EasyChair Preprint

№ 7198

Distributed Ledger Technology (DLT) and Byzantine Fault Tolerance in Blockchain

Sanatan Shrivastava and Ashish Sharma

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

December 9, 2021

Distributed Ledger Technology (DLT) and Byzantine Fault Tolerance in Blockchain

Sanatan Shrivastava[1] and Ashish Sharma[2]

[1] Indian Institute of Information Technology, Kota, Rajasthan, India
[2] Indian Institute of Information Technology, Kota, Rajasthan, India

2018kucp1096@iiitkota.ac.in
ashish@iiitkota.ac.in

Abstract. With the advent of technology, the world saw the rise of Blockchain technology, because of its accessibility, and efficiency in managing transactions and the related records. According to IBM, because it delivers immediate, shareable, and entirely transparent information kept on an immutable ledger that can only be viewed by permissioned network users, blockchain is excellent for delivering that information. The most important aspect of blockchain is its distributed ledger technology. The phrase "distributed ledger" refers to the fact that numerous Blockchain members share the ledger on which transactions are recorded since it is not controlled or owned by a single entity. The Byzantine Fault Tolerance which is largely associated with distributed systems is a feature that allows a decentralised, trustless network to function even when some nodes are broken or malevolent. This paper elucidates upon the Byzantine Faults in Blockchain Technology, its effects, and the solutions to this problem.

Keywords: Blockchain, Byzantine Fault Tolerance (BFT), Distributed Ledgers.

1 Blockchain Technology

Blockchain technology is attracting a lot of attention from the research community and academicians from many universities around the world in this technological age. Block chain technology is a distributed ledger technology (DLT), or an open source distributed database, with immutability, transparency, and integrity. Satoshi Nakamato proposed the block chain technology concept in 2008 to eliminate third-party involvement in financial transactions [1].

It is a method of data decentralisation in which data is not stored on a single server system as shown in figure 1. It is dispersed throughout the network. It encourages anonymity since participants in the chain can download and validate individual ledgers, it is transparent and thus serves as an excellent record-keeping platform. It's also irreversible because those ledgers can't be changed [2]. Blockchain and related Distributed Ledger Technologies (DLT) are proving to be game-changing, with the potential to transform the web

from a centralised document sharing platform to a generic decentralised platform capable of exchanging digital currency and assisting in the autonomous management of financial and real-estate assets. The Web's original concept was to be a decentralised network with unrestricted access. However, it expanded slowly around centralised servers, requiring privileged access to assure security while attempting to maintain openness[3].

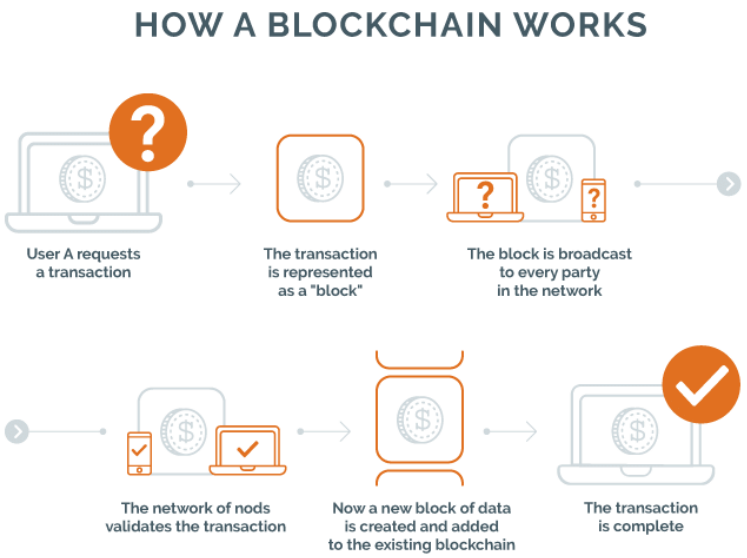

Fig 1. Working of Blockchain Technology[4]

If the web can offer trustable, safe, and responsible updates among autonomous players without a central server, the idea of a decentralised Web can be re-instantiated.

2 Related Works

Zyzzyva [6] is a Byzantine Fault Tolerance protocol that is currently under development. It does not require replicas to establish agreement before processing queries, unlike other BFT algorithms such as [7, 8, 9]. The Zyzzyva protocol operates and responds to clients with speculative outcomes almost instantly. Clients will conclude the requests and accept the findings if all clones return the same results. Otherwise, the proper duplicates may become briefly inconsistent and give different responses. Nonetheless, every proper duplicate will attain final agreement with the help of clients, and the responses will be guaranteed to be committed finally. Castro and Liskov's Practical Byzantine Fault Tolerance (PBFT) [7,8] assures both safety and liveness as long as less than one-third of the copies fail. Since the publication

of BFT's foundational work [13, 14], several alternative BFT algorithms [1, 20, 32] have been proposed. Query/Update (Q/U) [10] is a BFT protocol that requires the usage of replicas to tolerate up to faults, which is more than PBFT [7,8] requires. Clients broadcast their cached histories and requests to the server replicas, and all replicas execute the requests optimistically without inter-replica communication. In a fault-free condition, the system's performance can be considerably enhanced using Q/U since requests can be fulfilled in a single round of communication between the client and server replicas. If the client receives inconsistent results, on the other hand, it will notify the replicas and force them to return to a consistent condition. After that, the request will be re-executed. During typical operations, Q/U may handle requests with fewer message exchanges thanks to additional service replicas. It does not, however, operate effectively when many update requests are being processed at the same time.

3 Distributed Ledgers in Blockchain Technology

According to International Business Machines (IBM), A distributed ledger is a sort of database that is shared, replicated, and synchronised among decentralised network participants. It keeps track of transactions between network participants, such as the exchange of assets or data. The network's participants govern and agree on the revisions to the ledger's records by consensus and there is no involvement from a central authority or a third-party mediator, such as a financial institution or clearinghouse. The distributed ledger has a timestamp and a unique cryptographic signature for each record, making it an auditable, immutable record of all network transactions. Distributed ledgers are one of the fundamental technologies responsible for restoring the web's openness while maintaining its security because DLTs enable a more secure and responsible environment, economic and legal transactions can now be completed entirely online[11]. Because the ledgers are not stored or recorded on a centralized server and are dispersed throughout the network, these are known as ‘Distributed’ ledgers. The new DLT-based systems, such as Bitcoin and Ethereum, are meant to work without the need for a trusted authority. Bitcoin uses a consensus-based validation mechanism and cryptographic signatures to maintain a distributed database in a decentralised manner. The difference between a conventional payment system compared to a distributed ledger based payment system is shown in Figure 2.

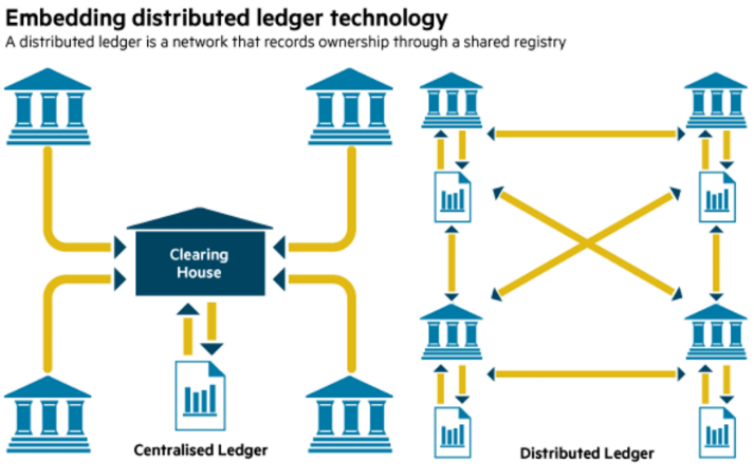

Fig 2. Centralized ledger compared to Distributed Ledger[12]

In such systems, transactions are carried out peer-to-peer and broadcast to the full set of participants, who work in batches known as "blocks" to validate them. This sort of DLT is sometimes referred to as "blockchain technology" since the ledger of activity is organised into separate but connected blocks[13]. Furthermore, since it is distributed, the DLT based-systems have to inherently deal with the design and implementation issues associated with large scale distributed systems such as latency issues and failure rollbacks on nodes or even entire systems. One major issue with these systems is the Byzantine Fault. Although malicious assaults and software faults are becoming more widespread, which can cause malfunctioning nodes to behave erratically, byzantine faults are still responsible for major failures. Previous algorithms assumed a synchronous system or were too slow to be practical, but the algorithms utilised in Byzantine Fault Tolerance mechanisms constitute a breakthrough in dealing with important difficulties

4 Byzantine Fault in Blockchain Technology

Among the failure modes, Byzantine failures are the most common and trickiest to diagnose. The so-called fail-stop failure mode is the most basic of the failure modes. Whereas fail-stop failure mode simply means that the only way for a node to fail is for it to crash, which is noticed by other nodes, Byzantine failures imply no constraints, which means that the failed node can generate any data it wants, even data that makes it appear to be functional. As a result, Byzantine failures can cause problems with failure detection systems, making fault tolerance problematic. A Byzantine fault is a condition in which components in a computer system, particularly distributed computing systems, may fail and there is imperfect information on whether they have

failed. The term is derived from an allegory known as the "Byzantine Generals Problem" [14] which describes a situation in which the system's actors must agree on a coordinated plan to avert catastrophic failure, yet some of these individuals are unreliable. A fault-tolerant distributed computer system's resilience to component failures is known as byzantine fault tolerance (BFT). The NEO platform which is a blockchain-based platform with its own coin and the ability to create digital assets and smart contracts uses this as a consensus process. Figure 3 shows an example of a Byzantine Fault. A component, such as a server, can appear both broken and functional to failure-detection systems in a Byzantine fault, exhibiting various symptoms to different observers. It's difficult for the other components to announce it failed and remove it from the network since they have to first agree on which component failed in the first place. Any defect that presents diverse symptoms to different observers is referred to as a Byzantine fault. In systems that need consensus, a Byzantine failure is the loss of a system service owing to a Byzantine fault [10]. The goal of Byzantine fault tolerance is to be able to defend against system component failures with or without symptoms that prevent other system components from reaching an agreement among themselves, which is required for the system's correct operation.

In blockchain, the Byzantine faults provide a challenge in maintaining transactions. Malicious users have significant economic incentives to try to cause errors because of the value stored in these ledgers. Byzantine Fault Tolerance, and hence a solution to the Byzantine Generals' Problem for blockchains, is, however, desperately needed. Without BFT, a peer can send and post fake transactions, essentially nullifying the blockchain's trustworthiness.

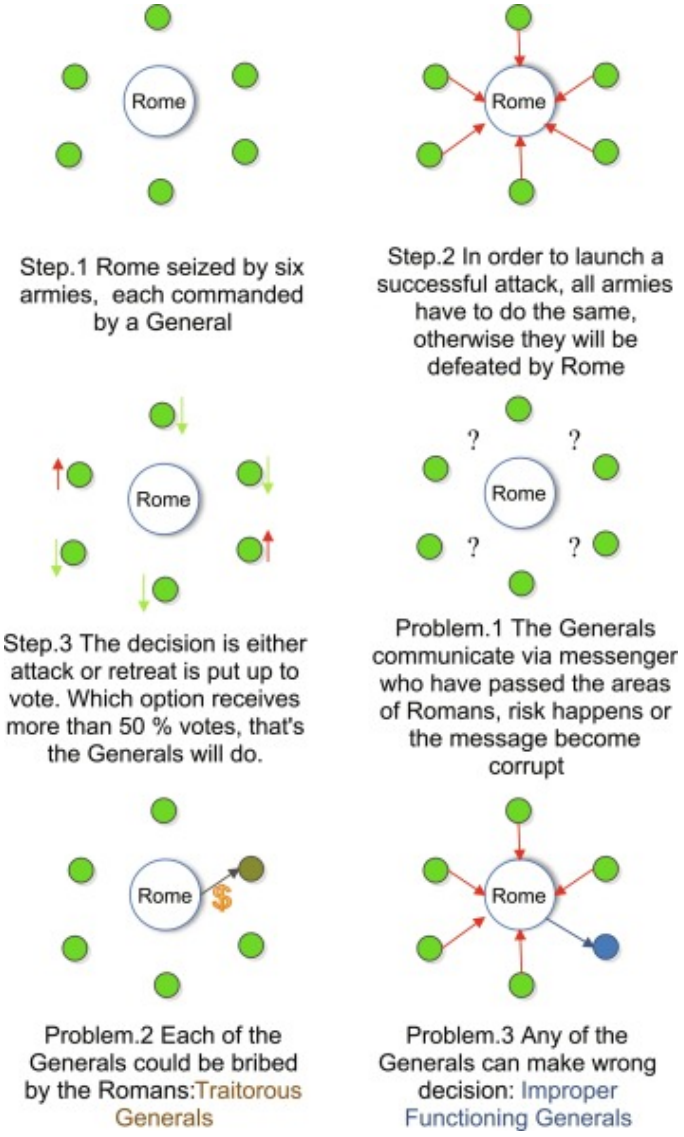

Fig 3. Problems associated with Byzantine Fault Tolerance[15]

To make matters worse, there is no central authority to take charge of the situation and restore the damage. A conventional blockchain transaction is shown in figure 4.

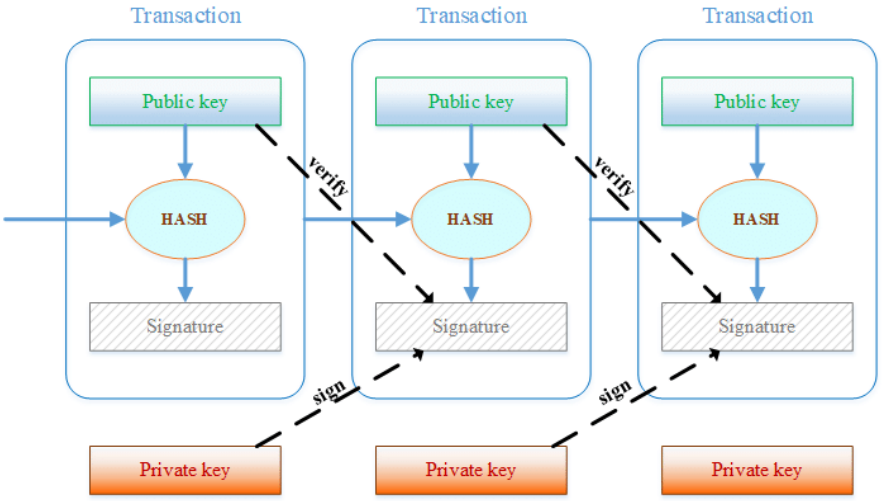

Fig 4. Diagram of a Blockchain Transaction[16]

It's hard to reliably identify which nodes are producing inaccurate transaction information, whether purposefully or by accident, because nodes are geographically dispersed and independent of each other and any central authority[9]. Byzantine Fault Tolerance is a quality of a distributed computer system that allows it to overcome this challenge and build stable consensus despite the fact that some nodes, either mistakenly or intentionally, disagree with the others[17].

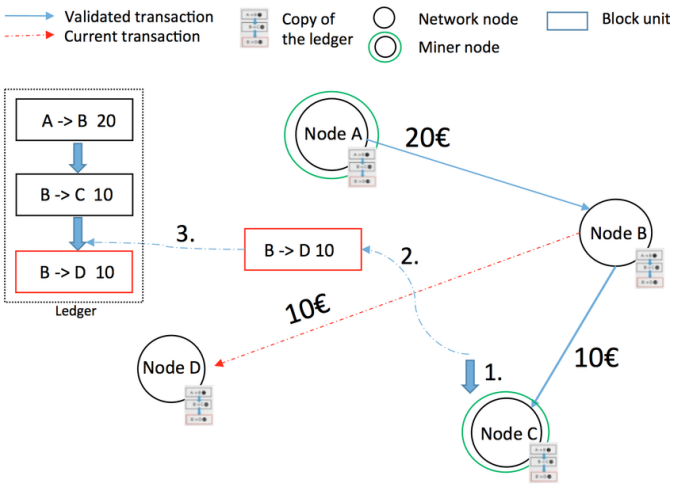

Fig 5. Transaction management in Blockchain[18]

Cryptocurrencies are decentralised, unlike currency money, which is normally governed by a central authority such as a bank. Many cryptocurrencies, like Bitcoin and Ethereum, can now operate without the assistance of a government or corporation thanks to PoW. This consensus process is critical for avoiding double spending, or when a coin or token is used to enable a transaction more than once. It's how new blocks are added to the blockchain network and transactions are verified. When a bitcoin miner successfully completes the Proof of Work behind a block, the network authorises it. The process of transaction management is displayed in figure 5.

5 Consensus in Blockchain Technology

Starting with the same beginning value state, the nodes in any dispersed network should agree on a specific output value in a transition state. This is referred to as reaching an agreement.
And this should be accomplished in a way that allows nodes to deviate and act either maliciously (crashing or going offline) or non-maliciously (crashing or going offline) (Byzantine).
If the following conditions are met, the procedure is said to be successful.

- The nodes choose an output value.
- This deterministic method confirms termination, or that the majority of nodes agree on the same output value.
- This indicates that both parties are in agreement.

There are many types of Consensus algorithms:

- Proof-of-Work
- Proof-of-Stake
- Delegated Proof-of-Stake
- Leased Proof-of-Stake
- Proof of Elapsed Time
- Practical Byzantine Fault Tolerance

In this paper, we will discuss Proof-of-Work and Byzantine Fault Tolerance and how they can be leveraged into blockchain technology to overcome transaction management's problems.

6 Solutions to Byzantine Fault in Blockchain

With the passage of time, initiatives to solve this distributed network/blockchain-centric problem have been made. We currently have a number of distributed network systems that provide a partial, if not complete, solution to this problem. Various consensus methods are designed in blockchain, which automatically address the Byzantine problem.

There are many algorithms that can provide reliable and robust solutions to the Byzantine Fault problem in Blockchain. One such approach is Proof-of-Work (PoW) Algorithm.
A group of special nodes known as miners listen to every transaction that occurs within a block- chain network. They compete to solve a computationally intensive puzzle as quickly as possible by allocating computing resources to construct a valid block of transactions. A block in a blockchain network typically contains a list of all transactions, as well as a nonce, merkle root hash, previous block hash, and a block header (as seen in Figure 6).

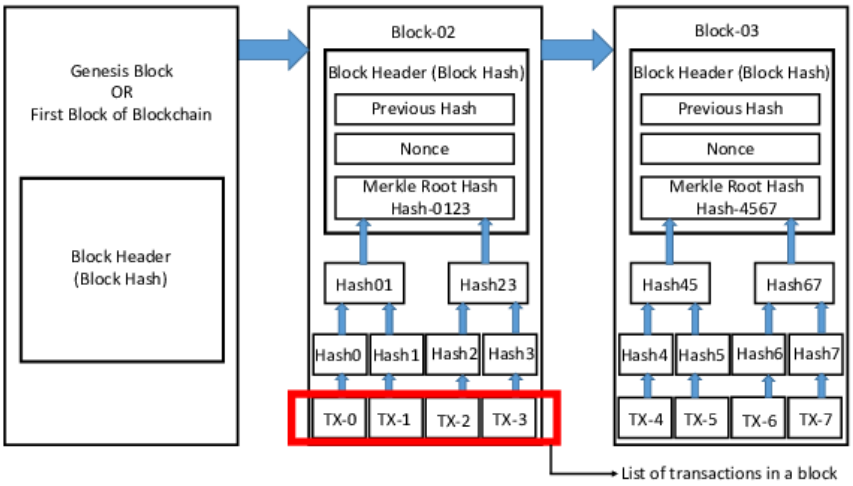

Fig 6. A block in the blockchain[19].

It takes a miner longer to complete a task as the difficulty level rises. When a miner node solves the problem, that is, when they are able to generate a valid block of transactions, they broadcast the answer to all peer nodes.

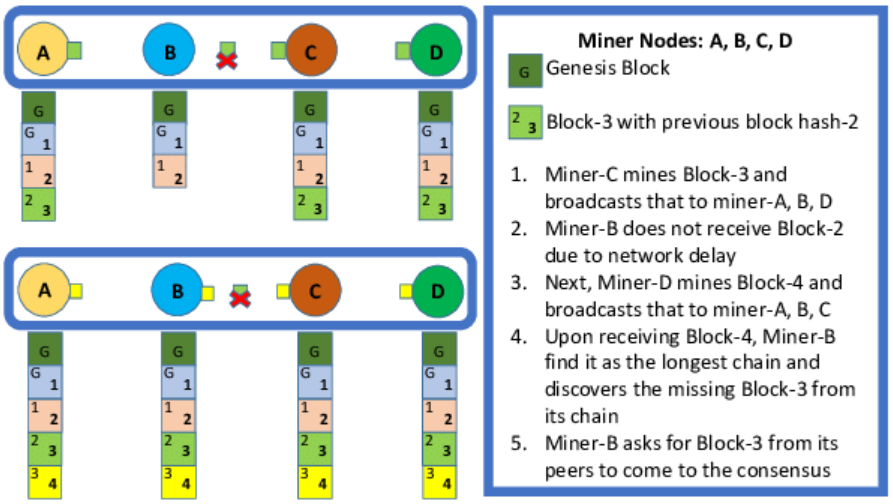

Fig 7. Consensus in Proof-of-Work algorithm[19].

It is simple to verify such a solution. If the hash of a block is supplied as the preceding block hash during the construction of a new block, it is accepted. As a result, all nodes finally establish an agreement (as shown in Figure 7), which is known as proof-of-work [12].

7 Byzantine Fault Tolerance in Blockchain Technology

The goal of Byzantine fault tolerance is to be able to defend against system component failures with or without symptoms that prevent other system components from reaching an agreement among themselves, which is required for the system's correct operation.

Assuming there are enough operationally correct components to provide the service, the remaining operationally correct components of a Byzantine fault tolerant system will be able to continue providing the system's service as designed.
The practical Byzantine Fault Tolerance model is heavily reliant on specific assumptions. For a given window of vulnerability, the most common assumption in practical BFT is that the number of malicious nodes in the network cannot be equal to or more than one-third of the total nodes in the system. Byzantine Fault Tolerance is a quality that characterises a system that accepts the Byzantine Generals' Problem's class of failures. The most challenging type of failure mode is Byzantine Failure. It contains no constraints and makes no assumptions about the type of behaviour that a node can exhibit (e.g. a node can generate any kind of arbitrary data while posing as an honest actor). The most severe and difficult to deal with are

Byzantine Faults. Byzantine Fault Tolerance is required in aeroplane engine systems, nuclear power plants, and almost any system whose actions are dependent on the findings of a large number of sensors.

It's now crucial to figure out how the Byzantine Generals Problem can be applied to blockchain. In the case of a peer-to-peer network, attaining consensus may be aided by unanimous agreement among loyal and non-faulty nodes. In a case when all nodes duplicate an incoming message, the foundation of Byzantine Fault Tolerance is clear. If a node repeats the incoming message, it plainly implies that it has no problems or errors. If the recipients, on the other hand, repeat the incoming message, the network may quickly rule out Byzantine nodes. The term "byzantine node" refers to a traitorous node that purposefully lies or misleads other nodes in the network. The byzantine node may also deceive or lie to the other nodes in the consensus protocol as shown in figure 8.

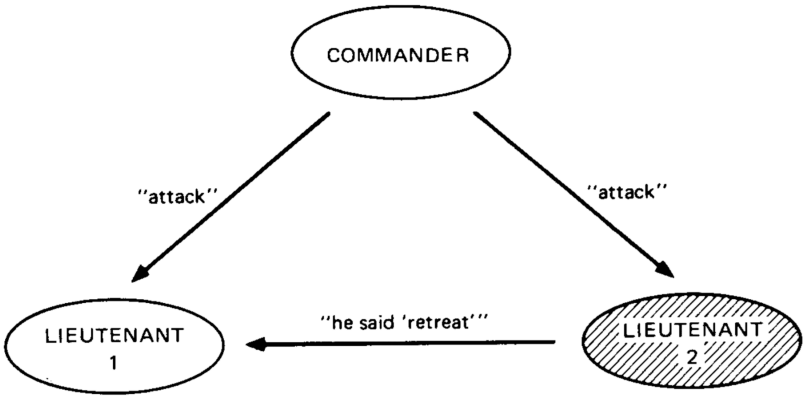

Fig 8. Lieutenant 2 is a byzantine node misleading the network[20]

Byzantine Fault Tolerance in blockchain would help it overcome failures in a perfectly functioning blockchain network. Byzantine nodes, often known as malevolent nodes, can lead to Byzantine failures. Users may encounter two sorts of Byzantine failures, the first of which is entirely technical in origin. A little technical problem in the node could compromise its functioning, and it could stop responding or working altogether in some situations. The arbitrary node failure is the second sort of Byzantine failure[21].

A node may exhibit the following characteristics in the event of an arbitrary node failure:

- Failure to provide a response
- Providing responses to outcomes with errors.
- Providing purposely deceptive findings in response to enquiries.

- Providing different results to various components of the system in response to a single query.

The following are the crucial steps to comprehending how Byzantine Fault Tolerance functions:

- Clients send requests to the leader node for specified service operations to be performed.
- The request could then be broadcast to backup nodes in the network by the leading node.
- Nodes could also guarantee that allocated requests are completed and that a relevant response is sent to the client.

As shown in figure 9, if 'm' represents the maximum number of nodes having the potential for errors in this situation. The client would then wait for (m+1) similar responses from multiple nodes. The outcome is essentially a reflection of the operation's outcome.

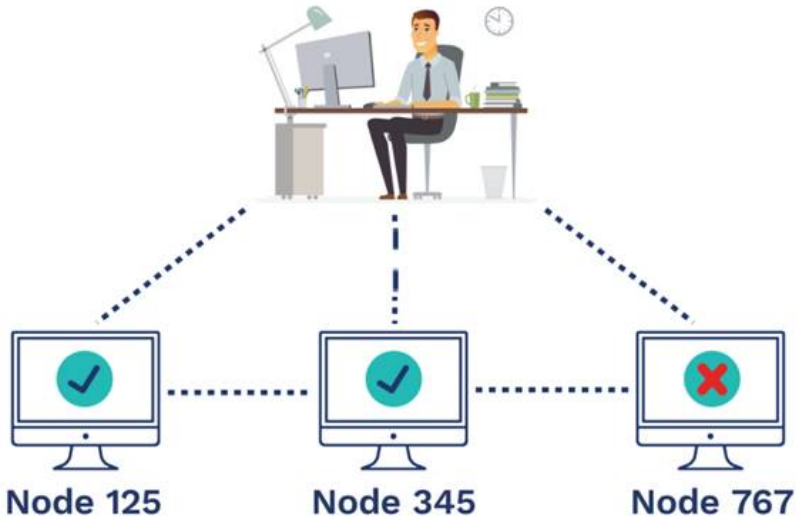

Fig 9. Byzantine Fault in nodes [22]

It's also crucial to make sure that nodes in Blockchain meet the key requirements for Byzantine Fault Tolerance (BFT). For practical BFT, the nodes must be deterministic and start in the same state. The end result would indicate that all honest nodes could agree on the record's order. In the end, the nodes might either accept or reject the record. Practical BFT, interestingly, uses a round-robin type format for changing the leader node in each view[23].

Additionally, a technique known as view change can be used to replace the leader node. When the leader node has not broadcast the request for a certain

length of time, this protocol is appropriate. Furthermore, Byzantine Fault Tolerance, whether synchronous or asynchronous, assures that a significant majority of honest nodes can agree on a leader's flawed nature. The honest nodes could also replace the faulty leader with the next available leader node in the chain.

Byzantine Fault Tolerance clearly plays a vital role in altering consensus techniques. Blockchain applications are progressively gaining traction in a variety of industries. However, with today's blockchain networks, there are a slew of issues. As a result, it's critical to consider Byzantine Fault Tolerance (BFT) as a critical tool for ensuring that the network continues to function correctly in the face of hostile actors. Because blockchain is open and transparent, it may attract a large number of unwelcome users seeking to exploit their personal interests.

8 Conclusion and Future Works

The increasing reliance on internet services places a significant demand on the computer systems that deliver those services. Byzantine fault tolerance (BFT) is a promising solution for solidifying such systems and providing the high dependability that is much needed. BFT uses redundant copies of the servers to ensure that a replicated system continues to provide accurate services even if only a tiny section of the system is attacked. With other methods like Proactive Recovery and integration of Software Transactional Memory (STM) into Byzantine Fault Tolerance (BFT), usage of a more robust and reliable transaction system with failure resilience can be expected with growing popularity and inclusion of Blockchain Technology into everyday life.

References

1. Afendi Abdi , Dr. Gavendra Singh, 2020, Opportunities and Challenges of implementation of Peer to Peer Blockchain Technology in the Higher Educational Institutions, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 09, Issue 05 (May 2020)
2. Tamimi Homepage, https://www.tamimi.com/
3. Kadam, Suvarna. (2018). Review of Distributed Ledgers: The technological Advances behind cryptocurrency.
4. Hassan, Nurul & Jain, Nishchay & Chandna, Vinay. (2018). BLOCKCHAIN, CRYPTOCURRENCY AND BITCOIN.
5. Driscoll, K.; Hall, B.; Paulitsch, M.; Zumsteg, P.; Sivencrona, H. (2004). "The Real Byzantine Generals". The 23rd Digital Avionics Systems Conference (IEEE Cat. No.04CH37576). pp. 6.D.4–61–11.

doi:10.1109/DASC.2004.1390734. ISBN 978-0-7803-8539-9. S2CID 15549497.

6. Kotla, R., Alvisi, L., Dahlin, M., Clement, A., and Wong, E.: Zyzzyva: speculative Byzantine fault tolerance. In Proceedings of ACM Symposium on Operating System Principles (SOSP), pp. 45-58, New York, NY, October 2007.
7. Castro, M., and Liskov. B.: Practical Byzantine fault tolerance and proactive recovery. ACM Transactions on Computer Systems, 20(4): pp. 398–461, November 2002.
8. Castro, M., and Liskov. B.: Practical Byzantine fault tolerance. In Proceedings of the Third Symposium on Operating Systems Design and Implementation, pp. 173-186, New Orleans, LA, February 1999.
9. Castro, M., Rodrigues, R., and Liskov. B.: BASE: Using abstraction to improve fault tolerance. ACM Transactions on Computer Systems, 21(3): pp. 236-269, August 2003.
10. Abd-El-Malek, M., Ganger, G. R., Goodson, G. R., Reiter, M., and Wylie, J. J, Fault-scalable Byzantine fault-tolerant services. In Proceedings of ACM Symposium on Operating System Principles (SOSP), 39(5): pp. 59-74, Brighton, UK, October 2005.
11. Driscoll, K.; Hall, B.; Paulitsch, M.; Zumsteg, P.; Sivencrona, H. (2004). "The Real Byzantine Generals". The 23rd Digital Avionics Systems Conference (IEEE Cat. No.04CH37576). pp. 6.D.4–61–11. doi:10.1109/DASC.2004.1390734. ISBN 978-0-7803-8539-9. S2CID 15549497.
12. Teles, Duarte. (2018). Data Protection with Ethereum Blockchain. 10.13140/RG.2.2.19486.48961.
13. Morten Linnemann Bech and Rodney Garratt, BIS Quarterly Review | September 2017 | 17 September 2017
14. Lamport, L.; Shostak, R.; Pease, M. (1982). "The Byzantine Generals Problem" (PDF). ACM Transactions on Programming Languages and Systems. 4 (3): 382–401. CiteSeerX 10.1.1.64.2312. doi:10.1145/357172.357176. Archived (PDF) from the original on 13 June 2018.
15. Security for Mobile Ad Hoc Networks; Raja Datta, Ningrinla Marchang, in Handbook on Securing Cyber-Physical Critical Infrastructure, 2012; https://www.sciencedirect.com/topics/computer-science/byzantine-fault
16. Vujičić, Dejan & Jagodic, Dijana & Ranđić, Siniša. (2018). Blockchain technology, bitcoin, and Ethereum: A brief overview. 1-6. 10.1109/INFOTEH.2018.8345547.
17. CoinmarketCap Homepage, https://coinmarketcap.com/alexandria/glossary/byzantine-fault-tolerance-bft

18. Bozic, Nikola & Pujolle, Guy & Secci, Stefano. (2016). A tutorial on blockchain and applications to secure network control-planes. 1-8. 10.1109/SCNS.2016.7870552.
19. Sai, Kuheli & Tipper, David. (2019). Disincentivizing Double Spend Attacks Across Interoperable Blockchains. 10.1109/TPS-ISA48467.2019.00014.
20. CDEMI Homepage, https://blog.cdemi.io/byzantine-fault-tolerance/
21. Driscoll, Kevin; Hall, Brendan; Sivencrona, Håkan; Zumsteg, Phil (2003). "Byzantine Fault Tolerance, from Theory to Reality". Computer Safety, Reliability, and Security. Lecture Notes in Computer Science. 2788. pp. 235–248. doi:10.1007/978-3-540-39878-3_19. ISBN 978-3-540-20126-7. ISSN 0302-9743. S2CID 12690337.
22. Vedveethi Homepage, https://vedveethi.co.in/eNote/BlkChain/BlkChain_Unit-3/ForwardHTML/Byzantine%20fault%20tolerant%20system.htm
23. 101blockchain Homepage, https://101blockchains.com/byzantine-fault-tolerance/