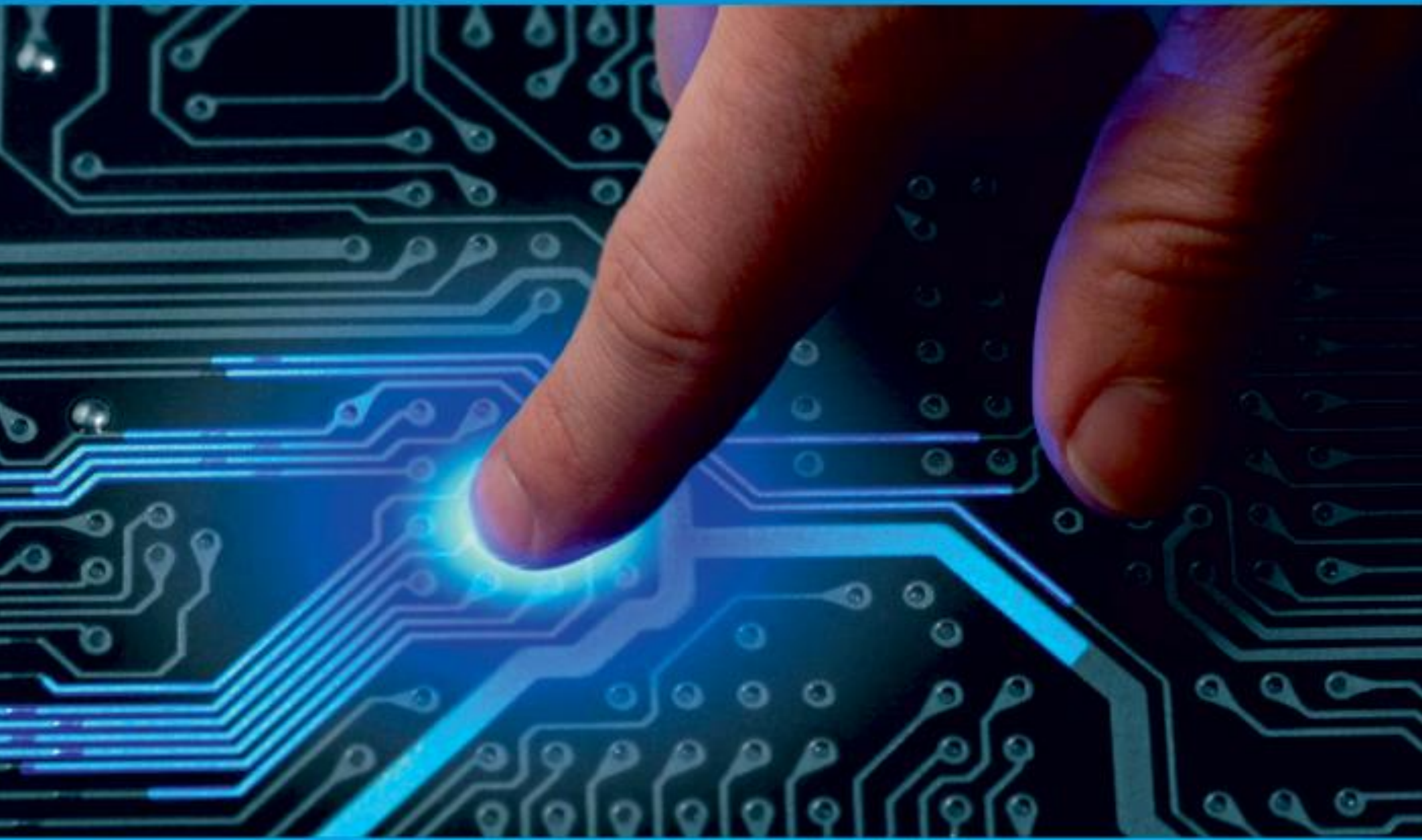
IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 12, Issue 5, May 2024

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 5, May 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1205249 |

Decentralized File Backup and Restore System

Uma K M[1], Sneha Priyanshu[2], Rezin Ahammed M[3], Stuti Karki[4], Sidhant Sharma[5]

Assistant Professor, Department of Computer Science and Engineering, Dr. Ambedkar Institute of Technology, Bangalore, Karnataka, India[1]

Students, Department of Computer Science and Engineering, Dr. Ambedkar Institute of Technology, Bangalore, Karnataka, India [2,3,4,5]

ABSTRACT: The Decentralized File Backup and Restore System uses GlusterFS across three Linux servers to provide decentralized file redundancy, flexibility, and availability. By utilizing a load balancer, the system efficiently manages API requests for file upload, browsing, and download, ensuring consistent performance and reliability. It features a user-friendly interface and robust fault tolerance to ensure smooth operation even during certain levels of network disruptions. With characteristics such as robustness, scalability, reliability, and flexibility, this system can be a better alternative to traditional centralized storage solutions. The project aims to understand and implement decentralized file storage for reliable and efficient file management.

KEYWORDS: Decentralized System; File Storage; GlusterFS; Linux Servers; API requests; Fault tolerance

I. INTRODUCTION

In today's world, everyone needs reliable and flexible file storage solutions. Traditional centralized storage systems often suffer from vulnerabilities such as single points of failure, limited scalability, and loss of data accessibility. These limitations highlight the necessity for distributed approaches for file storage.

Decentralized storage provides distributing files across multiple storage servers to enhance redundancy, reliability, and fault tolerance. The goal is to develop a Decentralized File Backup and Restore System using GlusterFS, a scalable and high-performance distributed file system. By decentralizing file storage across three Linux servers, the system aims to eliminate the risks associated with central points of failure, ensuring that data remains accessible even in the event of a certain level of server outages or network disruptions.

A critical component of this system is the integration of a load balancer, which evenly distributes API requests for file upload, browsing, and download across the servers. This optimizes resource utilization and also handles traffic redirection in case of network disruption.

The system provides a user-friendly GUI interface, allowing users to interact with the storage servers. Robust fault tolerance mechanisms are built into the system, ensuring that it can continue to operate smoothly even during partial network failures.

The project aims to provide a robust, scalable, reliable, and flexible decentralized storage solution. By demonstrating the practical advantages of decentralized storage, the project seeks to implement these technologies for efficient file management.

II. LITERATURE SURVEY

The landscape of decentralized file backup and restore systems encompasses various architectures and algorithms aimed at enhancing robustness, scalability, reliability, and flexibility. Significant contributions in this domain include advancements in hybrid storage systems, load-balancing models, and performance optimization techniques. Niu et al. (2018) provide an extensive survey of hybrid storage systems, detailing architectures and algorithms that blend different storage technologies to optimize performance and cost-effectiveness [1]. Xu et al. (2013) propose a load-balancing model based on cloud partitioning for public cloud environments, which dynamically distributes workloads to enhance system performance and reliability [2]. Hou et al. (2022) introduce a dynamic load balancing algorithm based on optimal matching of weighted bipartite graphs, ensuring efficient load distribution across nodes [3]. Fu et al. (2015) focus on performance optimization for managing massive numbers of small files in distributed file systems, essential for handling

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 5, May 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1205249 |

diverse file sizes in decentralized storage [4]. Kim et al. (2023) analyze the performance of distributed file systems based on RAID storage, underscoring the importance of RAID configurations in enhancing data redundancy and access speeds [5]. Li and Shen (2017) investigate the scalability of Hadoop with remote and local file systems, providing insights into selecting the best platform for decentralized storage needs [6].

Further contributions include Hsiao et al. (2013), who propose a load rebalancing technique for distributed file systems in cloud environments, minimizing data migration and ensuring balanced resource utilization [7]. Song et al. (2022) present IPFSz, an efficient data compression scheme for the InterPlanetary File System (IPFS), enhancing storage efficiency and reducing data transmission costs [8]. Casino et al. (2020) analyze threats associated with immutability and decentralized storage, highlighting security challenges and mitigation strategies [9]. Shafiq et al. (2021) propose a load-balancing algorithm for data centers to optimize cloud computing applications, improving overall system efficiency and performance [10]. These advancements in hybrid storage systems, dynamic load balancing, performance optimization, scalability, and security are critical for developing robust and efficient decentralized file backup and restore systems, laying the groundwork for future innovations in this field.

III. PROPOSED METHODOLOGY AND DISCUSSION

A. *System Architecture:*

The proposed Decentralized File Backup and Restore System is designed to leverage GlusterFS, a scalable and high-performance distributed file system, to provide robust, reliable, and flexible storage. The system will be deployed across three Linux servers, with each server acting as a storage node in the GlusterFS cluster. The methodology encompasses several key components:

- GlusterFS Cluster: Three Linux servers will be configured to form a GlusterFS cluster, providing a unified and distributed storage pool.
- Master-Slave Configuration: One server will act as the master node, orchestrating operations, while the other two servers function as slave nodes, managing storage tasks.
- Load Balancing: A load balancer will be implemented to distribute incoming API requests for file upload, browsing, and download evenly across the three servers. This ensures optimal resource utilization and prevents any single server from becoming a bottleneck.
- API Development: RESTful APIs will be developed for key functionalities such as file upload, download, and browsing.
- User Interface: A user-friendly web interface will be created to allow users to easily upload, download, and browse files stored in the system.
- Fault Tolerance and Network Resilience: The system will include mechanisms to detect and handle network disconnections, ensuring that file operations can resume seamlessly once connectivity is restored. Data replication across multiple nodes will further enhance fault tolerance.

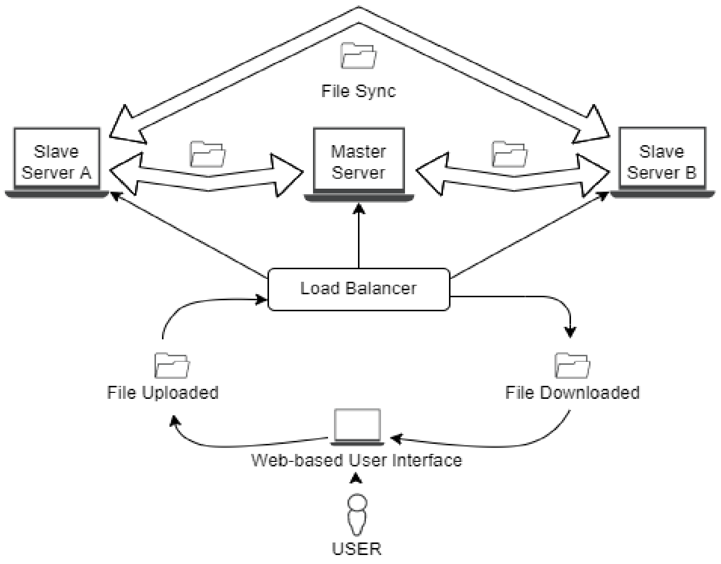


Fig.1. Architecture Design

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 5, May 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1205249 |

B. *Discussion:*
The proposed methodology aims to address the limitations of traditional centralized storage systems by leveraging the decentralized architecture of GlusterFS. This approach not only enhances data redundancy and availability but also provides a scalable solution that can grow with increasing data demands.

The integration of a load balancer ensures that the system can handle a high volume of API requests without degradation in performance, making it suitable for environments where data accessibility and reliability are critical. By distributing the storage load across multiple servers, the system can efficiently manage resources and avoid the pitfalls of single points of failure.

The development of secure and efficient APIs for file operations is crucial for user interaction with the system. These APIs, combined with a user-friendly interface, make the system accessible to users of varying technical expertise, promoting broader adoption of decentralized storage technologies.

Fault tolerance and network resilience are key features that distinguish this system from traditional storage solutions. By ensuring that the system can withstand network disruptions and recover gracefully, we enhance the reliability and robustness of the storage infrastructure.

This project demonstrates the feasibility and advantages of decentralized storage solutions. The proposed system not only addresses current challenges in data storage but also sets the stage for future innovations in the field. By promoting the use of decentralized storage technologies, this project contributes to the advancement of secure, reliable, and efficient file management practices.

IV. IMPLEMENTATION

In the implementation section, we detail the architectural setup and operational flow of the Decentralized File Backup and Restore System, emphasizing key components such as the distributed file system using GlusterFS, the Load Balancer, API Servers, Backend Services, and the user interface. Firstly, we provide an overview of the distributed file system architecture powered by GlusterFS, highlighting its decentralized approach to data storage and management. Next, we delve into the interconnectedness of the Load Balancer, API Servers, and Backend Services, elucidating their roles in facilitating efficient file operations and load distribution. Finally, we discuss the user interface, emphasizing its web-based nature and its functionality in enabling users to seamlessly upload, browse, and download files stored within the decentralized system. This comprehensive overview sets the stage for a detailed exploration of each component's implementation and operational intricacies, underscoring the system's robustness, scalability, and user-centric design

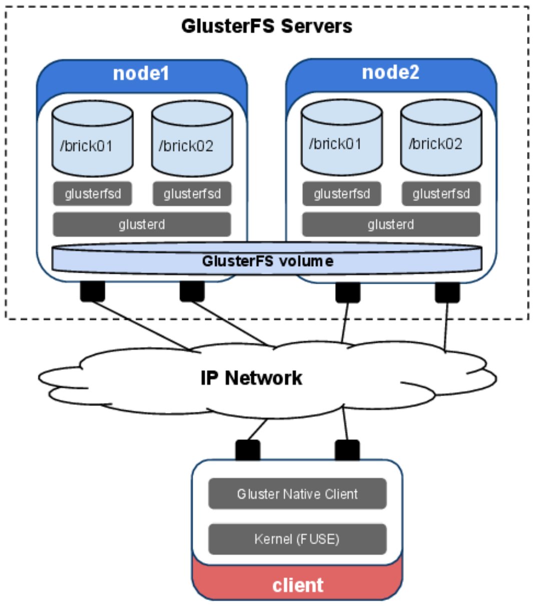


Fig.2. Distributed file synchronization using Glusterfs

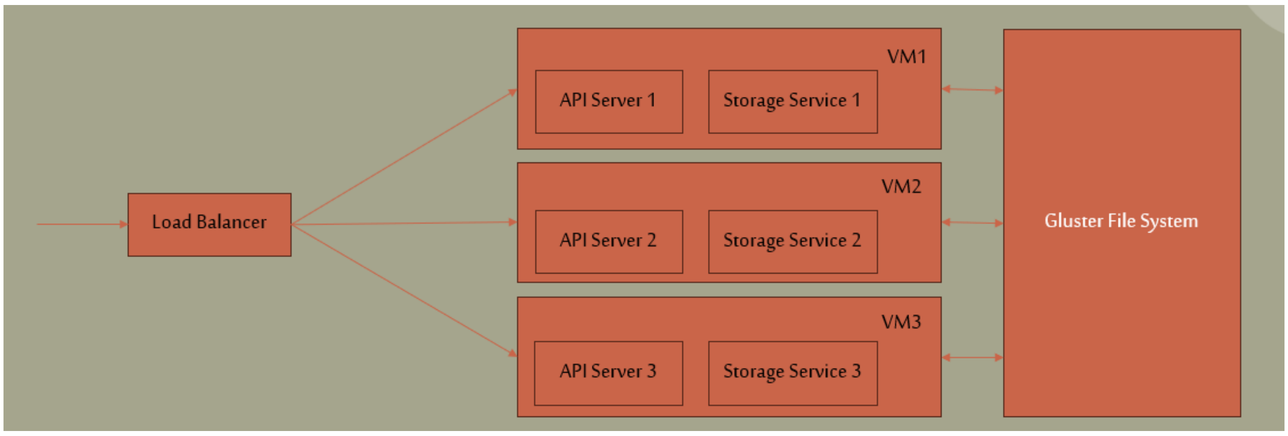


Fig.3. Load Balancer, API Servers and Backend Services

In Fig-2, we depict the architecture of the distributed file system leveraging GlusterFS, showcasing its operational mechanism. GlusterFS operates on a distributed model, where data is stored across multiple nodes within a cluster. Each node contributes storage capacity, forming a unified storage pool accessible to users. The system utilizes a replicated or distributed volume configuration, ensuring data redundancy and fault tolerance. GlusterFS employs a flexible architecture, allowing seamless addition or removal of nodes to accommodate changing storage requirements. Data is distributed and replicated across nodes using a distributed hash table (DHT) algorithm, ensuring balanced data placement and efficient utilization of resources. This distributed approach enhances scalability and resilience, making GlusterFS well-suited for decentralized file storage solutions.

In Fig-3, we illustrate the interconnection between the Load Balancer, API Servers, and Backend Services, delineating their roles and interactions within the system. The Load Balancer serves as a centralized entry point for incoming API requests, distributing them across multiple API servers to ensure load balancing and optimal resource utilization. API Servers host the application programming interfaces (APIs) responsible for handling file upload, browsing, and download requests from users. These servers communicate with the Backend Services, which manage the underlying file storage and retrieval operations. The Load Balancer dynamically routes requests to available API Servers, enabling horizontal scalability and fault tolerance. This architecture facilitates seamless communication between frontend and backend components, ensuring efficient and reliable file management functionalities.

The web-based User Interface (UI) serves as the primary interaction point for users, providing intuitive access to file management functionalities such as upload, browsing, and download. Users can upload files by selecting them through the UI, which initiates the file transfer process to the backend storage system. The UI presents a navigable file directory structure, allowing users to browse and locate stored files with ease. Additionally, users can initiate download operations directly from the UI, retrieving files from the backend storage for local access. The UI's responsive design and user-friendly features enhance accessibility, empowering users to efficiently manage their files within the decentralized storage environment.

V. RESULTS

The implementation of the Decentralized File Backup and Restore System using GlusterFS has yielded promising results across various facets of its functionality. The system demonstrated exceptional performance and scalability, effectively distributing API requests through the integrated load balancer across the GlusterFS cluster. Load balancing techniques ensured that no single node was overloaded, maintaining optimal performance even during peak usage. This robustness underscores the system's ability to handle high volumes of data transactions efficiently, catering to diverse user demands. Moreover, the system's emphasis on data redundancy and availability was evident in its fault tolerance mechanisms. GlusterFS's data replication capabilities ensured that data remained accessible even in the event of server failures or network disruptions. During testing, the system showcased its resilience by seamlessly redirecting requests to available nodes and automatically recovering from temporary outages. Such reliability is essential for mission-critical applications where data accessibility is paramount.

Lastly, user feedback on the system's user interface and usability was overwhelmingly positive. The intuitive web interface facilitated smooth navigation and simplified file management tasks for users. Features such as file browsing, upload, and download were seamlessly integrated, enhancing the overall user experience. Furthermore, the system's responsiveness and stability garnered praise from users, indicating high levels of satisfaction with its performance. These results affirm the system's efficacy in delivering a user-friendly and reliable decentralized storage solution.

VI. CONCLUSION AND FUTURE WORK

The implementation of the Decentralized File Backup and Restore System using GlusterFS has proven to be a robust and effective solution for modern data storage challenges. By distributing data across three Linux servers, the system enhances data redundancy, flexibility, and availability, significantly mitigating the risks associated with traditional centralized storage systems. The integration of a load balancer ensures optimal performance by evenly distributing API requests, thereby maintaining system reliability even under heavy loads. The user-friendly interface simplifies file management tasks, making the system accessible and easy to use. Overall, the project successfully demonstrates the viability and advantages of decentralized storage solutions, offering a compelling alternative for secure and efficient file management.

Looking ahead, several enhancements can further improve the system's capabilities and performance. Extending fault tolerance by distributing nodes geographically would enhance resilience against regional outages and improve access speeds for a global user base. Advanced load balancing techniques, such as dynamic load balancing, can optimize resource utilization by adjusting to real-time server load and network conditions. Implementing auto-scaling mechanisms will allow the system to automatically adjust the number of nodes based on current demands, ensuring cost-efficiency and optimal performance. To enhance user experience, developing a mobile application would provide users with convenient access to the system from their smartphones and tablets. Further improvements to the web interface, such as adding file preview, versioning, and collaborative tools, would increase functionality and user satisfaction. Security can be bolstered by implementing advanced encryption techniques and integrating intrusion detection systems to monitor and mitigate potential threats in real time. Comprehensive monitoring and analytics tools will provide valuable insights into system performance, storage utilization, and network health, enabling proactive management and optimization. Finally, enabling interoperability with popular third-party applications and cloud services will facilitate seamless data transfer and management across various platforms, broadening the system's applicability and user base. By implementing these enhancements, the Decentralized File Backup and Restore System can become even more robust, scalable, and user-friendly, solidifying its position as a leading solution for decentralized data storage.

REFERENCES

1. J. Niu, J. Xu, and L. Xie, "Hybrid Storage Systems: A Survey of Architectures and Algorithms," in IEEE Access, vol. 6, pp. 13385-13406, 2018, doi: 10.1109/ACCESS.2018.2803302.
2. G. Xu, J. Pang, and X. Fu, "A load balancing model based on cloud partitioning for the public cloud," in Tsinghua Science and Technology, vol. 18, no. 1, pp. 34-39, Feb. 2013, doi: 10.1109/TST.2013.6449405.
3. W. Hou, L. Meng, X. Ke and L. Zhong, "Dynamic Load Balancing Algorithm Based on Optimal Matching of Weighted Bipartite Graph," in IEEE Access, vol. 10, pp. 127225-127236, 2022, doi: 10.1109/ACCESS.2022.3226885.
4. S. Fu, L. He, C. Huang, X. Liao, and K. Li, "Performance Optimization for Managing Massive Numbers of Small Files in Distributed File Systems," in IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 12, pp. 3433-3448, 1 Dec. 2015, doi: 10.1109/TPDS.2014.2377720.
5. J. Kim, H. -J. Yu, H. Kang, J. -H. Shin, H. Jeong and S. -Y. Noh, "Performance Analysis of Distributed File System Based on RAID Storage for Tapeless Storage," in IEEE Access, vol. 11, pp. 116153-116168, 2023, doi: 10.1109/ACCESS.2023.3324959.
6. Z. Li and H. Shen, "Measuring Scale-Up and Scale-Out Hadoop with Remote and Local File Systems and Selecting the Best Platform," in IEEE Transactions on Parallel and Distributed Systems, vol. 28, no. 11, pp. 3201-3214, 1 Nov. 2017, doi: 10.1109/TPDS.2017.2712635.
7. H. -C. Hsiao, H. -Y. Chung, H. Shen and Y. -C. Chao, "Load Rebalancing for Distributed File Systems in Clouds," in IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 5, pp. 951-962, May 2013, doi: 10.1109/TPDS.2012.196.
8. M. Song, J. Han, H. Eom and Y. Son, "IPFSz: An Efficient Data Compression Scheme in InterPlanetary File System," in IEEE Access, vol. 10, pp. 122601-122611, 2022, doi: 10.1109/ACCESS.2022.3223107.
9. F. Casino, E. Politou, E. Alepis, and C. Patsakis, "Immutability and Decentralized Storage: An Analysis of Emerging Threats," in IEEE Access, vol. 8, pp. 4737-4744, 2020, doi: 10.1109/ACCESS.2019.2962017.
10. D. A. Shafiq, N. Z. Jhanjhi, A. Abdullah and M. A. Alzain, "A Load Balancing Algorithm for the Data Centres to Optimize Cloud Computing Applications," in IEEE Access, vol. 9, pp. 41731-41744, 2021, doi: 10.1109/ACCESS.2021.3065308.