The nuts and bolts of blockchain technology

Rohas Nagpal
Primechain Technologies Pvt. Ltd.
rohas@primechain.in

Abstract: Blockchain technology has earned the respect of Governments and banks around the world. This document provides a simple introduction to blockchain technology and briefly introduces terms such as cryptography, hash functions, proof-of-work, digital signatures, mining, merkle root. This document is intended for the novice reader and may suffer from errors inherent when a complex topic is (over?) simplified.

Depending upon the hemisphere you live in, you probably think that blockchain is either the Rajnikant or Chuck Norris of all technologies. Thanks to the massive media coverage and billions of dollars of investments, blockchain is a word that almost everyone has heard. But a lot of people don't understand the mathematics behind this revolutionary technology. Well, read on.

1. What is a blockchain?

Imagine a world without computer databases. There would be no ecommerce, no ATMs, no Internet banking, no Aadhaar or similar social security scheme, no Facebook, no Gmail, no WhatsApp!

Almost everything that makes the Internet so powerful and useful depends upon computer databases. The digital world relies very heavily on computer databases, even though most users are unaware of it.

Now imagine a database that is provably immutable / unchangeable and almost impossible to hack. That's a blockchain. At its core, a blockchain is an ordered and timestamp sequence of "blocks of information".

Blockchain technology, also called distributed ledger technology, derives its strength from strong cryptography and hash functions.

Blockchain technology was invented by the unknown inventor of the bitcoin crypto-currency in 2008. Simply put, the bitcoin crypto-currency runs on the bitcoin blockchain (simply referred to as The Blockchain). The Blockchain is a public blockchain where anyone can become a miner and details of every single bitcoin transaction are stored on each node.

Then there are other blockchain platforms such as - bitshares, ethereum, hyperledger, multichain, ripple and stellar.

Hyperledger and multichain can be used to public, private (e.g. the Government running a land registry) and consortium blockchains (e.g. a group of banks running a shared Know-Your-Customer or KYC platform).

2. The mathematics of it all

Sanya's a naughty young girl who's been grounded for a week. She wants to sneak out for desert with her friends but obviously can't let her dad know about it. She's not allowed to use her cellphone, so the only way for her to call her friends is using the good old landline in her dad's room.

Since she regularly gets grounded, she and her friends have worked out a simple system for sharing secrets. When she says, "*have you read the book I told you about*" she actually means "*let's sneak out tonight*". When she says something about "*page 10*" of the book, she means "*pick me up at 10 pm*". Continuing the logic, page 11 would mean 11 pm and so on.

So on the phone she asks her friend "*Have you read the book I told you about? Page 12 is really funny*", she means, "*Let's sneak out tonight, pick me up at midnight*".

What we have just seen is **cryptography** (and a rebellious teenager) in action in the real world.

The sentence "Let's sneak out tonight, pick me up at midnight" is *plain text* – what Sanya actually wants to convey. The sentence "Have you read the book I told you about? Page 12 is really funny" is the *cipher text* – something that an adversary (her dad in this case) should not be able to understand.

Encryption is the process of converting plain text to cipher text. The reverse process is *decryption*.

This science of encrypting and decrypting messages (*cryptography*) has been used for thousands of years. It is believed that when Julius Caesar sent messages to his generals, he replaced every A in his messages with a D, every B with an E, and so on through the alphabet. Only someone who knew the "shift by 3" rule could decipher his messages.

For example, if we want to encode the word "SECRET" using Caesar's key value of 3, we offset the alphabet so that the 3rd letter down, (D), begins the alphabet.

So starting with ABCDEFGHIJKLMNOPQRSTUVWXYZ

and sliding everything up by 3, you get

DEFGHIJKLMNOPQRSTUVWXYZABC

where D=A, E=B, F=C, and so on.

Using this scheme, the plaintext, "SECRET" encrypts as "VHFUHW". To allow someone else to read the cipher text, you tell him or her that the *key* is 3. This method is called *symmetric cryptography* and involves using the same key for encrypting as well as decrypting a message. This naturally poses a serious problem – what if an adversary gets hold of this key? At some point of time the sender and receiver need to exchange the key. That's when an adversary could get hold of the key. In modern cryptography, keys are extremely large numbers.

The *secure-key-exchange* problem was solved with the birth of *asymmetric key cryptography* – in which two different but related keys are used - the public key to encrypt data and the corresponding private key to decrypt the data.

If Sanya were to send an encrypted message to Karan, she would encrypt the message using his *public key* (which is available to the world). Once encrypted, the message can only be decrypted using Karan's *private key* (which would only be available to Karan).

Before we get into the nuts and bolts of how blockchains work, we need to understand some more concepts including **hash functions**. A one-way *hash function* takes an input (e.g. a PDF file, a video, an email, a string etc.) and produces a fixed-length output e.g. 160-bits.

The hash function ensures that if the information is changed in any way – even by just one bit – an entirely different output value is produced. The table below shows some sample output values using the sha1 (40) hash function.

Input	**Hash**
sanya	c75491c89395de9fa4ed29affda0e4d29cbad290
SANYA	33fef490220a0e6dee2f16c5a8f78ce491741adc
Sanya	4c391643f247937bee14c0bcca9ffb985fc0d0ba

It can be seen from the table above that by changing the input from sanya to SANYA, an entirely different hash value is generated. What must be kept in mind is that irrespective of the size of the input, the hash output will always be of the same size.

Two things must be borne in mind with regard to one-way hash functions:

1. It is computationally infeasible to find two different input messages that will yield the same hash output.
2. It is computationally infeasible to reconstruct the original message from its hash output.

Having understood hash functions, let's have a look at another interesting concept called **proof-of-work**. This is a way to reduce spam and denial of service attacks by requiring a computer to spend some time and processing power to solve something.

One such proof-of-work system that is used in blockchains is *hashcash*. The basic premise of *hashcash* is that if the sender of an email can prove that she has spent reasonable time and computational power to solve some puzzle, it can be believed that the sender is not a spammer. The logic is that spamming would be economically infeasible if a spammer had to spend non-trivial time and computational power for every single email being sent.

Let's develop an elementary proof-of-work system, based on hashcash, which can be used to control spam. Let's presume that *rohasnagpal@gmail.com* is sending an email to *info@primechain.in*. The sender must include something similar to the following in the header of the email:

rohasnagpal@gmail.com:info@primechain.in:06112016:xxxx

That's 4 pieces of information separated by colons. The first piece is the sender's email address, the second is the receiver's email address and the third is the current date in DDMMYYYY format (6th November, 2016 in this example). The fourth piece is something that needs to be calculated by the sender's computer. Let's call it a *nonce*.

The objective is to find an input that would result in a sha256 hash which begins with 4 zeros.

So we start the nonce at a value of 0 and then keep incrementing it (0, 1, 2, 3 …) and calculating the hash. Something like this:

Input	rohasnagpal@gmail.com:info@primechain.in:06112016:0
sha256 hash	2d87bf06373f4e91b43ab6180e30da0bf3f98efb44c5d5e2f7151b3179413bf6

Input	rohasnagpal@gmail.com:info@primechain.in:06112016:1
sha256 hash	cb3616e4ab0cee86badf0a598d1a151e06289c2c7e35f91554dc1ad7d128a99d

Input	rohasnagpal@gmail.com:info@primechain.in:06112016:2
sha256 hash	8d04a9e7ccd2c84549744c7fdbd48e3784ea3ab10020499a89349875726e3536

And so on till .. 76063

Input	rohasnagpal@gmail.com:info@primechain.in:06112016:76063
sha256 hash	0000b3c73f0cd6a92158b713fbade5f898dffeefc0a615d050b1ea391bd39906

Calculating this may not take a genuine sender a lot of time and computational power but if a spammer were to make these calculations for millions of emails, it will take a non-trivial amount of time and computational power.

At the receiver's end, the computer will simply take the following line from the header of the email and calculate the hash.

rohasnagpal@gmail.com:info@primechain.in:06112016:76063

If the hash begins with a pre-defined number of zeros (4 in this example), the email would not be considered spam. This will take the receiver a trivial amount of time and computational power since it just has to calculate the hash of one input. The date can be used as an additional validation parameter – e.g. if the date is within 24 hours of the time of receipt, the email will be approved for download.

A very important application of public key cryptography is a **digital signature**. In this, the signer first calculates the hash of the message she wants to digitally sign. Then using her private key and the hash, she creates a digital signature, using the relevant algorithm.

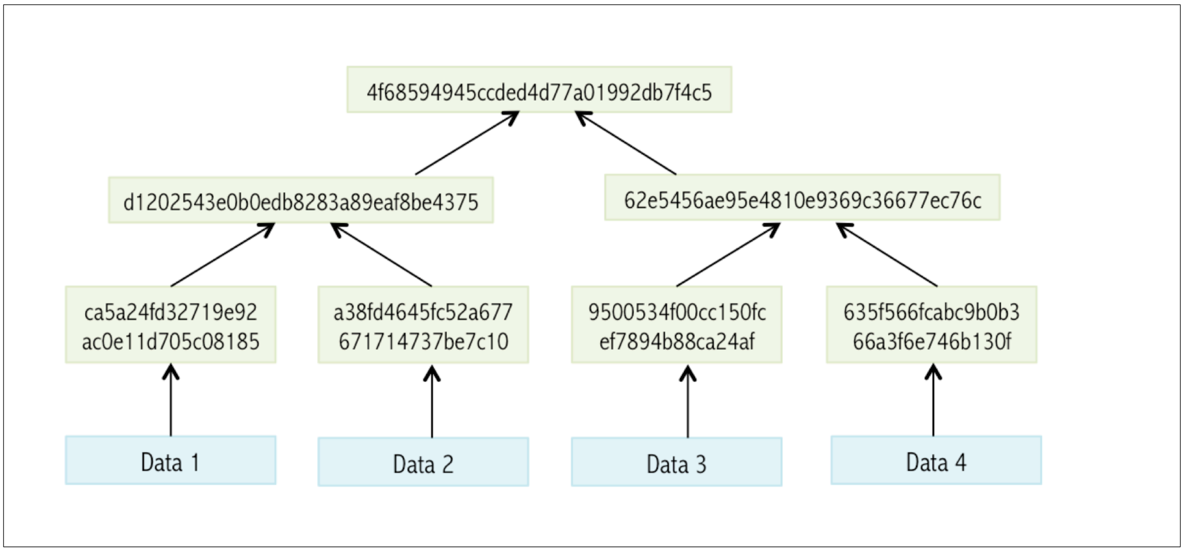
This digital signature is unique to the message.

The signer then sends the message and the digital signature to the receiver. The receiver re-computes the hash from the message. The receiver also computes another string using the digital signature and the signer's public key (using the relevant algorithm). If this string and the hash match, the digital signature is verified.

A **blockchain** is a public ledger containing an ordered and time-stamped record of transactions. In addition to preventing double-spending, the blockchain prevents the modification of previous transaction records.

A block of one or more new transactions is collected into the transaction data part of a block. Copies of each transaction are hashed, and the hashes are then paired, hashed, paired again, and hashed again until a single hash remains - the merkle root of a merkle tree. This is illustrated below:

Illustration 1: Merkle tree


4f68594945ccded4d77a01992db7f4c5
d1202543e0b0edb8283a89eaf8be4375
62e5456ae95e4810e9369c36677ec76c
ca5a24fd32719e92 ac0e11d705c08185
a38fd4645fc52a677 671714737be7c10
9500534f00cc150fc ef7894b88ca24af
635f566fcabc9b0b3 66a3f6e746b130f
Data 1
Data 2
Data 3
Data 4

4f68594945ccded4d77a01992db7f4c5 is the *merkle root* of the 4 transactions (or pieces of data) in the illustration above. This is stored in the block header. Additionally, each block also stores the hash of the header of the previous block.

This chains the blocks together and ensures that a transaction cannot be modified without modifying the block that records it and all following blocks. Transactions are also chained together. This is illustrated below:

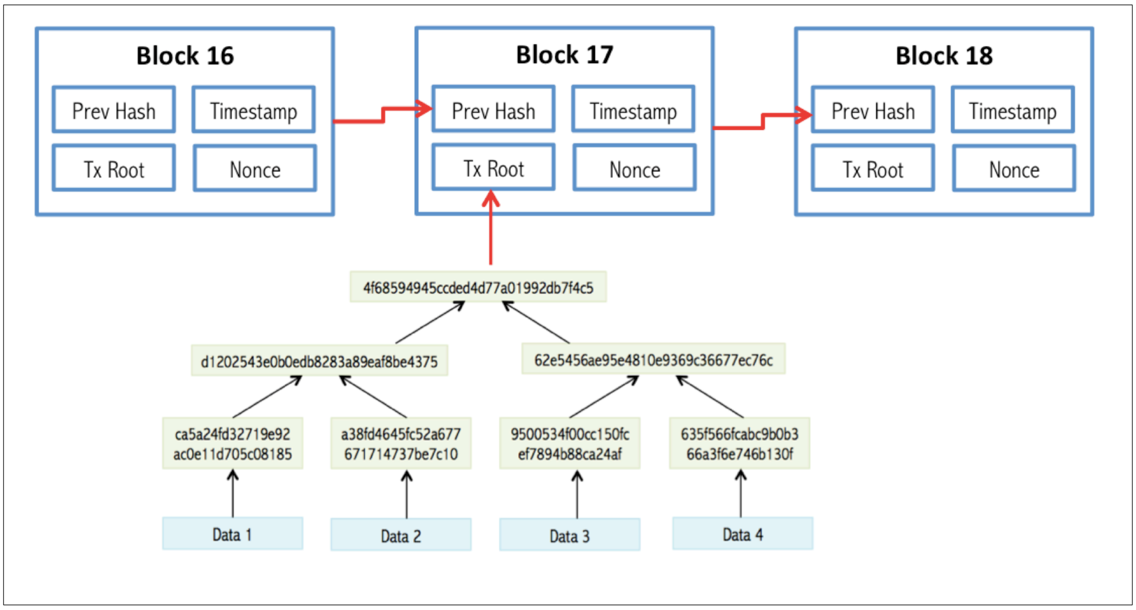

Block 16
Prev Hash
Timestamp
Tx Root
Nonce
Block 17
Prev Hash
Timestamp
Tx Root
Nonce
Block 18
Prev Hash
Timestamp
Tx Root
Nonce
4f68594945ccded4d77a01992db7f4c5
d1202543e0b0edb8283a89eaf8be4375
62e5456ae95e4810e9369c36677ec76c
ca5a24fd32719e92 ac0e11d705c08185
a38fd4645fc52a677 671714737be7c10
9500534f00cc150fc ef7894b88ca24af
635f566fcabc9b0b3 66a3f6e746b130f
Data 1
Data 2
Data 3
Data 4

Illustration 2: A blockchain

Blockchains use a *proof-of-work* technique similar (but more complex) than the one discussed earlier in this article. Since good cryptographic hash algorithms convert arbitrary inputs into “seemingly-random” hashes, it is not feasible to modify the input to make the hash predictable. To prove that she did some extra work to create a block, a *miner* must create a hash of the block header, which does not exceed a certain value.

The term *miner* must not be compared with a gold or coal miner in the real world. While a gold miner digs into the earth to discover gold, a blockchain miner uses computational power to calculate hashes. To add an entire block to the block chain, a *miner* must successfully hash a block header to a value below the target threshold.

The first-ever block is known as the *genesis* block. Each subsequent block is addressed by its block height, which represents the number of blocks between it and the genesis block.

New blocks are added to the block chain if their hash is at least as challenging as a difficulty value expected by the *consensus protocol* e.g. according to the bitcoin protocol, it should take 2 weeks for 2016 blocks to be generated. If the time taken is more or less than 2 weeks then the difficulty value is relatively decreased or increased every 2 weeks.

3. In conclusion

I believe that by 2020, blockchain technology will enable massive social upliftment and economic prosperity, the likes of which the world has never seen before.

Blockchains will minimise fraud and maximise efficiency, security & transparency in supply chains, healthcare, global money systems, financial technologies, democratic elections, auction of public assets, energy trading, electronic record authentication, delivery of Government services, IoT (Internet of Things) and more.

According to the *Reserve Bank of India* - "*With its potential to fight counterfeiting, the 'blockchain' is likely to bring about a major transformation in the functioning of financial markets, collateral identification (land records for instance) and payments system*.

According to the UK Government, distributed ledger technologies *have the potential to help governments to collect taxes, deliver benefits, issue passports, record land registries, assure the supply chain of goods and generally ensure the integrity of government records and services.... In summary, distributed ledger technology provides the framework for government to reduce fraud, corruption, error and the cost of paper-intensive processes. It has the potential to redefine the relationship between government and the citizen in terms of data sharing, transparency and trust. It has similar possibilities for the private sector*.

According to the Reserve Bank of Australia governor Glenn Stevens, *the blockchain technology could bring significant benefits to the global banking system*.

In a recent announcement, the Crown Prince of Dubai announced a strategic plan that would see *all Dubai government documents secured on a blockchain by 2020*.