new loan-trading system, and we should start a consortium with our competitors and clients where we all agree to use the same loan-trading system, because that will make my life easier and eliminate a lot of back-office costs,"**

the CEO would probably say things like

"Who are you?"

and

"Get out of my office"

and

"You're fired."

But in 2017, if you were a loan trader at a bank, or a blockchain consultant, or frankly a person off the street, and you walked into the office of a bank CEO and shouted the word

"blockchain!"

the CEO would hand you a sack of money. Lots and lots and lots of people took advantage of this. "Blockchain" was for a while the sexiest word in finance, and banks were tripping over themselves to announce blockchain initiatives.[31]

31 Not just banks. This stuff had a particular vogue at banks, but shipping companies also got in on it. Maersk has a blockchain platform (TradeLens, built with IBM) that it advertises for shipping and supply chain management. Meanwhile, the Australian Securities Exchange announced that it will replace a trading system with the blockchain, though it keeps delaying that.


Bloomberg Businessweek

Blockchain is a cryptographically secure, distributed database consisting of linked, time-stamped blocks of data such as transactions. It is based on decentralized, peer-to-peer consensus and an incentive-driven model of security. Each block in a blockchain contains a cryptographic hash of the previous block, which secures the integrity of the transaction data. The hash function, such as SHA-256, maps data to a bit string of a fixed size, a fingerprint of the original set of data. It is infeasible to reverse the function to recover the original data, and any change in the input data results in a completely different hash. Hashes are stored in a deterministic Merkle tree data structure. Each full node in the network independently checks new blocks. Creators of new blocks that are accepted by the rest of the network are rewarded. The technology allows parties to transact reliably with other parties about whom they have no information, diminishing the need for trusted intermediaries, and this **may cost you your job**.

Magazines got excited, too.

The idea here was sometimes vague, to be honest, but broadly speaking we're talking about *permissioned* blockchains, or *private* blockchains. Big public blockchains are generally not something a banker is going to like. Having all transactions be public is good for security (everyone can verify that everything is correct) but also bad (if your transactions are meant to be secret). It's also just sort of *icky* for regulation. "Who makes sure that your transaction records are correct?" a bank regulator will ask, and the bank will answer, "Well, we don't really know, but we think it's some mining pools in Russia," and the regulators will get nervous.

But many of the basic ideas of a blockchain—a ledger of every transaction that's demonstrably shared by every computer on the network—can be implemented *privately*. If you get together with 11 of your friends and agree that the 12 of you will do transactions with one another, keep a ledger, verify all the transactions, and use cryptographic hash functions to make sure the ledger isn't changed, then you can just do that. If you all trust each other and don't let anyone else join the network—or if you let only people you trust join the network—then you don't have to worry about malicious miners taking over your network. It's just you and your friends.

This has advantages for security, particularly for *explaining* security to bank regulators. You also don't have to make the blockchain public if you don't want to, and there are efficiency advantages. It's only 12 of you confirming transactions, so you can do it faster. Presumably you're doing this for a reason—you want the ability to do these transactions with each other—so you don't have to get paid for confirming transactions. You don't need mining or staking: Those are ways for public blockchains to reach a

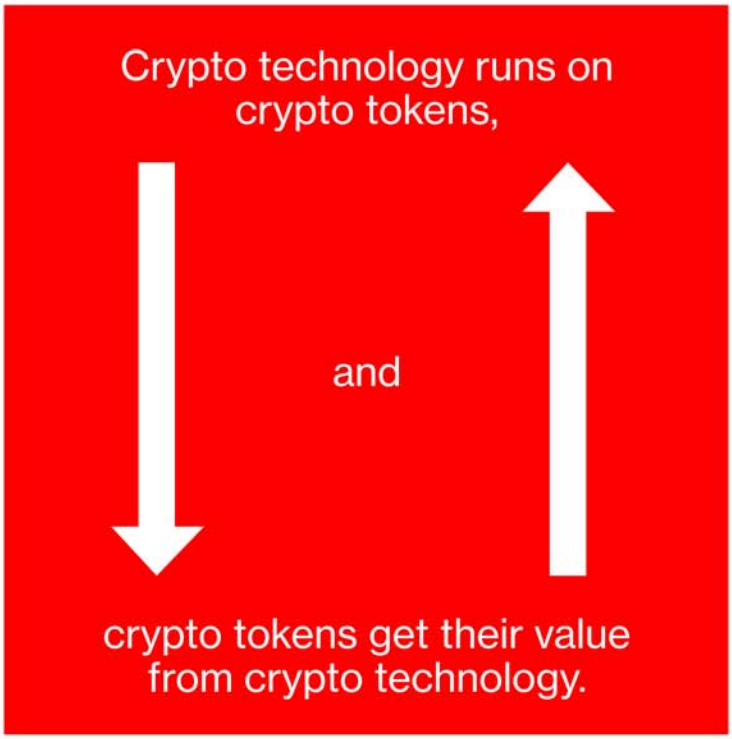
consensus among people who have to prove they have a commitment to the system. The 12 of you all know one another and built the system, so your consensus is good enough without any further proof. You can just vote on it.

D.
Web3

An important fact about Bitcoin is that it's both a technological method of sending money and the money itself. That is, Bitcoin is a computer system for sending Bitcoins, and a Bitcoin is the thing that the Bitcoin system sends.

Some of what goes on in crypto is *about the technology*: People use the ideas of blockchains and smart contracts and so forth to build software. Some of what goes on in crypto is *about the money*: People call up their brokers to place bets on the prices of crypto tokens going up.

But a lot of what goes on in crypto is *about both*:



Broadly speaking, the name for this is "web3." The idea is that the original web was the early development of the internet, when people built decentralized open community-controlled protocols for the internet. "Web 2.0," or "web2," was when big tech companies more or less took over the internet; now your experience of the internet is largely mediated through Facebook and Google and Apple and Amazon, and they make tons of

money from controlling the internet. No open decentralized project is going to compete with them. But web3 will be when people build decentralized open community-controlled protocols for the internet again *and also make lots of money*. Because the decentralized protocols won't be owned by big tech companies, but they won't be free and owned by no one, either. They'll be owned by their users.

Just kidding: They'll be owned by venture capitalists who buy tokens in these projects early on and who are the biggest boosters of web3. But also by their users.

i. Tokens and tokenomics

Think about what you get when you buy a Bitcoin. One thing you get is a unit of "digital cash": You can send the Bitcoin to someone else to buy a sandwich or whatever. If this digital cash thing takes off, then lots of people will accept Bitcoin for sandwiches, and this currency that you bought will be very useful.

But another thing you get is a *share* in the Bitcoin project. Not a share of actual stock, but still a chance to profit from the success of Bitcoin. If this digital cash thing takes off, then lots of people will want Bitcoin to use to buy sandwiches, and there will be a lot of demand for Bitcoin. But only 21 million Bitcoin will ever exist. So each Bitcoin will be more valuable as more people decide to use Bitcoin as their way to transfer digital cash.

That logic never quite made sense. A convenient currency for digital cash transfer has a stable value, and the *rising* value of Bitcoin makes it *less* useful as a currency: If your Bitcoin keep going up in value, you should not spend them on sandwiches. Bitcoin as an appreciating asset will be a bad currency.[32] Still, it worked well *enough*. Bitcoin is used enough for digital transfers of value that it became valuable, and early adopters got rich.

32 There's a famous story of the guy who spent 10,000 Bitcoin in May 2010 to buy two large pizzas as proof of the concept that you could use Bitcoin to buy pizza. Today those 10,000 Bitcoin would be worth about $200 million.

This is a key financial innovation of crypto:

Crypto built an efficient system to make *the customers of a business also its shareholders.*

Lots of crypto projects have this basic structure. Filecoin is a decentralized system for storing files, where you can pay to store files or get paid for storing files for someone else. You pay, or get paid, in Filecoin. Helium is a decentralized system for wireless hotspots, where you can get paid for running a hotspot or pay for access. You pay, or get paid, in Helium's tokens. And there are play-to-earn video games such as *Axie Infinity*, where you pay for tokens to play the game, and get rewarded in the game's

tokens. Participating–storing files, providing Wi-Fi, playing the game–makes you an investor as well as a user.

Actually, see "A Billion-Dollar Crypto Gaming Startup Promised Riches and Delivered Disaster," by Joshua Brustein, published on June 10 by *Bloomberg Businessweek*.

This is potentially powerful because crypto is in the network-effects business. Many crypto projects are useful mainly if lots of other people use them. Bitcoin is a useful payment system if lots of people have Bitcoin and accept it for payments. Ethereum is a useful computing system if lots of people build apps on Ethereum. If you build a decentralized financial exchange or a peer-to-peer marketplace or whatever else on Ethereum, it's useful if lots of people use it.

It's hard to build a network-effects business from scratch. Early users of a network-effects business won't get much out of it, because there's no network yet. You might as well just wait until there are more users. The economics of crypto tokens reverse that. Early users of a crypto network get tokens cheap, and if the network takes off later, then their tokens will be worth a lot. Now there's an advantage to being early. "Token effects," people sometimes call this.

Many claims made about web3 are just about taking this basic idea at face value, assuming that it's good, and applying it in vague ambitious ways. One issue for web3 is that for most consumers, the process of even signing up is going to be baffling–you generally don't just take out your credit card and start playing a game. You might need to buy Ether on an exchange, then use a bridge to connect to the game's own special wallet, then trade your Ether for another token you need to buy characters or fight battles or whatever.

Another problem is the urge to tokenize businesses with no obvious network effects. In July, *Esquire* published an article about how "The Crypto Revolution Wants to Reimagine Books":

> **What if you could own a stake in Harry Potter?**
>
> **What if the book series functioned like a publicly traded company where individuals could "buy stock" in it, and as the franchise grows, those "stocks" become more valuable? If this were the case, someone who purchased just three percent of *Harry Potter* back when there was only one book would be a billionaire now.**
>
> **Just imagine how that would affect the reading experience. Suddenly a trip to Barnes & Noble becomes an investment opportunity. Early readers could spot "the next big thing" and make a $100 contribution that becomes $10,000 or even $100,000 if the book's popularity grows. If readers could own a percentage of the franchise, they might then be incentivized to help that book succeed. They could start a TikTok account to promote the book via BookTok, or use their talents as filmmakers to adapt it to the screen. All of this stands to increase the value of their original investment.**

You'd feel like a chump reading poetry, wouldn't you?

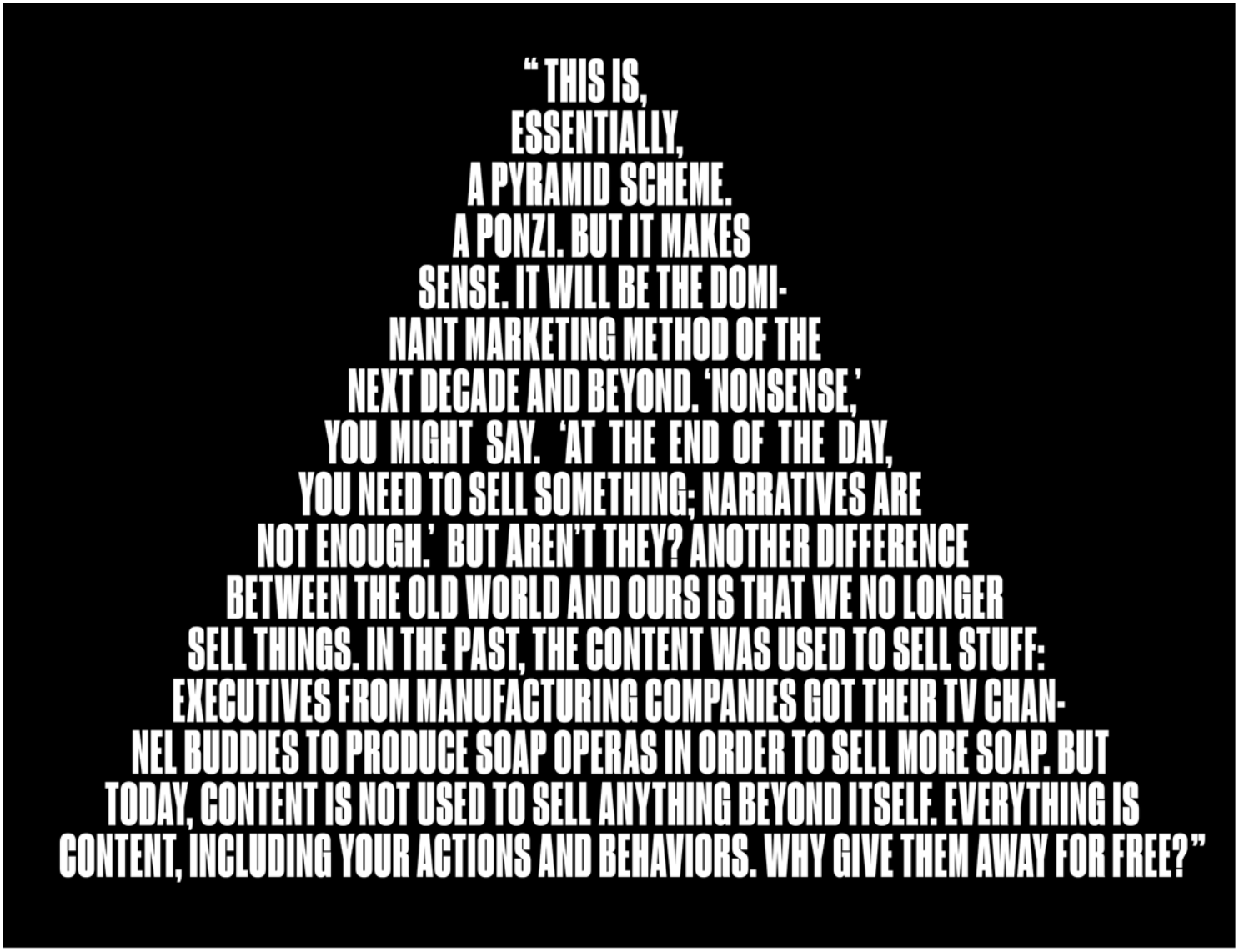
There 's grief of want, and grief of cold, —
A sort they call ' despair ; '

The bad way to put this is that every web3 project is simultaneously a Ponzi.[33]

33 Here I'm using "Ponzi" in the broad sense in which it's often used by people in markets: Investments where most current investors are just betting on the price going up because new buyers pile in. The underlying businesses or projects may be legitimate and real, but the investment model is about new buyers paying off old ones.

Why would you buy some tokens to join a web3 Facebook competitor? Well, if you like the product, go right ahead. But probably it's at least in part because you want to get rich off the tokens by selling them to someone else. Why do you think someone else will buy the tokens? Is it because you think they like the product? Or is it because you think *they* are planning to get rich by selling to a bigger sucker? Where does that end?

Dror Poleg, a writer on business and technology, wrote a blog post about web3 that I think about all the time. The title is "In Praise of Ponzis." It goes like this:



ii. DAOs

I should mention some other web3-ish concepts here. One is a DAO, usually pronounced "dow," which stands for "decentralized autonomous organization." DAOs aren't decentralized autonomous organizations. In the early days, people sometimes thought that's what they were. "This company runs automatically through smart contracts with no human intervention," they would say. "It's never been seen before in human history." But, no. The thing that runs automatically through smart contracts with no human intervention is a *smart contract*. A DAO is a way for *people* to get together to *vote* to control a pot of money or a protocol on the blockchain.

In other words, a DAO is a ... company? Like, just a regular company? It has shareholders, who put in money and control it. (Actually, they put in money and get back tokens that give them rights to govern the DAO.) And the shareholders can vote. DAOs are unlike regular companies in that the shareholders (token holders) tend to vote on more stuff: Often there's a chat room on an app called Discord, and people can propose ideas for the DAO, and there are procedures for holding a vote. Whereas US public companies let shareholders vote in only very constrained and symbolic ways, DAOs tend to let token holders vote on all sorts of stuff and often give them a fair amount of real control of the company. In this, DAOs look more like *partnerships* than public corporations.[34]

34 That's bad, by the way. In US law, the partners of a general partnership have unlimited liability for the debts of the partnership, and if you set up a business without filing any incorporation paperwork, then—oops, it's a general partnership. In May a lawsuit was filed against the members of a DAO that ran a system for trading and lending crypto. The system was hacked, and the lawsuit argues that the DAO members as general partners are on the hook for the damage. There are efforts to set up DAOs using normal legal techniques and getting limited liability, but it's early. Crypto is painfully learning lessons—incorporate your business to avoid huge personal legal trouble—that traditional finance learned centuries ago.

Some DAOs are the governance mechanisms for big decentralized finance, or DeFi, applications like crypto exchanges and are responsible for setting policies and parameters for how they work. Other DAOs are just weird larks. ConstitutionDAO made some headlines in 2021 for raising a bunch of money from crypto investors to buy a copy of the US Constitution. They failed to buy it, did some DAO voting stuff on what to do next, and ultimately returned most of the money (minus gas fees), and shut down. It was a quick way for people to pool their money online to have fun together. It was a Discord chat with a pool of money.

iii. Identity, reputation, credentials

In the Bitcoin white paper, Satoshi makes a privacy recommendation: "A new key pair should be used for each transaction to keep them from being linked to a common owner." It's free and easy to generate new Bitcoin addresses, so every time you accept a payment in Bitcoin, you should do it in a different address. The Bitcoin blockchain is public, sure, so everyone can see every transaction. But the goal is for all of *your* Bitcoin transactions to be separate, unlinkable, so no one can ever get a full picture of what you're up to on the blockchain.

Meanwhile, in Ethereum there's the Ethereum Name Service, or ENS, which we talked about earlier. This lets you register a domain name like "matthewlevine.eth" and use it across various Ethereum functions. "Use your ENS name to store all of your addresses and receive any cryptocurrency, token, or NFT," says its homepage. Venture capitalists sometimes use their ENS domain as their Twitter display name.

These are very different philosophies about what you're doing in the crypto world. Bitcoin is, philosophically, digital cash, anonymous and transactional. Ethereum is, philosophically, something like an open-source programming community, where *reputation* is what's valued. And to accumulate reputation, you need a consistent identity. You do all your Ethereum stuff under your real name, or at least your vanity plate.

Many of the grand claims that people make about web3 are about reputation and identity. The idea is that you can keep your identity on the blockchain in some immutable, decentralized, transparent, provable form, and then do good stuff with it. You have some crypto wallet that contains not just tokens that you *bought*, but also tokens you received for *doing* things. When you graduate from college, your college sends you a Bachelor's Degree Token, or a series of tokens specifying the courses you took and your grades and maybe what you learned. When you pass your driving test, the DMV sends you a Driver's License Token. When you go to a professional conference, the conference organization sends you an Attended a Conference Token. When you get promoted at your job, your employer sends you a Senior Blockchain Developer Token. When you help out on an open-source project, the leaders of the project send you a Thanks for Helping With Our Open-Source Project Token. When you post a lot on Reddit, other Reddit users send you Good Post on Reddit Tokens (these exist, and they're called Community Points). When you help your friend move, your friend sends you a Thanks for Helping Me Move Token. All of these tokens are verifiably signed by their issuers (your college, the DMV, Reddit, your friend), and the issuers have varying degrees of credibility and importance. (These tokens will also, in the general case, be *nontransferable*: You can't sell your college degree to someone else.)

And then if you're looking for a job, you'll show prospective employers your degree tokens and conference tokens and Reddit tokens and whatever else seems relevant.[35] And if you go on a decentralized dating app, perhaps you'll show prospective romantic partners your degree tokens and your cool-hobby tokens and the I'm a Good Romantic Partner in Many Ways Tokens that your exes sent you when you were together.[36] I don't know. These things are more often described as loose utopian sketches than specific programs. When I write them down they sound extremely *dystopian* to me, but perhaps you disagree.

35 And perhaps there will be *bad* tokens—an I Got Fired Token, an I Ran a Crypto Scam Token, etc.—that people can send to you to undermine your reputation, and prospective employers will have to consider how much credibility those deserve, etc. I say "you'll show prospective employers" your tokens, and I suppose there are different ways to imagine the public accessibility of all these things. Perhaps the tokens are publicly associated with your name, and anyone can see all of them; perhaps employers see only what you voluntarily send them.

36 Again, the possibility of *bad* tokens is interesting here.

Sometimes they're spelled out a bit more. In May, Vitalik Buterin published a paper, with E. Glen Weyl and Puja Ohlhaver, called *Decentralized Society: Finding Web3's Soul*. It called for a set of "non-transferable 'soulbound' tokens (SBTs) representing the commitments, credentials and affiliations of 'Souls.'" A soul, in this terminology, is not quite a *person*, though it is more like a person than it is like a Bitcoin address. A soul could be a person, or an institution (a university, etc.), but a person could also have multiple souls for multiple contexts.

"When issuing a tradeable NFT, an artist could issue the NFT from their Soul," says the paper, beautifully. It would be amazing if theology could be replaced, or perhaps *solved*, by cryptography. A person's soul is nothing more than the things and people that she loves, the people who love her, and the impact that she has on the world, and we've encoded it on the blockchain, here it is. It would be a bummer to lose the private key to your soul.[37]

37 Actually, a fascinating part of the *Decentralized Society* paper is about exactly this, *community-based* recovery of private keys. The idea is roughly that if your soul has a lot of connections to a lot of other souls, and you lose the private keys for your crypto accounts, there should be a recovery mechanism where those other souls can vouch that you are really you, and then you get the keys back (in a decentralized way).

For what shall it profit a man to gain a lot of Bitcoin and lose his own soul?

E.
Uncensorable Ledgers

Here's another way to describe what Satoshi did: He created a way to do *irreversible transactions* on computers.

i. Censorship resistance

Ordinarily, if there's some database on a computer somewhere, and it changes some data field, it can just change that field back to what it was; those are equivalent, easy things for a computer to do. But the blockchain makes it *hard* to change things back. If a Bitcoin moves from one address to another address, and that transaction is included in a block, and then a few more blocks are added afterward "confirming" the transaction, then it would take an unfathomable amount of computer *work*—running hash functions trillions and trillions of times—to rewind the blockchain to remove that transaction. Blockchain is one-way; its ledger is permanent.

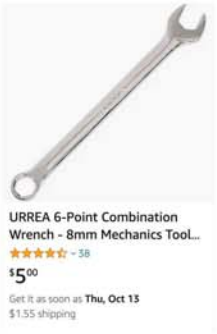
This is nice if you want your ledger to be really secure, hard to hack, backed up in multiple places, though honestly it's probably an overkill for those purposes. But it's *really* nice if you want your ledger to be immune from *government meddling*. Or, really, anyone's meddling. Crypto people call any interference with transactions "censorship." Bitcoin is "censorship-resistant." You might not trust the government, because, for instance, you live in a repressive dictatorship that controls the banking system and will seize your money from you unjustly.

Or you might have other objections. US banks, often though not always in conjunction with US government agencies, are very much in the business of *blocking payments*. They'll block payments to organizations designated as terrorists by the government, or to countries sanctioned by the government. But they'll also block payments to pornographic websites in response to public pressure campaigns. If you'd like to finance terrorism or do business in Iran or consume internet porn, this sort of thing might drive you to embrace crypto. But even if you don't like terrorism or Iran or porn, this sort of thing might make you *nervous*. What *other* forms of commerce might be shut down by the government, or by banks acting independently of the government?[38]

38 Firearms is one pretty obvious answer, but anything that creates strong feelings can be litigated through the payment system.

ii. Or not

The practical problem with censorship resistance is that the crypto world touches the real world at various points, and those contacts make it hard to be totally free from the pressures of outside influence. A meme in crypto is the "$5 wrench attack," named for an *xkcd* web cartoon pointing out that the way to steal someone's cryptocurrency isn't by using sophisticated methods to hack his laptop but rather by "hitting him with this $5 wrench until he tells us the password."



The other problem is that, while crypto generally creates decentralized and irreversible transactions, it also creates a *permanent public record* of those transactions. The government can just look at that record! It can't *reverse* the transactions, but it can do its best to make life unpleasant for the recipients.

You can send Bitcoin to anyone without anyone's permission. But at some point you'll want to *do* something with Bitcoin. You'll want to spend it; if you have a lot of it, you'll want to spend it on real estate or yachts or jewelry or art.[39] A financial system cannot be *entirely* self-contained; you have to be able to turn your money into actual stuff.

39 If you want to spend it on NFTs, then I suppose you can keep it in the crypto system forever, but what sort of life is that?

And that's where they get you. The normal way to turn your Bitcoin into dollars is through a centralized crypto exchange, such as the apps for buying and selling crypto that you saw advertised during the Super Bowl. They're the main "fiat off-ramp," the place that lets you sell your Bitcoin (which are hard to spend) for dollars (which are easy to spend). Centralized crypto exchanges are *extremely* censorable, since they tend to be run by wealthy CEOs who would prefer to remain wealthy and not in prison, and so they'll do things like ask you for your driver's license before letting you open an account. If you're on the government's "do not open an account" list, they won't open an account for you.

Also, if your *Bitcoin* are on a bad list, they won't turn them into dollars for you. If you come by 100,000 Bitcoin in the wrong way—by hacking someone's Bitcoin account, or by doing a ransomware attack, or by getting them from an address that the government has blacklisted for some good or bad reason—and transfer them into an exchange, the exchange will ask you where they came from. Also the government, and the exchange, can trace the provenance of your Bitcoin and see if any of them were involved in known bad transactions (hacks, ransomware, etc.). And if they were, the exchange can block you from taking your money out and can call the police.

Forbes contributor and rapper (left).

Earlier this year a couple were arrested and accused of trying to launder 119,754 Bitcoin stolen from the Bitfinex exchange in a 2016 hack. (She was a YouTube rapper and a *Forbes* contributor—colorful character!) That's billions of dollars of Bitcoin at today's prices. But these people were not living particularly large, because it turns out billions of dollars of stolen Bitcoin can be extremely difficult to spend and extremely easy to track. Most of the allegedly stolen Bitcoin never left the account where they first landed after the hack. Some were sent to crypto exchanges and converted into dollars, but the exchanges kept meticulous records of who was opening the accounts and withdrawing the money and quickly froze withdrawals. Some were exchanged for gold at a precious-metals dealer, but they had to show a driver's license and give a real home address for that transaction, so the FBI saw who got the gold. Some of the money was spent on a Walmart gift card, and if you're buying a Walmart gift card with your billions of dollars of stolen censorship-resistant currency, then your money laundering is not going well.

F.
Digital Scarcity

We've talked about how Satoshi's essential technological innovation was that he found a way to make numbers on computers *scarce*. And a weirdly important generalization of Bitcoin is that it is a way to create electronic scarcity.

i. Wait, what?

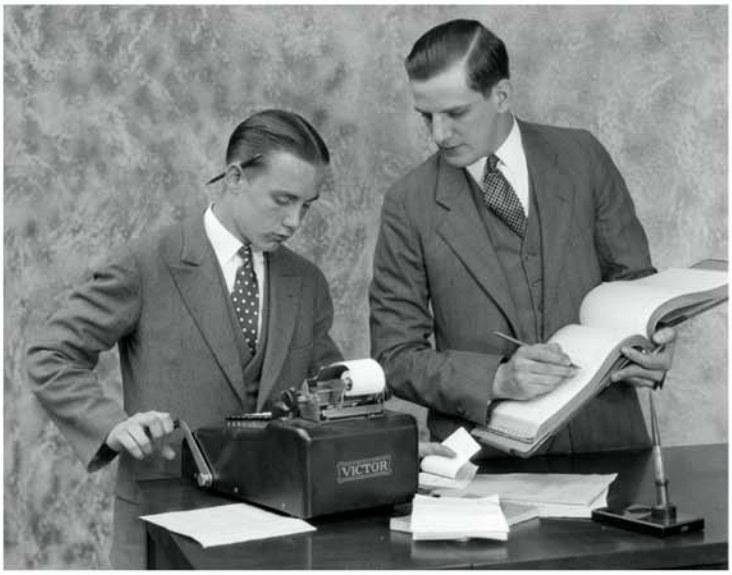
Is that good? For digital cash it is, sure. Dollars are electronic ledger entries that are scarce because a complicated system of banking regulation makes them scarce; banks *can* make new dollars just by changing a number in a database, but there's a lot of ceremony involved in making sure they do that in the right way. If you want a decentralized permissionless system of money without trust or regulation, you need some way to limit how much money there is, and Bitcoin solved that problem.

The broader idea of scarce digital assets is weirder, though. The normal condition of human existence is that good stuff is scarce, and companies make money selling it to us because we can't get infinite amounts of it for free. This isn't always true of everything, but it's generally true of the things that you notice paying for. (In general you can breathe as much air as you want for free, though science fiction occasionally has fun imagining dystopian worlds where air is scarce, expensive, and controlled by corporations. Crypto people sometimes have fun imagining those worlds, too, but thinking they're utopias.)

It's hard to mine coal

or make a chair

or train a tax accountant.



There isn't an unlimited supply of those things, so they cost money. Every so often a scarce thing becomes abundant—a vast oil field is discovered, a robot is invented that can make 100 chairs per hour—and prices drop, and the people who were previously in the business go broke.

So there's a natural intuition that scarcity creates value and abundance destroys it. This is, of course, wrong. You'd rather have more stuff than less. As a consumer, abundance is good. But you can't get rich selling stuff that's infinitely available for free. And you want to get rich.

The modern world has developed ways to get rich by selling stuff that, in theory, is infinitely available because it's electronic. Microsoft Word is not

really scarce: The amount of physical resources (electricity, computing power, disk space) involved in making one more copy of that computer program and putting it on my computer is tiny in the scheme of things, and when I pay for a copy of Word, I'm paying Microsoft Corp. for the effort involved in coming up with Word in the first place, not for the marginal copy. And Microsoft is, in part, in the business of making those copies scarce—of making it *not* trivial to copy its programs, with copy-protection schemes and subscription plans—so that it can charge for them.

Or, if you regularly read Bloomberg online, thank you for paying to get past our paywall. Some of your money is going toward paying for our servers to deliver one more impression of an article to your browser. Not very much of it, though. Most of it is going to me, haha, thanks!

One possible future is that the world will be increasingly like that, at least for some people.[40] Technological progress will make the basic necessities increasingly abundant and also less fun, the physical world will become more homogenous and boring, and everyone will spend more of their time online. Their friendships and romances and family life will occur on computers; their lives will get meaning from stuff that happens on computers.

40 To be clear, there are many possible futures that are much more dystopian. "Actually clean air becomes increasingly rare, valuable, and rationed" would be a future where people don't care very much about the metaverse.

One possibility here is that this will just be nice and egalitarian: Everyone can find their own communities and live in the limitless abundance of the internet. The other possibility is that in an internet world the essential goods will be *positional*. Being in the *cool* internet chat room will be desirable the way living in a fancy house in a good neighborhood is desirable now. Having a *cool* online avatar will be desirable in the way that wearing a nice watch is desirable now. And if crypto is a way to make those things *scarce*, to make the desirable avatars a limited edition available only to trendsetting early adopters and rich people, then you can make money selling them.

This all seems bad to me,

BUT WHAT DO I KNOW?

ii. Rare monkey JPEGs

I guess we have to talk about NFTs again.

Remember, an NFT is just a token with a number. If you buy a Bitcoin, your Bitcoin is identical to anyone else's Bitcoin.[41] If you buy an NFT, it has a number. There will be some series of NFTs—some ERC-721 token series with a name, let's call them Tedious Tamarins—and each NFT in that series will have a number, and Tedious Tamarin No. 63 will be distinguished from Tedious Tamarin No. 64 by having a different number.

41 Bitcoin do not have numbers, but they do have provenance: You can trace where your Bitcoin came from, back to when they were originally mined. One could imagine a world where different Bitcoin had different values, where a Bitcoin that's traceable back to Satoshi's addresses is worth more than a regular Bitcoin, while a Bitcoin that's traceable to a cache of Bitcoin stolen from an exchange in a hack is worth less. That's not the world we live in, for the most part, though the stuff about Bitcoin stolen in a hack being worth less is basically true.

I'm being as trivial as possible, but of course everything on the internet is distinguished from everything else on the internet by having a different series of numbers. That's all internet culture is, numbers encoding pictures and sounds and text and arguments. There's no reason in principle why the number 63 should not encode a tamarin who looks bored but *really cool*, while the number 64 encodes a tamarin who looks bored and kinda schlubby. And so you might pay more for No. 63 because he looks cooler.

What this means *technically* is interesting. The paradigmatic NFT is some piece of digital art: Tedious Tamarins is a series of 1,000 digital drawings of slightly different monkeys with bored expressions, and different tamarins will be worth more depending on how cool they look. (Commonly an NFT series will have a handful of possible attributes—a tamarin might be smoking a cigarette, or wearing a funny hat, or yawning, or closing its eyes, and tamarins will be worth more depending on how many attributes they have and how rare those attributes are. You could imagine the value of different NFTs within a series being based on purely aesthetic considerations, but, mostly not.)

But what does it mean to say that the NFT *is* a piece of digital art? The art does not live on the blockchain. As the well-known software engineer and hacker who goes by the name Moxie Marlinspike wrote in a blog post:

> **Instead of storing the data on-chain, NFTs instead contain a URL that points to the data. What surprised me about the standards was that there's no hash commitment for the data located at the URL. Looking at many of the NFTs on popular marketplaces being sold for tens, hundreds, or millions of dollars, that URL often just points to some VPS running Apache somewhere. Anyone with access to that machine, anyone who buys that domain name in the future, or anyone who compromises that machine can change the image, title, description, etc for the NFT to whatever they'd like at any time (regardless of whether or not they "own" the token). There's nothing in the NFT spec that tells you what the image "should" be, or even allows you to confirm whether something is the "correct" image.**

If you buy an NFT, what you own is a notation on the blockchain that says you own a pointer to some web server. On that web server there's *probably* a picture of a monkey, but that's none of the blockchain's business.

Meanwhile the intellectual-property rights to that picture of a monkey are *certainly* none of the blockchain's business. It's not uncommon for the person or company selling the NFT series to 1) own the IP rights to the pictures of the monkeys and 2) promise to transfer those rights, or some of them, to individual holders of the NFTs. But if that happens, it happens off the blockchain; those promises are or aren't enforceable through the normal legal system. And it's not all that uncommon for the person selling the NFT series *not* to own the IP rights. Early in the NFT boom, it was common for people to make NFTs "of" the *Mona Lisa* or the Brooklyn Bridge. It's just a token with a number; why can't that number refer to "the Brooklyn Bridge"?

In some ways this technical silliness makes NFTs more culturally interesting. If you buy an NFT, all you're getting is a very thin signifier, some fragile pointer to some piece of digital art. And yet people do pay a lot of money for NFTs with the right sort of monkey picture. These NFTs

may not represent ownership in any particularly binding sense, but they represent a *feeling* of ownership. And they also represent a form of *community*.

Worth a shot, no?

The most famous NFT collection is of course the Bored Ape Yacht Club, a series of 10,000 JPEG images of monkeys on Ethereum, some of which sell for millions of dollars. There's no physical yacht club, but there are (IRL) *parties* for Bored Ape owners. Various celebrities, art dealers, and venture capitalists own apes, and owning an ape is a way to join an exclusive club. And that club has its norms, and those norms include "It is cool to use a picture of your ape as your Twitter profile picture" and "It is *not* cool to use a picture of an ape you *don't* own as your Twitter profile picture." The technological and legal connections between blockchain and JPEG and ownership are a bit thin, but the connections are enforced culturally.

iii. The metaverse

I plan to go to my grave not knowing what "the metaverse" is, so I'm certainly not going to explain it to you. But one thing it is is that you can buy some real estate on the internet. Like, there will be a picture of a house on the internet, and you can have an avatar on the internet that lives in the house, and the avatar can walk from your internet house to the internet store, where it can buy some internet groceries. It will be like a computer game, but more all-consuming and much more boring.

How much should your house in the metaverse cost? Two plausible answers might be "It should cost as much as a house, it's a house," or "It should be basically free, it's just a file on a computer, there's essentially infinite space in the metaverse, and nobody has to actually nail together the house." The second answer strikes me as correct, but that's no fun for the people selling houses on the internet.

So, scarcity. The internet houses are NFTs. The goal is to make some of them scarce and prestigious. Having an internet house next door to Elon Musk's internet house is desirable, let's assume, and only a couple of internet houses can be next door to Elon Musk's internet house. Why? They're all on computers; you could put a million internet houses next door to Elon's internet house, but you don't. You make the positional goods scarce.

This is dumb, but, I mean, look at Facebook (Meta, now). What are the odds that, 50 years from now, the center of the world economy will be digital goods? What are the odds that we will all spend most of our time

and money seeking status and romance and connection and entertainment on the internet? If most goods end up being digital, if most people make their money by producing digital goods, then monitoring and metering the distribution of those goods will be an important economic function. Digital scarcity.

I'm sorry to be This Type of Guy, but it's hard not to think of the movie *The Matrix*. You know the premise: Everyone is a sack of meat in a bath of nutrients with their brains plugged into a remarkably realistic simulation of late-'90s America. Why a remarkably realistic simulation of late-'90s America? Why does Neo, the main character, have to go work at a boring desk job if he's just a brain in a vat being fed a soothing simulation by the machines? Agent Smith explains:

DID YOU KNOW THAT THE FIRST MATRIX WAS DESIGNED TO BE A PERFECT HUMAN WORLD WHERE NONE SUFFERED, WHERE EVERYONE WOULD BE HAPPY? IT WAS A DISASTER. NO ONE WOULD ACCEPT THE PROGRAM. ENTIRE CROPS WERE LOST. SOME BELIEVED THAT WE LACKED THE PROGRAMMING LANGUAGE TO DESCRIBE YOUR "PERFECT WORLD." BUT I BELIEVE THAT, AS A SPECIES, HUMAN BEINGS DEFINE THEIR REALITY THROUGH MISERY AND SUFFERING. SO THE PERFECT WORLD WAS A DREAM THAT YOUR PRIMITIVE CEREBRUM KEPT TRYING TO WAKE UP FROM.

So they simulated late capitalism instead. Even in a world where all the goods are digital and available in limitless abundance, you still have to have a (simulated) desk job to pay for them. Digital scarcity.

III
The Crypto Financial System

Let's step back a bit and abstract from what we've discussed so far. Crypto is:

1. A set of tokens, which are worth fluctuating amounts of money. We can say that these tokens are financial assets, like stocks and bonds.[42]
2. A novel set of ways to create new tokens and distribute them and try to make them worth money.
3. A novel way of *holding* financial assets: Instead of the databases that people use to hold stocks and bonds, you can own your crypto on the blockchain.
4. A novel way to write contracts and computer programs (computer programs that are contracts, and contracts that are computer programs).

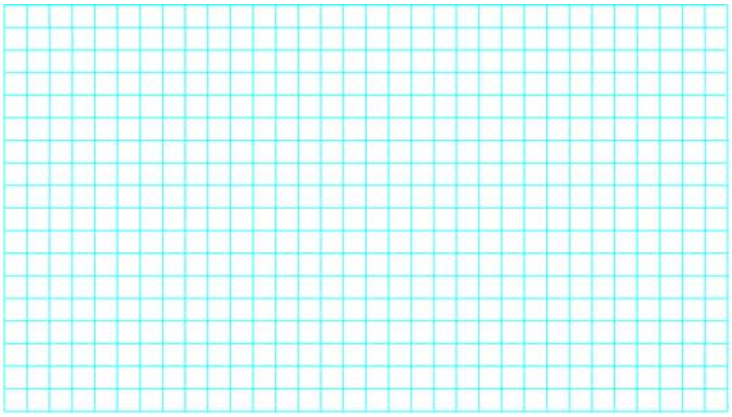
42 Well. That's debatable. Classically, a "financial asset" means a contractual claim on the cash flows of some person or entity: A share of stock represents a claim on the profits of a company, a bond represents a claim on repayment from a company or government, etc. My financial asset represents a liability (or equity issuance) of somebody else; each financial asset has both an owner and an obligor. Some crypto, I would argue, looks a lot like that; it's an equity claim on the value of some crypto-y business. But a lot of crypto is consciously *not* like that. Bitcoin is "digital gold." It's specifically *not* a financial asset; owning a Bitcoin doesn't represent a claim on anyone else. A Bitcoin exists as an independent thing that you can own, not a contractual relationship between parties like stock or a bond. In the text I use "financial asset" in the extremely loose sense of, like, "a thing with a fluctuating price that you can see on your computer screen and that hedge funds can trade." But cryptocurrencies aren't *technically* financial assets, or not always anyway.

If you read only that description, you might object: "Yes, fine, but what is crypto *for*? What do these tokens *do*? *Why* are they worth money?" Nothing in that description answers those objections. I suppose the last one does, in a sense—"Crypto tokens are for building smart contracts for trading crypto tokens"—but it's not a very *good* answer, because it's entirely self-referential. "Yes, fine, but why are you trading the tokens in the first place?"

What *are* you? What do you *do*?

But for now I want to set this objection to the side. If you're a certain sort of financial person—a financial engineer, an arbitrageur, a market-structure enthusiast, a builder of high-frequency trading systems—this abstract set of facts is incredibly, incredibly beautiful. You wake up one day, and there's just a whole other financial system. It's full of smart people building interesting things, and it's full of idiots making terrible mistakes. People have built brilliant new ways to make financial bets that you can use, and they've built insane new ways to make financial bets that you can exploit. How can you not want to join in? It's so *interesting*, so intellectually appealing, such a blank canvas for all of your aesthetic views about how markets should work. Also, ***so many idiots are getting rich; why shouldn't you?***

There are other appealing properties when you compare this system to traditional finance. The crypto system is, philosophically, one of *permissionless innovation*. The workings of the major blockchains are public and open-source; if you want to build a derivatives exchange or margin-lending protocol or whatever in Ethereum, you just do it. You don't need to set up a meeting with Vitalik Buterin to get his approval. You don't need to negotiate access and fees with the Ethereum Corp.; there is no Ethereum Corp. Anyone can try anything and see if it works.



If you're a smart young person coming from traditional finance, this feels liberating. If you're used to spending months negotiating credit agreements with prime brokers and setting up direct access to trade on a stock exchange, the idea that you can just *do* stuff in crypto, with no preliminaries, is amazing. Obviously it's a bit alarming as well: Some of those long, slow processes in traditional finance are there to prevent money laundering or fraud or ill-considered risk-taking. But, empirically, a lot of them aren't really preventing any of those things, or aren't doing so in an optimal way. A lot of them are just How It's Always Been Done. Nothing in crypto is How It's Always Been Done; it's all too new.

And so people invented a financial system for crypto. It runs alongside the traditional financial system, though they touch at many points. In some ways it looks a lot like a copy of the traditional financial system. In other ways it looks totally different. In some ways it's a streamlined and modernized and innovative evolution of the traditional system. In other ways it's a chaotic and stupid devolution of the traditional system, a version of traditional finance ("TradFi," as crypto people call it) that unlearned important historic lessons about fraud and leverage and risk and regulation.

It's so fun. Let's talk about it.

A.
Your Keys, Your Coins, Your Hard Drive in a Garbage Dump

i. Holding crypto

Maybe the first thing to say about the crypto financial system is that the traditional financial system is deeply *intermediated*, and the crypto system is not. If you have money, your bank tracks your money for you; if you have stock, your broker tracks your stock for you; etc.

One dumb, simple thing that this means: If you have money in the bank, your bank has to give it to you. If you forget the PIN code for your ATM card or if you forget the password for your online banking, you'll have a hard time taking out money, and that will be inconvenient for you. But the bank owes you the money; they can't just be like, "Aha, got you, the money is ours now." There's some process by which you can go into the bank and prove that you are who you say you are, and they're like, "Fine, we'll reset your password, it's Test1234, don't forget this time."

Forgot your username or password?

Forgot your password?

Forgot Password

Forgot Password

Forgot username/password? >

Forgot ID/Password?

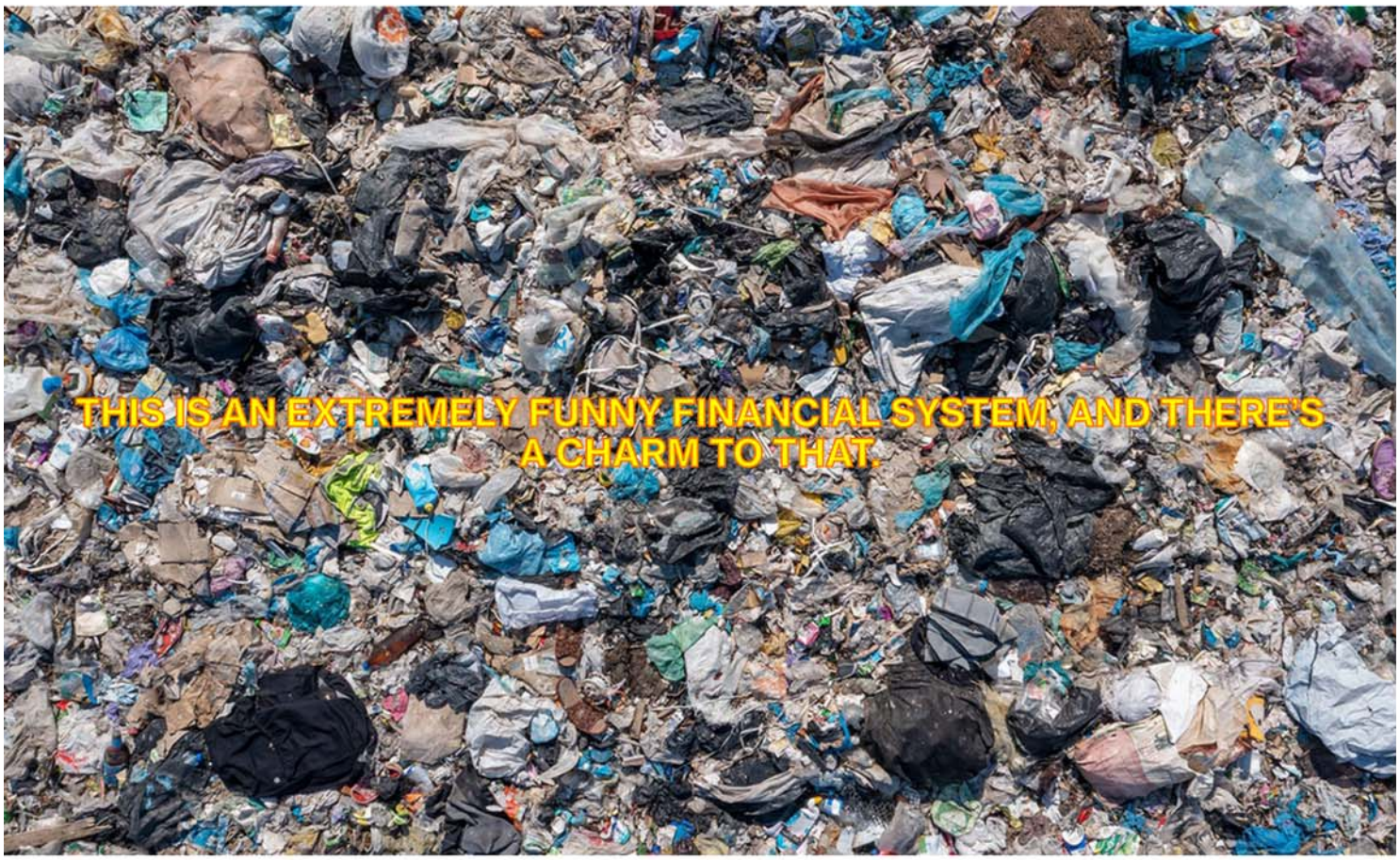
Crypto doesn't *necessarily* work like that. Owning Bitcoin means 1) having a public Bitcoin address with some Bitcoin in it and 2) possessing the private key to that address. If you have the public address/private key pair, then you own the Bitcoin; you can transfer them to someone else on the blockchain. If you don't have that pair, then you can't. If you lose your private key or lose track of your public address, there's no one to recover your password for you or give you back your Bitcoin. They're just gone.

There are ways to, you know, not lose your keys. Mainly people use "software wallets," which generate and keep track of their addresses and private keys and allow them to sign transactions and send and receive crypto online. The wallets may be desktop or phone apps, or extensions on a browser. Most modern wallets require you to keep track not of private keys, but rather a "seed phrase" of, typically, 12 random-looking words. Maybe: "army truth speak average" and so on.

The phrase can be used as a seed to generate lots of public-private key pairs, so the wallet can create lots of different addresses that can all be recovered from a single seed phrase. (People often speak of wallets "holding" crypto, but really what they have are these keys for various addresses; the crypto is only ever on the blockchain.) Then you write the seed phrase down on a piece of paper, which is easier to write than a long random number.

There's still hope, maybe?

But this is a developing technology, and there's a long history of people losing, forgetting, or throwing away their private keys or seed phrases. A guy in Wales named James Howells periodically pops up in the news, because he threw away a hard drive with the private key for 8,000 Bitcoin. He's pretty sure he knows the garbage dump that has his hard drive, and for years he's been waging a campaign to dig up the dump and sift through the garbage. If he finds the hard drive and if it still works, then he'll have Bitcoin worth about $150 million, which he can use to pay all the garbage diggers. In one sense it would be much, much better to have a financial system in which the bank could reset his password and give him back the 8,000 Bitcoin, instead of digging up a garbage dump. In another sense



ii. *Not* holding crypto

If you're a portfolio manager at an institutional investor, and you want to buy Bitcoin, and you go to your compliance and operations people and say, "I want to buy Bitcoin, and I will write our private keys down on a Post-it that I will keep next to my computer," they will say no. If you propose better security measures, they might still say no: This stuff is too new, too scary. If you say, "MegaBank Custody Services will hold our Bitcoin for us, and we'll just have a book entry on their ledger saying we have Bitcoin," then compliance might be a bit more comfortable, but that requires MegaBank to offer that service.

But what if you go to compliance and say, "I'm just going to enter into a bet with a hedge fund on the price of Bitcoin. For every dollar that Bitcoin goes up, the hedge fund will pay us $5; for every dollar that it goes down, we'll pay them $5." Then you *don't have to own crypto at all.* All you have is an over-the-counter derivative with a hedge fund. Compliance knows what that is; that's an understandable thing. There are no weird custody issues there, because there's nothing to keep custody of. There are no weird "coins" to worry about, just a contract with a hedge fund to pay you money. You can write contracts on the price of corn,

on interest rates,

and on hurricane damage,

so why not Bitcoin? You do due diligence on the hedge fund, you get comfortable with the credit and collateral terms, and then you sign the contract. And then, *economically*, you own some Bitcoin–your investment goes up when Bitcoin goes up, and down when Bitcoin goes down–without the worries of *really* owning Bitcoin. No keys to lose.

And so, in traditional finance there's a big business offering those instruments. CME Group Inc. offers Bitcoin futures, which are basically the bet that I outlined above: You pay me $5 for every dollar that Bitcoin goes up, and I pay you $5 for every dollar it goes down. It's a trusted,

centralized, traditional-finance way to bet on the price movements of Bitcoin.

Still, not everyone can buy futures, which require a lot of money and are not offered by some retail brokerages. In the US financial system, pretty much the easiest thing to invest in is *stocks*. So wrapping Bitcoin in a *stock* would increase its appeal. The easiest way to do this would be a cash Bitcoin exchange-traded fund, a pot of money that trades like stock on a stock exchange and invests the money in Bitcoin. People keep trying to do this, but the US Securities and Exchange Commission remains skeptical and hasn't approved cash Bitcoin ETFs, though they exist in some other countries. The US has, however, approved Bitcoin *futures* ETFs, which invest in Bitcoin through futures contracts. Two layers of abstraction: a Bitcoin, wrapped in a futures contract, wrapped in a stock, and delivered to your brokerage account.

B.
CeFi

i. Fiat on-ramps

OK, I said one way to own a Bitcoin is to write your private key on a Post-it note. But where did you *get* that Bitcoin?[43] The main way people in most of the world get into crypto is that they exchange dollars (or euros, pounds, yuan, etc.–what crypto people call "fiat") for crypto. And how they do that is a tricky question.

43 In the early days of Bitcoin, a reasonably plausible answer would be "I mined it": Every Bitcoin ever created has come from mining, and early Bitcoin was in part a hobbyist mining operation. In modern crypto, you're not going to get very far just by mining crypto on your home computer.

One simple way is for you to find someone with crypto and say, "Hey, can I buy some of your crypto?" They say sure, and you arrange a deal. In researching this article, I set up an Ethereum wallet and texted a friend to ask if he had any Ether that I could buy. He replied sure, and I said, "Send me $20 worth" and gave him my public address. He sent me $20 of Ether, and then he texted me to say, "Done, Venmo me $20." I did, and our transaction was complete.[44] He sent me the Ether before I sent him the dollars, so he took some credit risk, and in fact I was away from my phone when he sent it, so he ended up taking my credit risk for several hours. When I did get his text, I briefly considered that it would be *very* funny if, as part of my research for this article, I *stole* $20 of Ether from him, but that seemed mean.

44 Bloomberg's code of journalistic ethics forbids journalists who write about crypto—including me—from owning more than a "nominal" amount of crypto for research purposes. (This rule itself is arguably biasing—it forbids crypto enthusiasts from writing about crypto—but never mind.) In researching this article, I bought $20 of Ether from a friend by Venmo, and $100 of Ether from Coinbase; this also led to Coinbase giving me, like, $5 of free Bitcoin as a new-account bonus. In seeking permission to do this, I told my bosses that 1) I planned to lose all my money, but 2) if I accidentally made any profit, I would donate it. (I did spend, like, $8 of Ether on the matthewlevine.eth domain.)

I'm not going to lie: This was fun to do and fun to write about–it felt sillier and more exciting than my automatic monthly Vanguard investments–but it's not a good way to run a financial system.[45] *In general*, if you get a text asking you to send crypto to a string of letters and numbers, you should throw your phone into the ocean.

45 Incidentally, the day after I did that trade, I met with the team behind a decentralized finance protocol and was like, "Well, how would *you* quickly turn dollars into Ether in New York?" And they had no great answers and were like, "We're working on it." And I was like, "Well, I texted a friend for Ether and then Venmoed him the money." And they were like, "Yeah, that's probably what we would do."

Instead, the main way that normal people buy crypto is through *crypto exchanges*, specifically *centralized* crypto exchanges. Crypto exchanges are companies–Coinbase, Gemini, Binance, FTX, Kraken, and Bitfinex are some big ones–that accept regular money for crypto. You wire an exchange $100, and it gives you $100 worth of Bitcoin, perhaps minus a fee.

In the olden days, the stereotype was that a lot of crypto exchanges were run by criminals or incompetent teenagers, or incompetent teenage criminals. The standard crypto-exchange transaction was 1) you exchanged your dollars for Bitcoin to buy heroin, and then 2) the exchange got hacked and lost your Bitcoin before you could even buy the heroin.

Modern crypto exchanges are less like that. For one thing, they're more careful and technically adept, so they're less likely to lose your Bitcoin. For another thing, though, they're big companies, regulators are aware of them, and they try to be good corporate citizens. In their role as on-ramps and off-ramps between traditional currencies and crypto, they do the same sorts of anti-money-laundering and know-your-customer checks that traditional banks and brokerages do. If you show up at Coinbase–a US public company!–with a sack of dollar bills that you got from dealing heroin and try to convert them into Bitcoin, Coinbase will turn you away and probably report you to the police. The centralized exchanges are very much part of the regulated financial system these days. The days of crypto being a zone of utter lawlessness are mostly gone.

This is a series of trade-offs. Roughly speaking, the crypto exchanges of the olden days let you trade dollars for Bitcoin without asking any questions, but they might steal your Bitcoin: When the exchanges were unregulated and crime-positive, the odds of them *doing crime to you* (or having crime done to them) were pretty high. The modern crypto exchanges ask a lot of questions and make it difficult for you to move tons of money in secret, but they probably won't steal your Bitcoin.

ii. Custodians

So you've opened an account at an exchange and sent the exchange $100 to buy $100 worth of Bitcoin. What does the exchange give you for your $100? One possibility: It gives you 0.005215 Bitcoin. It sends you some instructions on how to set up a Bitcoin wallet, it asks you for a public Bitcoin address, it converts $100 to Bitcoin at the current market price, and it sends you that number of Bitcoin at your public address. And then you access those Bitcoin using your private key for that address.

This is suboptimal for the exchange. For one thing, dollar and Bitcoin transactions have different time frames and finality. If you fund your account with a bank transfer or a credit card, and then you buy $100 of Bitcoin, and then you call your bank and say, "I've been defrauded, I don't recognize that charge," there's a decent chance the bank will take the $100 from the exchange and give it back to you.[46] Meanwhile, the Bitcoin transfer to your wallet is fast and irreversible.

46 This is true of any other transaction: If you buy a meal from a restaurant, you can try the same scam, and in fact big retailers lose some amount of money to "chargebacks." This is part of why Satoshi invented Bitcoin, to avoid this sort of reversibility. But the problem is probably bigger for crypto exchanges: They're dealing with large, purely financial transactions. Bitcoin is volatile, and if it goes down after you buy it, you might try to get your money back, and if you *do* report it as fraud to your bank, your bank is historically more likely to be suspicious of a crypto exchange than it is of a big-box retailer.

For another thing, you know the exchange will get some customers who don't write down their wallet's seed phrase (or lose it or forget it) and then can't access the Bitcoin. And they'll call the exchange's customer service number and say, "I lost the password to my Bitcoin account, can you reset it?" And the exchange will say, "No, it doesn't work that way, also we don't have a customer service number." And the customers won't like it. "I paid you $100 for Bitcoin, and I don't have the Bitcoin," they'll say, and blame the exchange and complain to regulators and law enforcement and the press.[47]

47 And, especially, to their bank or credit card company.

Now, these problems are annoying but solvable, and modern crypto exchanges do some amount of this, acquiring crypto for "self-custodying" customers. But there's a simpler possibility that is also quite popular. The exchange could hang on to your Bitcoin for you. Instead of sending you

0.005215 Bitcoin on the Bitcoin blockchain, it could go out and buy 0.005215 Bitcoin and put them into its own Bitcoin wallet. Being a professional Bitcoin exchange, it could put in the effort to keep these Bitcoin safe and not lose the keys.

And then instead of sending you 0.005215 Bitcoin, the exchange just keeps a database of its customers and their account balances. And your entry in the database includes your name, your driver's license number, your account number, your email address, your phone number, your password,[48] your mother's maiden name, and your account balance, and the exchange writes 0.005215 in the balance field.

48 Just kidding. A hash of your password.

And then when you log in to your account, it displays "0.005215 Bitcoin" as your account balance, and you think you own 0.005215 Bitcoin. And you're not exactly wrong. But really what you own is a claim on the exchange for 0.005215 Bitcoin. You don't own them directly, and you don't control the private key. You just have an entry on the ledger of the exchange.

You know. Like a bank!

If you have a bank account, the bank owes you money, and you trust it to keep a record of that; if you have a crypto exchange account, it's the same, but the exchange owes you Bitcoin. One thing this means is that if you lose your password, you can call the exchange, and it can reset it for you. The customer service can be a bit better.

There are some obvious downsides. One big one: It's like a bank! If you got into Bitcoin because you don't trust banks and you want to be in control of your own money, it's somewhat weird, philosophically, to just go and trust a crypto exchange to keep your money for you.

These days the big crypto exchanges seem to be mostly law-abiding, and you can get rich enough running a legitimate crypto exchange that it seems silly to steal the money instead. But another downside is hackers. A crypto exchange has a giant pot of money, and it has to move that money around a lot to deal with customer transactions. It's an appealing target for hackers looking to steal private keys. Again, modern crypto exchanges spend a lot of money on information security, but that wasn't always the case, and there's a long history of Bitcoin exchanges being hacked. Or "hacked." When all the Bitcoin in an exchange's wallet get stolen, it can be hard to tell, sometimes, whether they were stolen by outside hackers or by the exchange's CEO.

Also, while I suppose an exchange is less likely to lose its private keys than the average customer is, it can happen. In 2018 the CEO of Quadriga Fintech Solutions Corp. died in somewhat mysterious circumstances while on vacation in India. At the time, the company's QuadrigaCX was Canada's largest crypto exchange, and it was apparently run entirely off of its CEO's laptop. When he died, he took all of Quadriga's private keys with him, meaning its customers' Bitcoin were lost forever. Or that's what it would've meant, except that before he died he also stole all the customers' Bitcoin, so the wallets whose keys disappeared with him were empty anyway. When crypto exchanges are bad, they tend to be bad in all ways at once.

iii. Also exchanges, though

Centralized crypto exchanges are on-ramps to crypto for people with dollars and other traditional currencies, but they're also exchanges. If you have some Bitcoin in your Coinbase account, and you'd rather have Ether, you can sell your Bitcoin for Ether. If you want to actively trade among cryptocurrencies, to make bets on which will go up more, you can do that on an exchange.

In traditional finance there tends to be a division between "exchanges" and "brokerages." If you want to buy stock, you open an account at a brokerage such as Charles Schwab, Fidelity, or Robinhood, and you send your broker an order to buy stock. The stock exchange is a place for big brokerages and institutions to trade stock; retail customers need an account with a broker to access the exchange.[49] There are many layers of intermediation.

49 In practice, modern US stock trading is even more intermediated than this, and your order may get sent to an electronic trading firm and not the stock exchange.

In crypto that's not generally true: Big crypto exchanges such as Coinbase or FTX let anyone open an account and trade crypto directly on the exchange, and you wouldn't normally connect to the exchange through a broker.[50] Every step that goes into making a trade happen–getting your money into your account, taking your buy order, matching your order with someone's sell order, settling the trade, putting the crypto in your account, keeping track of your account–is done by the exchange.

50 Although traditional retail stock brokerages are increasingly getting into the business of buying crypto for their customers.

Let me spend a bit more time on one of those functions: providing leverage. If Bitcoin isn't exciting enough for you, you can find an exchange that will let you borrow money to buy more of it. You put in $100, the exchange lends you $900, you get $1,000 worth of Bitcoin. If Bitcoin goes up 10%, you double your money; if Bitcoin goes down 10%, you lose everything. Bitcoin goes up and down by 10% a lot, so this is an exciting way to gamble.

Traditional finance also provides leverage, but it's a complex and intermediated system involving brokers and clearinghouses. Crypto exchanges are more integrated, so in many cases a crypto exchange is basically in the business of managing market risk. Say instead of Bitcoin going down 10%, it falls 15%; you're not only down your original $100, but now you *owe* $50. Crypto exchanges have to decide when to make you post collateral–put up more money–to ensure that you're good for your losses and when to liquidate your position so you don't lose more than you can pay back.

A crypto exchange may have customers with big leveraged bets on Bitcoin rising (they're "long," in the language of finance) and customers with big

leveraged bets against Bitcoin (they're "short"). If Bitcoin moves too far in one direction too quickly, then the long (or short) customers will be out of money, which means there won't be money to pay back the short (or long) customers on the other side. The exchange has to think about how volatile its assets are, set leverage limits so blowups are unlikely, and monitor leverage levels to ensure no one is in imminent danger of blowing up. If someone *is* likely to blow up, the exchange has to seize their collateral and sell it, ideally in an intelligent way that doesn't destabilize the market too much. And in periods of high volatility the exchange might shut down trading rather than deal with all this. That's a lot of centralized decision-making.

C.
Stablecoins

Bitcoin is good at keeping track of who has Bitcoin. This is technologically interesting and also useful insofar as Bitcoin are a store of wealth. And if Bitcoin were the dominant currency in the world—if it were "digital cash," and you could use it to buy stuff and the prices of things were set in Bitcoin—then it would be even more useful. But it isn't. You use dollars or pounds or yen or euros to buy stuff, and the price of Bitcoin in dollars (etc.) is very volatile.

One thing that would be cool is if crypto could keep track of *who has dollars*. Then you could get the benefits of crypto (decentralization, smart contracts, skirting the law) along with the benefits of dollars (your bank account isn't incredibly volatile, you can buy a sandwich). A stablecoin is a crypto token that's supposed to always be worth $1.[51] If you have a stablecoin, then you have $1 on the blockchain. You hope.

51 Or one unit of some other traditional currency, though I'll stick to dollars.

i. Collateralized

The simplest sort of stablecoin is what's sometimes called a "fully backed" stablecoin; popular examples include USDC and Tether. The idea here is:

1. **SOME REASONABLY TRUSTWORTHY INSTITUTION SETS UP A STABLECOIN FACTORY TO ISSUE STABLECOINS ON ONE OR MORE POPULAR BLOCKCHAINS.**
2. **YOU GIVE THE ISSUER $1.**
3. **IT GIVES YOU BACK ONE STABLECOIN.**
4. **IT PUTS THE DOLLAR SOMEWHERE SAFE.**
5. **IF YOU EVER WANT YOUR DOLLAR BACK, YOU GIVE THE ISSUER ONE STABLECOIN, AND IT GIVES YOU BACK THE DOLLAR.[52]**

52 Or maybe not "you," but an institutional investor. This description makes a stablecoin issuer an on-/off-ramp between regular currency and crypto, which is risky for the reasons we discussed earlier (regulatory, credit, etc.). If you want to hand the issuer a stablecoin and get back a dollar, it will probably at least want to do know-your-customer checks to make sure it's OK to give you dollars. Or the issuer might limit redemptions to some list of large institutional investors, rather than letting just anyone show up with a stablecoin. This is probably fine: If there are enough institutions that can redeem stablecoins for dollars, then *they* will buy stablecoins for around a dollar, and *you* can just sell your stablecoins to them.

Your stablecoin lives on some blockchain[53] and can be traded and used like any other token on that blockchain. If you have 10,000 dollar stablecoins on the Ethereum blockchain and you want to buy some Ether, you can buy $10,000 worth of Ether with your stablecoins without putting more dollars in. And if you have $10,000 worth of Ether, you can sell it for 10,000 dollar stablecoins. Having 10,000 dollar stablecoins is like having $10,000, but the stablecoins live on the blockchain–in your crypto wallet–rather than in a bank account.

53 Most of the big stablecoin issuers issue on multiple blockchains. Tether lives mostly on Tron and Ethereum. USDC lives mostly on Ethereum but is also on Solana, Tron, Avalanche, and (to a lesser extent) four other blockchains.

This is useful if, for instance, you don't have a bank account. Or if you don't live in the US, and it's hard for you to set up a dollar-denominated bank account. Or if you plan to use the $10,000 to buy more crypto later, and you don't want to move it back and forth between the regular and crypto financial systems. Or if you want to send the $10,000 to a smart contract, which can deal directly with your crypto wallet but which has a hard time talking to a bank. Banks are suspicious of crypto, and crypto is suspicious of banks, so it's always a bit painful to connect the crypto system to a bank. "This smart contract will send me dollars if the Jets win this weekend": no, bad, doesn't work. "This smart contract will send me stablecoins if the Jets win": yes, fine. Stablecoins are "wrapped" dollars, dollars that live on the blockchain.

More generally, this is useful if you think the crypto financial system is *better* than the traditional one. If sending tokens over a crypto blockchain is faster and cheaper than sending dollars by interbank transfer, then stablecoins are a better way to send dollars. If the blockchain lets you develop interesting derivatives contracts and trading applications in a quick and permissionless way and the traditional financial system doesn't, you'll want to use stablecoins instead of regular old dollars.

One important point about the collateralized-stablecoin model is that it requires you to trust the issuer. The *dollar-ness* of the stablecoin happens, as it were, entirely off the blockchain. As a crypto matter, what you have is a receipt for $1 from some institution that you trust. If that institution incinerates all the dollars, that receipt shouldn't be worth a dollar.

You may remember Brock Pierce as a young Gordon Bombay in *D2: The Mighty Ducks*.

One of the longest-running and funniest controversies in crypto is about where Tether, the biggest stablecoin, keeps its money. Tether is replete with colorful characters (the *Mighty Ducks* guy, etc.), and they go around boasting about how transparent they are without actually saying where the money is. They also go around promising to publish an audit but never do it. They probably have the money, more or less, but they seem to be going out of their way to *seem* untrustworthy. Still, people trust them.

- **You can collateralize other things**

The collateralized-stablecoin model is a way to *wrap* noncrypto assets and put them on the blockchain. You could imagine all sorts of assets getting wrapped.

For instance, what if you wanted to trade stocks, but in crypto? You might want to do this for reasons similar to why you might want to have dollars, but in crypto. You *like* the crypto system, the smart contracts, the permissionless innovation, the decentralized exchanges, frankly the lack of regulation. But you also like stocks, which represent ownership in productive enterprises and are also a very popular tool for speculation. The crypto system doesn't talk all that well to the stocks system. Your broker is suspicious of crypto, and crypto is suspicious of your broker (less and less so, but still). Putting the stocks on the blockchain lets you trade the things you want (stocks) the way you want (in crypto).

Conceptually, one way to do this is that some fairly trustworthy institution buys a bunch of Tesla Inc. stock and holds it in its vault. And then it issues "wrapped Tesla" tokens on some blockchain: Each wTSLA token corresponds to one share of Tesla that the institution has in its vault, and you can trade them on the blockchain exactly as you would Ether.

And in fact this exists. FTX, a leading centralized crypto exchange, offers "tokenized stocks" on the Solana blockchain, though not to US customers; Binance, another leading exchange, offered tokenized stocks for a while but then stopped. If you're going to try this, you'll want to run it by your local securities regulator. It's legally sensitive to find new ways to sell people new, not-quite stocks.

But if it works it's interesting. For a crypto exchange, it's a way to keep your retail gamblers gambling on your platform: Someone who wants to bet on Bitcoin and Ether and Tesla stock can do all of it on one crypto exchange.

More broadly, though, it's a way for the crypto financial system to *ingest* the traditional financial system. Have a financial asset? Put it in a box and issue tokens about it. Now it's a crypto asset. If the crypto financial system is *good*—if the computer programs, payment rails, and institutional structures of crypto have competitive advantages against the programs, rails, and structures of traditional finance—then some people will prefer to trade their stocks or bonds or other financial assets in the crypto system.

ii. Algorithmic

The fully backed stablecoin model has problems. One is that you might not trust the issuer of a backed stablecoin. Another is that you might not *want* to trust any issuer. An issuer of a fully backed stablecoin is, by necessity, using the US dollar financial system. It's keeping the backing dollars in a bank or in other traditional-finance dollar instruments. It might be subject to the regulatory pressures of that system.

What you want is something that's worth a dollar but exists purely on the blockchain. Can that be done?

● **Good algorithmic**

Sure! And with a fairly simple and traditional bit of financial engineering.

It starts with leverage, or just borrowing money. Leverage is a way to amplify the risks and returns of betting on crypto: Instead of putting in $100 and buying $100 worth of Bitcoin and making $10 if Bitcoin goes up 10%, I put in $100 and borrow $100 and buy $200 worth of Bitcoin and make $20 if Bitcoin goes up 10%. Or I lose $20 if Bitcoin goes down 10%. Or I lose everything if Bitcoin goes down 50%.

For me, that's a high-risk, high-reward proposition. But what if I borrowed the money from you? What does that proposition look like for you? Well, if Bitcoin goes up 10%, you get back $100: I sell my Bitcoin for $220, give you back $100 and keep the rest. And likewise:

> If Bitcoin goes up 20%, you get back $100. (I sell for $240, give you back $100, and keep $140.)
>
> > If Bitcoin goes down 20%, you get back $100. (I sell for $160, give you back $100 and keep $60.)
> >
> > > If Bitcoin goes down 49.5%, you get back $100. (I sell for $101, give you back $100 and keep $1.)

Let's stop there, for no particular reason.

At every point that I've named so far, you get back $100. You put in $100, and you get back $100, no matter what. For me this trade is very risky. (If

Bitcoin fell 49.5%, I lost 99%.) For you this trade is very safe. Very stable. What you have is a stablecoin. You put in $100 and get back $100.

This basic idea is called an "algorithmic stablecoin." You put in $100 and get back a thing that's worth $100, with that value guaranteed by a larger amount of a volatile cryptocurrency. I've described this as just a direct loan from you to me (a contract), but ordinarily this would be done as a smart contract, a computer program on a blockchain.

Vitalik describes a rudimentary form of this in the original Ethereum white paper, calling it a "hedging contract." Here's a slightly different example that involves no dollars at all: Say you and I each put 1,000 Ether into a smart contract, and that when we do it, 1,000 Ether is worth $1 million. On some preset maturity date, the contract will send you $1 million worth of Ether, and I will get whatever Ether is left. If Ether doubled in value in that time, that means you'll get 500 Ether worth $1 million, and I'll get 1,500 Ether worth $3 million. On the other hand, if Ether's dollar value fell 50%, you get all the Ether in the pool so you get your $1 million back, and I get nothing. You get back $1 million no matter what. The smart contract, though, *never held any dollars at all*. There's no bank account, no Treasury bills, no trusted central intermediary, but you have a claim with a steady value in US dollars. We have in a way manufactured dollars purely out of crypto.

A few points. First, I'm being far too cute here: In my original example, if the price of Bitcoin falls by 51%, my $200 worth of Bitcoin will be worth $98, and I won't have enough money to pay you back. Your supposed stablecoin is worth 98¢.

It's not easy to manufacture stablecoins out of extremely unstable assets. Your algorithmic stablecoin has some risk of becoming unstable. But you can do a little better than the crude version I've proposed here. You could have margin calls, for instance, so that if Bitcoin falls by more than 20%, the smart contract sells the remaining Bitcoin to pay you back immediately.[54] (Actual algorithmic stablecoins—Dai, the stablecoin of MakerDAO, is a big one—work this way.)

54 Or you could just have more overcollateralization: Instead of putting up $200 worth of Bitcoin as the asset backing $100 of stablecoins, we put in, like, $400. (I have to put in $300 for you to lend me $100.) This isn't a popular solution. People say it is "capital-inefficient," meaning roughly that they find more leverage more exciting.

Second, I am omitting *interest*. In the real world, if I'm borrowing $100 from you to buy Bitcoin, I'm not just paying you back $100; I'm paying you back with interest, $101 or whatever. An algorithmic stablecoin—much like a dollar in a bank account—can potentially generate interest.[55]

55 Honestly, so can fully backed stablecoins! Tether is earning interest on its cash, so why not pay that to holders of Tethers? Early stablecoin development happened in a very low-rate environment, and a norm developed of not paying interest, but that seems unlikely to last forever.

Third, note that there are two sides to this trade. Some people want stablecoins and will put money (or Ether, Bitcoin, etc.) into this sort of smart contract to get stablecoins; others want leverage and will borrow money from this sort of smart contract to take leveraged positions in risky crypto assets. There are different appetites for risk, so there's a trade to be done.

Fourth: Do you know what a bank is? Not in crypto, just in the world. Here's what a bank is. Some people want to borrow money to make investments. The main investments are 1) starting or expanding a business and 2) buying real estate. They borrow money from a bank; they get leverage. If their business does well or their house's price goes up, they pay back the money to the bank, with interest. The bank has the *senior claim* on their business or house–if there isn't enough money for everyone, the bank gets paid first. Sometimes the bank loses money, but mostly it gets paid back on most of its loans.

The bank, then, is just a pool of these loans, just a lot of senior claims on a lot of businesses and houses. These loans are called the bank's "assets." Then the bank itself goes and borrows money. A bank with $10 billion of assets–$10 billion of mortgages and business loans–might have $1 billion of its shareholders' money ("equity" or, for a bank, "capital") and borrow the other $9 billion. It borrows the $9 billion from its customers in the form of deposits. A bank deposit is, formally, a loan to the bank; if you deposit $100, the bank owes you that $100. And it uses that $100 to fund loans to businesses and homebuyers.

Wait, though. I said (way, way) earlier that a dollar is just an entry in the bank's database. When you have a dollar, what you have is an entry on the books of the bank–that deposit is the dollar. The deposit *is* the dollar, and yet it's also the *debt* of the bank. Your entry on the bank's database shows both that the bank owes you a dollar and *is* the dollar that you own. Isn't that incoherent? How can the bank owe you the dollar, if you have it already, right there in your account?

The modern banking system is a machine that takes in risky assets at one end, takes senior claims on them (lending money against those assets, with the right to be paid back first), repeats that move a few times (taking and issuing senior claims on the senior claims[56]), and spits out *dollars* at the other end. A dollar is distilled from risky assets.

56 In practice, this can be repeated many times. Banks would sometimes buy senior tranches of collateralized debt obligations made up of senior tranches of mortgage-backed securities made up of senior loans on houses; if you deposited money in that bank, you had a senior claim (the deposit) on a senior claim (the CDO tranche) on senior claims (the MBS tranche) on senior claims (the mortgages) on houses.

The stablecoin thing is nothing new. It's just *banking*, but banking in a particularly clear way,[57] purified in smart contracts on the blockchain.

57 And also banking in a largely *unregulated* way, at the moment. And without deposit insurance.

● Bad algorithmic

To summarize the previous section: If you have a large quantity of risky tokens, you can with a little financial engineering issue a smaller quantity of claims on those tokens and call them stablecoins. But recall also one of the simplest lessons of Bitcoin, which is that you can make up an arbitrary token that trades electronically, and people might pay you money for it. (Worth a shot, no?)

These two insights—you can make up a token and it will be worth money and you can use claims on risky tokens to make a stablecoin—can be combined in a natural way to create a disaster. Here's the disaster:

1. I make up a cryptocurrency. Call it Sharecoin. I list it on exchanges and try to sell it to people for some money. (Maybe Bitcoin or Ether or dollars or Korean won—it doesn't matter.)
2. I make up another cryptocurrency. Call it Dollarcoin. I set up a smart contract saying that one Dollarcoin can always be exchanged for *$1 worth of Sharecoin*. I can do this, because I just made up Sharecoin, and the smart contract can issue any old amount of it. If Sharecoin trades at $20, the smart contract will give you 0.05 Sharecoins per Dollarcoin. If it trades at $0.001, the smart contract will give you 1,000 Sharecoins. It doesn't care; it can make all the Sharecoins it wants.
3. Conversely, if you want Dollarcoins, you buy $1 worth of Sharecoins and deliver them to the smart contract to get back a Dollarcoin.
4. As long as there's *some* price for Sharecoin, the smart contract can issue a dollar's worth of Sharecoins for any Dollarcoin that someone brings to exchange.
5. "See, a Dollarcoin should always be worth a dollar," I say, "through the power of algorithms."

The flaw in this logic is in Step 4: There's absolutely no reason for Sharecoin to be worth anything at all—I just made it up!—and so no reason for a Dollarcoin to be worth a dollar. But of course everything in crypto was made up, by somebody, in the recent past, so this objection is not as compelling as you might think. People might believe in this story or just in the general vibe of Sharecoin and Dollarcoin. They might buy Dollarcoin and treat it as worth a dollar, and buy Sharecoin and treat it as a valuable component of a thriving ecosystem.

At some point the process reverses. People start to want dollars rather than Dollarcoins, so some of them sell Dollarcoins for dollars on the open market. This pushes the price of Dollarcoin slightly below $1, perhaps to 99¢. Other people get nervous, so they go to the smart contract —which is supposed to keep the price of a Dollarcoin at $1—and trade Dollarcoins in for $1 worth of Sharecoins. Then they sell those Sharecoins, which pushes down the price of Sharecoin, which makes more people nervous. They trade even more Dollarcoins for Sharecoins and sell those. This pushes the price of Sharecoin lower, which creates more nervousness, which leads to more redemptions at lower Sharecoin prices and even more Sharecoin supply flooding the market. This is a well-known phenomenon in traditional finance (it happens when companies issue debt and commit to paying it back with stock), and it has the technical name "death spiral." It's as bad as it sounds.

A couple of algorithmic stablecoins have death-spiraled, the most famous one being TerraUSD. Terra was a blockchain ecosystem with a native currency (like our Sharecoin) called Luna and a stablecoin called TerraUSD. Billions of dollars of TerraUSD were issued, backed by algorithmic conversion into $1 worth of Luna. TerraUSD was popular, because stablecoins are popular and also, to be fair, because you could get 20% interest rates on TerraUSD. The total amount of TerraUSD reached $18.5 billion, the market capitalization of Luna rose above $40 billion, the system all worked, and then it didn't. A quick death spiral hit in May, and Terra unraveled completely. By the end of the month, TerraUSD was

trading below 2¢, zillions of Luna had been issued and were trading at essentially zero, and the whole Terra blockchain had burned to the ground. Terra's founder, Do Kwon (colorful character!), was tweeting in September that he wasn't "on the run" from South Korean authorities. Those authorities responded that he was "obviously on the run."

D.
DeFi

The thing about centralized crypto exchanges is that they're *centralized*. Broadly speaking, you have to trust the people running the exchange to run it in a good way: not to steal customer money, not to get hacked, not to take advantage of their knowledge of customer orders to trade ahead, not to blow up by allowing too much leverage, etc. Sometimes that trust is misplaced and the exchange does steal the money. Sometimes the exchange is honest, careful, and well-regulated by the responsible national authorities, but still, you're in *crypto*:

Also, you're in crypto: You want *smart contracts*. A financial-products exchange can be thought of as a computer program. Most stock exchanges long ago got rid of trading floors with human traders and are now just computer servers matching electronic buy and sell orders. Certainly every crypto exchange works like that. A centralized crypto exchange is fully electronic; it has computers that keep ledgers of customer assets and run the programs matching orders and moving assets between customers.

The blockchain is already designed to keep a ledger of customer assets, so why keep your assets on an exchange's ledger? And you can run computer programs on Ethereum or most other modern blockchains: Why not run the exchange programs there?

A venue for trading tokens that isn't a company but is instead a set of smart contracts on a blockchain is called a decentralized exchange, or DEX. And the broader idea—of having a financial system, with lending and derivatives and everything else, that runs as smart contracts on a blockchain—is called decentralized finance, or DeFi.

A few more points of terminology. Decentralized finance is DeFi, so centralized finance—meaning specifically the bits of *crypto* that use centralized trusted intermediaries, mainly exchanges and lenders—is

"CeFi"

And I will sometimes refer to DeFi things as protocols. A protocol is a set of smart contracts—the computer programs that run on the blockchain and do stuff—or at least a set of rules for creating them. A decentralized exchange isn't an exchange in the traditional sense: It certainly isn't a building in New York where traders meet in person to shout orders at each other, and it also isn't a data center in New Jersey where an exchange company keeps its computer servers to match orders with each other. It's a protocol, a set of smart contracts that let people move cryptocurrencies around. There might be a company involved, and surely someone is making money somewhere. But even if the company goes away, the smart contracts will keep running as long as the blockchain does.

i. Some background on exchanges and market makers

● CLOB

Here's what an exchange is. People send it orders: "Buy 100 shares at $100," "Sell 100 shares for $102," etc. The exchange is, at its heart, a system for matching those orders, finding a buyer for each seller and vice versa. The orders come in at different times. When the exchange gets an order to buy 100 shares at $100, it puts the order on its order book—just an electronic list of orders that haven't executed yet. When it gets its next order, to sell 100 shares for $102, it looks at its order book to see if there are any orders to buy 100 shares at $102. Nope, not yet; the $100 buyer doesn't want to pay $102, and the $102 seller doesn't want to accept $100. It puts the sell order on the order book, too. The $100 buy order and the $102 sell order "rest" on the order book. Then another order comes in, to buy 100 shares at $102. The exchange sees in the order book that, aha, yes, there's an order to sell 100 shares at $102. So the matching engine matches the $102 buy and the $102 sell order; they pair off with each other and do their trade. And then the orders are removed from the order book—they've been filled—and the exchange waits for the next order.

In general, a centralized exchange will have lots of orders resting on its order book at any time. All the resting buy orders will have lower prices than all the resting sell orders: If a buy order has a higher price than a resting sell order, those two orders will pair off and execute with each other. (If I want to pay $103 and you'll accept $102, the exchange has found a mutually beneficial trade.) And this way of running an exchange is usually called a central limit order book, or CLOB.

You could certainly build a smart contract to do this on the blockchain: The contract would take orders, keep them on a central limit order book, and execute them against each other. And smart contracts like this do exist. But most DEXes don't work this way.

● Market makers

You may ask: "Where do all these resting orders come from—who's going around thinking up these specific prices they want to pay for a stock or prices they want to sell for?" Ordinary people might not bother with this. If they think a stock is a good investment, they will often send in *market orders*, just: "Buy from whoever's selling at whatever the best price is."

Resting orders come mainly from professional investors called market makers, who help make fast and efficient trading possible. Their job is to sell stock or crypto or whatever to people who want to buy, and buy from people who want to sell, and collect the "spread," the difference between their bid (or buying) price and their offer (or selling) price. If you send a market order to buy, you buy immediately—but you pay a bit of money (the

spread) to the market maker for that service. The market maker, meanwhile, is in the business of providing that service and collecting that spread; it's not really betting that prices will go up or down.

In modern stock and crypto markets, market makers are *also* largely computer programs, and their programs are pretty simple:[58]

58 This is oversimplified but not by that much. It's close to the standard Jack Treynor model of the dealer function.

1. POST A BID AND AN OFFER FOR A STOCK.
2. IF SOMEONE SELLS YOU STOCK AT YOUR BID PRICE, LOWER YOUR BID AND OFFER SLIGHTLY: IF SOMEONE IMMEDIATELY "HITS YOUR BID," THEN YOU MIGHT WORRY THAT YOUR BID WAS ACTUALLY TOO HIGH AND YOU COULD'VE PAID LESS. ANYWAY, NOW YOU OWN SOME STOCK AND WANT TO GET RID OF IT, SO YOU MIGHT AS WELL PUT IT ON SALE.
3. IF SOMEONE BUYS STOCK FROM YOU AT YOUR OFFER PRICE, RAISE YOUR BID AND OFFER SLIGHTLY, FOR SIMILAR REASONS.
4. KEEP DOING THIS. THE SYSTEM IS SELF-CORRECTING: THE MORE STOCK PEOPLE WANT TO SELL YOU, THE LESS YOU PAY THEM FOR IT; THE MORE THEY WANT TO BUY FROM YOU, THE MORE YOU CHARGE THEM. HOPEFULLY IT ALL BALANCES OUT, AND YOU END UP *FLAT*; YOU SELL ALL THE STOCK YOU BOUGHT AND BUY ALL THE STOCK YOU SOLD. AND YOU COLLECT THE SPREAD ALONG THE WAY.
5. ALSO, YOU'RE PROBABLY KEEPING AN EYE ON GENERAL MARKET CONDITIONS, AND RAISING AND LOWERING YOUR BIDS AND OFFERS BASED ON WHAT'S HAPPENING IN OTHER MARKETS.

Market makers in US stocks are often called high-frequency traders, or sometimes even flash boys, and part of what that means is that they're constantly changing the prices of their orders as their information changes. And the result is that, when you want to buy a share of stock, you can send in a market order, and you will quickly trade with a market maker at a price that's pretty much the market price, one that's updated continuously to reflect supply and demand and all available information.

● **Nope!**

This doesn't work on the blockchain. The problem with the blockchain is that it's a slow computer: Ethereum runs computer programs by sending them to thousands of nodes to confirm transactions, and that takes time. You wouldn't want to run a self-driving car on the Ethereum Virtual Machine. It turns out you wouldn't want to run a high-frequency trading platform there either.

Centralized exchanges—in traditional finance and in crypto—have lots of very fast computers with very fast connections that allow market makers to constantly update their prices in response to trades and new information. Traders build fancy fiber-optic lines to get their orders to the exchange a few milliseconds faster than the competition. Speed matters, and they update their orders many times a second. This is harder when the

computers are on the blockchain; it's also more expensive. Every action in Ethereum requires gas fees, and sending a message to a smart contract saying, "Cancel my buy order at $100.001 and put in a new order at $100.002," would use Ethereum's computer power and cost gas.

It turns out this is nearly fatal for CLOB market-making in Ethereum. "Market making is very expensive," Vitalik Buterin wrote, "as creating an order and removing an order both take gas fees, even if the orders are never 'finalized.'"

ii. Automated market makers

● Mechanics

So Vitalik proposed a different idea. Here it is:

> **The mechanism would be a smart contract that holds A tokens of type T1, and B tokens of type T2, and maintains the invariant that A * B = k for some constant k (in the version where people can invest, k can change, but only during investment/withdrawal transactions, NOT trades). Anyone can buy or sell by selecting a new point on the xy=k curve, and supplying the missing A tokens and in exchange receiving the extra B tokens (or vice versa). The "marginal price" is simply the implicit derivative of the curve xy=k, or y/x.**

That's just a comment that he wrote on Reddit; later a whole business model grew out of it. Here's the simplified version. I decide to be a market maker in some token pair, say Ethereum and USDC (a dollar-denominated stablecoin). Let's say one Ether currently trades for 1,600 USDC ($1,600). I set up a smart contract—again, that's a computer program on the blockchain, but in this case it might be easier to think of it as a pot of money, *managed* by the computer program. I can deposit both Ether and USDC into this pot. Anyone who wants to buy Ether (with USDC) or sell Ether (for USDC) can come to my smart contract and make that trade with it, and the contract will adjust its prices to reflect supply and demand.

But it will do it in an absurdly simple manner that doesn't require much in the way of computation or updating prices. It will just try to keep one number constant.

When I set up the contract, I deposited *equal values* of Ether and USDC. If one Ether was worth 1,600 USDC, I put in 100 Ether and 160,000 USDC (each set of tokens worth $160,000). The program *multiplies* those two numbers: 100 times 160,000 is 16 million. And then it will just hold that product constant. That is, the number of Ether multiplied by the number of USDC will always be 16 million.

A trader will try to buy, say, 10 Ether. That would leave 90 Ether in the fund. Sixteen million divided by 90 is 177,777.78. The pot has 160,000 USDC now, so it will need 17,777.78 more USDC to keep the product constant. That's what my smart contract charges. The trader has to pay me 17,777.78 USDC—1,777.778 USDC per Ether—to do this trade. Someone wants to buy Ether from my smart contract, so my smart contract automatically raises its price for Ether. But it doesn't have to go around constantly updating prices and posting new prices on central limit order books: It just advertises its constant product, and that updates prices on its own.[59] It's an automated market maker (AMM).

59 This thing I described is a *constant product market maker*, where the product of the two tokens is set to be a constant. There are other functions that automatic market makers can use.

That's cool! I don't know, it's cool. This isn't a thing that really exists in traditional finance. It was developed, almost by accident, as a workaround to a novel *problem*—that the computers are too expensive—in decentralized finance.[60]

60 You might be asking: "Wait, how do I know the price charged by the smart contract is *correct*? Is the computer program ever checking Ether's price quoted on, say, Coinbase to make sure its price makes sense?" No. Insofar as demand for Ether by people interacting with the contract reflects demand for Ether in the rest of the world, then the price created by the contract *is* the price. Sometimes different smart contracts and different centralized exchanges could have different prices for Ether, but if they get too out of whack, people can try to buy Ether where it's cheap and sell it where it's expensive for a quick profit and close that gap.

● LPs

In my description above, I assume that I want to be a market maker in some currency pair, so I set up a smart contract to do it on my own. In reality, the way DEXes normally work is that AMM smart contracts *pool* liquidity. If you want to be a market maker in stocks, it helps to be a large financial institution. In crypto, anyone who wants to be a market maker in a pair of tokens can contribute that pair to a liquidity pool (and get back "liquidity tokens" representing their stake in that pool); they're called "liquidity providers." They collect fees in crypto for providing liquidity in the pool.

There's a risk, though. In general, if you provide liquidity in an asset (in DeFi or centralized finance, in crypto or otherwise), and the asset price steadily goes down, you'll find yourself buying more and more of it, and it will be worth less and less. In an AMM liquidity pool, if the price of one token keeps going up against the other token, the pool will have more and more of the token worth less and less. This is a risk that all market makers take; in traditional finance it's sometimes called "adverse selection."

In DeFi, though, it delights in the name "impermanent loss." I don't know why, but I love it. There's nothing particularly impermanent about it—unless the price bounces back, of course—but that's what it's called.

iii. Lending

Exchanges—providing liquidity between different tokens—are a key function of DeFi, but there are others. One important function is lending.

● Secured

Most DeFi lending is what you would call, in traditional finance, *margin lending*. You have a volatile asset (Ether, say), and you want to borrow some money (a dollarlike USDC stablecoin, say), using that volatile asset as collateral. Perhaps you're borrowing that money to buy more volatile assets, or perhaps you're borrowing against your Ether to buy a sandwich or a house without selling your Ether.

If you went to a broker for a margin loan against $100 of stock, the broker might lend you $50. It's "overcollateralized," because the stock is worth more than the loan, which makes the broker feel safe. If the stock then fell to $70, the broker might send you a margin call: "Pay down $15 of this loan, or I'll liquidate your stock." That way it's less likely the broker will end up holding the bag for your loss if the stock keeps falling. If you failed to meet the margin call, the broker would sell your stock, clear, say, $65 for it, keep enough money to repay the loan, and give you what's left.

This is pretty straightforward to automate with smart contracts. There are robust DeFi lending systems, such as Compound: Lenders put up tokens and earn interest, borrowers borrow tokens (overcollateralized by other tokens) and pay interest, and automatic liquidation provisions make sure that if the value of the collateral goes down, it's sold to pay back the loans.

● Unsecured

What's much harder in DeFi are the main businesses of traditional finance: *unsecured* lending and lending secured by physical assets (such as houses). DeFi is good at lending crypto secured by crypto. The collateral lives on the

blockchain; the loan lives on the blockchain; they're connected by smart contracts on the blockchain. It's all pretty neat.

But this sort of lending crypto against crypto doesn't *do* much. It's mostly useful for crypto speculation–you lend Ether, get USDC, and use it to buy even more Ether. By contrast, a mortgage lets you buy a house and then pay for it with your future income. An unsecured business loan helps you build a business and then pay off the loan with the business's future earnings.

Finance, at its heart, is about moving future wealth into the present by borrowing, or moving present wealth into the future by saving.

There are deep philosophical reasons that crypto is bad at this. An unsecured loan is essentially about *trust*. It's about the lender trusting that she'll be repaid not out of a pool of collateral but out of the borrower's future income. She has to trust that the borrower will have future income and that he will pay.

Relatedly, an unsecured loan requires *identity*. You need to know who's borrowing the money, what their payment history looks like, what their income is. The default approach in much of crypto is pseudonymity: Anyone can set up any number of crypto wallets or Bitcoin account numbers, and they're generally not tied to a name. If you borrow money against crypto collateral, all your lender needs to know is that the collateral is on the blockchain. If you borrow money against your future income, your lender needs to know who you are.

That said, there's nothing in principle to *prevent* unsecured lending in DeFi, and there are projects such as Goldfinch that do it. You get together a smart contract, where people deposit money in exchange for tokens, and a DAO, a decentralized autonomous organization that controls the money. The tokens give them a share in the DAO's profits and the right to vote on who gets the money. People propose potential borrowers, token holders consider them and vote, and if the holders vote yes then they make a loan. As with any DAO, the token holders can get together in a chat room and consider "off-chain" information if they want. (They could make the borrower send in a driver's license and two bank statements.) You can build some incentive mechanism to punish members for proposing borrowers who default and reward them for proposing good borrowers. Decentralized finance is made up of smart contracts, but it's also made up of people, and if they want to do unsecured lending, they can.

We've talked about web3 as a source of online reputation, about "soulbound" tokens that could be used to verify a person's actions,

connections, and characteristics. A soul, in this terminology, is something close to a credit report: It's a list of stuff that a person has done that should make you trust them, a decentralized and blockchained source of reputation information. The right assortment of degrees and endorsements might make you feel good enough about a person—or rather, a "soul," an address on the blockchain—that you give them an unsecured loan. And then I suppose if they default you can take their soul.

iv. Tokenomics of DeFi

The basic mechanism of DeFi is that you put some tokens up in a smart contract to generate fees or interest. You put your tokens in an automated market-making contract to get liquidity-provider fees, you put your tokens in a lending protocol to get interest, etc. Generally this is referred to as "locking" your tokens—because they're being used for lending or whatever, you can't get them back for some period—and people often talk about DeFi protocols in terms of their TVL, or "total value locked." Mechanically, the way this generally works is that you send your tokens to the smart contract, and it sends you back other tokens that are, basically, receipts for the tokens you locked up. Instead of having Ether and USDC, you have liquidity-provider tokens saying you've put some Ether and USDC into a smart contract.

What do you do with those liquidity-provider tokens? Well, in theory, you hold them until you want your locked tokens back, and then you hand them in and get back the underlying tokens. The liquidity-provider tokens are just a receipt. But at some point people realized that those tokens represent something of value; we can do something with them. If you have a token representing a receipt on some Ether that you locked into a smart contract, then that's *almost* as good as having some Ether, and someone might give you some money for it. Now you can borrow against those receipt tokens in DeFi, too.

Everything is like this. In proof-of-stake Ethereum, you can stake Ether to earn staking rewards. A protocol called Lido runs a big staking pool. If you stake your Ether with Lido, it will give you back a token called stETH, basically a receipt for your staked Ether. The Ether that you staked with Lido will earn staking rewards, while the stETH that you get back can be sold or invested in other DeFi protocols to earn more money.

More broadly, the business of DeFi is about reusing tokens as much as possible. You have some tokens, you lock them up in a smart contract that does a thing and pays you a return, the smart contract gives you some sort of receipt token, and you turn around and lock up those receipt tokens in another smart contract that does another thing and pays you some more. People talk about "yield farming," the process of bouncing between DeFi protocols to try to earn the maximum yield, reusing tokens, and getting paid in the protocols' own tokens to make as much money as possible.

This can create a self-reinforcing cycle, by which I kind of mean a Ponzi. It goes like this:

1. There's a protocol that does some stuff with Ether or stablecoins or whatever.

2. If you put your Ether or stablecoins or whatever into the protocol (for lending or liquidity provision or whatever), it will give you some of its own tokens. This is no problem; it can print those tokens for free.

3. If you put *those* tokens back into the protocol, locking them up rather than selling them, it will give you even more of those tokens. It'll pay you 10% interest every hour, if you want. Who cares, the tokens are free.

4. "Buy this token, it pays 10% interest every hour," the promoters of the protocol can say, more or less accurately. They can quote an APY—an annual percentage yield, a normal finance concept that's much beloved in DeFi yield farming—with an enormous number of digits, and people will get very excited.

5. So they'll buy the token, and it will go up. Or they'll put their Ether or stablecoins into the protocol to earn its tokens, and the protocol's total value locked will go up.

6. "Look at this protocol, its TVL is huge and rising, its token has doubled in price this week," people will say, and they'll buy more of it.

7. People keep getting paid comical interest rates *in the token*, which is fine as long as the token price keeps going up,

or stays flat, or goes down at a slower rate than the interest rate. Even though the market is being flooded with tokens, people still seem to be making money, and will do so as long as new money comes in.

8. The amount of tokens issued rises inexorably toward infinity, the amount of new money coming in does not, and tragedy ensues.

The greatest of these protocols is probably OlympusDAO, which is run by a pseudonymous founder called Zeus (colorful character!), has a group of loyal investors called OHMies, and at its peak offered yields of 7,000% and was worth $3 billion, according to a *CoinDesk* article titled "Olympus DAO Might Be the Future of Money (or It Might Be a Ponzi)."[61] Since then it's lost about 99% of its value, so there's your answer.

61 It also sparked an even funnier copycat called Wonderland on the Avalanche blockchain, which offered yields of more than 80,000% and was partly run by a pseudonymous person who, it turned out, also co-founded Quadriga. (Colorful character!) Last year, I needed physical therapy for my knee, and I ended up bonding with my therapist by talking about crypto. He was heavily invested in Wonderland, which he cheerfully described as a Ponzi, but one that was making him a lot of money.

Olympus became particularly famous for the "(3, 3)" meme, based on the notation of game theory. The idea is that the payoff in a two-player game can be written as "(X, Y)," where X is what I get, and Y is what you get. These outcomes could be dollars, but often they're written as abstract utility numbers: A payoff of (3, 3) is better for both of us than a payoff of (2, -1), without worrying too much about what those numbers mean. Olympus's pitch was that if everybody buys OHM and locks it up, then everybody's payoff will be good—i.e., (3, 3)—while if everybody does bad things like *sell* their OHM, then everybody's payoff will be (-3, -3). (Those numbers are abstract and unitless, and actually the payoff was (-99%, -99%).)

This, of course, exactly describes any Ponzi: As long as people keep investing new money and not withdrawing it, everyone will get richer (on paper), but it will unravel when people start taking their money out. Olympus always struck me as charmingly forthright about what it was up to. It's the future of money, because as long as everyone keeps buying, it will keep going up! People have tried that one before.

v. Some arbitrages

In traditional finance, people devote their careers to finding arbitrages, circumstances in which they can buy something at a low price and instantly sell the same thing at a high price. This is hard to do, because traditional finance is very competitive, and people aren't regularly leaving $20 bills on the sidewalk. Even if you think you've spotted an arbitrage—the same thing trading at different prices in different places—you have to worry about some legal or operational reason that you can't actually move between those places. (Maybe there's a 5% tax to move the stock offshore. Maybe short selling isn't allowed, etc.)

Most of what people call arbitrage in traditional finance is buying one thing at a low price, simultaneously selling a slightly different thing at a higher price, and hoping they turn out to be the same thing. Or buying one thing at a low price, selling the same thing a bit later, and hoping it turns out to be at a higher price.

Meanwhile, decentralized finance is new enough that pricing anomalies exist, but efficient enough that everything happens visibly on a virtual computer running public code, so you can reliably exploit them. There are magical possibilities.

● Flash loans

Let's say you're a smart young person and you discover an arbitrage. Stock XYZ is trading at $10 on Exchange A and $11 on Exchange B. You can buy it on Exchange A for $10 and sell it on Exchange B for $11, making an easy $1 profit.

That's fine, now you have $1. But you want to buy 10 million shares on Exchange A for $100 million and sell 10 million on Exchange B for $110 million. Then you would have $10 million, which is much more worth doing.

The only problem with this plan is, while you're a smart young person, you don't have $100 million. Why would you?

In financial theory the solution is simple: You borrow the $100 million at the market rate of interest, buy the stock, sell it, pay back a little interest, and keep your profits. In practice no one is going to lend you $100 million. I mean, if you really have found a perfect arbitrage, maybe you'll be able to raise $100 million. You can work your connections, cold-call some hedge funds, maybe get some meetings where you present your strategy and say, "See, it's a perfect arbitrage. I just need $100 million. Are you in?" And they'll just do the strategy and leave you out. Maybe they'll pay you a small finder's fee.

Or you could start small—do your arbitrage for $1,000, make $100, do it again for $1,100, etc.—until you have a huge bankroll, but that's not great either. The longer it takes you, the more likely it is that the clever arbitrage you've spotted will go away. In particular, the more you *do* the trade, the more likely someone else is to notice and do it in big size and make the arbitrage go away.

In crypto finance the situation is different. If, say, you spot Ether trading at two different prices on different decentralized exchanges, you can just write a program that does the following:

1. Borrows $100 million from some decentralized lending protocol, such as Aave
2. Uses the $100 million to buy a token on Decentralized Exchange A
3. Sells the token on Decentralized Exchange B for $110 million

4. Returns the $100 million to the lending protocol (plus a small fee)
5. Sends the leftover $10 million to you ...
6. ... *all in the same transaction that executes all at once.*

The lender in Step 1 can make the return in Step 4 a *condition* of the loan; the loan and the payback occur simultaneously, in the same computer program executing in the same block of the blockchain. As far as the lending protocol is concerned, there's no credit risk: If any of the intermediate steps fail—if it turns out you're wrong about the arbitrage, and you can't sell the token for more than you paid for it, etc.—then the whole thing never happens, and the loan isn't made. The lending protocol isn't evaluating your creditworthiness, your résumé, or your track record as an investor. It's just making sure that the code works.

This is clever and neat and feels like a good way to build a financial system. It's an egalitarian, decentralized, permissionless, computerized way for anyone who spots an arbitrage to be able to exploit and close it. It should make markets more efficient and prices more accurate.

The problem is, in crypto, "an arbitrage" often means "a mistake in a smart contract."[62] Somebody writes some contract that has some bug that occasionally lets a user put in one token and get back two. Then somebody notices and writes a program to use flash loans to put in 1 billion tokens and get back 2 billion tokens and blow up the smart contract entirely. People sometimes leave money lying around in crypto, and crypto has built very efficient ways for other people to take their money. It's not clear that *that* is a good way to build a financial system.

62 Of course, this is true in traditional finance, too. Somebody notices a flaw in a loan agreement or credit-default-swap contract, buys up a bunch of the bond or CDS, then goes to court to try to extract money from the person who wrote the contract wrong. This, however, is a lengthy and risky process, and one of the main risks is that the court will say, "Oh, come on, that's not what they meant, get out of here." Whereas a smart contract will never say that.

● MEV

It gets stranger. Let's say you notice that a token is trading at $10 on Decentralized Exchange A and $11 on Decentralized Exchange B. So you send an order to Exchange A to buy 1,000 tokens for $10,000 and a simultaneous order to sell 1,000 tokens for $11,000. What happens to those orders?

In the US stock market, whole books have been written about that question. People get really mad about it. Your order goes to your broker, who routes it to different exchanges at different times through different pipes. High-frequency electronic traders who see your order execute on one exchange might "race ahead" to another exchange to push up the prices. There are controversies about "co-location," where the high-frequency traders pay fees to the stock exchanges to keep their computers in the same room as the exchange's computers, so they can get a tiny speed advantage by connecting to the exchange through a fairly short wire.

In crypto the answer is different. It's easiest to understand if you start with how trades worked on Ethereum, before the switch to proof of stake. When you made a trade, your transaction would be broadcast to the entire network and included in the list of transactions that miners were working on executing but that hadn't yet made it into a block.

When a block was finalized, miners would include your transaction in the block. But the miners decided which transactions got included in a block and in what order, and they also earned gas fees for executing transactions. Users could specify how much they wanted to pay for gas, and transactions with higher gas fees were usually prioritized.[63] This created a trade:

63 From the Ethereum documentation: "For transactions that need to get preferentially executed ahead of other transactions in the same block, a higher tip will be necessary to attempt to outbid competing transactions." In modern Ethereum, blocks tend to be "built" by block builders, who are separate from validators, and who are in part building blocks to maximize gas fees; then they are validated by the validators.

1. You find an arbitrage and send some orders to decentralized exchanges to do that arbitrage.
2. I see those orders on the network and think, "Hey, that's a good trade."
3. I send *the same orders* to the exchanges to do the same transaction.
4. I pay a higher gas fee to get priority over you, so that I can do the trade ahead of you.
5. By the time *you* get to do the trade, it's no longer available. I bought everything available at the good price. Ha ha.

NOT these Flash boys.

This is usually called MEV, which originally stood for "miner-extractable value," because one winner in this scenario was the miner who got to charge higher gas prices to people who wanted to front-run trades or avoid

getting front-run. It's the subject of a 2019 paper, by Philip Daian et al., titled "Flash Boys 2.0," a reference to the high-frequency traders in US stock markets who purportedly race ahead of other traders' orders to extract value from them.

Rather than *solve* this concern about traditional markets, crypto *made it explicit*: Time priority was subject to an explicit "gas auction," where whoever paid the most to the transaction orderers got to go first. Sure, yes, in crypto you could get front-run all the time by more sophisticated electronic traders, but in a transparent and decentralized way.

As Ethereum moved to proof of stake, MEV was rebranded "maximal extractable value." There's still money to be squeezed from traders by the people maintaining the network, but the mechanics have changed. In today's Ethereum, there's a division of labor between block builders (who compile and order transactions) and validators (the replacement for miners, who do the proof-of-stake work to make blocks official), and if you're doing arbitrages you can send your transactions privately to block builders. You can still pay for speed and priority, but you can't see everyone else's trades to front-run them.

E.
Reinventing 2008

In 2008, Satoshi Nakamoto invented Bitcoin. One thing that seems to have motivated him was a distrust of banks and financial intermediaries. This was understandable, because it was 2008. The modern banking system was at a low ebb. Banks had taken risks that few people understood and ended up losing tons of money on supposedly safe investments. High levels of leverage in the banking system—and in the more opaque and less regulated "shadow banking" system—made those systems fragile; a bank that borrowed $30 for every $1 of shareholders' equity would go bust if the value of its assets fell 4%.

The high leverage and lack of transparency caused contagion. An asset's price would fall, the highly leveraged banks and funds that held it would get margin calls, and they'd have to sell whatever they had on hand. That would cause more prices to fall, which would lead to more margin calls, which would bankrupt some of the banks and funds, which would lead to more fire sales and more price drops. Meanwhile, the lenders to those banks and funds, who thought their money was safe, would have losses; many were also highly leveraged and might go bust. All of this was opaque enough that even banks and funds that *hadn't* taken big risks or lost a lot of money were treated with suspicion by lenders, which could cause them to fail, too. Ultimately the banking system was bailed out by massive infusions of money from central banks.

All of this was gross.

Satoshi didn't like it, and he built a new payment mechanism that escaped the need to trust banks. It was irreversible and decentralized: Everyone was responsible for their own mistakes; no one could be bailed out by central banks printing money. It was transparent: Every transaction was recorded on the blockchain; there were no hidden chains of leverage.

In a deep sense it wasn't built on debt at all. To some Bitcoiners, the central sin of the banking system is that every bank deposit is a liability—it's a debt, payable on demand, from the bank to its depositors—and so every dollar that you keep in the bank *necessarily* adds to the leverage of the system.[64] Dollars *are debt*, and debt got kind of a bad rap after 2008. Meanwhile, Bitcoin are *not debt*. They're just Bitcoin. They exist in themselves, on the blockchain, rather than being liabilities of banks.

64 This complaint is not limited to Bitcoiners. After the 2008 crisis there was a lot of interest in the "Chicago Plan" of narrow banking without fractional reserves. In the Chicago Plan, every bank deposit is backed 100% by "real dollars" (reserves at the Federal Reserve), and lending is financed out of equity.

In early 2009, when Satoshi mined Bitcoin's genesis block, this message resonated with a lot of people. A new financial system with transparent and irreversible transactions, with no special power for governments or big banks, had an appeal.

Over time, though, there was another, much more obvious appeal to Bitcoin. Its price kept going up. If you bought a Bitcoin for $100, you could soon sell it for $1,000. This got a lot of people very interested in crypto, not for philosophical or monetary-structure reasons but because getting rich is nice.

Many of these people came from traditional finance, because they saw that crypto finance was fun, it was wide open, it allowed for permissionless innovation, and everyone was getting rich.

These people—the people who left TradFi for crypto because the money was better—didn't necessarily have strong philosophical commitments to all that Satoshi stuff. They weren't like, "Leverage is bad, banks are evil, monetary soundness is what matters." Some came from banks. They were there to make money. One way to make money is by finding good trades, finding cheap ways to borrow money, and then borrowing as much as possible to put into those trades.

Back in the ~$850 halcyon days.

And so somehow crypto had itself a 2008! In 2022 the crypto financial system rediscovered what the traditional system had discovered in 2008. It was honestly kind of impressive.

i. Terra

In May 2022, you'll recall, the algorithmic stablecoin TerraUSD collapsed. If you had money in Terra/Luna, odds are that you lost roughly all of it. Many of the people who lost money were regular retail savers who'd been suckered by TerraUSD's promises of stability (and of a safe 20% interest rate) or regular retail cryptocurrency investors who speculated on Luna and lost. But not all of them.


Kyle Davies and Su Zhu.

One victim of the Terra collapse was a hedge fund called Three Arrows Capital, run by two former Credit Suisse Group AG currency traders. (Colorful characters, and also former TradFi guys.) "Speaking from an undisclosed location" in July, 3AC's founders explained to Bloomberg News that what they'd "failed to realize was that Luna was capable of falling to effective zero in a matter of days and that this would catalyze a credit squeeze across the industry that would put significant pressure on all of our illiquid positions."

Meanwhile, the Federal Reserve was raising rates, and speculative assets generally were losing value. It turns out that crypto is basically a speculative asset and that it's not particularly a hedge for stock market volatility. Everything in crypto went down. Bitcoin was worth more than $67,000 at the peak in October 2021; it fell below $20,000 in June 2022. Ether went from $4,800 to less than $1,000. The total market value of all cryptocurrencies fell from about $3 trillion in late 2021 to about $1 trillion in June 2022. Two-thirds of all crypto wealth just vanished.

That's just a very traditional story, isn't it? Leveraged hedge funds piled into crowded trades that seemed, on the basis of a fairly short series of

historical data, to be safe. This made the trades unsafe, so the hedge funds lost money. So their lenders sent them margin calls. So they were forced to sell off other, better assets—assets that were more liquid and could be sold to meet the margin calls—which made those better assets bad, too. "In a crisis, correlations go to 1," traders say. Losses on bad trades force leveraged hedge funds to sell good assets, and so everything goes down at once.

ii. Contagion

3AC was a leveraged hedge fund. When it blew up, the people who loaned it money didn't get their money back. 3AC went into a complicated cross-border insolvency process that will presumably *eventually* recover some money for its creditors, but they'll lose at least some of their money, and it will take a while to get back the rest. Who are these creditors?

Well! A little of everyone, really. Documents filed in 3AC's insolvency process reveal that the hedge fund was borrowing from DeFi platforms but also from an assortment of big-name centralized, or CeFi, crypto lenders, borrowing platforms, and exchanges.

The DeFi platforms mostly did fine: They had collateral, they had automatic liquidation mechanisms, they liquidated the collateral, and they got their money back. The centralized lenders did less well. It turns out a lot were less strict about demanding collateral than you might have wanted. 3AC was one of the biggest and best-known hedge funds in crypto. Working with 3AC was a stamp of approval for many lending platforms. It was prestigious to say, "Our customers include Three Arrows." Also, 3AC was viewed as a smart fund, doing clever low-volatility arbitrage trades with good risk management rather than taking wild gambles. So if 3AC came to a lending platform and said, "Hey, we'd like to borrow $500 million unsecured," the platform might say yes.

Oh boy, did they. Voyager Digital, a crypto brokerage that let customers buy, sell, borrow, and lend crypto, is a public company listed in Canada. It had $2.3 billion of assets at the end of June, about $650 million of which was unsecured loans of USDC and Bitcoin to 3AC. Oops! It went bankrupt, too.

Celsius Network is the same basic idea but worse. It offered customers willing to lend out their crypto up to 18% interest on deposits, with pretty vague descriptions of how it earned that yield. CEO Alex Mashinsky (colorful character) once explained to *Bloomberg Businessweek* that it's ridiculous that banks take deposits, use them to make loans, and then *don't* pay 18% interest. "Somebody is lying," Mashinsky said. "Either the bank is lying or Celsius is lying." Only one possible answer! Celsius had also loaned

3AC money, though that was the least of its problems, and it was in some of the same trades as 3AC, which blew up when 3AC did. It also went bankrupt.

The leverage of these platforms is pretty astonishing. Celsius was levered about 19 to 1: It had almost $95 of debt (mostly customer deposits) and about $5 of equity for every $100 of assets. Voyager was levered 23 to 1. A fairly small loss could wipe it out entirely, and did. That some banks were levered 30 to 1 going into the 2008 financial crisis became a matter of intense scandal, and post-crisis reforms require much higher capital levels for banks. Also, banks mostly invest in mortgages and stuff! These guys were investing in crypto loans, hugely volatile stuff with nothing in the way of long-term, through-the-cycle history! And they were doing it with 5% capital ratios.

iii. Non-contagion

In many ways this looks like 2008. But it's striking how *little* effect the loss of $2 trillion of crypto wealth had on anything else. The 2008 crisis in the banking and shadow banking system led to a global recession, a foreclosure crisis, and real political instability. The 2022 crisis in crypto seems to have been pretty walled-off from real-world effects. Two trillion dollars of market capitalization were lost without much of a visible impact outside crypto.

Why? Part of the answer is about who lost money and how they thought of the money they lost. A lot of people who put money into crypto were using their gambling money, and when their bets didn't pay off, they thought, "Ah, well, that was fun, too bad." Almost everything about the world of crypto screams "high risk" to anyone who knows at all what to look out for. And so, if you do know what to look for, you take your crypto risks with money that you can afford to lose and in ways that account for the risks. You don't take your life savings, lever them up 10 to 1, and invest everything in Dogecoin.

The great lesson of 2008 is that the real systemic risk is in the *safest* assets. The problem isn't banks and investors buying insane risky securities that promise 50% returns and then go to zero. The problem is banks and investors buying AAA rated bonds that promise an extra 0.03% of yield and borrowing 95% of the money they use to buy those bonds—then finding out those securities shouldn't have been rated AAA. People invest money they can't afford to lose—and often money they borrowed—in safe assets, and when those assets lose money, the system breaks.

By 2022 the crypto financial system was working on creating safe assets. That's what Celsius and Voyager and TerraUSD promised, safe and stable

ways to earn high returns without a lot of volatility, but in crypto. Some people were taken in by those promises and invested their life savings; some hedge funds bet on those promises and levered up those bets. But mostly, look, the guy who promises 18% yields and says, "Either the bank is lying or Celsius is lying" isn't going to persuade *that* many people to entrust him with their life savings.

Part of the answer, though, is about the traditional financial system. Traditional big financial companies *have* been dipping their toe into crypto, but for the most part the traditional and crypto financial systems have stayed pretty separate. You don't hear a lot about *banks* keeping 20% of their assets in Bitcoin, and in fact banking regulators have been rather stern about letting banks own crypto. So when crypto prices collapsed, banks and other financial institutions weren't particularly harmed.

This matters a lot. If you want to buy a house or open a store, you go to a bank for a loan. When banks are in crisis, as they were in 2008, they will be less likely to lend you money. So you won't buy a house or open a store. There will be less credit, less economic activity, less growth in the real economy. Governments bailed out banks in 2008, not because they love bankers but because banks matter for the rest of the economy.

The crypto financial system is very fun and cool and has invented a lot of interesting stuff. But it's mostly not where people go to get money to buy a house or open a store. Bitcoin and Ethereum and DeFi could all vanish tomorrow without a trace, and most businesses that make stuff in the physical world would be just fine.

IV
Trust, Money, Community

Let me tell a couple of stories about crypto and society.

A.
Trust

One story goes like this. We live in a world of trust. That trust pervades everything we do. We're spoiled; the institutions we deal with every day are trustworthy. Not all of them, not all the time, not in every way, but quite a lot of them to a high degree. We put money in the bank, and when we go to take it out, it's there.

Crypto—Satoshi Nakamoto and his disciples—said:

NO, NO, NO. TRUST IS BAD. DON'T TRUST YOUR BANK. USE IMMUTABLE CODE. VERIFY EVERY TRANSACTION FOR YOURSELF, OR DOWNLOAD OPEN-SOURCE CODE AND VERIFY THAT IT WORKS CORRECTLY, AND THEN USE IT TO VERIFY EVERY TRANSACTION FOR YOURSELF, OR AT LEAST USE A NETWORK IN WHICH THAT'S POSSIBLE AND IN WHICH ECONOMIC INCENTIVES DEMONSTRABLY MAKE IT LIKELY THAT IT WILL HAPPEN. AND DO ALL OF THIS IN A SYSTEM THAT'S RESISTANT TO CHANGES, THAT CAN'T BE CONTROLLED BY GOVERNMENTS OR BANKS, THAT'S IMMUNE TO THE RULES OF WIDER SOCIETY.

This had an appeal, and crypto became very valuable, and people looked to put their money to work in crypto—and they trusted people. Over and over again, they trusted Quadriga and Terra and Voyager and Celsius and dozens of other projects that failed or ran off with their money or got hacked. They just fell over themselves to trust people.

Clockwise: Gerald Cotten (Quadriga), Do Kwon (Terra), Stephen Ehrlich (Voyager), and Alex Mashinsky (Celsius).

Why did they do this? Well, there's a common thread in these kinds of things. The people who are easiest to pull in are often the ones who think they're the most independent-minded and cynical. "Either the bank is lying or Celsius is lying," Celsius told people, flattering their unjustified belief that they knew the real score.

But there's something else, too. The people who trusted Celsius weren't tripped up *only* by their belief that they were outsmarting the system, though there was that. They also...they thought Celsius was a bank? It looked sort of like a bank. It did things, with crypto, that were banklike. They were familiar with how banks work. They understood that *banks are*

safe, that if you put money in a bank you can get it back. They looked at Celsius and thought, "Well, this is a big thing, it has a nice website, it's available to Americans. Surely if it was a problem someone *would've done something about it*."[65]

65 Why didn't anyone do anything about it? Why were these platforms allowed to take money from retail customers, run high levels of leverage, and blow themselves up? I think the basic answer is that regulatory authorities are still very much catching up to how crypto works and who should regulate it. There's a very well-understood system for regulating a bank: There are bank examiners who know how banks work and who visit banks regularly to make sure they're working properly. In crypto, big platforms are often decentralized and vaguely stateless, and even in the US there's a lot of jurisdictional jousting over which regulators get to regulate crypto and how they should do it. If you launch a terrible crypto platform, it might take regulators a long time to get around to understanding what you're doing and figuring out how to stop you. And crypto has been very much in a move-fast-and-break-things mode, so it broke a lot of things before the regulators could catch up.

There's something a bit alarming about this. Crypto is in a way *about* rejecting the institutions of society, about being trustless and censorship-resistant. But it quietly free-rides on people's deep reservoir of trust in those institutions. People are so used to trusting banks that, when Celsius told them not to trust banks, they said, "Ah, yes, OK," then trusted Celsius to work like a bank, to be regulated like a bank. They didn't worry about Celsius's opacity and leverage. They didn't do their own due diligence on its loans and audit its DeFi positions and demand irrefutable proof of its soundness. It promised to pay them back, and that was good enough for them.

But there's something hopeful about it, too. Trust in institutions is so strong and resilient that all of crypto's bluster can't stamp it out. "Not your keys, not your coins, put your trust only in verifiable code," crypto evangelists yelled, and people heard them and said, "Yes, that is nice, but I'm busy, I'm going to trust these nice strangers with my Bitcoin."

Crypto, in its origins, was about abandoning the system of social trust that's been built up over centuries and replacing it with cryptographic proof. And then it got going and rebuilt systems of trust all over again. What a nice vote of confidence in *the idea of trust*.

B.

Money

There's a related story about money. One way to think of money is that it's a system of social credit. Society has mechanisms—capitalism, politics, etc.—to allocate resources, with a rough heuristic of: "The more good stuff you do for society, the more good stuff you get for yourself." Money is a rough way of keeping track of that. If you do good stuff for other people, they give you money, which you can use to buy good stuff for yourself.

Another way to think about money is that it's some sort of external objective fact. If you have money, it's *your* money, and society has nothing to say about whether you can keep it or what you can do with it.

Crypto starts from the second view: Your Bitcoin are yours immutably; they're controlled only by your private key, and no government or bank can take them away from you. But the history of crypto since Satoshi has undermined this view. If you got your Bitcoin illegitimately, the government can trace them and stop you from spending them. There are still gatekeepers—crypto exchanges and fiat off-ramps and banks—that decide what you can do with your money. Crypto might be immutable and "censorship-resistant," but its interactions with the real world are not.

Not just that. Crypto isn't even immutable, not really. In 2016 an important smart contract on the Ethereum blockchain called the DAO got hacked. (DAO is now a generic term, but this was *the* DAO, the first of its name.) There was a flaw in the contract that allowed a hacker to drain a lot of money from it, and he did. Ethereum was a new technology, so the hack was a big deal.

> "The descriptions didn't matter; only the code did." June 17, 2016

This hack was controversial; there was controversy about whether it was even a "hack." Some people said: "Look, if the code of the smart contract allowed the hacker to do this, then it was allowed. There's no external standard of validity, just the code, and if it happened in the code, it's fine. If we reverse this transaction, we will destroy the essence of the blockchain, which is the irreversibility of transactions."

Other people said: "No, that's nuts. This was a big hack, and lots of people lost money." It was clear enough—to humans, anyway—that this was not how people intended the smart contract to work, even if it was in fact how it worked. *Sometimes the code is wrong.*

The Ethereum network decided to roll back the blockchain and reverse the hack. That's hard to do. You couldn't just amass a bunch of computer power yourself and hack the Ethereum blockchain and reverse transactions. But if *everyone in Ethereum* agrees to do it, they can.[66] Cryptocurrency isn't money that's *totally immune to censorship*, that's atomic and individual and immutable. It's money that's *controlled by consensus*, much like dollars are. It's a different form of consensus—proof-of-work mining, proof-of-stake validation, decentralized communities, DAOs, Discord chats—but the thing that gives you the money and makes the money valuable is that consensus.

66 Or enough of them did. In fact a minority of dissenters split off, "forking" the Ethereum blockchain and starting their own network (now called "Ethereum Classic").

MONEY IS A SOCIAL FACT, EVEN WHEN THE MONEY IS BITCOIN OR ETHER.

C.
Community

Here's another, more speculative story.

The most valuable thing in human life, this story begins, is connection. Being with your friends, making friends, feeling esteemed by your peers: These are the things that give life meaning.

"Sure, sure, sure, whatever," you say, because this feels fuzzy and fake. But look how rich Mark Zuckerberg is! In 1999, if you'd said, "A giant contributor to US gross domestic product is the friends we made along the way," that wouldn't have made any sense. Now, Facebook is worth almost half a trillion dollars—though, to be fair, it's changed its name to Meta Platforms Inc. to mark its pivot away from friendships.

And to mark its pivot to the metaverse. I don't know what the metaverse is. But I gather that it means something like: Our lives, our social lives, our intellectual lives, our professional lives, our aesthetic lives, the things that we do all day that give our lives meaning, will take place increasingly on interconnected computers. Our reality will be intermediated increasingly by computers and the internet.

Human social life moving to the internet has economic value, it turns out, though if you explain the mechanism, it seems remarkably trivial. "If people talk to their friends about vacuums, and you show them an ad about vacuums, they'll probably buy a vacuum." "If we intermediate between people's friendships, we can serve ads."

A key lesson of crypto is: A bunch of people can get together online and *make their community have economic value, and then capture that value for themselves.* If you explain the mechanism for *that*, it sounds even worse. "Well, see, there's this token of membership in the community, and it's up 400% this week. Also the tokens are JPEGs of monkeys."

But look, pretty soon, what *are* we going to sell to each other? Online communities are valuable. There's money to be made.

D.
Finance

There are lots of online communities, though. One is Bored Ape Yacht Club, a self-selected club where you become a member by buying an expensive membership token. The value of that community is, I guess, you feel cool and exclusive? Maybe you befriend a celebrity or a venture capitalist, bonding over your apes.

Or there's social networking. Facebook is valuable; make a Neo-Facebook; give people a token; let them keep the value for themselves. "Advertisers can get your data only if they pay you in tokens," you tell them, or "You can earn tokens for posting, which you can then use to pay other people for posting." Why not?

Or gaming. "If you buy a laser in this game, it's an NFT, and it's yours to keep forever. Maybe you can use it in another game." Why not? These are standard claims about web3 that leave me mostly cold. I don't want to be in the advertising-data-selling or computer-game-arms-dealing businesses.

But other online communities are DeFi? Like, in some crude sense, what decentralized finance is is a big community of people who get together to pretend to trade financial assets—or, rather, who trade financial assets in a sort of virtual world. They've built derivatives exchanges and secured lending protocols and new ways to do market-making, but instead of trading stocks or bonds they trade tokens that they made up. And those tokens are valuable, in part because they're linked to other online communities (you can use DeFi to buy Ether that you then use to buy NFTs to become a Bored Ape owner), but also in part because DeFi is itself an online community, or cluster of communities, and the tokens it trades are points in that community. If you build a cool trading platform or execute a cool trade, you'll earn tokens, which you can spend on other cool trading platforms or trades. Talented financial traders are willing to work on projects to get those tokens. If you had some of those tokens, you could hire those traders.

A problem, and an advantage, of crypto is that it financializes everything. "What if reading your favorite book made you an investor in its stock." Feh, it's a story that only a venture capitalist could love. On the other hand, it's a story that venture capitalists love. A minimalist case for crypto is:

"It's an efficient way to get venture capitalists to put money into software projects."

Or it's a Ponzi. The web3 vision of having the customers of every project also be its investors works well in times of speculative excess, but it's disastrous in a crash. "All our customers have a stake in our success" is great when token prices go up, but it also means that all your customers become poorer when token prices go down, which makes it hard to attract customers.

But we've only really seen the boom. The problem with making every product also a Ponzi is that you can't be sure if your customers are there for the product or the Ponzi. When it collapses, you can. If they're still there–if they still use your product without getting rich off the token–then that means your product is promising. If not, well, you ran a Ponzi.

The great speculative frenzy of crypto and web3 over the past few years drew a lot of money and attention and talent into the crypto world. A great deal of that money and attention and talent went strictly to optimizing the speculative frenzy, to tweaking the tokenomics and leveraging the bets so people could make as much money as possible without actually building anything. Some of it probably went into building, though.

Now the speculative frenzy has, if not disappeared, at least cooled. Now if you're trying to raise money for a web3 project, it should probably do something, besides issuing a token that goes up. If it creates value for people, if the product is something people want, then the tokens will take care of themselves. Crypto people have a lot to prove on that front. One reason the 2022 crisis in crypto didn't spark a contagion is that crypto has so few connections to things that matter to people. They play games and gamble on the blockchain, but they don't have mortgages there.

Perhaps this is all a self-referential sinkhole for smart finance people, but honestly it would be weird if that's all it ever turned out to be. If so many smart finance people have moved into the crypto financial system, if they find it so much more enjoyable and functional and productive than the traditional financial system, surely they'll eventually figure out how to make it, you know, *useful*.

Here's another way to tell this story. There's the real world, and people do stuff in the real world. They grow food and build houses.

Over many centuries a financial system grew up, as an adjunct to the real world. That financial system enabled people to do more stuff in the real world. They could build railroads or semiconductor factories or electric cars, because they could raise money from strangers to fund their activities. They could buy bigger houses, because they could borrow money from banks. They could also trade out-of-the-money call options on GameStop, because that's fun and you can make memes about it, but that's an accidental feature of a financial system that mostly does serious stuff in the real world.

By 2008, or 2022, that system looked pretty abstract. When you think about modern finance, you often think about things like those GameStop options, or the system of payment for order flow that enables their trading, or synthetic collateralized debt obligations referencing other CDOs referencing pools of mortgage-backed securities. There's a house there somewhere, under the CDO-squareds. All the sophisticated modern finance can be traced back, step by step, to the real world. Sure, it's a lot of steps now. But the important point is that sophisticated modern finance was built *up*, step by step, from the real world. The real world came first, then finance, then the more complicated epiphenomena of finance.

Crypto, meanwhile, has built a financial system from first principles, pure and pleasing on its own, unsullied by contact with the real world. (I exaggerate: The basic function of *sending money* using crypto, Satoshi's original goal, is fairly practical. But, otherwise.) That's interesting as an object of aesthetic contemplation, and I've enjoyed contemplating it, and I hope you have, too. And it's attracted a lot of finance people who also enjoy contemplating it, and getting rich. And their task is to build back down, step by step, to connect the elegant financial system of crypto to the real world. You've built a derivatives exchange, cool, cool. But can a real company use it to hedge a real risk facing its real factory? You've built a decentralized lending platform, awesome. But can a young family use it to buy a house?

And the answer is, you know, maybe, give it time. The crypto system has attracted a lot of smart people who want to solve these problems, in part because they're intellectually interesting problems and in part because solving them will make these people rich.

But another part of the answer might be that the real world—growing food, building houses—is a smaller part of economic life than it used to be, and that manipulating symbolic objects in online databases is a bigger part. Modern life is lived in databases. And crypto is *about* a new way of keeping databases (on the blockchain).

If you build a financial system that has trouble with houses but is particularly suited to financing video games—one that lets you keep your character on the blockchain, and borrow money from a decentralized platform to buy a cool hat for her, or whatever, I don't know—then that system might be increasingly valuable as video games become an increasingly important part of life. If you build a financial system whose main appeal is its database, it will be well-suited to a world lived in databases. If the world is increasingly software and advertising and online social networking and, good Lord, the metaverse, then the crypto financial system doesn't have to build all the way back down to the real world to be valuable. The world can come to crypto.

Want more?
Subscribe to Matt Levine's *Money Stuff*, a daily newsletter on all things Wall Street and finance.

Editor: Pat Regnier

Development: Peru Dayani, James Singleton

Production: Emily Engelman, Steph Davidson, Thomas Houston, Michael Frazer, Justin McLean, Bernadette Walker

Designers: Albert Hicks IV, Alexander Shoukas

Photo editors: Aeriel Brown, Donna Cohen, Ryan Duffin, Dietmar Liz-Lepiorz, Leonor Mamanna

Audio: Mark Leydorf

Copy editors: William Elstrom, Nicholas Mullan, Minette Valeriano

Editorial assistance: Jim Aley, stacy-marie ishmael, Olga Kharif, Margaret Sutherlin, Folder Studio

Photos: Alamy (21), AP Images (1), Bloomberg (5), Classic Stock (1), Everett Collection (1), Getty Images (64), Juliana Tan (1), Library of Congress (1), Nasa (1), Netflix/Everett Collection (1), Reuters (2), Shutterstock (4), Trunk Archive (1), YouTube (1). Videos: Getty Images (3)

AMLO's Tren Maya Risks Mexico's Environment Near Cancun

June 3, 2022

US-China Cold War Has Japan Caught in the Middle

May 9, 2022

Lehman Brothers' Collapse: The Bankruptcy That Took Over a Decade to Unwind

May 18, 2022

Bloomberg New Economy Announces 2022 Class of Catalysts

Aug. 3, 2022

CPI Indicators Give Global Look at Food Inflation

May 19, 2022

Chanel, Givenchy Bags Stolen in Los Angeles Burglary Ring

July 1, 2022

Elite Equestrians Criticize Watchdog As Sex Abuse Scandals Plague Industry

March 10, 2022

3M Hid Its Toxic Chemical Contamination in Belgium Over a Decade

June 10, 2022