Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 4 (2024)

Deep Analysis of Blockchain and Treechain

Gaddalpati Sai Krishna,Vavilapalli Lokesh,Gada VarunBalaji,Tanushree L ,Bindu Garikapati
Koneru Lakshmaiah Education Foundation

Abstract:- Blockchain and Tree Chain represent two distinct approaches to decentralized ledger systems, each with its own set of advantages and limitations. Blockchain, characterized by its linear structure akin to an array, utilizes hash algorithms like [2] SHA-256 for securing data integrity. On the other hand, Tree Chain, resembling a tree data structure, introduces faster transaction speeds through Consensus Code Allocation and load balancing algorithms. While Tree Chain boasts superior time complexity due to its tree-based structure, it may not be suitable for highly distributed or confidential networks. Hence, while Blockchain remains a robust choice for broader applications, Tree Chain presents a promising solution for shorter networks seeking increased throughput and rapid data transfer.

Keywords*:* BlockChain, Tree Chain, Hash algorithms,hierarchical, branching, Consensus codes, parent-child relationships, proof of work, proof of stake, multi-chain consensus.

1. Introduction

Blockchain the superior leading technology of security, blockchain enables us to make transactions in a very secure manner without help of other online wallets, banks and third partyapplications. It provides decentralized based system which enables every user in the network has equal rights in transactionaldata. It was originally designed as the underlying technology for the cryptocurrency Bitcoin, but its applications have expanded beyond cryptocurrencies to various industries which developed in a very enormous way such that it become very trustworthy technology which won a million hearts in part of secure technology.

Blockchain in terms of its fundamental building blocks: In Blockchain is commonly described as combination of blocks, each block consists of three main parts first part consists of data , then the next part consists of present Hash which is created by different Hash algorithms some of the popular hash algorithms used by the ultimate bitcoin are SHA-256(Secure Hash Algorithm) which creates a fixed length hash value of 256 bits similarly another well-known hash algorithm used by Ethereum and other ETH-based cryptos.Final Part of block consists of previous hash which was generated by hash function of previous block.

These three combinedly known as a ledger, this ledger is shared among every user in the network and every ledger is updated whenever other block is created i.e., whenever new data gets added. In this process every single user consists of data which makes it much more secure. Ledger which is present at every user in the network is encrypted with their public keys so, that they can only be decrypted with user appropriate private key. If any intruder tries to infiltrate or tamper with data, the hash in that data block gets changed which leads to disturbance in whole block chain and it gets notified, and the intruder gets caught or we can take appropriate measures according to situation.

Centralized System: A centralized system is simply like a client-server architecture where a single entity or a central authority has control and authority over the entire system. In a centralized system, decision making, data storage, and resource allocation typically occur at a central point. For example, if we consider a client server architecture server has the main authority and client are in form of peer-to-peer network connected to server, here any user can be added or deleted without any knowledge of other existing user data in network if the main server gets attacked or disturbed it causes high damage to every system connected to that server. It can be affected by Dos or Distributed Dos, so it’s less secure than decentralized system.

Tree chain: Tree Chain is a revolutionary blockchain system that functions similarly to blockchain, but it offers faster transaction speed. If we consider array data structure and tree data structure block chain is like array and tree chain is somewhat like tree data structure so in Treechain we can perform operation with less time complexity (log n) compared to block chain. Here Validators not only calculate KWM for their own key but also for the public keys of potential validators. The results are ordered based on the calculated KWM in descending order. This introduces randomness and 1 unpredictability, making it hard for a single entity to control the network easily. Consensus codes are vital in Tree Chain, defining which validators are responsible for verifying specific transactions. This contributes to the system's hierarchical structure. Although all validators store the entire chain, Tree Chain uses a load balancing algorithm to randomize the selection of validators for committing transactions and blocks to the ledger. This consensus algorithm doesn't demand extra computational power, reducing the delay in storing new blocks to near real-time.

2. Objective

In our study, we examine the evolution of blockchain technology from its origin with Bitcoin in 2008 to more advanced architectures designed to address scalability and efficiency challenges. The foundational concept of blockchain as a distributed ledger technology has proven valuable for its decentralization, security, transparency, and immutability. However, as blockchain applications have expanded beyond cryptocurrencies, limitations in traditional linear blockchain structures have become apparent, particularly in handling high transaction volumes and resource-constrained environments like the Internet of Things (IoT).

These challenges have driven innovation in blockchain architectures, leading to the development of alternative structures such as tree blockchains. Our research focuses on comparing linear blockchain, characterized by its sequential chain of blocks, with tree blockchain, which introduces a hierarchical structure allowing for multiple branches. This comparative analysis aims to shed light on the strengths and limitations of each approach, providing insights into their suitability for various applications and the potential for addressing current blockchain scalability issues.
This research paper aims to conduct a thorough Deep analysis of linear and tree blockchains. By examining their design principles, functionalities, and potential use cases, we can shed light on the key differences between these two architectures. This analysis will provide valuable insights for developers and businesses navigating the labyrinth of blockchain architectures, empowering them to make informed choices about the most suitable architecture for their specific applications.

3. Literature Survey

A. Blockchain Background:

Blockchain technology was first introduced in the year 2008 by an anonymous person or group using the pseudonym Satoshi Nakamoto. It was introduced as an underlying technology of a sensational cryptocurrency Bitcoin at the beginning, it is now one of the most secured network we use in our daily lives.
Blockchain was mainly bought into picture in order to eliminate the limitation of centralized architecture in centralized architecture the nodes connected to central server user to share data simultaneously one after another, they used to update the data seeing other nodes data getting updated and it used to be managed by central server if server gets any issue then all the nodes connected to server face problem to overcome this problem blockchain came up with decentralized architecture in which data gets updated in every node's local copy connected in a network, in this network every node is connected with other node in a peer to peer manner if one of the node's local copy updates data others get updated too.[4] This enables high security of data and secure transmission.
Block chain is simply a chain of blocks in which a block is added to the chain in a distributed way as a ledger the data is stored in an efficient way as we seen in above introduction, the data added to block is then verified and committed in the decentralized network permanently which makes it more persistent and efficient.[3] Here Ledgers in the blockchain can be called as immutable ledgers due to if an intruder may try to access one of blocks data, then the blockchains hash value gets changed it causes disturbance in blockchain as the next block of blockchain consist of this block previous value the disturbance can be notified.[4] [3]These blocks are mined by certain group of people called miners, each minor tries to generate a hash value for the block which just got new

data to be added, each blockchain uses a target range specific hash values and minors try various nonces to generate such hash value. This process requires high computational work to be done in order to mine data, after generating a hash value in that specific range the block gets verified by other miners then finally gets added to blockchain.

B. Types of Blockchain

[1] There are four existing types of blockchain public, private, hybrid, and consortium. The first type is referenced as Public blockchain,[3] these are decentralized and open to anyone. They are used in scenarios where high transparency is required.[1] The second type is, Private blockchain these are restricted to an only accessible by a specific group or organization. They are used in scenarios where high privacy and security this type is faster than public blockchain as the number of blocks to be mined are limited than public blockchain. Next type is Hybrid blockchain in these are combination of both public and private can be represented as semi-decentralized and involve multiple organizations working together. They are used in scenarios where a balance between transparency and privacy is required. These are more optimal than other types in blockchain. The Last type is Consortium blockchain.[1] These are also closed networks, but only a set of groups are allowed to validate transactions or data in each blockchain network.[1] They provide high privacy and security are required.

C. Treechain

Treechain can be represented as modified version of blockchain in which the nodes can be represented in a tree data structure manner. As in blockchain the block gets appended after performing validation of hash value is done by a single validator for single longest ledger in a linear data structure way as the data in the block gets added to all ledgers connected in the decentralized System it becomes more complex to perform. To over come this type of disadvantage treechain runs parallel blockchains in which each chain has a separate ledger to validate, and the block gets appended faster with this type of parallel computation process.[6]
Treechain is a technology which is introduced to reduce the limitations of blockchain, it can be replicated a lightweight consensus algorithm for low end devices which increases the efficiency of processing compared to block chain. As we know in blockchain while a hash gets generated to public key of a node/System the miners try to generate hash for that block performing a high computational work which increases transaction load of that block and also throughput raises. Here treechain comes into picture, it states that we use a key weight metric to determine upper bound throughput of a block according to different throughputs processed by all miners. The lower bound throughputs are ignored and the miner who succeeds in producing upper bound throughput appends the block to blockchain without further computations.
]This feature makes treechain faster and more efficient than blockchain, the cost of computations to be performed also decreases as we append block to blockchain in a single computation which produces negligible transaction dela

4. Methods

Blockchain Creation and Management Algorithm

Algorithm Define Block Class

Attributes: index: int previous_hash: string timestamp: string

transaction_type: string amount: float hash: string

Algorithm 2: Calculate Hash

calculate_hashindex, previous_hash, timestamp, transaction_type, amount concatenated_value ← str(index) + previous_hash + timestamp + transaction_type + str(amount) hash ← SHA-256(concatenated_value) return hash

Algorithm 3: Create Genesis Block

genesis_create_block index ← 0 previous_hash ← "0" timestamp ← current_timestamp() transaction_type ← "Genesis Block" amount ← 0 hash ← calculate_hash(index, previous_hash, timestamp, transaction_type, amount) genesis_block ← Block(index, previous_hash, timestamp, transaction_type, amount, hash) return genesis_block

Algorithm 4: Create New Block

create_new_blockprevious_block, transaction_type, amount start_time ← current_time() index ← previous_block.index + 1 previous_hash ← previous_block.hash timestamp ← current_timestamp() hash ← calculate_hash(index, previous_hash, timestamp, transaction_type, amount) new_block ← Block(index, previous_hash, timestamp, transaction_type, amount, hash) end_time ← current_time() time_taken ← end_time - start_time return new_block, time_taken 1)

Algorithm 5: Blockchain Initialization and Genesis Block Creation

blockchain ← empty list genesis_block ← genesis_create_block() blockchain.append(genesis_block) previous_block ← genesis_block

Algorithm 6: Data Structures for Plotting

block_indices ← empty list time_taken_list ← empty list

Algorithm 7: Simulate Credit and Debit Transactions

transactions ← [("credit", 100.0), ("debit", 50.0), ("credit", 200.0), ...] for transaction_type, amount in transactions do-è2:new_block, time_taken ← create_new_block(previous_block, transaction_type, amount) blockchain.append(new_block) previous_block ← new_block block_indices.append(new_block.index) time_taken_list.append(time_taken)

Algorithm 8: Plot Time Taken for Each Block

plot(block_indices, time_taken_list, marker='o', linestyle='-', label='Time Taken') xlabel('Block Index') ylabel('Time Taken (seconds)') title('Time Taken to Create Each Block') grid(True) legend() show()

E. Treechain Creation and Management Algorithm

Algorithm 9: Node Class Definition

Attributes: val: string connections: list fingerprint: string

Algorithm 10: Calculate Fingerprint

calculate_fingerprintval, connections_fingerprints combined_value ← val + ".join(connections_fingerprints) fingerprint ← SHA-256(combined_value) return fingerprint

Algorithm 11: Create Node

create_nodeval node ← Node(val, [], ") return node

Algorithm 12: Fingerprint Tree

fingerprint_treenode if not node.connections then-è2:node.fingerprint ← SHA-256(node.val) return node.fingerprint else 3:connections_fingerprints ← [fingerprint_tree(conn) for conn in node.connections] combined_value ← node.val + ".join(connections_fingerprints) node.fingerprint ← SHA-256(combined_value) return node.fingerprint

Algorithm 13: Time Fingerprint Measurement

time_fingerprint_measurementnode start_time ← current_time() fingerprint_tree(node) end_time ← current_time() time_elapsed ← end_time - start_time return time_elapsed

Algorithm 14: Generate Node Instances and Hierarchical Connections

top ← create_node("Top") childA ← create_node("Child A") childB ← create_node("Child B") childC ← create_node("Child C") childD ← create_node("Child D") childE ← create_node("Child E") childF ← create_node("Child F") top.connections ← [childA, childB, childC] childA.connections ← [childD, childE] childB.connections ← [childF]

Algorithm 15: Measure Time for Fingerprint Computation

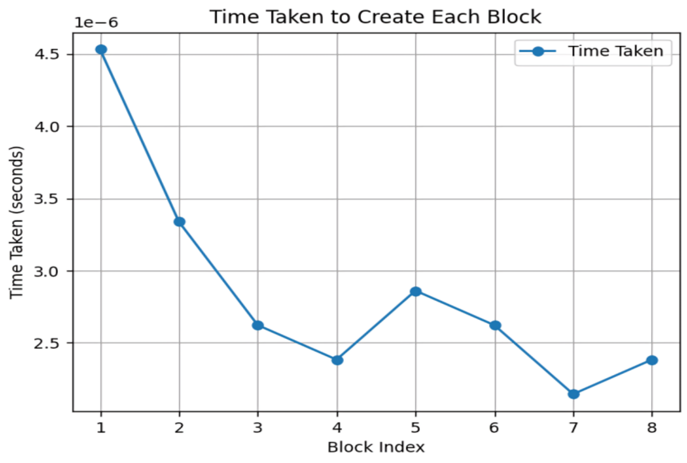
1: node_list ← [top, childA, childB, childC, childD, childE, childF] time_elapsed_list ← [] for node in node_list do-è

2: time_elapsed ← time_fingerprint_measurement(node) time_elapsed_list.append(time_elapsed)

Algorithm 16: Plot Time Complexity

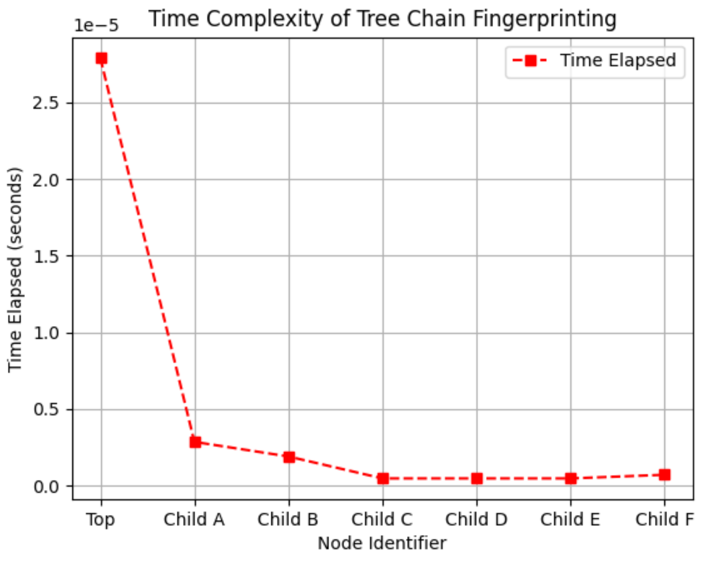
node_identifiers ← ["Top", "Child A", "Child B", "Child C", "Child D", "Child E", "Child F"] plt.plot(node_identifiers, time_elapsed_list, marker='s', linestyle='–', color='r', label='Time Elapsed') plt.xlabel('Node Identifier') plt.ylabel('Time Elapsed (seconds)') plt.title('Time Complexity of Tree Chain Fingerprinting') plt.grid(True) plt.legend() plt.show()

Fig-1 Time Complexity of Blockchain



Graphical Analysis: The time analysis of the creation of blockchain is shown in fig:1 x-axis represented as Blocks, y-axis is the time taken for creation of each block. The time to create hashes ranges from around 4.5 to 2.4 seconds, The graph's curve suggests a nonlinear relationship between block number and time taken.

Figure 2 Treechain Time complexity representation:



Graphical Analysis: Here is the time analysis of the creation of Childs in Treechain, here The x-axis of the graph represents the number of nodes (or Childs) in the Treechain. The y-axis, on the other hand, showcases the time it takes to create these nodes. Notably, the graph highlights the time taken for hash creation, which falls within a range of 2.7 to 0.2 seconds. This is a significant advantage compared to block formation times in a blockchain system. The linear time complexity showcased in the graph translates to a significant efficiency advantage for Treechain. Unlike blockchain, where block formation times can become a bottleneck for scalability, Treechain's hashing process scales efficiently with increasing data volume. The graph provides compelling evidence that Treechain offers faster creation times compared to blockchain. This speed advantage is attributed to the linear time complexity of its hashing process, making Treechain a potentially more scalable and efficient solution for specific applications.

5. Results

The ever-growing demand for secure and scalable data management has sparked innovation in distributed ledger technologies. While blockchain has established itself as a powerful tool, its reliance on complex block formation processes can create bottlenecks for high-volume applications. Treechain emerges as a potential contender, offering a compelling alternative with its focus on speed and efficiency.

The Speed Advantage of Treechain:

The analysis of Treechain's fingerprint creation time, as revealed by the graph, showcases a significant advantage over blockchain. The linear time complexity of Treechain's hashing process translates to faster creation times for nodes compared to the complex cryptographic procedures involved in blockchain's block formation. This translates to a clear edge for Treechain in scenarios where rapid data processing and scalability are paramount.

Reduced Complexity for Streamlined Operations:

Beyond speed, Treechain offers a simpler and less computationally intensive approach compared to blockchain. By focusing on efficient hashing of individual data elements, Treechain avoids the overhead associated with creating and linking intricate block structures. This reduction in complexity can be particularly beneficial for resourceconstrained environments or applications dealing with high data turnover.

A Complementary Approach, Not a Replacement: It's important to acknowledge that blockchain technology offers a robust and well-established foundation for various use cases. Treechain doesn't aim to replace blockchain entirely but rather presents itself as a complementary option for specific scenarios where speed and efficiency are critical.

The Future of Distributed Ledgers: Specialization is Key: The emergence of Treechain highlights the growing trend towards specialization within the realm of distributed ledger technologies. Different applications may require varying levels of security, scalability, and performance. Treechain offers a compelling option for those prioritizing speed and efficiency, while blockchain continues to excel in areas demanding robust security and established infrastructure.

In conclusion, Treechain's faster creation times and reduced complexity make it a promising contender in the race for efficient data management. Its emergence strengthens the notion that the future of distributed ledgers lies in specialization, with different technologies catering to the specific needs of diverse application

6. Discussion

The evolution from linear blockchains to more advanced architectures like Tree-chain represents a significant advancement in distributed ledger technology. This progression addresses many of the challenges faced by early blockchain implementations while opening up new possibilities for blockchain applications.

1. Addressing Scalability Challenges

One of the most pressing issues in blockchain technology has been scalability. Linear blockchains, while secure and decentralized, have struggled to meet the increasing demands of global adoption. The Tree-chain architecture's approach to this problem is innovative and promising:

- Parallel processing: By allowing multiple chains to operate simultaneously, Tree-chain significantly increases the number of transactions that can be processed in a given timeframe.

- Dynamic scaling: The self-scaling feature of Tree-chain is particularly noteworthy. This adaptive capacity could be crucial in handling sudden spikes in transaction volume, a common occurrence in financial markets or during peak usage periods of decentralized applications (dApps).

- Efficient validator selection: The two-level randomization system for selecting validators, combined with the use of consensus codes, offers a more streamlined approach to transaction validation. This could potentially reduce the computational overhead associated with consensus mechanisms in linear blockchains.

These advancements in scalability could pave the way for blockchain technology to be more widely adopted in high-throughput applications, such as payment systems, supply chain management, and IoT networks.

2. Energy Efficiency and Sustainability

The energy consumption of blockchain networks, particularly those using Proof of Work consensus, has been a significant point of criticism. Tree-chain's approach offers a promising solution:

- Reduced computational requirements: By assigning transaction processing responsibilities based on transaction characteristics, Tree-chain potentially reduces the need for energy-intensive competitive mining.

- Efficient resource utilization: The parallel processing nature of Tree-chain allows for more efficient use of network resources, potentially leading to lower overall energy consumption.

- Alignment with sustainability goals: As organizations and governments increasingly focus on reducing carbon footprints, Tree-chain's energy efficiency could make it a more attractive option for large-scale blockchain implementations.

This shift towards more energy-efficient blockchain architectures could help address environmental concerns and make blockchain technology more palatable for environmentally conscious stakeholders.

3. Implications for Blockchain Adoption and Use Cases

The development of architectures like Tree-chain could significantly expand the potential use cases for blockchain technology:

- Real-time applications: The near real-time transaction settlement capability of Tree-chain opens up possibilities for applications that require immediate finality, such as high-frequency trading or real-time supply chain management.

- IoT and edge computing: The improved scalability and efficiency make Tree-chain more suitable for IoT networks, where a large number of devices need to process transactions quickly and with minimal energy consumption.

- Microservices and decentralized applications: The parallel nature of Tree-chain could enable more complex and interconnected decentralized applications, potentially leading to new paradigms in software architecture.

However, it's important to note that linear blockchains will likely continue to play a crucial role in scenarios where maximum security and decentralization are paramount, such as in cryptocurrencies or sensitive record-keeping systems.

4. Challenges and Future Research Directions

While Tree-chain offers many potential benefits, several challenges and areas for future research emerge:

- Security analysis: Rigorous security audits and formal verification of Tree-chain implementations will be crucial to ensure they can match the security guarantees of established linear blockchains.

- Complexity management: The increased complexity of Tree-chain architecture may pose challenges in terms of implementation, maintenance, and governance. Research into simplifying the management of such systems will be valuable.

- Interoperability: As the blockchain ecosystem becomes more diverse, developing robust interoperability solutions between different blockchain architectures will be crucial. This includes facilitating seamless communication between linear blockchains and Tree-chain systems.

- Hybrid systems: Future research could explore the potential for hybrid systems that combine elements of both linear and tree-like structures, leveraging the strengths of each to create even more robust and versatile blockchain solutions.

- Specialized architectures: The insights gained from Tree-chain development could inform the creation of even more specialized blockchain architectures, tailored to specific use cases or industries.

5. Long-term Implications for the Blockchain Ecosystem

The development of architectures like Tree-chain signals a maturation of the blockchain space:

- Diversification: The blockchain landscape is likely to become increasingly diverse, with different architectures coexisting to serve various needs. This diversification could lead to a more robust and adaptable blockchain ecosystem.

- Specialization: As different architectures prove their worth in specific scenarios, we may see increased specialization in blockchain solutions, with particular architectures becoming the go-to choice for certain industries or applications.

- Innovation acceleration: The success of new architectures like Tree-chain could spur further innovation in the field, potentially leading to even more groundbreaking approaches to distributed ledger technology.

- Mainstream adoption: By addressing key limitations of early blockchain systems, architectures like Tree-chain could accelerate the mainstream adoption of blockchain technology across various industries.

In conclusion, the evolution from linear blockchains to Tree-chain represents a significant step forward in blockchain technology. While linear blockchains have laid a strong foundation and will continue to play a crucial role in certain applications, architectures like Tree-chain offer solutions to pressing challenges in scalability, efficiency, and adaptability. As the technology continues to mature, we can expect to see a rich ecosystem of blockchain solutions, each tailored to specific needs and use cases. This diversity and ongoing innovation will be key to realizing the full potential of blockchain technology across a wide range of industries and applications.

References

[1] https://imiblockchain.com/types-of-blockchain/.
[2] National Institute of Standards and Technology, “Secure Hash Standard”, FIPS180-3 (2008)
[3] H. F. Atlam, A. Alenezi, M. O. Alassafi, and G. Wills, “Blockchain with internet of things: Benefits, challenges, and future directions,” International Journal of Intelligent Systems and Applications, vol. 10, no. 6, pp. 40–48, 2018.
[4] K. Christidis and M. Devetsikiotis, “Blockchains and smart contracts for the internet of things,” Ieee Access, vol. 4, pp. 2292–2303, 2016
[5] O. Alphand, M. Amoretti, T. Claeys, S. Dall’Asta, A. Duda, G. Ferrari, F. Rousseau, B. Tourancheau, L. Veltri, and F. Zanichelli, “Iotchain: A blockchain security architecture for the internet of things,” in 2018 IEEE Wireless Communications and Networking Conference (WCNC). IEEE, 2018, pp. 1–6.
[6] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, “Lsb: A lightweight scalable blockchain for iot security and anonymity,” Journal of Parallel and Distributed Computing, vol. 134, pp. 180–197, 2019
[7] Ali Dorri,Raja Jurdak,Tree-Chain: A Fast Lightweight Consensus Algorithm for IoT Applications Queensland University of Technology (QUT)
[8] https://en.wikipedia.org/wiki/Blockchain

[9] https://sdewes.org/jsdewes/pid10.0436
[10] https://www.jmir.org/2021/8/e17475
[11] Prabhat Kumar, Randhir Kumar, Gautam Srivastava, Govind P. Gupta, Rakesh Tripathi, Thippa Reddy Gadekallu, Neal N. Xiong. "PPSF: A Privacy-Preserving and Secure Framework Using Blockchain-Based MachineLearning for IoT-Driven Smart Cities" , IEEE Transactions on Network Science and Engineering, 2021
[12] Arpit Namdev, Harsh Lohiya. "Chapter 8 Design and Analysis of an Algorithm Based on Biometric Block Chain for Efficient data sharing in VANET" , Springer Science and Business Media LLC, 2023