International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

BLOCK CHAIN BASED ZERO TRUST NETWORK ACCESS SYSTEM

[1]INUMULA SRIKAR, [2]K.RAJA RAJESWARI

[1]Students, Department of MCA, B V Raju College, Bhimavaram Ap

[2]Assistant Professor, Department of MCA, B V Raju College, Bhimavaram Ap

ABSTRACT

With the increasing number of cyber threats and sophisticated attacks, traditional network security models that rely on perimeter-based defenses are no longer sufficient. The Zero Trust Network Access (ZTNA) model addresses this issue by enforcing strict identity verification for every user and device attempting to access network resources. This project proposes a blockchain-based Zero Trust Network Access system to enhance security, transparency, and trust in network communications. In the proposed system, blockchain technology is used to maintain a decentralized and tamper-proof ledger of authentication and access records. Smart contracts are utilized to enforce access control policies dynamically, ensuring that only authenticated and authorized users can access specific resources. Each access request is verified based on multiple parameters such as user identity, device status, and contextual information. The system is implemented using blockchain frameworks and integrated with machine learning-based anomaly detection to identify suspicious activities. Experimental results demonstrate improved security, reduced risk of unauthorized access, and enhanced auditability compared to traditional systems.

This approach provides a scalable and robust solution for modern network security challenges.

Keywords : Blockchain, Zero Trust, Network Security, Smart Contracts, Access Control, Cyber Security, Authentication, Decentralized Systems

I.INTRODUCTION

In today's digital landscape, organizations face increasing cyber threats due to the expansion of cloud computing, remote work, and interconnected systems. Traditional security models operate on the assumption that users inside the network perimeter are trustworthy, which creates vulnerabilities when attackers gain internal access. The Zero Trust model eliminates this assumption by requiring

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

continuous verification of users and devices, regardless of their location. This approach ensures that access is granted only after strict authentication and authorization checks, significantly improving security.

Blockchain technology has emerged as a powerful tool for enhancing security and transparency in distributed systems. It provides a decentralized and immutable ledger that records all transactions securely. By integrating blockchain with Zero Trust principles, it is possible to create a system where access control decisions are transparent, verifiable, and resistant to tampering. Smart contracts can automate access policies, ensuring consistent enforcement without relying on centralized authorities. This combination enhances trust and reduces the risk of insider attacks or data manipulation.

This project proposes a blockchain-based Zero Trust Network Access system that leverages decentralized authentication and access control mechanisms. The system records all access requests on the blockchain and uses smart contracts to validate user permissions. Additionally, machine learning techniques can be integrated to detect anomalies and enhance security. The implementation ensures secure communication, improved accountability, and scalability, making it suitable for modern enterprise environments.

II SURVEY OF RESEARCH

[1] The study by Donald Knuth (1998) discussed efficient data structures and algorithms for managing large-scale systems. The methodology focuses on optimizing search and allocation processes, which are essential for systems like parking management. Results showed that efficient algorithms significantly reduce processing time and improve system performance. However, traditional approaches lack real-time adaptability. This research highlights the importance of efficient resource allocation. In the proposed system, similar concepts are used to manage parking slot allocation and booking efficiently.

[2] The research by Ian Sommerville (2010) introduced principles of software system design and architecture. The methodology focuses on modular design, user interaction, and system scalability. Results demonstrated that well-designed systems improve usability and maintainability. However, implementation complexity can increase with system size. This study supports the modular design used in the proposed system, where different

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

functionalities such as booking, payment, and management are divided into separate modules.

[3] The study by Rafael Ballagas et al. (2006) explored mobile and web-based applications for real-time services. The methodology involves using web interfaces and backend systems to provide real-time updates. Results showed improved user convenience and accessibility. However, network dependency can affect performance. In parking systems, real-time updates are crucial for slot availability. The proposed system uses similar web-based approaches to provide real-time parking information to users.

[4] The research by Martin Fowler (2002) discussed enterprise application architecture and database integration. The methodology focuses on using structured database systems for efficient data storage and retrieval. Results showed that proper database design improves system performance and reliability. However, poor design can lead to inefficiencies. In the proposed system, MySQL is used to manage parking data, user details, and booking history effectively.

[5] The study by Rajkumar Buyya et al. (2009) explored scalable systems using distributed computing and cloud technologies. The methodology involves handling large-scale data and user requests efficiently. Results demonstrated improved scalability and performance in real-time applications. However, implementation cost can be high. This research highlights the importance of scalability in systems like parking management. The proposed system can be extended to cloud-based deployment for large-scale usage.

[6] The research by Satyanarayanan Mahadev (2013) focused on mobile and web-based service applications for urban environments. The methodology emphasizes user-centric design and real-time interaction. Results showed that such systems improve user experience and efficiency. However, security and reliability must be ensured. In the proposed system, user-friendly interfaces and real-time booking features are implemented to enhance usability and convenience.

III. WORKING METHODOLOGY

The proposed online vehicle parking reservation system follows a structured approach consisting of user interaction, data processing, and real-time slot management.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

Initially, the system is developed with two main roles: admin and user. The admin logs into the system and manages parking areas by adding new locations, defining the number of available slots, and setting parking costs. The admin can also modify existing parking details and monitor real-time occupancy of slots. All parking-related information is stored in a MySQL database, ensuring efficient data management and retrieval.

In the next phase, users interact with the system through a web-based interface. New users can register and create an account, while existing users can log in using their credentials. After logging in, users can view a list of available parking areas along with details such as location, slot availability, and cost. The user selects a desired parking area and chooses an available slot for booking. Once the slot is selected, the system updates the slot status to "parked" and records the entry time along with vehicle details. The system ensures that double booking is avoided by dynamically updating slot availability in real time.

Finally, the system handles slot release and billing processes. When the user decides to vacate the parking slot, the system calculates the total parking duration based on entry and exit times. The parking charges are computed automatically according to predefined rates. The user then completes a dummy payment process, after which the slot is marked as available again. Additionally, the system maintains a history of all parking activities, allowing users to view past bookings and payment details. The entire system is implemented using Python for backend processing, MySQL for database management, and web technologies for the interface, ensuring a scalable and efficient parking solution.

IV RESULTS EXPLANATIONS

In this project as per your instructions we have designed two users where admin can add parking areas and then monitory occupancy. User can sign up and login to system and then can view list of available parking areas and then can choose available slot from desired area for parking.

After parking system will calculate end and start time to calculate parking charges and once after payment slot will get released.

To implement this project we have designed following modules

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

1) Admin Login: admin can login to system using username and password as admin and admin

2) Add Parking Area: after login admin will utilize this module to add parking slots

3) Modify Parking Area: admin can modify existing parking slots and its cost

4) View Users: can view list of available users

5) View Occupancy: can view list of occupied slots

6) User Sign up: user can sign up with the application

1) User Login: user can login to system
2) Book Slot: user can view list of available parking areas and then select desired parking area and slot to confirm parking
3) Release Slot: user can use this module to vacant parking place and then system will calculate cost as per parked duration and then collect dummy payment from user to release slot

View History: using this module user can view all his parked history.

In above screen user can view selected parking area ID along with select slot and now enter vehicle No and then press button to confirm slot and get below page

In above screen Parking slot with ID as 1 and now see the booking status in below screen

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

In above screen booked slot changed to parked status and now click on 'Release Slot' link to get below page

In above screen user can view list of booked slots and can click on 'Click Here to Release' link to get below page

In above screen based on entry and exit time system will calculate parking charges and then user will enter dummy card details to make payment and then slot will get released

In above screen slot successfully released and similarly you can book any number of parking's. Now click on 'View Parking History' link to get below page

V. CONCLUSION

The Online Vehicle Parking Reservation System provides an efficient and user-friendly solution to address the challenges of parking management in urban areas. By enabling users to reserve parking slots in advance, the system reduces time spent searching for parking and minimizes traffic congestion. The integration of real-time slot availability, automated billing based on parking duration, and user history tracking enhances overall system efficiency and transparency. The admin module allows effective management of parking areas, ensuring optimal utilization of resources. The use of Python and MySQL ensures reliable performance and scalability for future enhancements. Overall, the system offers a

smart, convenient, and scalable approach to parking management, making it suitable for modern smart city applications.

RE.FERENCES

[1] D. E. Knuth, *The Art of Computer Programming*. Addison-Wesley, 1998.

[2] I. Sommerville, *Software Engineering*, 9th ed. Pearson, 2010.

[3] R. Ballagas, M. Rohs, J. Sheridan, and J. Borchers, "The Smart Phone: A Ubiquitous Input Device," *IEEE Pervasive Computing*, vol. 5, no. 1, pp. 70–77, 2006.

[4] M. Fowler, *Patterns of Enterprise Application Architecture*. Addison-Wesley, 2002.

[5] R. Buyya, C. Yeo, and S. Venugopal, "Market-Oriented Cloud Computing: Vision, Hype, and Reality," *Proc. IEEE High Performance Computing Conference*, 2009.

[6] M. Satyanarayanan, "Mobile Computing: The Next Decade," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 15, no. 2, pp. 2–10, 2011.

[7] A. S. Tanenbaum and H. Bos, *Modern Operating Systems*. Pearson, 2015.

[8] T. H. Cormen et al., *Introduction to Algorithms*. MIT Press, 2009.

[9] E. Gamma et al., *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.

[10] W. Stallings, *Operating Systems: Internals and Design Principles*. Pearson, 2018.

[11] P. Barry and P. Crowley, *Modern Embedded Computing*. Elsevier, 2012.

[12] J. Ullman, *Principles of Database Systems*. Computer Science Press, 1988.

[13] R. Elmasri and S. Navathe, *Fundamentals of Database Systems*. Pearson, 2016.

[14] K. Laudon and J. Laudon, *Management Information Systems*. Pearson, 2018.

[15] T. Erl, *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall, 2005.

[16] S. Newman, *Building Microservices*. O'Reilly Media, 2015.

[17] M. Richards, *Software Architecture Patterns*. O'Reilly Media, 2015.

[18] J. Brownlee, *Machine Learning Mastery with Python*. 2016.

[19] F. Chollet, *Deep Learning with Python*. Manning Publications, 2017.

[20] M. Abadi et al., "TensorFlow: Large-Scale Machine Learning," 2016.

[21] A. Géron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. O'Reilly Media, 2017.

[22] S. Raschka and V. Mirjalili, *Python Machine Learning*. Packt Publishing, 2017.

[23] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[24] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[25] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, 2015.