Love, Trust and Crypto

Professor Raghavendra Rau

14th November 2022

Introduction

In this lecture, I will talk about how crypto can solve problems of trust. Let's start with the question of how trust originated. Before 1950, during the pre-industrial revolution, most people lived in small villages or towns. Everyone knew everyone, and trust was all at the local level. You trusted your butcher because you knew him. You trusted your baker because you've been dealing with her for a very long time. Most people did not venture extremely far away from their villages or their towns, so they almost never had an occasion to meet people who were complete strangers to them.

Everything began to change during the industrial revolution. The introduction of steam power and the increasing mechanisation of tasks involved large fixed costs. In turn, to cover these fixed costs, businesses had to increase scale. And that meant that people had to work with other people whom they had never met before.

The problem only worsened after 1840, when mass production started becoming common. The earliest factories involving assembly lines and electrical energy were not located in small villages; they were located in towns. People started moving from villagers to towns, where they knew no one. They could no longer rely on locally produced sources of trust but would have to depend on intermediaries. For example, I might not be able to trust someone who I want to work with because I've never met her before, but I would trust the bank that witnesses the contract between me and this person. In the 1960s, with the introduction of automation electronics and computers, people started ignoring the identity of specific intermediaries. I did not need to trust my specific bank; I trusted the banking system.

Unfortunately, today, the world is even more complex. Scandals, crises, and other failures of the capitalist system have led to an increasing reluctance of people to trust the system. Financial intermediaries today are usually at the bottom of the trust tables year after year. What then is a source of trust in today's world? I'm going to argue that trust today can be created by technology.

Crypto

What do we know about crypto? Well, there is a lot of complicated jargon. People talk about distributed ledger technologies, blockchains, consensus protocols, smart contracts, shared databases and so on ad nauseum. It is difficult to understand how it works and it's difficult to understand why it is important.

At this stage, it is useful to understand the relationship between blockchain and crypto. crypto is the general phrase that applies to a lot of distinct types of transactions. For example, when we talk about money, we refer to cryptocurrencies. When we talk about contracts we can talk about smart contracts or decentralized finance. The underlying technology behind all of this though is based on a blockchain.

Blockchains

What is a blockchain? Let us start with the big picture. Blockchains are useful under three circumstances:

- The users want to be anonymous – use cryptography to protect anonymity
- The data is unstructured – use hashes to represent the data
- The data needs to be indelible – no one can alter the data without everyone finding out – use proof-of-work to validate data.

How Does a Bank Ledger Work?

You pay some money to your friend in the same bank as you. How does the bank record this?

The bank makes a ledger entry. Each ledger entry contains some information:

- The originating person (+ Account number)
- The destination person (+ Account number)
- The transaction detail (How much money is being transferred)

This is linked to your bank accounts. Another entry is made in each account:

- The starting balance
- How much is transferred
- The ending balance

A simple ledger entry might be

- Mr Black writes a cheque to transfer five bitcoins (BTC) from his account to Ms. Green
- Here Mr Black is the originator
- Ms. Green is the destination

And the ledger entry might look like this:

Origin	Destination	Amount
Mr. Black	Ms Green	5 BTC

The actual account might look like this:

Date	Starting balance	Amount CR/DR	Ending balance
14 Nov	10 BTC	5 BTC (DR)	5BTC

What problems might you envisage here?

1. Perhaps your bank manager (who keeps the records) diverts the money to her own account, so *she falsifies the transaction.*
2. Perhaps Mr. Black does not have enough money in his account in the first place – *so the cheque bounces.*
3. Or perhaps Mr. Black sends the money to Ms. Green but then turns around and spends the same cash to buy a coffee – *so he double spends.*
4. Or perhaps Mr. Black does not want everyone (or even his bank manager) to know he needed to send funds to Ms. Green – *he wants to be anonymous*

How does a blockchain help?

- No one has control of the ledger, so Ms. Red cannot divert the money to her own account. In general, no one can falsify anything on the chain – even though no one person is responsible for the ledger.
- There is no way for Mr. Black to spend the same money twice.
- And best of all, everyone's identity can be kept completely secret.

So, let us summarize:

- The blockchain is a way to store data when you do not trust your counterparty
- Trust is ensured by technology
- If you trust your counterparty, you do not need a blockchain
- If you do not need to store data, you do not need a blockchain

And there are distinct types of blockchains:

- If everyone is allowed to read entries, it is a public blockchain, otherwise private.
- If everyone is allowed to write entries, it is a permissionless blockchain; otherwise, it is permissioned.

Type of counterparties			
Known			Unknown
Trusted	Untrusted		
	Need to let outsiders check the data later	No need to let outsiders look at the data	
No need for a blockchain	Public permissioned blockchain	Private permissioned blockchain	Permissionless blockchain

But the bottom line is simple: **Blockchains need to solve three problems**

1. No falsification
2. No double-spending
3. Anonymity

They do this using three technologies:

1. Cryptography
2. Hashing
3. Mining.

Solving Problems of Love Using Blockchains

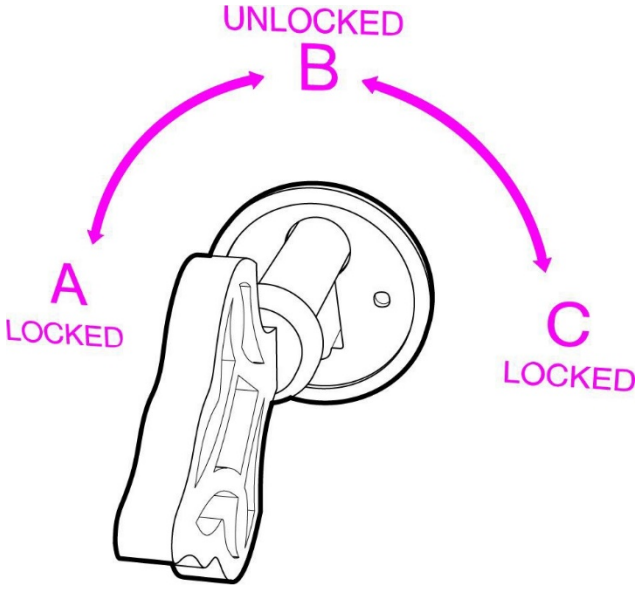
Romeo and Juliet want to send letters to each other. What problems do they face?

- They need to be sure that no one else can read their letters.
- They need to be sure that the letter is indubitably coming from only the two of them (not forged by Juliet's cousin, Tybalt)
- The letter is not garbled during sending.
- They need to be sure that Romeo (or Juliet) is not writing the same letter to five other girls (boys).

Problem 1: No one else should be able to read the letters.

Solution: Imagine a lockbox with two keys. The lockbox can be locked in one of two ways:

- It can be locked clockwise (from position B to C in the figure below)
- It can be locked anticlockwise (from position B to A in the figure below).



Key 1 can only turn clockwise (from A to B to C) and Key 2 can only turn anticlockwise (from C to B to A). Juliet keeps key 1 for herself and distributes many copies of Key 2 to everyone. Because Juliet is the only person who has a copy of Key 1, we can call this her private key. The second key which she distributes to

everybody is a public key.

Now suppose Romeo wants to send Juliet a letter. He puts the document in the box and uses a copy of her public-key to lock it. Remember, Juliet's public-key only turns anticlockwise, so he turns it to position A. Now the box is locked. The only key that can turn from A to B is Juliet's private key, the one she's kept for herself.

That is it! This is what we call public-key encryption: Everyone who has Juliet's public-key can put documents in her box, lock it, and know that the only person who can unlock it is Juliet.

Problem 2: Romeo needs to be sure that the letter comes from Juliet (not forged by her cousin Tybalt).

Solution: Juliet locks her box with her private key (turns it clockwise to position C). The only one who could do this is Juliet – her key is the only one that turns clockwise. So, it's pretty much guaranteed that Juliet put the letter in the lock box. This is called her digital signature. But now the problem is anyone with Juliet's public key can unlock the box by turning the key anti-clockwise. So, Juliet puts her lock box into a *second* lock box which is Romeo's lock box. She locks the second outer lock box with Romeo's public key. So, Romeo is the only one who is able to unlock the outer lock box with his private key, take out the second inner lock box, and unlock it with Juliet's public key.

Problem 3: Even if someone else gets the letter, the sender should be secret.

This is pretty much guaranteed by the public - private key encryption scheme. Romeo does not actually send a letter to Juliet by name, he sends it to her public key. As long as there's no association between the public key and a real person's name, the transaction is completely anonymous.

How on earth do you make such keys? Mathematics.

Consider the two problems:

- What is 17959 times 33851?
- Is 643712231 prime?

Which calculation is easier?

Just in case, you are curious, 17959 times 33851 is 607930109. 643712231 is not prime. 643712231 = 20261 × 31771 ... but both of those ARE prime.

In the best encryption schemes, such as the RSA algorithm, the public key consists of two numbers where one number is multiplication of two large prime numbers. The private key is also derived from the same two prime numbers.

If somebody can factorize the large number, the private key is compromised.

Therefore, encryption strength totally lies on the key size and if we double or triple the key size, the strength of encryption increases exponentially.

Problem 4: The letter is not garbled during sending

Solution: Hashing: Construct a summary of the letter.

The summary should have three characteristics:

1. Regardless of the length of the original data, the hashed summary always has the same length
2. A particular data input will always result in the same hash
3. Two different data inputs cannot result in the same hash – the no collision property

What on earth is hashing? It sounds vaguely illegal.

Not at all. In 1954, Hans Peter Luhn filed a U.S. patent on a "Computer for Verifying Numbers", specifically: Credit card numbers and Social Security numbers. These numbers were being used everywhere. But the numbers were difficult to remember, and they could be transcribed incorrectly or deliberately falsified. What was needed was a means of quickly verifying whether an ID number was valid.

Luhn's algorithm worked as follows:

- *Start with a 10-digit number.*
- *Double every second digit*
- *If any result is 10 or greater, add up the digits of that result to get a single-digit number (for example, "16" becomes 1 + 6 = 7)*

- *Add up all ten digits of the new number*
- *Multiply by 9*
- *Take the last digit of that result*

This recipe produces a single digit “check” number. In Luhn’s original formulation, a 0 indicated the original number was valid. In later versions, the check was simply appended to the original number as a final digit.

Hashing constructs a similar summary of any field. Consider a word: **adam.** Let us give each letter a numeric value: a = 1, b = 2, and so on.

Obviously, A and a will have different values and we will need to give values to the commas, periods, dashes, numbers themselves and all the other special symbols. But to make it easy for us, let us assume that there are only small letters in our text.

Then **adam = 1 4 1 13**

Set the maximum length of the hash. Let us suppose this is 10.

adam only takes four characters, so we add 1 bit, pad out the remaining length by zeros, and add the number of characters at the end. So adam becomes 1 4 1 13 1 0 0 0 4

We can handle longer phrases by chopping them up into shorter fragments that are never more than ten characters and then using mathematical operations to get the fragments into the same length of hash.

There are lots of hash calculators freely available on the Internet (please check the references for an example) and you should play with them to get a sense of how they work.

Problem 5: They need to be sure that Romeo (or Juliet) is not writing the same letter to 5 other girls (boys).

The blockchain contains a history of every transaction ever recorded on the blockchain. To understand what that means, consider the Yap stones used as currency in the South Pacific. The Yap stones were not actually found on the island of Yap. They were cut at great expense from a separate volcanic island and brought back to the island of Yap. However, the stones were large and once they were installed outside your house, it was exceedingly difficult to move them. So how would you use them as currency? Well, everybody in an island keep track of who owned which stone and when transfers happened. For example, if you spent a portion of your stone to buy something, the island's collective memory would say this stone was formerly owned by Raghu, but he sold a portion of that stone to somebody else and now that Bunny owns part of the stone. Not that this is not super difficult to do. The island's population is relatively small, so there are not that many transactions to keep track of.

The same sort of verification of transaction has to be done on the blockchain itself. But who verifies the transaction? The answer is a group of people called the miners. What do the miners do? Well, and the Bitcoin protocol the system sets an allowable solution to a problem. A mathematical problem. For example, the system can say the only allowable solution must have three leading zeros. The greater the number of zeros the more difficult it is to solve the problem. The miners try to construct a hash for a set of transactions, a block, so that their hash fits that pattern.

So, what does the process consist of? A minor arbitrarily selects a set of Bitcoin entries. He then proceeds to hash that block after adding a trial number. If the resulting hash fits the network rule, that is it has required number of leading zeros, he immediately broadcast set to the network, and collects a fee. if the trial number doesn't work, he tries a different trial number and restarts the process. This process is called proof of work. It is laborious, time consuming, and environmentally unsustainable.

But wait! Suppose a nefarious miner comes along later and alters the transaction in a previous block. How can this be stopped? The answer is every block is chained to the previous block. Specifically, we add the header from the previous block and hash the current block *plus* the header from the previous block together. What this means is if you change an intermediate block, its header will change. That means the next block will no longer be connected to this block - it has the old header in it. In order for the blockchain to be consistent, the miner has to correct this block as well with the new header he has just generated. Then he has to go on and correct the next block, and the next, and the next. All the while, new blocks are being added to the end of the blockchain. It is impossible for any one miner to add blocks fast enough to overtake all the other blocks being added to the end by the other miners all around the world. And that is why blockchains, or at least active traded blockchains, are impossible to hack.

What can we conclude from all this? Well, one thing is for sure. If blockchains had existed at the time of Shakespeare, Romeo and Juliet might never have been written, and that would indeed have been a tragedy.

References and Further Reading

Hash calculator Available at https://xorbin.com/tools/sha256-hash-calculator

Vryonis, P., 2013, Explaining public-key cryptography to non-geeks

(Available at https://blog.vrypan.net/2013/08/28/public-key-cryptography-for-non-geeks/)