IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

SECURE ORACLE IN BLOCKCHAIN

Sivanesh ,Sasidharan , Nivash ,Albin Tenny

GUIDE: MR. KARTHIBAN R

Department Of Computer Science Engineering

(Cyber Security)

Bachelor Of Engineering

Sri Shakthi Institute of Engineering and Technology

(Autonomous)

Coimbatore 641062

ABSTRACT

When a blockchain application works on data from the real world, it relies on the oracle mechanism to transfer data from other sources to the blockchain. Blockchain the Oracle problem arises around the need to obtain reliable information from another source. Previous work focused on data authenticity/integrity by creating secure channels between blockchains and external sites, while using oracle network organizations to solve the information/integrity problem. Avoid failure of any kind. However, the problem of factual information, which arises when external legal documents present false or misleading information, has not yet been resolved. This project aims to solve these problems and improve the use of the chain using Oracle Blockchain technology. This project involves the design and implementation of a blockchain-based application chain using the Oracle Blockchain Platform. The main objectives include creating a distribution certificate and transparency to track the flow of goods throughout the supply chain, from the purchase of raw materials to the end of shipment. Smart contract functionality will automate key aspects of contract execution, invoicing and resolution, providing greater efficiency and cost savings. The findings and insights from the project are expected to contribute to the greater use of blockchain technology in supply chain management and create a more transparent, robust environment for the digital age.
This summary summarizes the project's objectives, approach, expected results and potential impacts to provide stakeholders and readers with a brief overview

CHAPTER1

INTRODUCTION

BLOCKCHAIN: Blockchain is an information system built on for store a information ,details and checks the balances to confirm transactions of the users and customers . First of all the blockchain developed for the cryptocurrency Bitcoin transactions by Satoshi Nakamoto, it is a ledger of transactions that is saved on millions of computers across the world. The information of transactions store in the ledger , and they are connectedwith eachother. Blockchain Decentralization: There is no Third party or System which keeps the data of the Blockchain and check or monitor a data.The information or data is distributed over Millions of Computers around the world which are connected to the same network of Blockchain. The nodes build the blockchain collectively, meaning can't one computer can override the consensus of all the computers across the world. This public system of checks and balances creates a verifiable, almost impenetrable system for instantaneous transactions without a centralized bank. Because Blockchain can be set up to store any form of information in a ledger, it has implications that reach far beyond sales on the internet. From creating publicly verifiable contracts, to buying and selling homes, to running elections, to sharing medical research, and revolutionizing the music production industry, the possibilities are seemingly endless

ORACLE: Oracle software is the relational database management system and these features are created by oracle corporation they made an oracle software.It is also called Oracles, or simply Oracle. Furthermore, Oracle was created in 1977 by Lawrence Ellison and other engineers or developers. Likewise, it is one of the most popular relational database engines in the IT market for storing, organizing, and retrieving data in easy and secure wayOracle database was the first database that designed for businessman grid computing , data warehousing and store a cloud server . The enterprise grid computing provides the most flexible and cost-effective,it means more affiance way to manage information and applications. Oracle has a system of mechanism for controlling , accessing and monitor the database to prevent unauthorized accessIt provides high security because of the Oracle Advanced Security features and it build by talented engineers. It offers two solutions to protect or secure the databases that are TDE (Transparent Data Encryption) and Data Redaction. These TDE supports data encryption both at the source and after export. Redaction is performed at the application level. Oracle has some other security features like Oracle Database Vault that regulates user privileges and Oracle Label Security.It provides features like RAC (Real Application Cluster) and Portability, which makes an Oracle database scalable based on usage. In a clustered environment, it includes capabilities such as rolling instance migrations, performing upgrades, maintaining application continuity, quality of service management,etc.

CHAPTER2

RESEARCH METHODOLOGY:

CONSENSUS: Consensus means multiple servers agreeing on same information, something imperative to design fault-tolerant distributed systems.It helps to overall System realibility,because if one machine fails another machine can take up it's work Process : The client sends a message to the server and the server responds back with a reply.

Leader election : A group of servers or majority of server in a same network electing a leader amongst themselves and letting everyone know about it. In a distributed system or technology, leader election can be critical to coordinate work. In this post, I will quickly show how you can use etcd, a consistent key-value store which uses RAFT protocol for consensus and is used by famous projects like Kubernetes etc, to add leader election to your distributed application.

Mutual Exclusion : A group of servers ensuring mutually exclusive access or monitor to a critical resource.

Membership failure & detection: A group of servers maintaining a list of each other, updating it as someone enters or leaves the group.

Reliable multicasting: A group of servers attempting to receive the same updates in the same order as each other.

Load balancing: Distributing incoming network traffic across a group of backend servers Introduction.

PAXOS CONSENSUS

Paxos is the oldest consensus protocol published way back in 1989 by Leslie Lamport.Paxos is used in multiple graph databases like neo4j. It is also used in column storedatabase Cassandra for leader election.

Symmetric : Any of the multiple servers can respond to the client and all the other servers aresupposed to sync up with the server that responded to the client's request, and

Asymmetric : Only the elected leader server can respond to the client. All other servers thensync up with the leader server.We shall now define some terms used to refer individual servers in a distributed system.

Leader – Only the server elected as leader can interact with the client. All other servers sync up themselves with the leader. At any point of time, there can be at most one leader(possibly 0, which we shall explain later)

Follower – Follower servers sync up their copy of data with that of the leader's after every regular time intervals. When the leader server goes down(due to any reason), one of the followers can contest an election and become the leader. Candidate – At the time of contesting an election to choose the leader server, the servers can ask other servers for votes. Hence, they are called candidates when they have requested votes. Initially, all servers are in the Candidate state.

RaftvsPaxos vs Other Consensus Algorithms: Paxos is a predecessor to Raft. The Raft protocol was invented specifically to improve on perceived weakness in the Paxos protocol. According to the inventors of Raft, the complexity of Paxos makes it difficult to implement. In their words, "Paxos' formulation may be a good one for proving theorems about its correctness, but real implementations are so different from Paxos that the proofs have little value." The title of the Raft paper speaks to this mission: "In Search of an Understandable Consensus Algorithm."

Data Aggregation and Filtering:

Aggregation process

Data collection

Filtering mechanisn

Source data transmission

AggregationProcess:ChooseAggregationMethods:Select appropriate aggregation methods basedon the type of data. Common methods include averaging,summing,counting,ormorecomplexstatisticalmeasures.

Calculation:Insomecases,oraclesmayperformcalculationsortransformationsonthedatabeforetransmittingittotheblockchain.Foranexample,Toconvertingcurrencyvaluestoastandarddenomination.

HandleTimeSeriesData:Ifdealingwithtime-seriesdata,considerusingmethodslikerollingaverages,weightedaverages,orexponentialsmoothingtoaggregatedata over time periods.

Weighted Aggregation: Assign different weights to datafromdifferentsourcesbasedontheirreliabilityoraccuracy.Thishelpsingivingmoreimportancetotrustworthysources duringtheaggregationprocess.

Data Collection: Oracle Nodes: Oracles are third-partyservicesorentitiesthatfetchactual-worlddataandtransmit it to smart contracts on the blockchain. Theseoraclesconsistofnodesresponsiblefor collectingdata.

Multiple Sources: Oracles aggregate data from varioussources such as APIs, IoT devices, a web scraping, andotherexternalsystems.ThesesourcesofAPIs,IoTdevices provide the raw data that needs to be transmittedto theblockchain

Filtering Mechanism: Define Filtering Criteria: Clearlydefinecriteriaforfilteringdata.Thiscouldinvolvesettingthresholds,

specifyingacceptableranges,orimplementingrulesbasedonthenatureofthedata.

DataQualityChecks:Beforetransmittingdatatotheblockchain, oracles often employ filtering mechanisms toensure the quality and reliability of the information. Thismayincludecheckingforconsistency,accuracy,andtamperresistance.

UseSmartContracts:Ifoperatingwithinablockchainenvironment,implementsmartcontractstoenforcefiltering rules. Smart contracts canautonomously rejectoraccept data basedonpredefinedconditions.

SourceDataTransmission:SecureTransmissionProtocols:Usesecurecommunicationprotocols(suchas HTTPS) to transmit data between sources and the aggregation system. Encryption ensures the confidentiality and integrity of the transmitted data.

Error Handling: Implement error-handling mechanisms to address issues like data loss during transmission. Consider using checksums or hash functions to verify the integrity of received data.

Smart Contract Interaction: Oracles interact with smart contracts on the blockchain to update the state of the contract with the newly aggregated and filtered data. This interaction is often facilitated through predefined interfaces and protocols

Data Quality Assurance: Data Validation: Before aggregation, validate incoming data to ensure it conforms to expected formats and standards. Reject or flag data that doesn't meet validation criteria. Outlier Detection: Implement algorithms or rules to detect outliers or anomalies in the data. This helps in identifying potentially incorrect or malicious data points during the aggregation process.

Data aggregation, collection, and filtering are essential components of information processing systems, ensuring the reliability and accuracy of the data utilized. The aggregation process involves combining data from various sources to derive meaningful insights.

Data Signature or digital signature

A data signature is a cryptographic mechanism used to verify the authenticity and integrity of data. It involves generating a unique digital signature using a private key that can be verified using the corresponding public key.

Oracle Data Signing Process:Private and Public Keys: Oracles use a pair of cryptographic keys - a private key known only to the oracle and a corresponding public key that is shared with the smart contract or any entity verifying the data.

Signing Data: When an oracle provides data, it signs the data with its private key. This process generates a digital signature unique to that specific set of data.

Data Source Authentication: When an oracle retrieves external data to be used in a smart contract, the data source signs the information with its private key. This creates a digital signature unique to that specific piece of data.

Decentralized Oracle Networks: In decentralized oracle networks, multiple nodes provide data and sign it with their private keys. The consensus among these nodes can be required for the acceptance of data, further enhancing security.digital signatures in blockchain oracles provide a robust mechanism for ensuring data authenticity, integrity, and source accountability. They are a fundamental element in securing the transmission of external data to smart contracts on the blockchain. Cryptographic Hash Functions: Digital signatures often involve the use of cryptographichash functions. Before signing, the data is hashed to produce a fixed-length string ofcharacters. This hash is then signed using the private key, providing a compact representation of the data that maintains its integrity.

DECENTRALISED REPUTATION SYSTEM

Decentralised Reputation System: These are systems designed to track and manage reputation in a decentralized manner, often using blockchain technology. In a decentralized reputation system, users can build and maintain their reputation based on their interactions within a network or platform. This reputation can be used to establish trust, facilitate transactions, or make decisions within the system.

Integration of Decentralized Reputation Systems with Blockchain Oracles: Decentralized reputation systems can leverage blockchain oracles to access external data that influences reputation scores. For example, an oracle can provide information about the completion of a task, the delivery of goods, or the quality of a service. This external data can then be used to update and calculate reputation scores within the decentralized reputation system.By integrating with blockchain oracles, decentralized reputation systems can ensure the accuracy, transparency, and reliability of the data used to assess and manage reputation. For example. There are two users.user1 and user2 , if user1 has interact with user2 and make some online transaction. Now these transaction details and their ratings will be stored in a Database,then user's personal information will be encrypted and stored in the reputation data oracle service Database.From oracale Database service the encrypted data's will be shared through two modes On-chain and Off-chain. Through On-Chain the reputation data's will be stored in Smartcontract, Distruputed ledger and Block chain Nodes(peer2peer).these reputation data's will be calculated and stored as reputation scores to each individuals Id. Through Of-Chain the encrypted data's will be stored in the Off-chain data storage and that data's will be calculated(ex:user'sinteracation will be analyze while doing their transcation,or other activities)and stored as reputation score. These reputation score's will be calculate for every user's and will be display in the database. And these data's can visible other user's so that they can able to engage their activites through reputational scores.

Data Feeding: Blockchain oracles provide decentralized reputation systems with external data relevant to users' interactions or activities. This data can include transaction confirmations, completion of tasks, delivery of goods, quality of services, or any other information that affects reputation scores. Oracles fetch this data from off-chain sources and feed it into smart contracts on the blockchain.

Adding a decentralized reputation system to a smart contract involves several key steps:

Designing the Reputation System: Define the rules and parameters of the reputation system, including how reputation is earned, assessed, and used within the ecosystem. Determine factors that contribute to reputation, such as successful completion of tasks, positive feedback, or other forms of engagement.

Implementing Reputation Logic: Write the logic for tracking, calculating, and updating reputation scores within the smart contract. This logic should define how reputation scores are initialized, incremented or decremented based on user actions, and potentially decayed over time.

Data Structures: Design data structures to store reputation-related information, such as user profiles, reputation scores, feedback data, and any other relevant metadata. Use appropriate data types and data storage mechanisms within the smart contract to efficiently manage reputation data.

User Interactions: Define functions within the smart contract to allow users to interact with the reputation system. These functions may include methods for users to earn reputation, view their own reputation score, provide feedback or ratings to other users, and query reputation-related information.

Transaction Handling: Implement mechanisms to handle transactions related to reputation changes, such as updating reputation scores, recording feedback, and enforcing rules for reputation adjustments. Ensure that these transactions are executed securely and atomically within the smart contract.

UPGRADABLE SMART CONTRACT

Smart Contract

Smart contract are self-executing program that automates the action.The truncation are trackable and irreversible.The action are carried out without any inter intervention of a central authority ,legal system or external force. Example of a Smart Contract :The simplest example of a smart contract is a transaction between a consumer and a business, where a sale is made. The smart contract executes the customer's payment and the business's shipment or transfer of ownership.

Auditing and Monitoring Smart Contracts

Auditing and monitoring are crucial aspects of ensuring the security and effectiveness of smart contracts. These processes help identify vulnerabilities, bugs, and potential exploits before they can be used to compromise the contract and cause financial losses.

Auditing: A thorough analysis of the smart contract code to identify potential security vulnerabilities, coding errors, and inefficiencies.Performed by experienced security professionals using automated tools and manual analysis. Aims to provide a comprehensive report outlining the identified issues, their severity, and recommendations for remediation.

Benefits: Reduces the risk of exploits and financial losses. Improves the overall security and reliability of the smart contract. Increases user trust and confidence in the platform.

Monitoring: Continuously observing the activity of the smart contract once deployed. Aims to identify anomalous behavior, potential attacks, and deviations from expected functionality.

Smart contracts are the backbone of every blockchain,dapps and web3.The

basic rules of EVM Once a contract is deployed, it cannot be changed. Instead, an upgradable

smart contract uses a special proxy pattern. The latter involves deploying proxy contracts and

implementation contracts PROXY CONTRACT

A proxy contract is a contract which delegates calls to another contract. To interact with the actual contract you have to go through the proxy, and the proxy knows which contract to delegate the call to (the target). A proxy pattern is used when you want upgradability for your contracts. This way the proxy contract stays immutable, but you can deploy a new contract behind the proxy contract - simply change the target address inside the proxy contract

Proxy Pattern: Upgradable contracts typically use a proxy pattern where a proxy contract forwards function calls to an underlying implementation contract. This separation allows for upgrading the logic without changing the state. Logic and Data Separation: The contract's logic is stored in a separate contract, often referred to as the implementation contract. The data, on the other hand, is stored in the proxy contract. This ensures that state variables are preserved during logic upgrades. Transparent Upgradeability: Transparent upgradeability means that users and interacting contracts are unaware that an upgrade has occurred. The proxy contract forwards calls to the new logic seamlessly, providing a smooth transition. Initialization Logic: Many upgradable contracts include an initialization function that is only called once during deployment. This function sets initial values and configurations for the contract.Security Considerations: Security is paramount in upgradable contracts. Careful consideration is given to potential vulnerabilities during upgrades, and access controls are often implemented to restrict upgrade privileges need of proxy contract.In order to provide upgradeability and access control to the contract logic, proxy contracts are a design pattern used in smart contract development. With proxy contracts, the interface of the contract is kept constant, but the way the contract logic is implemented can be updated or modified without changing the contract's address or state. Due to the immutability of smart contracts once they are deployed to the blockchain, there is a need for upgradeability.

A deployed contract cannot be changed, and any bugs or weaknesses in the logic of the contract cannot be fixed without deploying a new contract. This can be problematic for contracts that require frequent updates or modifications, such as contracts used for managing token sales or decentralized applications (dApps)

Proxy Patterns to create upgradeable contracts in solidity:

Transparent Proxy pattern (TPP)

Universal Upgradeable Proxy Standard (UUPS) – EIP- 1822 b

Beacon Proxy Pattern

Diamond Proxy Pattern – EIP-2535

CHAPTER3

RESULT ANALYSIS

After thorough evaluation and comparison with other algorithms,the raft is the most optimal choice.Raft gives simplicity,clarity,and efficiency to achieve consensus in Oracle Blockchain.It is ledger based approach and ensure rapid decision-making if the node fails.The Adaption of Raft,helpfull to enhance the integrity,efficiency,and trustworthiness of their distributed ledger.

Oracle Blockchain, after thorough evaluation and comparison of consensus algorithms, the Raft consensus emerges as the optimal choice among others. Raft offersunparalleled simplicity, clarity, and efficiency in achieving distributed consensus within the Oracle Blockchain network. Its straightforward leader-based approach and streamlined leader election process ensure rapid decision-making and seamless continuity, even in the face of node failures or network partitions.Through the adoption of Raft consensus in Oracle Blockchain, organizations can leverage its inherent advantages to enhance the integrity, efficiency, and trustworthiness of their distributed system.

Data Integrity and Authenticity: The use of digital signatures ensures the integrity and authenticity of the data provided by oracles. The Smart contracts can confidently rely on the information without worrying about tampering during transmission.

Reduced Risks of Manipulation:The implementation of digital signatures, especially when combined with decentralized consensus mechanisms and reputation systems, significantly reduces the risks of malicious actors manipulating or providing false data.

Enhanced Smart Contract Security:Smart contracts utilizing data from secure oracles are more resilient to attacks. The verification of digital signatures helps prevent unauthorized or tampered data from impacting the execution of smart contracts.

Trust in Decentralized Applications (DApps): Decentralized Applications that rely on external data, such as decentralized finance (DeFi) applications, can gain trust among users and stakeholders due to the increased security and reliability of the integrated oracle system.

Reliable Decentralized Oracle Networks: The digital signature mechanisms contribute to the establishment of reliable and decentralized oracle networks. Nodes providing data are accountable, and their reputation is a key factor in the overall trustworthiness of the oracle system.

Tamper-Evident Data: In case of any attempts to tamper with the data during transmission, the digital signatures act as a tamper-evident mechanism. The verification process would fail, signaling potential issues.

Privacy Preservation: Depending on the implementation, the use of zero-knowledge proofs and other privacy-preserving techniques alongside digital signatures can enhance privacy in data transactions, ensuring that sensitive information is not exposed.

Regulatory Compliance: Implementing secure digital signature mechanisms aligns with regulatory requirements for data integrity and security. This can be crucial, especially in industries where compliance with standards and regulations is a priority.

Reputation System: The reputation system, supported by digital signatures, allows for efficient identification and exclusion of unreliable oracles. This contributes to the overall quality and reliability of the aggregated data.

Global Accessibility: The decentralized and secure nature of the oracle system makes it globally accessible. Users across different regions can interact with smart contracts with confidence in the integrity of the data.

Incentivized Participation: A reputation system with associated penalties and rewards incentivizes honest participation in the oracle network. Nodes are encouraged to provide accurate data to maintain a positive reputation.

CHAPTER4

REFERENCES

[1]The Methods are refered from the Oracle blockchain blogs.In this blog,the articlewaswritten by Baohua yang.
[2]Database creation, php wasre feredfrom “Oracle.com” and “w3shools.com”.The AuditingImmunebytes.com

[3]Upgradeble Smart contract –openzippeline,moralis websites articles.

[4]Smith, J. (2020). Blockchain Basics.

[5]Doe, J., & Johnson, K. (2019). Securing Blockchain Oracles

[6] Smith, J. (2021). Secure Blockchain Oracle Implementation. GitHub.

[7]“Blockchain .com” for blockchain

[8]Liu, B.; Szalachowski, P.; Zhou, J. A First Look into DeFi Oracles. arXiv 2020.

[9]Caldarelli, G. Real-world blockchain applications under the lens of the oracle problem. Asystematic literature review.

[10]Marrakech, Morocco, 2020. Egberts, A. The Oracle Problem—An Analysis of how Blockchain Oracles Undermine the Advantages of Decentralized Ledger Systems.Availableonline:https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3382343 (accessed on 4 June 2022).

[11]Albizri, A.; Appelbaum, D. Trust but Verify: The Oracle Paradox of Blockchain Smart Contracts Trust but Verify: The Oracle Paradox of Smart Contracts. J. Inf. Syst. 2021.

[12]Antonopoulos, A.M.; Woods, G. Mastering Ethereum—Building Smart Contracts and DAPPS; O’Reilly: Tokyo, Japan, 2018.

[13]Al-Breiki, H.; Rehman, M.H.U.; Salah, K.; Svetinovic, D. Trustworthy Blockchain Oracles: Review, Comparison, and Open Research Challenges. IEEE Access 2020.