SHA-3 and the use of Hashing in Cryptocurrencies.

Abstract:

The latest hashing algorithm in the Secure Hash Algorithm (SHA) family, the SHA-3. In the world of Blockchain and Cryptocurrencies currently depends on hashing to process new blocks of data which contain the transactions carried out on the target blockchain network. Nowadays the term "blockchain" is loosely used in everyday jargon however the true meaning behind block chain comes from the cryptographic operation that connects the blocks by inserting the hash of the previous block into the current block thus forming a chain.

Considering the current state of the Bitcoin cryptocurrency where currently there are 617488 blocks and the chain is constantly growing keeping in mind the very first block was processed on the 9th of January 2009 shows the growth curve over the years and ultimately the increased difficulty in mining subsequent blocks in the future, ultimately highlighting the importance of using an efficient and brute-force proof hashing algorithm. These concepts will be discussed to give an in-depth understanding and highlight the importance of hashing in blockchain networks.

Introduction

Hashing is a one-way algorithm this means that once a value is hashed there is no way to "un-hash" the value, this is what is meant by a one-way function. Each hashed value will return a value of a fixed length combined of various characters, being more precise a string of bits.

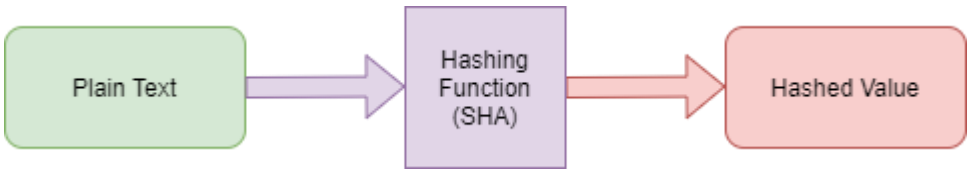

Figure 1 Diagram of a one-way hashing function

For example, a cryptographic hash function increases the security and efficiency of a digital signature scheme when the digest is digitally signed instead of the message itself. In this context, the collision resistance of the hash function provides assurance that the original message could not have been altered to a different message with the same hash value, and hence, the same signature. [6]

How is a block chain formed?

Each block chain network has its "block of origin", as the name implies a block chain is a forever growing list of data that are linked to each other and uses a similar concept as a linked list where in a block chain records cannot be inserted in the middle of a list but can only be appended to the end of an already existing chain.

Following the block of origin, each new block includes the information about its parent this allows for traversing the block chain network from the latest block to the starting block, thus defining a clear structure.

First Impressions

Difficulty

An important factor to consider is the difficulty of hashing, as the underlying hardware improves with each year the hashing rate delivered by end users or mining pools increases hence there has to be a mechanism in place which will increase the difficulty of hashing otherwise the blocks would be calculated too quickly and number of blocks would expand exponentially.

To prevent this a mechanism is introduced which increases the difficulty of mining the blocks, the implementation of this mechanism is different for each blockchain. However, a very common approach is to adjust the difficulty after discovering 2016 new blocks. The time it took to mine those 2016 blocks is taken into consideration and based on that a new difficulty is calculated, it is important to note that the difficulty can also be decreased if the pool hash rate delivered by the miners is lower as this would increase the computational time it takes to mine a single block.

In Figure.2 below we can see a chart which demonstrates the difficulty or the TH – Total Hash Rate for the past 4 years in the Ethereum Network, the hashing algorithm in this particular network is SHA-3.

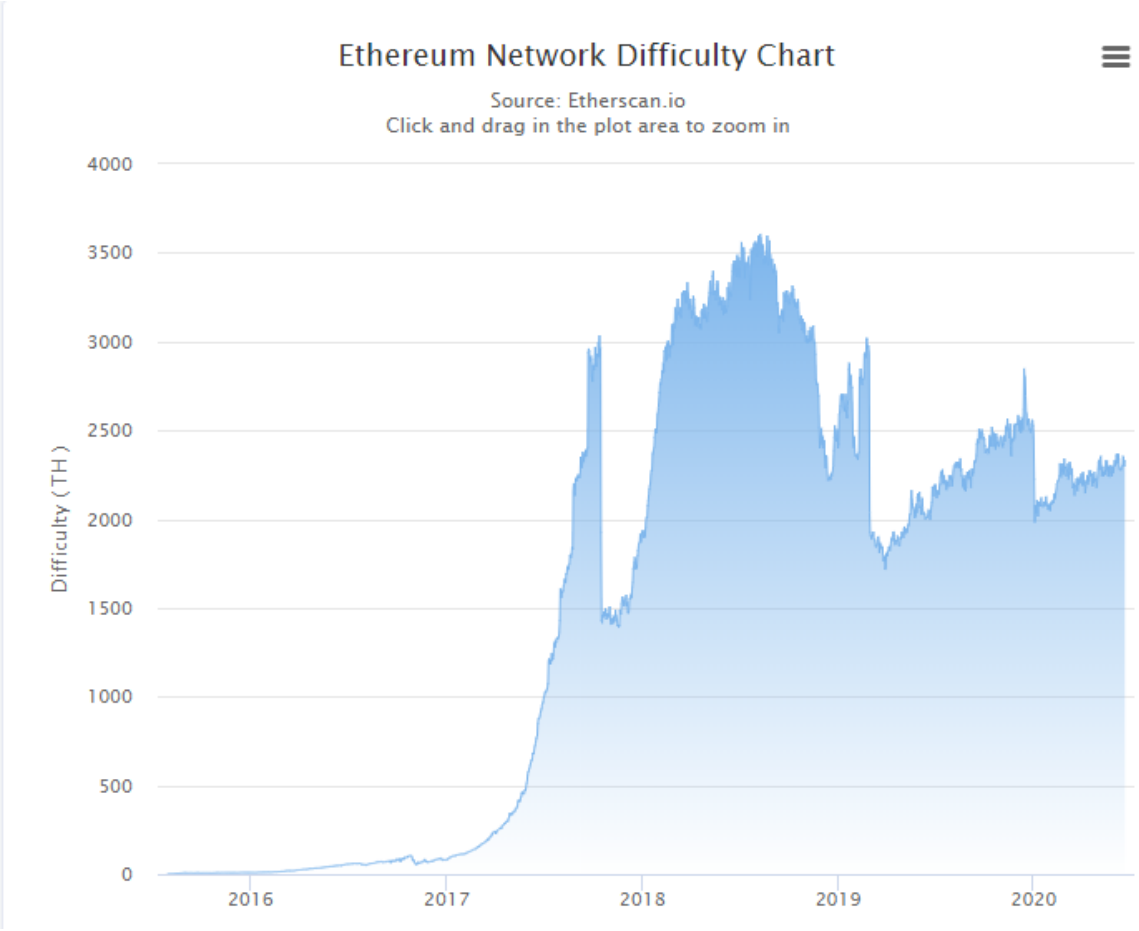

Figure 2 Ethereum Network Hashing Difficulty Chart, Source: https://etherscan.io/chart/difficulty

SHA-3 vs SHA-2

The SHA-3 algorithm has been chosen as a successor to ensure a more secure and robust hashing function exists for the foreseeable future. Currently SHA-2 is considered safe considering the current computational power available and the 256-bit or 512-bit complexity of the hash. If the structure of the output is not known iterating through all the possible combinations could take hundreds of years when considering plain brute forcing.

However there are other methods than brute forcing that can potentially be used to crack hashes such as collision attacks which have been successfully carried out on the SHA-1 hashing function, since both SHA-2 and SHA-1 are built using the Merkle–Damgård structure a similar attack could potentially be used on the SHA-2 in the future.

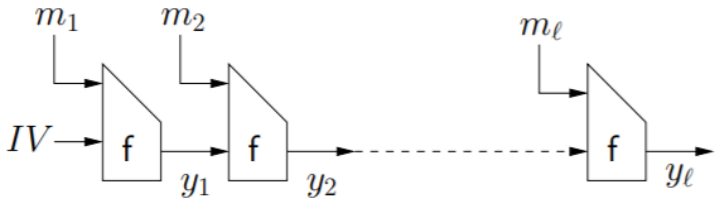

Figure 3 n Merkle-Damgard Construction Source: https://iacr.org/archive/crypto2005/36210424/36210424.pdf

For this reason, the SHA-3 hashing function chosen as the successor of SHA-2 is built on top of a different structure called the Sponge function which belongs to the Keccak-family.

The construction of the sponge function is based on a similar logic to the Merkle-Damgard structure however it introduces some additional complexity to the hashed output.

The sponge construction is a simple iterated construction for building a function F with variable-length input and arbitrary output length based on a fixed-length transformation or permutation f operating on a fixed number b of bits. Here b is called the width. The sponge construction operates on a state of b = r + c bits. The value r is called the bitrate and the value c the capacity. [1]

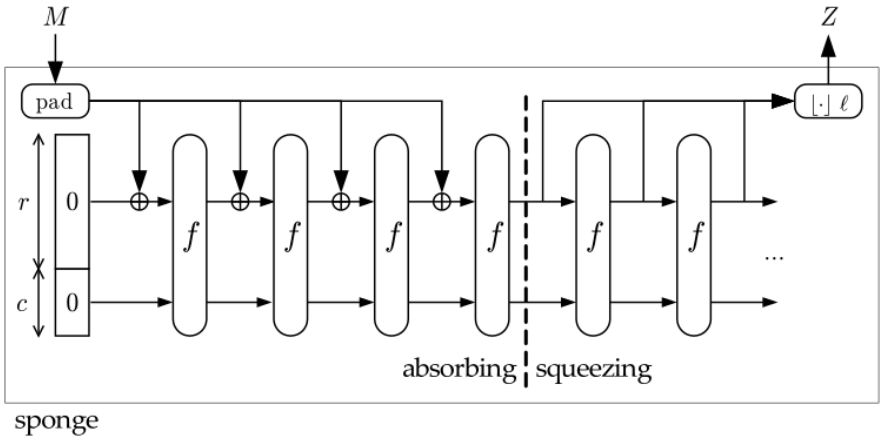

Figure 4 Sponge Function Construction Source: https://keccak.team/sponge_duplex.html

To calculate the KECCAK$[r,c]$ sponge function with the above mentioned parameters capacity c and bitrate r, if we apply the sponge construction to KECCAK-f$[r+c]$ and by applying a specific padding to the message input.

Table 1 Table of parameters for SHA3 functions

	r	*c*	Output length (bits)	Security level (bits)	Mbits	*r*
SHA3-224	1152	448	224	112	01	`0x06`
SHA3-256	1088	512	256	128	01	`0x06`
SHA3-384	832	768	384	192	01	`0x06`
SHA3-512	576	1024	512	256	01	`0x06`

Examples of hashes:

The text message that will be hashed is “Use of Hashing in Cryptocurrencies”, let’s see and compare the SHA-3 digest for 224, 256, 384 and 512 hash sizes.

Table 2 Hashes generated using a SHA3 Hash Generator Source: https://www.browserling.com/tools/sha3-hash

Hash Size	**Output**
224	41494182432150ced8f91307580908f58c8f09f91e5917a1a31d4095
256	f3a14350aeadf909a71d3d2d59ccaac30c6196bcf290a9e79c69c7f326d6a09a
384	133e1bb5a88e2f7ccfe2457abab355ac690057060066fc8183ecec5b017c800c7554f18efac013c22eccb6bbe4df79ef
512	91016410cbf3cf5ee841491211eb2fdb574259837a3e3295470452912f3281d265a2a3c7ee588bc7860d587b1ecb06eaff8708d28ec4989672a90ffd635f5ced

For the same plain text, we have obtained four different hashes, each with a different output hash size, inspecting the output digest there is no correlation between each of the output sizes. However, if an entity was to verify these checksums by once again calculating the hash of the sentence “Use of Hashing in Cryptocurrencies” the hashes produced for each output sizes would be an exact match.

Blockchain Concepts

In-depth look at a block

Understanding the concept and use of hashing in cryptocurrencies helps to further understand the structure of the block chain and the importance of every block mined. The main advantage of blockchains is their transparency which helps to prove the whole concept of decentralization and ensures the integrity of the network.

The transparency offered by the blockchains gives anyone the possibility to inspect every single block that has been mined since the beginning of a blockchain network, in Figure.5 below we can see a randomly chosen block that has been mined on the Ethereum network together with the hash of the block itself.

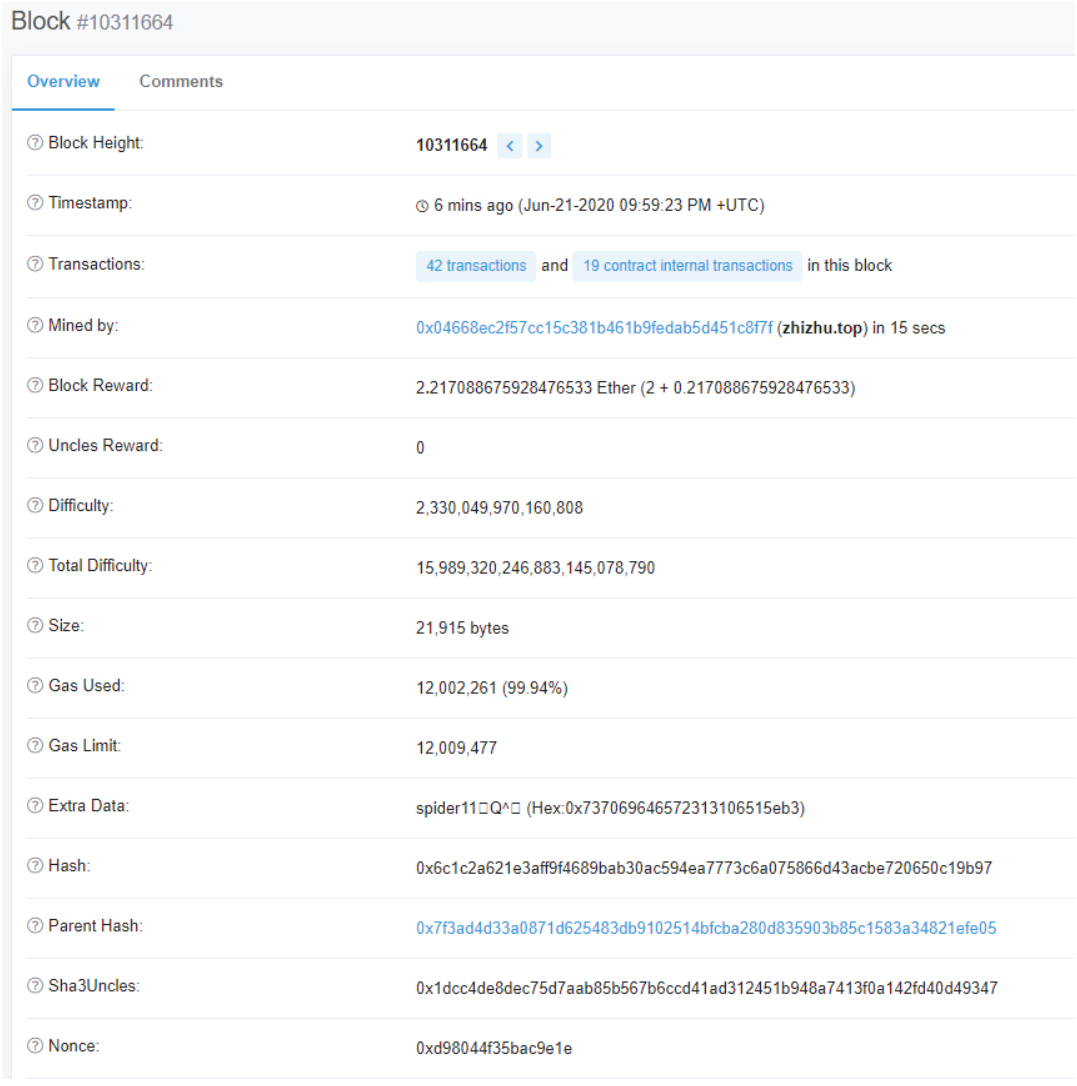

Figure 5 Date stored in a hashed block Source: https://etherscan.io/block/10311664

Some of the key data points as seen in Figure 5:

Block Height – The block number in the chain

Timestamp – The time at which the block was fully mined by a user or a pool of users

Transactions – Each block is a collection of transactions that occurred in the network during a particular period of time, after a block is completed all the transactions that occurred during this time period can be seen upon inspecting that block.

Block Reward – Each mined block offers a reward to the user or pool of users that provided the computational power to obtain its hash, this incentivizes the users to offer their processing power to the network.

Difficulty – As the difficulty of each block depends on the difficulty of previous block, this information is crucial for each subsequent block to calculate its difficulty.

Hash – The hash of the current block's header

Parent Hash – The parent hash is always included in each new block to from the chain.

Nonce – A very important value which represents the amount of transactions in the block, knowing the exact number of transactions is important as it prevents a potential replay attack from occurring. [2]

Hash Rate Performance

As the difficulty to mine a block varies another important indicator to a blockchain network is the hash rate of the network itself. In Figure 6 below we can see the corresponding network hash rate since the beginning of its existence, observing a large spike in the hash rate for the year of 2018 where coincidentally the cryptocurrency was at its peak value.

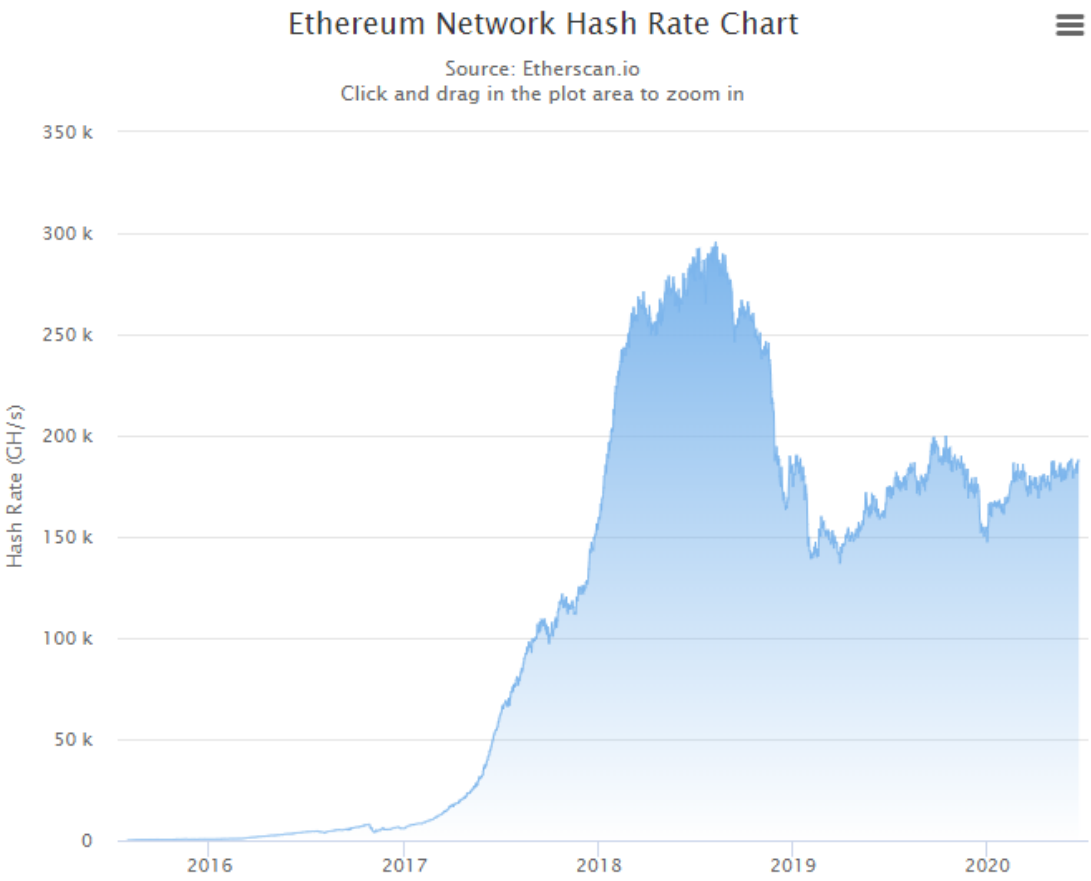

Figure 6 Ethereum Network Hash Rate Chart Source: https://etherscan.io/chart/hashrate

Looking at the numbers, during 2018 they hash rate available in the network peaked at 295,000 Giga Hashes per second (GH/s), where one GH translates to 1,000,000,000 hashes per second. This means that each second an incredible 295,000,000,000,000 (295 TH/s) hashes are being calculated in order to create and mine the blocks that later become part of the blockchain.

An average modern-day graphics card can calculate around 52 MH/s [3], taking the peak network hash rate of 295 GH/s this incredible computational effort requires roughly 5.67 million graphic card units running around the clock for the sole purpose of ensuring the integrity of the blockchain. This is also the main point of decentralization as currently there is no individual hardware that could produce a hash rate high enough to endanger a blockchain network, this leads to the 51% attack principle.

Calculation:

295,000 GH/s -> 295,000,000 MH/s

$$\frac{295{,}000{,}000}{52} = 5{,}673{,}076 = \sim 5.67\ million$$

What is a 51% attack?

The possibility of an individual or a pool of miners to control over 50% of the hash rate in a blockchain network, this would allow the attacker(s) to take over partial control of the network as they would be the ones responsible for mining and securing newly created blocks. In previous section the structure of each block was discussed, it is known that transactions are confirmed as part of mining a block, this means that the attacker(s) could manipulate all new transactions that are completed while the network is under the control of the attacking entity. [4]

How is this possible?

The goal of the blockchain network is to verify the validity of transactions on the network, if most of the parties confirm a block by calculating its hash, the block is considered as valid. In a case where over 50% or the majority of the hashing power is coming from a single source, they are the responsible party for validating a block. For this reason, the hash rate of a network is key to its integrity and fraud prevention.

Complex Structures

Block Structure Revisited

As shown previously in the block overview section, each block stores the transactions which have been carried out on the network and much other information, when taking this into consideration calculating the hash of all this data is very impractical.

For this exact reason, Merkle trees are at the core of what makes a blockchain operate. While it would be possible to perform a complete calculation of all this data without taking advantage of Merkle trees by including all transactions in massive block headers, in practice this computational requirement could only be fulfilled by a scarce number of powerful super-computers around the globe hence causing a vast scalability challenge.

By using Merkle trees, it is possible to build Ethereum nodes that run on all computers and laptops large and small, smart phones, and even internet of things (IoT) devices [5].

In Figure.7 below we can see a representation of a Merkle Tree and how it is structured, how exactly does it bring value to a blockchain network and how does it solve scalability issues?

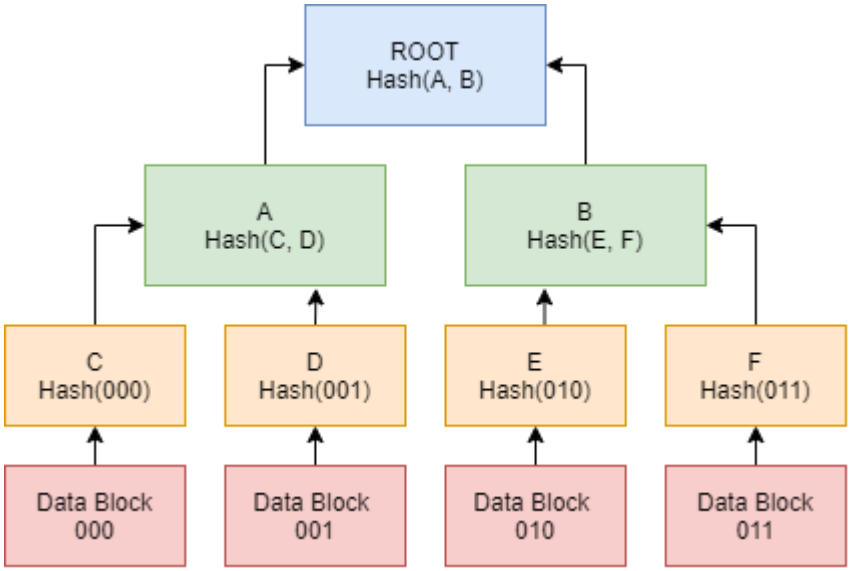

Figure 7 Example of a Merkle Tree

A Merkle tree or commonly known as a hash tree, in its basic form is a hash-based data structure which represents a way of hashing a large number of data blocks together for efficient data verification. This operation relies on splitting the data blocks into nodes, where each leaf node contains a hash of the data block, this process is then repeated for each parent node and continued until the total number of hashes remaining becomes only one, the root hash of all the hashes otherwise known as the Merkle root.

This means that the Merkle root contains all the hashes of all the transactions that have been completed as part of a block, the validity of the transactions is hence guaranteed by the root hash.

Example of a Merkle tree application:

If there are 100 transactions in a block of value 1, a unique root hash will be generated. A malicious operation then attempts to alter one of these transactions by changing the value to 0, this operation will affect the root hash which is based on the hash of all the hashes of the transactions.

All the other nodes on the blockchain will then be able to validate this malicious operation against their own root hash and invalidate the attempt of manipulating the transaction.

Merkle Proofs

Understanding the concept and structure of Merle tree leads to solving of the scalability and mass adoption issue and the introduction of what is known as a Merkle proofs. A Merkle proof consists of a data block, the Merkle root hash of the tree, and the roots child node which consists of all the hashes going up along the tree from the data block to the root.

The party attempting to obtain the proof can verify that the hashing, at least for that branch, is consistent from the data block all the way up throughout tree, hence they can validate that the given chunk does indeed reside at the given position in the tree. This allows the blockchain to be virtually unbound by any size restrictions as it allows for verification of even the smallest data sets within the tree without exposing any other data. [5]

Referring to the Example of a Merkle tree application, the blockchain offers the possibility to directly authenticate a value or a data set in the tree by passing its unique key. Here the Merkle root is trusted by all the involved parties as it has been validated by a vast amount of computational power using the hashing mechanism. An unknown party at some points wishes to authenticate one of the contracts within the tree in order to validate its value is “1” by passing the unique contract ID “50” requesting a Merkle proof, if the response to the Merkle proof request is successful, the value “1” is said to exist within the location of contract ID “50” as seen in Figure.8.

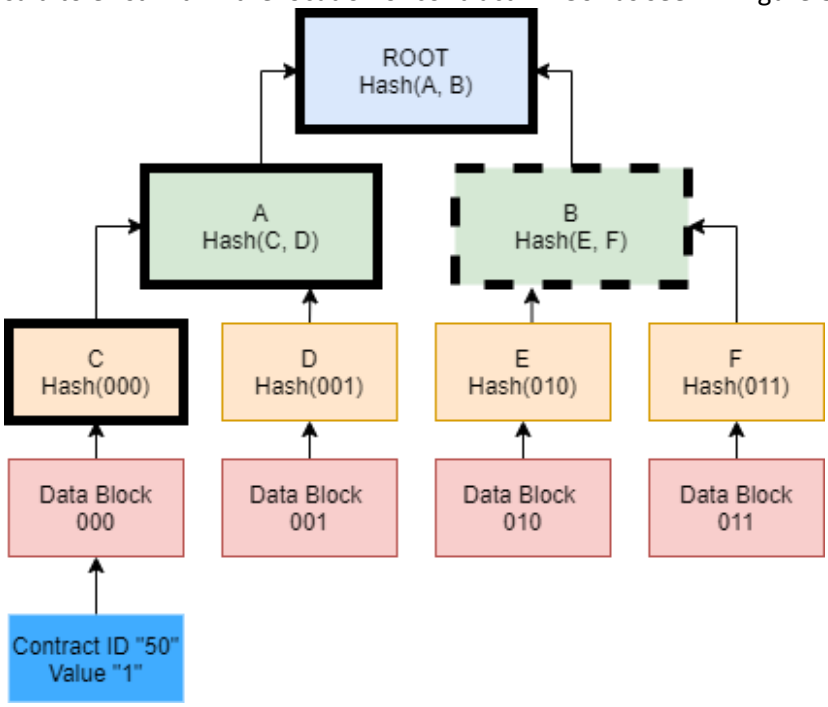

Figure 8 Example of a Merkle Proof

This simple example demonstrates how Merkle proofs can be used to solve scaling issues in blockchain applications by making extensive use hashing, proving its importance.

Conclusions

This paper highlights the functionalities and the key role, hash functions play in the world of cryptocurrencies and subsequently blockchain networks, from the very first block of origin to the very latest block mined just seconds ago.

Currently the adoption of SHA-3 and hash functions belonging to the Keccak family is growing within the cryptocurrency space with SHA-2 still very much holding most of the market share, as blockchains are ever growing structures being future-proof is a key factor and the advantage of using SHA3 over SHA2 is prominent. However, as it stands SHA-2 is resilient to any attacks and that will not change in the nearest future to its advantage SHA-2 has been around since 2001 which means it has been thoroughly tested by the community.

On the other hand, the main disadvantage of SHA-3, is being a relatively new hashing function only released in late 2015 thus leaving potential room for unexpected attacks that could be exploited against the algorithm in the future, alongside security the performance of the hashing function has to be considered to ensure a satisfactory hash rate is produced by the miners of the cryptocurrency.

This brings us to the Difficulty and Hash Rate which are the fundamental pieces of the puzzle that work with each other in perfect harmony to establish the blocks and in result the block chain. These two parameters are the main influencing factor behind the rate at which the block chain network grows, this poses certain advantages and disadvantages for the involved parties.

Advantages:

- The value of the cryptocurrency is stable as the block's mining difficulty is always adjusted to match the networks total hash rate.
- Prevention of the blockchains exponential growth, which could result in extremely high block numbers after a short period of time.

Disadvantages:

- This also means that over-spending on latest hardware is not necessarily worth it as the mining difficulty can be quickly adjusted rendering the hashing rate of the hardware not enough for significant income.
- As the block becomes more difficult to mine, the profitability of the operation diminishes, this can lead to overall network hash rate reduction.

Summarizing the above key points, the use and importance of hashing within the cryptocurrency space is invaluable and the block chain could not exist without it. Beginning with the use of hashing within the Merkle Trees to facilitate faster transaction authentication to using the hash function on the block headers themselves to maintain the integrity of the blockchain as it expands over the years.

In the coming years we can expect new improvements to hashing functions and their underlying structures such as the sponge function structure which is being used for the SHA-3 function Family.

Sources:

Blockchain.com

[1] https://keccak.team/keccak_specs_summary.html Sponge Structure and Pseudo Code

https://arxiv.org/pdf/1902.05320.pdf - GPU Accelerated Keccak (SHA3) Algorithm

[6] https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf

http://bitcoin.sipa.be/ - Mining difficulty

https://iacr.org/archive/crypto2005/36210424/36210424.pdf

[2]https://eth.wiki/en/faqs/glossary

[3] https://minerstat.com/hardware/amd-rx-5700-xt

[4] Zibin Zheng, Hong-Ning Dai, Mingdong Tang, Xiangping Chen : Blockchain and Trustworthy Systems,

[5] Vitalik Buterin : https://blog.ethereum.org/2015/11/15/merkling-in-ethereum