Components of a Blockchain

Kostubh Agarwal [20872383]

Executive Summary

A blockchain is a decentralized database, made up of a network of computers, referred to as nodes[1]. More specifically, it is a distributed ledger – a record of history, that is collectively agreed upon and upheld by these nodes. A block is a unit of storage. And a chain is the cryptographic mechanism by which the blocks are secured and linked together. By analogy, a block is like a train carriage, and the chain is the coupling connecting the carriages. Altogether, the blockchain can be thought of as an ever-expanding cargo train that transports data, and cannot be stopped or derailed. This technical brief covers the core components that make up a blockchain, focusing primarily on Bitcoin[2]. Bitcoin is the first blockchain and it serves as a framework for a cryptocurrency[3].

Introduction

Trust, or the lack of it, has always played a role in human interaction. As a species hard-wired for survival, humans are selfish and often untrustworthy. Within small communities, this is less of an issue, as people can be held accountable. However, in larger communities, this is not the case. In an increasingly digital environment, people are interacting from opposite ends of the globe. While this holds, it is also the case that an apartment resident may never interact with their neighbour across the hallway. Digitally, we are closer than ever, even though physically we may not be. In a world where interacting with strangers is commonplace, systems of accountability are critical. The most common systems of accountability are referred to as third parties. Ideally, third parties are neutral players that mediate interactions between two people, without having any incentives of their own. Many establishments, businesses and technologies exist solely for this purpose. For instance, e-commerce retailers rely on payment gateways to validate, oversee and ensure that legitimate financial transactions take place between buyers and sellers. In the real world, third parties don't always fulfill their ideological definition — they can be selfishly motivated and sometimes even untrustworthy. Even if the third-party service is a software product, humans can still shut down, manipulate, or add fraudulent data with a few keystrokes. By that standard, software today is not truly autonomous. True autonomy requires that the software cannot be altered or removed once deployed. Until the invention of the blockchain, this has not been possible. Blockchains offer an alternative structure, where software can be fully autonomous. This is possible because blockchains are immutable and permissionless. Chris Dixon, a partner at a16z, refers to blockchains as "computers that make commitments" [7]. These commitments can be self-executing rules, protocols, and/or algorithms baked directly onto the blockchain — ensuring full autonomy.

---

[1] for simplicity's sake this paper assumes every node is a mining node
[2] this paper is primarily based on the Bitcoin blockchain, although many others exist
[3] digital currencies which exist on a Blockchain

Transactions / Messages

Assuming a network of nodes pre-exists, to start a blockchain, a user must trigger a transaction. In the case of a cryptocurrency, this may be a statement asserting that one party has sent some amount of cryptocurrency to another. For Bitcoin, the first transaction, known as the “genesis block” [1] was simply a deliverance of 50 Bitcoin to the user who mined[4] the first block. With a general-purpose blockchain like Ethereum, this may be a piece of code that is to be executed when a specific set of conditions are met — a smart-contract[5]. Whether it be a Bitcoin transaction or an Ethereum message, the underlying principles are the same.

```
00000000   01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000010   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000020   00 00 00 00 3B A3 ED FD  7A 7B 12 B2 7A C7 2C 3E   ....;£íýz{.²zÇ,>
00000030   67 76 8F 61 7F C8 1B C3  88 8A 51 32 3A 9F B8 AA   gv.a.È.Ã^ŠQ2:Ÿ¸ª
00000040   4B 1E 5E 4A 29 AB 5F 49  FF FF 00 1D 1D AC 2B 7C   K.^J)«_Iÿÿ...¬+|
00000050   01 01 00 00 01 00 00 00  00 00 00 00 00 00 00 00   ................
00000060   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000070   00 00 00 00 00 00 FF FF  FF FF 4D 04 FF FF 00 1D   ......ÿÿÿÿM.ÿÿ..
00000080   01 04 45 54 68 65 20 54  69 6D 65 73 20 30 33 2F   ..EThe Times 03/
00000090   4A 61 6E 2F 32 30 30 39  20 43 68 61 6E 63 65 6C   Jan/2009 Chancel
000000A0   6C 6F 72 20 6F 6E 20 62  72 69 6E 6B 20 6F 66 20   lor on brink of 
000000B0   73 65 63 6F 6E 64 20 62  61 69 6C 6F 75 74 20 66   second bailout f
000000C0   6F 72 20 62 61 6E 6B 73  FF FF FF FF 01 00 F2 05   or banksÿÿÿÿ..ò.
000000D0   2A 01 00 00 00 43 41 04  67 8A FD B0 FE 55 48 27   *....CA.gŠý°þUH'
000000E0   19 67 F1 A6 71 30 B7 10  5C D6 A8 28 E0 39 09 A6   .gñ¦q0·.\Ö¨(à9.¦
000000F0   79 62 E0 EA 1F 61 DE B6  49 F6 BC 3F 4C EF 38 C4   ybàê.aÞ¶Iö¼?Lï8Ä
00000100   F3 55 04 E5 1E C1 12 DE  5C 38 4D F7 BA 0B 8D 57   óU.å.Á.Þ\8M÷º..W
00000110   8A 4C 70 2B 6B F1 1D 5F  AC 00 00 00 00            ŠLp+kñ._¬....
```

Figure 1: The Bitcoin Genesis Block [1]

Hashes and Hashing Functions

Hashing is a technique that generates a fixed-bit string value based on the input of an arbitrary set of data. It is commonly used as a cryptography technique used to secure and verify data. The fixed-bit string is known as a hash. A hash is simply a unique piece of identification for some set of original data. It is like a barcode. A barcode on an apple will identify that apple, but on its own, the barcode is not the apple. Similarly, the original data cannot be reproduced from a hash. Hashing functions are unidirectional. On the contrary, a hash differs from a barcode, by the precision by which it represents data. A hashing algorithm is designed so that any change in input, even as simple as an addition of space in a sentence, will generate a completely new and unique hash. Additionally, hash functions are unpredictable. It is virtually impossible to predict what fixed-bit string will be generated from a given set of data. Collectively, these properties enable hashes to verify data — Because if a piece of data has been tampered with, its corresponding hash would not match the pre-validated hash. In a blockchain, a wide variety of hash functions are used. Every piece of data in the blockchain has gone through a hash function at least once. The blockchain acts as a database for these hashes.

```
function(data) ⟶ 48037298457328947235...3294732984
```

Figure 2: An example of a hash function. The output is known as a hash.

---

[4] Refer to the Proof of Work section

[5] A concept describing a piece of code that automatically executes upon a set of pre-defined conditions, first introduced by Nick Szabo [15]

Digital Signatures

In a blockchain, every user has both a private key and a public key. This is an elaborate system to help digitally sign data. Digital signatures are critical for declaring ownership, verifying, and protecting data, whether that be digital assets, messages, or cryptocurrency transactions. For Bitcoin, the digital signature mechanism consists of both a private and digital key. A public key is public and a private key is private, for a given person. Together these generate and verify digital signatures.

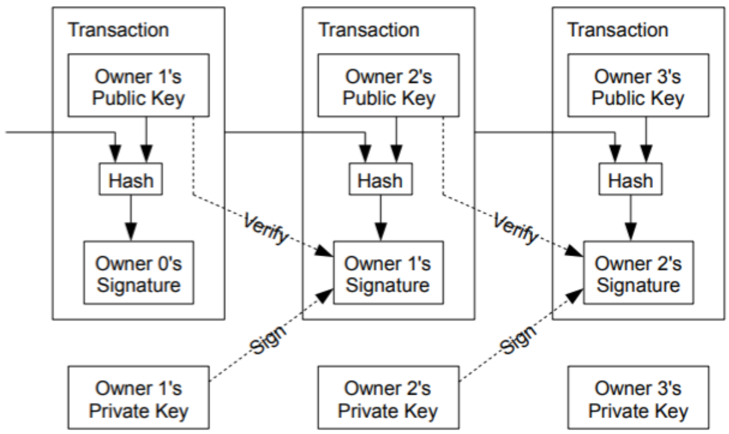

Figure 5: Digital signature system employed by Bitcoin. [1]

To digitally sign a transaction/message, the private key along with the data of the message/transaction is input into a hash function.

```
function(privateKey, message) → digitalSignature
```

Figure 3:A hash function that generates a digital signature

The output is a unique digital signature. A digital signature is dynamic, changing every time a transaction/message is to be signed. The hash is unique for every transaction and cannot be replicated. The digital signature now has to be verified by the public key. The signature, message/transaction and public key are then input into a hash function that produces a boolean value — validating a digital signature.

```
function(digitalSignature, message, publicKey) → T/F
```

Figure 4: A hash function that generates a boolean value, verifying a digital signature

After being digitally signed and verified, the transaction is broadcast to the network of nodes.

Proof of Work[6]

Decentralized databases have existed since the beginning of the internet. Back in the early nineties, webpages were likely to run one's own server, located in their own home. AWS, Azure, and other centralized cloud services were a decade away. Decentralization is not new, but what separates a blockchain from the early internet ecosystem, is the peer-to-peer consensus mechanism by which a common set of data is regulated, governed, and upheld — while being open to anyone and everyone. Consensus systems, although effective at distributing risk, are prone to corruption. That is why it is critical that nodes[7] in a blockchain are not concentrated in any one location. In real life, government identification often serves as a prerequisite for someone to participate in a consensus system, like a democratic election. The government maintains a record of all of its citizens and prevents such forms of corruption. Unfortunately, the same principle does not apply to the cyber realm. Any internet user can create infinite digital identities, whilst remaining completely anonymous or pseudonymous. That is why the blockchain includes a mechanism known as Proof of Work to ensure resistance to such threats.

In the case of proof of work, this cost is directly tied to CPU cycles, which is tied to CPU power, tied to electrical consumption. To add a new block, a user must solve a complex computational puzzle involving re-applying hash functions until a certain number of the first digits of a hash function are zero. The probability of guessing the correct hash is very unlikely. Thus the CPU has to go through a highly intensive guess-and-check process until this is achieved. This puzzle is similar to brute force password guessing. Since the transactions may be collected by multiple nodes, proof of work introduces a lottery scenario, where contribution to the network is proportional to the computational power being deployed. The first node to complete their computational puzzle gets to publish the block, containing transactions, to the network. Overall, proof of work serves as a barrier to entry that prevents one user from having majority representation[8].

Time-stamps

There arises a scenario where one transaction is valid, but in a collective, with other transactions, it is not valid. For instance, if user A has $100 total, but creates 3 different transactions claiming to pay each user $100: D, C, E. These are published to all the nodes in the network. Following this, three different blocks are mined, each including one of these three transactions. The three blocks then are published to the nodes in the network. Which block should the network validate and add to the chain? Which two blocks should be invalidated because of invalid transactions?

This scenario is representative of the double-spending problem. The blockchain prevents double-spending because it maintains a historic track record, containing information about all of the transactions of cryptocurrency that have ever taken place on it. In this scenario, two of the three

[6] This section will cover proof of work, proof of stake is an alternative mechanism to achieve the same goal.
[7] These mechanisms are only applicable to the nodes that mine blocks. There are non-mining nodes that have no effect on the consensus mechanism.
[8] A general explanation. The security measures preventing attackers from corrupting the blockchain are expanded on in The Block-chain section.

transactions are invalid, as they would be considered double-spending. So the question arises of which block to approve. To simplify things, the blockchain has a policy of accepting blocks in order of chronology. Chronology is upheld by adding time-stamps to every block. This serves as proof that a block was accepted by the majority of nodes at a given time and maintains chronological order. To maintain a tight cryptographic link between blocks, the time-stamp along with the previous block's time-stamp is run through a hash function before being added to the block. So only the first block, including the first transaction, would be approved. The other 2 blocks would be discarded.

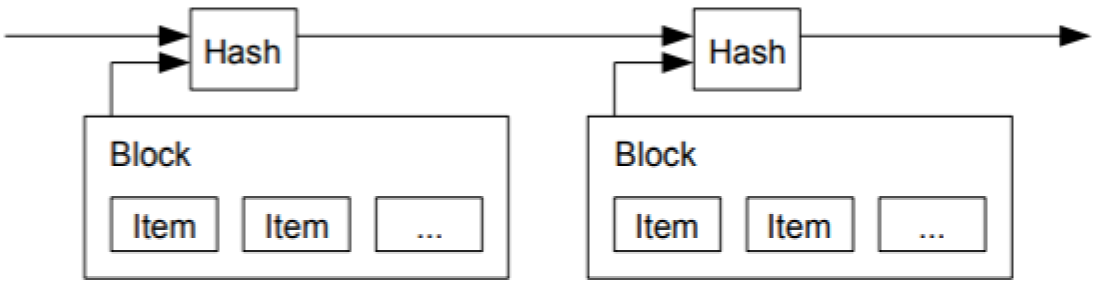

Figure 6: Illustration of the Bitcoin time-stamp server [1]

The Blockchain

A block is a unit of storage on the blockchain. So far, it includes hundreds of digitally signed transactions, a timestamp, proof of work, and the previous block's hash. All of these pieces of data are hashes. All of these components are then run through another hash function which generates a hash for the entire block.

```
function(proofOfWork, timeStampHash, transaction, previousBlocksHash) →
1289189138...28932
```

Figure 7: Hash encapsulating all of the hashes included in the block.

This hash is representative of all the data within a block. If any of the previous pieces of data are even minutely tampered with, an invalid hash will be generated.

All of the components of the block then have to be verified by the majority of nodes before being added to the blockchain. 51% at least. The contents of a block must not conflict with the existing blockchain — the history that has already been agreed upon. Only then can it be added to the chain. If there are any conflicts, the block will be rejected.

Another critical aspect of the design is the fact that only the longest blockchain is maintained. Every block is representative of a lot of CPU computation. The blockchain as a whole is representative of the collective, ever-expanding CPU power. For an attacker to attack the blockchain — to change, remove, add data would require a longer blockchain to be created to replace the existing one.

To “outpace” the “honest chain” the attacker would need to do a magnitude equivalent of all the proof-of-work ever completed for the entire blockchain, since its birth — the collective work of millions of nodes over potentially many years. Furthermore, an incentive structure has been created such that it would be more rewarding to generate new blocks than to replace the honest chain with one that is corrupted. This introduces a concept known as tokenomics — A complex field of study regarding the economics of digital tokens, and more specifically the inbuilt incentive systems. These incentive systems make it more rewarding to support a network than to hack it.

Even still, if an attacker can assemble enough CPU computation to overwrite the longest chain, and is not motivated by the built-in incentive structure if a blockchain is still hacked users have one more defence. They can simply fork and re-deploy the shorter chain. A copy / paste. Blockchains rely on the collective support of their nodes, which are users. If people don’t believe in the validity of a blockchain, they would simply leave, rendering it useless.

Conclusion

Overall, the blockchain is a haven for trust and data. Once data has been added to the blockchain it is permanent — immutable. Blockchains like Ethereum enable users to create autonomous programs. Bitcoin has set all its monetary policy in advance. If users decide to trust the policies of a given blockchain, they can have full confidence that these will not change. Ever. On the other hand, blockchains are permissionless. There is no central authority. Rather the transparent nature of the Blockchain makes accountability a collective responsibility. Anyone and everyone can see what is on the blockchain. Anyone and everyone can see how the blockchain works. Anyone and everyone can leverage the blockchain. Rather than bestowing trust upon third parties, humans can now trust a protocol, which is autonomous, abundantly transparent, and accessible.

References

[1] Nakamoto, S. (2008) Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf

[2] Buterin, Vitalik. "Ethereum Whitepaper." *Ethereum.org*, 2013, ethereum.org/en/whitepaper/.

[3] 3Blue1Brown. "But How Does Bitcoin Actually Work?" *YouTube*, 7 July 2017, www.youtube.com/watch?v=bBC-nXj3Ng4.

[4] "Bitcoin Whitepaper - Programmer Explains." *Www.youtube.com*, www.youtube.com/watch?v=MCxOwPlVVgA. Accessed 15 Jan. 2022.

[5] "Blockchain.com Explorer | BTC | ETH | BCH." *Www.blockchain.com*, www.blockchain.com/explorer?view=btc. Accessed 15 Jan. 2022.

[6] Ferriss, Tim. "The Quiet Master of Cryptocurrency — Nick Szabo (#244)." *The Blog of Author Tim Ferriss*, 4 June 2017, tim.blog/2017/06/04/nick-szabo/.

[7] TechCrunch. "Chris Dixon: Crypto Networks and Why They Matter." *YouTube*, 13 May 2020, www.youtube.com/watch?v=2wxtiNgXBaU. Accessed 6 Dec. 2020.

[8] "Decentralizing Everything with Ethereum's Vitalik Buterin." *TechCrunch*, techcrunch.com/video/decentralizing-everything-with-ethereums-vitalik-buterin/. Accessed 15 Jan. 2022.

[9] Dixon, Chris. "What Is Blockchain: Computers That Can Make Commitments." *Andreessen Horowitz*, 27 Jan. 2020, a16z.com/2020/01/27/computers-that-make-commitments/#:~:text=Blockchains%20are%20computers%20that%20can. Accessed 15 Jan. 2022.

[10] "Hash Value | Practical Law." *Content.next.westlaw.com*, content.next.westlaw.com/w-003-3374?__lrTS=20200728124753789&transitionType=Default&contextData=(sc.Default)&firstPage=true. Accessed 15 Jan. 2022.

[11] "Hashing and Digital Signature in Blockchain." *101 Blockchains*, 16 May 2021, 101blockchains.com/hashing-and-digital-signature-in-blockchain/.

[12] *How SHA-256 Works Step-By-Step - Qvault*. 8 July 2020, www.google.com/url?q=qvault.io/cryptography/how-sha-2-works-step-by-step-sha-256/&sa=D&source=docs&ust=1642229281830861&usg=AOvVaw0JRhJVTH5L5m5tFipnIQ03. Accessed 15 Jan. 2022.

[13] "Introduction to Hashing and Its Uses." *2brightsparks.com*, 2019, www.2brightsparks.com/resources/articles/introduction-to-hashing-and-its-uses.html.

[14] N-able. “SHA-256 Algorithm.” *N-Able*, 12 Sept. 2019, www.n-able.com/blog/sha-256-encryption.

[15] Szabo, Nick. “Smart Contracts.” *Www.fon.hum.uva.nl*, 1994, www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html.

[16] Wikipedia Contributors. “History of Bitcoin.” *Wikipedia*, Wikimedia Foundation, 14 Oct. 2019, en.wikipedia.org/wiki/History_of_bitcoin.