A Trustworthy and Consistent Blockchain Oracle Scheme for Industrial Internet of Things

Peng Liu, Youquan Xian, Chuanjian Yao, Peng Wang, Li-e Wang, and Xianxian Li

***Abstract*—A blockchain provides decentralization and trustlessness features for the Industrial Internet of Things (IIoT), which expands the application scenarios of IIoT. To address the problem that blockchains cannot actively obtain off-chain data, the blockchain oracle is proposed as a bridge between the blockchain and external data. However, the existing oracle schemes make it difficult to solve the problem of low quality of service caused by frequent data changes and heterogeneous devices in IIoT, and the current oracle node selection schemes are difficult to balance security and quality of service. To tackle these problems, this paper proposes a secure and reliable oracle scheme that can obtain high-quality off-chain data. Specifically, we first design an oracle node selection algorithm based on a Verifiable Random Function (VRF) and reputation mechanism to securely select high-quality nodes. Second, we propose a data filtering algorithm based on a sliding window to further improve the consistency of the collected data. We verify the security of the proposed scheme through security analysis. The experimental results show that the proposed scheme can effectively select high-quality nodes, reduce data differences, and improve the quality of service of the oracle. In the oracle network with malicious nodes accounting for 10%, the data accuracy rate is increased by about 4%, and the data variance is reduced by about 45% on average.**

***Index Terms*—IIoT, blockchain, oracle.**

I. Introduction

BLOCKCHAIN is a distributed technology for recording and storing data. It has gained significant attention in recent years due to its technical features, such as decentralization, trustlessness, and traceability. Based on these features, blockchain can act as a trusted intermediary between enterprises in IIoT, creating a trust base for different enterprises and promoting cross-domain collaborative manufacturing [1], [2]. However, blockchain is a deterministic and closed system, incapable of actively obtaining external data. To resolve this limitation, blockchain oracle[1] is proposed as the bridge between blockchain and external data [3].

However, the features of IIoT, such as rapid data changes, network, and device heterogeneity, pose challenges to the quality of service of the oracle. For example, when the smart contract needs to obtain real-time sensor data in IIoT, due to the existence of the above problems, the data obtained by different nodes may vary greatly, which brings difficulties to the data aggregation process and affects the service quality of the oracle. To improve the service quality of the oracle, some research schemes apply the reputation mechanism to the oracle and improve the quality and credibility of the obtained data by encouraging and selecting high-quality nodes. ChainLink [4] is one of the classic representatives. It evaluates the service quality and credibility of the oracle node by establishing a reputation mechanism. The service provider with a higher reputation will receive considerable benefits, thereby improving the service quality of the oracle. Taghavi et al. [5] used reinforcement learning to score nodes and select high-quality nodes to complete tasks to improve service quality. However, since the non-anonymous node selection process in the above schemes is open and transparent on the blockchain, it may be subject to malicious attacks (such as Sybil attack [6], Targeted attacks [7], etc.) and return incorrect data, endangering the security of the system. Therefore, to ensure security in the node selection process, a node selection algorithm based on VRF is proposed to ensure anonymity security in the node selection process with its unpredictable but verifiable characteristics [8], [9]. However, the ensuing problem is a completely random node selection scheme, which is difficult to guarantee the quality of service of the oracle. Therefore, a key issue is how to design a trustworthy oracle scheme to improve the quality of service of the selected nodes while ensuring the anonymous security of the node selection process, and improving the data consistency of the oracle in complex scenarios such as IIoT. Although extensive research has been conducted on blockchain oracles, no study has been conducted to completely solve the above problems.

Manuscript received 10 January 2024; revised 22 April 2024; accepted 5 May 2024. Date of publication 10 May 2024; date of current version 16 October 2024. The research was supported in part by the National Natural Science Foundation of China (Nos.62166004, U21A20474, 62262003), the Guangxi Science and Technology Major Project (No.AA22068070), the Basic Ability Enhancement Program for Young and Middle-aged Teachers of Guangxi (No.2022KY0057, 2023KY0062), Innovation Project of Guangxi Graduate Education (Nos. XYCBZ2024025), the Key Lab of Education Blockchain and Intelligent Technology, the Center for Applied Mathematics of Guangxi, the Guangxi "Bagui Scholar" Teams for Innovation and Research Project, the Guangxi Talent Highland Project of Big Data Intelligence and Application, the Guangxi Collaborative Center of Multisource Information Integration and Intelligent Processing. The associate editor coordinating the review of this article and approving it for publication was M. Tornatore. *(Corresponding author: Xianxian Li.)*

The authors are with the Key Laboratory of Education Blockchain and Intelligent Technology, Ministry of Education, and the Guangxi Key Lab of Multisource Information Mining and Security, Guangxi Normal University, Guilin 541004, China (e-mail: liupeng@gxnu.edu.cn; xianyouquan@stu.gxnu.edu.cn; yaochuanjian@stu.gxnu.edu.cn; wangp@gxnu.edu.cn; wanglie@gxnu.edu.cn; lixx@gxnu.edu.cn).

Digital Object Identifier 10.1109/TNSM.2024.3399837

[1]The official website of Ethereum gives a more detailed explanation of the oracle: https://ethereum.org/en/developers/docs/oracles/.

Considering the above issues, this paper proposes a blockchain oracle scheme for IIoT that considers security, credibility, and quality of service. Firstly, we design a node selection algorithm that combines a reputation mechanism and a VRF to select nodes with a high reputation secretly. Then, we propose a data filtering algorithm based on a sliding window to improve the consistency and efficiency of data collection and improve the service quality of the system. We believe that the scheme is not only suitable for the IIoT but also can be used as a general framework for the oracle, providing new ideas for improving the service quality of the blockchain oracle.

The main contributions of this paper are summarized as follows.

- We design a node selection algorithm that combines reputation mechanisms and VRF to address the tradeoff between node selection security and service quality. This algorithm ensures the anonymity and randomness of node selection while improving the service quality of selected oracle nodes, thus enhancing the security and service quality of the oracle network.
- To address the issue of low-quality real-time data obtained from heterogeneous oracle nodes, we design a data filtering algorithm based on sliding windows. It reduces the time difference and variance of the data obtained between nodes, incentivizes oracle nodes to obtain data from the data source as quickly as possible, and improves the consistency and efficiency of data acquisition.
- The security of the proposed scheme is verified through security analysis, and simulation experiments demonstrate that our scheme can effectively improve the security and service quality of the oracle network. When there is a 10% malicious node occupancy rate in the network, our proposed scheme increases the data accuracy by approximately 4% and reduces the average data variance by about 45%, compared to conventional schemes.

The remaining parts of this paper are organized as follows. Section II presents some related work that is relevant to our research. Section III introduces the necessary background knowledge, such as oracle and verifiable random functions. Section IV introduces the system structure and process of the proposed oracle scheme. Section V introduces the details of related algorithms. In Section VI, experimental results demonstrate the significant advantages of our proposed oracle scheme in terms of service quality, security, and robustness. Section VII provides a security analysis of the proposed scheme. Finally, Section VIII concludes the paper.

II. Related Work

The following will introduce the existing decentralized oracle schemes.

A. Voting-Based Oracle

Augur [10] is the first proposed decentralized oracle prediction market platform for Ethereum. It distributes data request events to all participants in the market, and most people agree that the answer is rewarded, otherwise, it is punished. Astraea [11] divides participants into three categories: submitters, voters, or verifiers. Under the assumption of rational people, the honest behavior of each participant is the Nash equilibrium solution of the system. On this basis, Cai et al. [12], [13] jointly determined the reward of voters through a lightweight scoring mechanism and a nonlinear voting weight scaling mechanism to avoid herding [14]. Although the voting-based method is easy to implement, it requires determining results and a long voting period, which is not suitable for returning data sources with frequent result changes, such as real-time sensor data in IIoT.

B. Hardware-Based Oracle

Trusted Execution Environment (TEE) can protect the core code of the program from interference by other malicious programs, while avoiding data leakage [15]. Zhang et al. [16] designed the Town crier oracle scheme, which performs authentication and data request processing in TEE to ensure the security and credibility of acquired data. Based on the Town crier, Woo et al. [17] improved the availability of oracle networks in the event of a single point of failure or the emergence of malicious nodes through the Byzantine Fault Tolerant (BFT) consensus mechanism. Liu et al. [18] provided trusted vaccine anticounterfeiting data for the blockchain through a low-cost Microcontroller Unit (MCU) that supports TEE. Although the concept of TEE is promising, in the actual industrial IoT scenario, heterogeneous devices are difficult to guarantee support for TEE, and there are already many attacks on TEE [15].

C. Cryptography-Based Oracle

The oracle schemes based on cryptography can be roughly divided into two categories: Transport Layer Security (TLS) and threshold signature. The representative scheme of TLS is DECO [19], which is based on zero-knowledge proof and TLS, allowing the oracle node to prove that the data comes from the specified Web data source without relying on trusted hardware. The threshold signature scheme can ensure that the data is not disclosed before aggregation is complete. Only by collecting the signature fragments that meet the threshold can an effective signature be generated to prevent the "freeloading" problem [4]. DOS Network [8] and Lin et al. [9] first select a set of nodes through a VRF to obtain data, and then sign the data with a threshold signature algorithm. Similarly, Manoj et al. [20] obtained trusted agricultural data for the blockchain using a threshold secret-sharing scheme. The above scheme can provide security trust from cryptography for data, but TLS needs a trusted certificate authority, which is contrary to the concept of blockchain decentralization. In addition, the low reliability of devices in the heterogeneous Internet of Things will also lead to a decrease in the availability of threshold signatures, and it is difficult for threshold signatures to reach a consensus on frequently changing data [21].

D. Reputation-Based Oracle

Although the first three methods have been extensively studied, they are difficult to deal with rapidly changing data

TABLE I
COMPARISON OF RELEVANT LITERATURE

Literature	Anonymous security	Quality of service
[10], [11], [13], [16], [18]–[20], [26]	✗	✗
[4], [5], [17], [22]–[24]	✗	✓
[8], [9]	✓	✗
Ours	✓	✓

and heterogeneous network devices in the IIoT. The reputation mechanism selects and motivates nodes with high reputations to perform tasks by evaluating nodes. Its fewer constraints make it easier to adapt to the complex and changing environment of the IIoT. ChainLink [4] evaluates the service capability and credibility of the oracle node by establishing a reputation mechanism. Service providers with higher reputations receive significant benefits, ensuring high availability and data authenticity. For the lazy behavior of nodes, Du et al. [22], based on an auction mechanism, improved the enthusiasm of individual feedback data through more reasonable incentives to meet the specific delay constraints of smart contracts. Taghavi et al. [5] used reinforcement learning to score nodes and select high-quality nodes to complete tasks to improve service quality. In addition, there has been a lot of research on reputation mechanisms in blockchain. For example, Beh-Raft-Chain [23] and RepuCoin [24] improve the security and efficiency of blockchain by considering the reputation of nodes when selecting consensus nodes. However, the non-anonymous node selection process, whether based on reputation-weighted or random, can easily lead to node collusion or becoming the target of malicious attacks. Moreover, the Matthew effect [25] makes it difficult for new nodes to compete with the original nodes, resulting in fewer new nodes willing to join, which is not conducive to the improvement of oracle service quality. Goel et al. [26] proved that random node selection can effectively prevent collusion among nodes, under the reasonable assumption that miners always implement smart contracts honestly (otherwise they risk not receiving block rewards). Although the VRF-based node selection algorithm [8], [9] can reduce the risk of nodes being predicted or exposed to targeted attacks in the selection process, the quality of the selected nodes cannot be guaranteed.

Although blockchain oracles have been extensively studied, no study has addressed the tradeoff between security and service quality in oracle node selection. Table I analyzes the existing research from the perspective of anonymous security and quality of service. In addition, few studies have explored the relationship between the heterogeneity of oracle nodes and networks and data quality. Therefore, this study aims to ensure the anonymity and randomness of the oracle node selection process while selecting high-quality nodes and to reduce the problem of low data quality caused by the heterogeneity of nodes and networks.

III. PRELIMINARIES

A. Oracle

Smart contracts and blockchains are similar to a closed system that cannot access external information, which means that interactions are limited to the data available on the blockchain. This is an open practical problem, known as the oracle problem, which is defined as how to transmit real-world data to the blockchain [3].

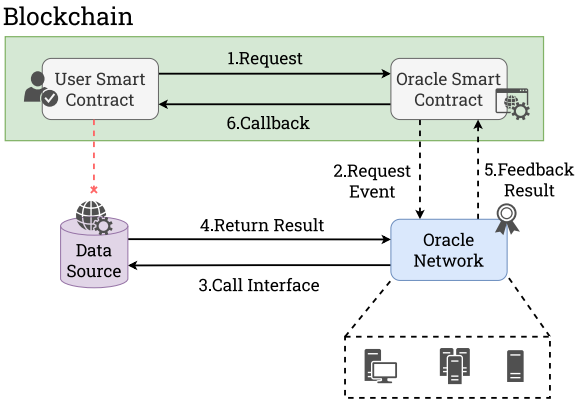


Fig. 1. Overview of oracle system.

The main steps of the solution are illustrated in Fig. 1: 1) The user contract initiates an on-chain request by calling the function of the oracle contract, and then the oracle contract records the event for the oracle according to the user contract's request; 2) After the request event is recorded in the blockchain event log through the consensus, any node in the oracle network listens for request events from the oracle contract; 3) The oracle node requests data or executes operations from third-party services based on the request event; 4) All nodes in the oracle network that have completed the request processing aggregate the obtained results through a specified aggregation mechanism to obtain a final feedback result; 5) The oracle network uploads the final feedback result to the oracle contract; 6) After obtaining the feedback result, the oracle contract returns it to the user contract in the form of a callback.

It should be noted that oracle contracts and networks are typically deployed and maintained by their initiators, such as the Chainlink Foundation. Smart contracts, which necessitate external data, referred to as user contracts, merely need to invoke the interfaces offered by oracle contracts to obtain trusted external data. User contracts are designed and deployed on the blockchain by individual developers or service providers for users.

In the context of IIoT, an oracle has a wide range of potential applications. For instance, an insurance smart contract could use data from the IoT to conduct investigations and gather evidence. This could involve verifying whether flights that purchased flight insurance arrived on time or if the warehouse's smart door lock was locked at the time of default. Therefore, designing a trusted oracle for the existing problems in IIoT is of practical value.

B. Verifiable Random Function

Verifiable Random Function (VRF) [27] is a type of pseudorandom function that can generate corresponding pseudorandom numbers and non-interactive proofs based on data input, and anyone can verify the correctness of the random numbers through the proof. VRF can be represented as a triad of algorithms with polynomial-time complexity $(VRF_{Setup}, VRF_{Generate}, VRF_{Verify})$.

$(sk, pk) \leftarrow VRF_{Setup}()$ generates a private key *sk* and its corresponding verification public key $pk = g^{sk}$, where g is a generator element of the cyclic group; $(y, \pi) \leftarrow VRF_{Generate}(sk, x)$ means using the private key *sk* to encrypt any message x to obtain a pseudorandom string y and its non-interactive proof π; $0/1 \leftarrow VRF_{Verify}(pk, x, y, \pi)$ means using the verification public key *pk* to verify the effectiveness of the proof π, that is, whether y is the pseudorandom string obtained by encrypting the message *x* using *sk*.

For any given key pair (*sk*, *pk*) and input *x*, VRF guarantees that there is no other valid output y' and proof π' that can make VRF_{Verify} valid. In addition, VRF also has unpredictability, that is, its output is unknown to anyone before the private key *sk* is made public. By combining this feature with existing distributed threshold signature schemes, a decentralized random number generation algorithm can be constructed, providing a reliable source of random numbers for probabilistic dependent distributed systems [8]. Furthermore, using VRF to select consensus nodes in the blockchain can effectively reduce targeted attacks after the node identity is revealed [28].

IV. System Design

In this section, we provide a detailed description of the proposed oracle scheme. The goal of this scheme is to provide trusted, high-quality off-chain data for the IIoT blockchain and maintain the trust foundation for cross-domain collaboration among IIoT participants.

A. System Structure

As shown in Fig. 2, the proposed oracle solution in this paper consists of four parts, on-chain oracle contract, data aggregation module, data collection module, and node selection algorithm. The on-chain oracle contract is responsible for processing requests from user contracts and recording corresponding request events. The aggregation module can process the data returned by the oracle according to user-defined aggregation rules to produce a final result. The data collection module is responsible for collecting and verifying the results submitted by the oracle. The node selection algorithm is responsible for secretly selecting a subset of available oracle nodes from the IIoT network to process specific data request events.

1) Oracle Contract: The on-chain oracle contract not only addresses requests from user contracts but also undertakes tasks such as generating request events, message delivery, and managing oracle nodes. In this paper, the oracle contract primarily comprises the following four sub-contracts:

- *Registration Contract:* A participant (with a blockchain account/wallet) needs to pledge tokens at the contract to join the oracle network and become an oracle node to obtain task rewards.[2] After successfully joining, the public key corresponding to the node account will be recorded in the registration contract, and other participants can query the public key from the contract to verify the identity of the node publishing data. In addition, when a node commits malicious acts, the registration contract can confiscate its deposit and use it to reward other honest nodes.
- *Message Contract:* As the oracle task initiator, the user contract needs to be implemented by calling the function of the message contract when requesting data from the oracle. Once the call is successful, the message contract will generate and record the corresponding data request event. The off-chain oracle nodes will continuously listen for request events from the message contract, request data from the specified data source, and feedback on the final result of the message contract. After verifying the results from the oracle network, the callback function of the user contract is invoked to return the results, enabling the user contract to perform subsequent computations after successfully obtaining the data.
- *Payment Contract:* Similar to commercial projects such as ChainLink [4], when a user contract needs to invoke an oracle contract to obtain the required data, it needs to pay a certain fee to the oracle service. The fee is partly used to reward the oracle node that honestly feeds data, and the other part is used to pay for the cost of running the oracle contract and the maintenance overhead of supporting the oracle network. Whenever the request event is processed and the data is successfully feeds, the payment contract will pay the corresponding service fee to the oracle node that provides the correct data this time. The final source of funds actually comes from the blockchain account that invokes the user contract.
- *Reputation Contract:* The reputation of an oracle node is calculated by considering its historical behavioral data, including the average response time to events, the accuracy of feedback data, and the total number of requests processed. The reputation contract will record the above data and calculate the reputation value of the node according to specific calculation rules. Fast and accurate data feedback will increase the reputation value of the oracle node, while the opposite may cause its reputation to decline. A higher reputation in the node selection module can help the oracle node increase its probability of being selected. Considering both oracle performance and data quality, the reputation value R_i of any oracle node O_i is represented as:

$$R_i = (logS_i)\left(\frac{\alpha}{T_i} + (1-\alpha) \times A_i\right), \alpha \in (0,1) \quad (1)$$

Here, T_i, A_i, and S_i represent the average response time $\frac{TotalUsedTime}{S_i}$, response data accuracy $\frac{CorrectNumberOfTimes}{S_i}$, and total service times of node O_i $(1 \leq S_i)$, respectively. α is a system hyperparameter used to adjust the preference of the node selection algorithm for nodes. Its increase and decrease represent the weight of improved node performance (faster response time) and data quality (higher correctness) in

[2]It is essentially a common staking-based method. The source of virtual tokens depends on the specific implementation of the blockchain and the incentive mechanisms. Therefore, this article does not discuss their source, but only assumes that participants already hold certain token assets when registering.

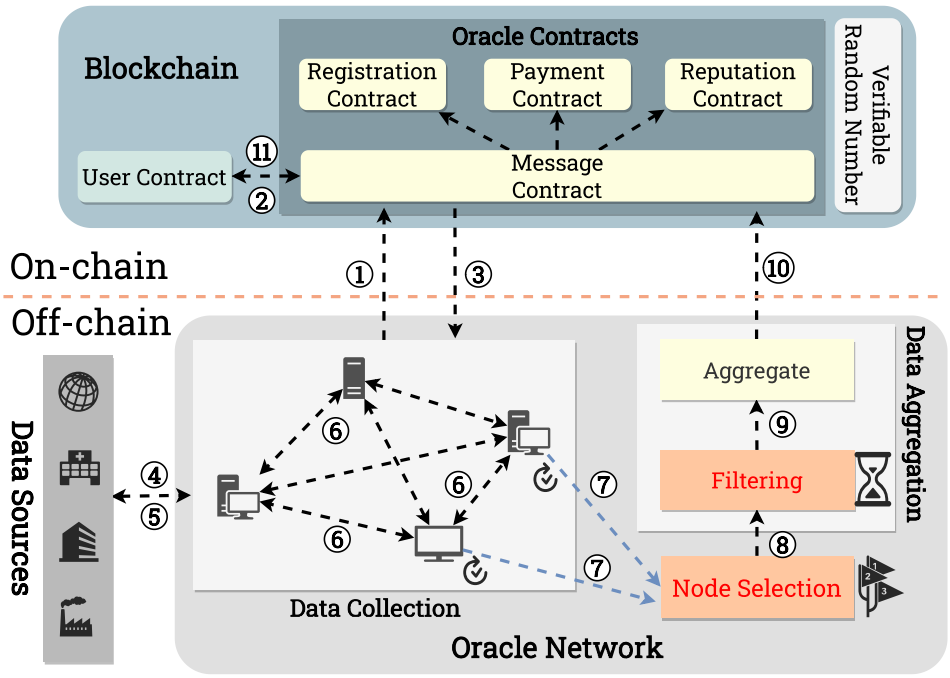


Fig. 2. Overview of our oracle scheme.

the reputation value, respectively. In addition, $logS_i$ can significantly improve the reputation value of nodes in the early stages of joining the network, while preventing unlimited reputation growth with increasing service times S_i and avoiding the Matthew effect. When the reputation drops to a certain value, the tokens pledged by the node in the registration contract are confiscated to encourage oracle nodes to provide higher-quality services.

2) Node Selection Module: The node selection algorithm can implicitly select a subset of registered oracle nodes to process specific data request events on the blockchain. The algorithm is fully decentralized and oracle nodes continuously listen to data request events on the chain and calculate their priority in the request event based on the current verifiable random number. Higher priority means that the node's response to the request event is given priority in the subsequent sorting process. Oracle nodes can decide whether to provide services based on their priority in the event. In addition, due to the inherited properties of the verifiable random function, each node's priority is unpredictable but verifiable to other nodes. That is, before providing feedback, the outside world cannot predict its specific priority in a particular request event; after providing feedback, the outside world can verify whether the published priority is true. These features maintain the anonymity of participating nodes and effectively prevent targeted attacks on the operation of the oracle service. We will discuss the random node selection algorithm and priority calculation method in detail in Section V.

3) Data Collection Module: The data collection module runs on each oracle node and obtains data from the specified data source according to the requirements of the data request event. In addition, nodes continuously collect and verify the feedback results of data request events broadcasted by other nodes (including timestamp, proof of data source for data authenticity, submitter's identity, and their priority as the current request event handler), to obtain a qualified result set. Its main function is to prevent the "freeloading" problem. In the solution to the "freeloading" problem, this paper adopts the two-phase threshold signature scheme based on ChainLink. The biggest feature of this scheme is that all feedback results are initially submitted in ciphertext form. Only when the feedback results of the threshold t oracle nodes for the same request event are aggregated and signed into an effective signature, all participating nodes will publicly reveal their private keys to decrypt and disclose the feedback results. This mechanism effectively prevents cheating behavior by nodes trying to reduce data acquisition costs by copying feedback data from other nodes.

4) Data Aggregation Module: The data aggregation module aggregates the feedback results from the oracle nodes according to the rules defined by the user and returns them to the oracle contract. Different aggregation strategies need to be used for different data and business needs. For example, for data with frequent fluctuations in a short period, the median or mean is usually used as the final result. For data with longer change cycles, it is usually accepted after undergoing consistency checks. This module may be implemented on-chain as a smart contract or off-chain using consensus protocols, depending on the specific application scenario, Fig. 2 illustrates the off-chain aggregation scenario. Before aggregation, two steps need to be completed: 1) Sorting all feedback results according to the priority of the corresponding nodes and selecting the results from the top t nodes with the highest priority; 2) Using a sliding time window-based data filtering algorithm to filter out data with a more concentrated

time distribution from the data selected in the previous step, to improve the quality of subsequent aggregation results. We will discuss the data filtering algorithm in detail in Section V-B.

B. System Flow

The oracle workflow is as shown in Fig. 2:

1) The participant pledges a certain of tokens to the registration contract and uploads its public key pk_i to join the oracle network to become an oracle node O_i, pk_i will be its unique identifier. After successful registration, the reputation contract will assign an initial reputation value R_i to the node, indicating that the node will begin to have a probability of being selected in the subsequent node selection process.
2) A user contract can send a data request q to the oracle contract after paying a service fee for a reward in the payment contract. The oracle contract records the data request event $E = (q, d, f, t, w)$ on the blockchain. Here, q can serve as the unique identifier of E, d represents the set of data sources specified by the user contract, f represents the reward that the user contract needs to pay to all oracle nodes upon a successful request, t represents the minimum number of nodes required to process the request, and w represents the maximum width of the time window (i.e., the maximum time difference between feedback result timestamps).
3) Upon listening about a new request event E on the blockchain, any oracle node O_i calculates its priority $L_{q,i}$ based on its private key sk_i, reputation value R_i, the latest verifiable random number ξ_r known in the current blockchain, and the unique identifier q contained in the request event. It then decides whether to participate in the service for the request event E. Note that the node selection process is not mandatory, and nodes can keep their priorities confidential without any impact. If the priority $L_{q,i}$ of a node is relatively high, i.e., there is a greater probability that it will provide feedback data for the request and receive corresponding rewards, the node will be more inclined to participate in the service.
4) Assuming that node O_i decides to respond to the request event E, it must select a specific data source from set d based on the result of taking the modulus of its priority $L_{q,i}$ with the size $|d|$ of the data source set, and request data. Based on the verifiability of the proposed node priority, other nodes can confirm whether O_i honestly obtained the data from the specified data source by verifying O_i's public $L_{q,i}$ and data source proof in the feedback result.
5) The data source processes the request from node O_i and returns real-time data $x_{q,i}$ as the result. According to the method proposed in [19], O_i needs to generate a proof $p_{q,i}$ for the communication process, proving that $x_{q,i}$ is obtained from the specified data source at time $ts_{q,i}$.
6) Node O_i broadcasts the result $I_{q,i} = (m_{q,i}, pk_{q,i}, \eta_{q,i}, p_{q,i}, ts_{q,i}, L_{q,i})$ to the oracle network, where $(sk_{q,i}, pk_{q,i})$ is the temporary key pair generated by O_i for this submission. $\eta_{q,i}$ represents the partial signature of q obtained by threshold signing using $sk_{q,i}$. $m_{q,i}$ represents the ciphertext obtained by partially encrypting $x_{q,i}$ using $sk_{q,i}$.
7) Nodes participating in the service will continue to collect the results broadcast by other nodes and verify $p_{q,i}$ until t $\eta_{q,i}$ from different nodes are aggregated by any node at least. If any node O_j collects feedback results that meet the threshold requirement of t, it can compute a group signature η_q and make it public. Once node O_i detects a valid group signature η_q, it will make its temporary private key $sk_{q,i}$ public so that other nodes can decrypt $m_{q,i}$ to obtain $x_{q,i}$. After decrypting all feedback results $I_{q,i}$, node O_j sends the corresponding $K_{q,i} = (x_{q,i}, p_{q,i}, ts_{q,i}, L_{q,i})$ to the node selection module.
8) The node selection module receives the results sent from several data collection nodes. It will perform verification and deduplication on these data and sort them according to priority $L_{q,i}$ to obtain feedback results from the t nodes with the highest priority.
9) Based on a sliding time window data filtering algorithm, the data with timestamp $ts_{q,i}$ selected by the t nodes can be further filtered to obtain the final data K_q that can be aggregated. After executing the aggregation logic on K_q, the data aggregation module writes the aggregation results and the information of the selected oracle nodes to the message contract.
10) After receiving and validating the feedback from the data aggregation module, the message contract invokes the reputation contract to update information such as the nodes' average response time T_i, response data accuracy A_i, and total service times S_i. If the data returned by a node is excluded during data filtering or identified as an outlier during the aggregation process, it will be considered as returning erroneous data, leading to a decrease in its response data accuracy. Finally, the payment contract will reward the oracle nodes that provide correct data.
11) Finally, the message contract feeds the aggregation results back to the user contract by callback, so that the user contract can complete the subsequent calculation process.

V. The Proposed Scheme

A. Node Selection

After the participants register as oracle nodes with the registration contract, they can become potential candidates for processing subsequent data requests. As mentioned earlier, the node selection algorithm in this paper is mainly implemented based on priority sorting, and each oracle node has a different priority for different oracle events. Therefore, the core of the node selection algorithm is how to design a random priority calculation method. It should take into account the credibility of the node in the calculation process, and allow nodes with higher credibility to have a certain probability of receiving a higher priority while maintaining randomness. Furthermore,

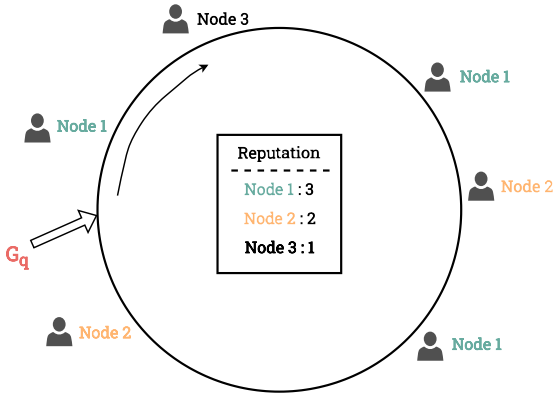


Fig. 3. Diagram of oracle node selection algorithm.

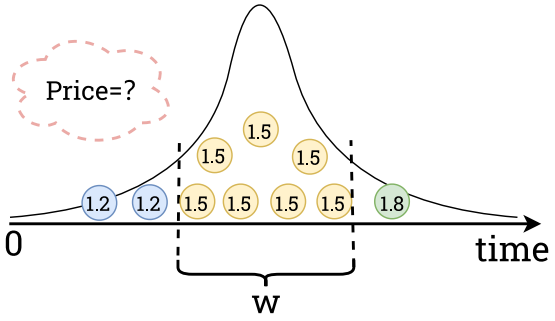


Fig. 4. Diagram of data filtering algorithm.

the priority of a node is unknown before it is published, but can be quickly verified after it is published to ensure the anonymity of the node during the work process.

This paper's random node selection algorithm is based on the idea of the consistent hashing function [29]. Given the hash function H, its range is organized into a virtual circle, called a hash ring, in a clockwise direction. As shown in Fig. 3, the hash function can map request events and all oracle nodes to different positions on the hash ring, and then oracle nodes determine their priority based on the distance between their position on the hash ring and the position of the request event on the hash ring. The closer a node's position is to the request event's clockwise distance, the higher its priority. In short, only the top t nodes with the closest hash values need to be selected based on their distance on the hash ring.

Given a hash function H, assuming the latest verifiable random number maintained in the current blockchain is ξ_r. When node O_i receives a new data request event E from the blockchain, it first hashes the unique identifier q and the random number ξ_r of E to obtain the corresponding position of E on the hash ring:

$$G_q = H(q \| \xi_r) \tag{2}$$

Since both q and ξ_r are public, all participants can compute G_q. As reputation needs to be taken into account when randomly selecting nodes, in our scheme, the number of positions a node occupies on the hash ring is determined based on its reputation. For example, when the reputation of O_i is $R_i = 10$, it will be mapped to 10 different positions on the hash ring, and it only needs to select the position with the smallest clockwise distance to G_q to calculate the priority for this request. The higher the reputation of a node, the more positions it occupies on the hash ring, and the greater the probability of obtaining a high priority. When there are more oracle nodes, the probability of selecting high-reputation nodes can be increased while ensuring randomness. When O_i needs to calculate its position on the hash ring, it signs q and ξ_r with the signature function *Sign* and its private key sk_i to obtain:

$$g_{q,i} = Sign(q \| \xi_r \| sk_i) \tag{3}$$

Since ξ_r cannot be predicted in advance until the moment it is generated [8], O_i cannot be predicted to compute the high priority $g_{q,i}$. Then, node O_i needs to query its reputation value R_i from the reputation contract on the chain and use $g_{q,i}$ as the generator to compute the set of all positions mapped to the hash ring for this request:

$$Y_{q,i} = \{H\left(g_{q,i}^k\right) \mid 1 \le k \le \lceil R_i \rceil\} \tag{4}$$

Finally, calculate the shortest distance from all positions in $Y_{q,i}$ to G_q. The shorter the distance, the higher the priority of the node in processing the request event E. Therefore, the priority of node O_i can be expressed as:

$$L_{q,i} = \frac{1}{min(\mid G_q - y \mid)}, y \in Y_{q,i} \tag{5}$$

B. Data Filtering

Due to the heterogeneity of nodes and the differences in geographical location in the IIoT, the delay in obtaining data varies among different nodes. In scenarios where data frequently changes, data consistency may be poor. The data filtering algorithm in this paper focuses on filtering the data before aggregation. It filters to obtain a portion of the data with a more concentrated temporal distribution according to the width of the time window specified by the oracle contract. Assuming that the data aggregation module selects $\{K_{q,i} \mid 0 \le i < t\}$ and corresponding timestamps $\{ts_{q,i} \mid 0 \le i < t\}$ based on the priority of the feedback results submitted by nodes, and the time window width limit is w. The filtered data set K_q should meet the following three requirements: 1) The range of time stamps of the data is no greater than w, to avoid large differences between two feedback results; 2) The number of data contained in the time window should be as many as possible; 3) When the amount of data is the same, the variance of the data timestamps should be as small as possible, making the overall data acquisition time closer to the expected value, i.e., the distribution is more concentrated.

As shown in Fig. 4, the selected oracle nodes get different results from the data source at different times. We use a time window of width w to move from left to right, count the variance of the data timestamp in all time intervals, and filter out the data in the time interval with the least variance. The specific implementation of the reference algorithm is Algorithm 1.

The data filtering algorithm can motivate the oracle nodes to obtain data from the data source as quickly as possible. Assume that the delay of all nodes to the data source follows a normal distribution $N(\mu, \sigma^2)$ with a mean of μ and a standard

deviation of σ [30]. That is, the delay of a small number of nodes to the data source may be lower or higher, but according to the concentration of the normal distribution function, the delay of the majority of nodes is concentrated in the interval closer to the average delay μ. Therefore, to improve the probability of getting rewards, i.e., the probability of getting feedback data selected by the sliding time window, all nodes should get data from the data source and get feedback as soon as possible. Suppose a node deliberately delays processing the request, then even if it has a higher priority in this service, its feedback result is likely to be excluded from the sliding time window due to a larger time variance. In addition, if w is set too small, it is difficult to ensure the security and credibility of the system. At this point, we can set a lower threshold ζ to reduce this risk. Tasks with aggregation numbers below the threshold will fail and increase w by half before restarting.

The above algorithm is relatively easy to implement when using the on-chain aggregation method, only the corresponding calculation process needs to be implemented in the smart contract. However, this method has obvious efficiency problems. The aggregation mechanism is mainly divided into on-chain and off-chain methods. In the on-chain aggregation method, the nodes write the results directly into the blockchain through the oracle contract after obtaining the results. When the aggregation condition is met, the oracle contract aggregates all the results according to the aggregation rules. When the off-chain aggregation method is used, the final result is obtained and fed back to the oracle contract by the oracle nodes through a special off-chain protocol. The difference between on-chain and off-chain aggregation is that the former is easy to implement, but will incur large on-chain computation and storage costs, while the latter is the opposite. If the number of participating oracle nodes is large, the submission process of on-chain aggregation may generate many transactions, which may consume a lot of time in the blockchain consensus phase and significantly affect the real-time performance of the oracle. Therefore, the off-chain aggregation method is more suitable for IIoT scenarios.

To run the above node selection algorithm and data filtering algorithm in a distributed manner during the off-chain aggregation process, this paper proposes an implementation solution based on the reputation distributed consensus algorithm Raft [23], as shown in Fig. 5. First, it is necessary to form a temporary distributed consensus network by combining t oracle nodes that participate in the data request event processing. When a node O_i publishes its feedback result, the following situations may exist: 1) Other nodes have not returned results yet, then O_i creates a new consensus network by broadcasting information and electing itself as the leader in the network; 2) There is already a consensus network for this service in the network, then O_i joins as a following node. Suppose a consensus network already contains t nodes, when O_i tries to join the network, if its priority $L_{q,i}$ is higher than the lowest priority member O_j in the current network, it can join successfully, and O_j will be removed from the network, otherwise, the joining fails. Note that the leader node in the temporary consensus network is not always fixed, and the existing leader node may be removed due to attacks or low priority, which may lead to a new round of leader node elections.

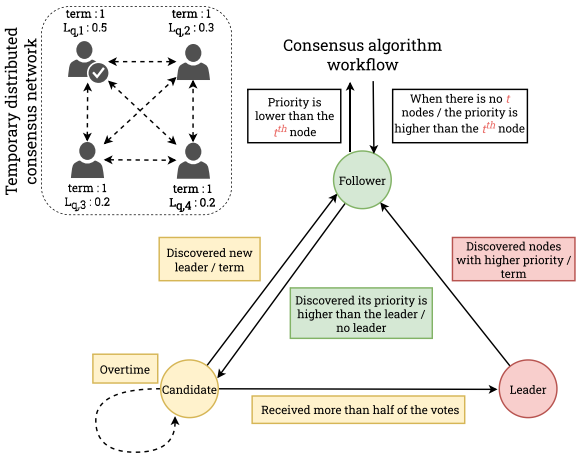


Fig. 5. Diagram of distributed network construction.

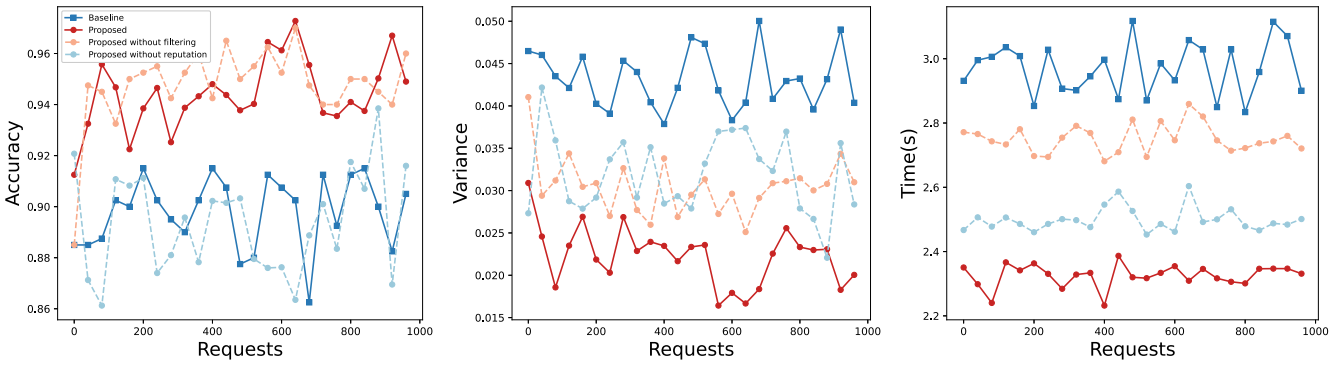
Algorithm 1: Data Filtering Algorithm Based on Sliding Window

Input: Feedback results of nodes $K_{q,0}, \ldots, K_{q,t-1}$, Timestamp of data $ts_{q,0}, \ldots, ts_{q,t-1}$, Time window width w

Output: Contains the subset of results with the most data and the smallest time variance K_q

```
l,r = 0,0;
maxnum, minvar = 1,0;
K_q = {K_{q,0}};
while r ≤ t − 1 do
    if ts_{q,r} − ts_{q,l} ≤ w then
        num = r − l + 1;
        avg = Σ_{i=l}^{r} ts_{q,i}/num;
        var = Σ_{i=l}^{r} (ts_{q,i} − avg)^2/num;
        if maxnum ≤ num or (num == maxnum and var < minvar) then
            maxnum, minvar = num, var;
            K_q = {K_{q,i} | l ≤ i ≤ r};
        r = r + 1;
    else
        l = l + 1;
return K_q;
```

Secondly, all nodes in the temporary consensus network need to go through multiple rounds of consensus to complete the off-chain aggregation process: 1) In the first round, the consensus is reached with t feedback results $\{K_{q,i} \mid 0 \leq i < t\}$, and the consensus result is used as the input of Algorithm 1 to obtain K_q; 2) In the second round, the consensus is reached with the K_q obtained by each node, and the consensus result is used as the input of the aggregation strategy to obtain the aggregation result. Finally, the current leader node is



(a) Accuracy of data (b) Variance of data (c) Time of obtaining data

Fig. 6. The accuracy, variance, and time of the data obtained by each scheme when there are 10% of the malicious nodes.

responsible for submitting the aggregation result to the on-chain oracle contract, and all nodes in the temporary consensus network exit the off-chain aggregation process.

VI. Performance Evaluation

In this section, we verify the performance and security advantages of the proposed solution through simulations. Specifically, we first examine the performance of different solutions, then investigate the performance of the proposed solution under various proportions of malicious nodes, and finally analyze the reputation update and parameter effects on the solution's performance, demonstrating our research results.

We simulated a smart contract system using the Golang language that continuously publishes data requests and receives feedback data. We use distributed oracles to provide data collection services for blockchains, where the delay of the oracle nodes is randomly extracted from $N(2,0.6)$ with a limit of $[0,+\infty)$sec. At the same time, to simulate real-time sensor data, a real-time data source is designed to generate a floating point number $P \in (0, 1)$ per second. Malicious nodes are set to upload a random number $K \in (0, 1)$ instead of collecting data P. In an oracle network with 100 oracle nodes and 10% malicious nodes, we run 1000 oracle tasks. Baseline is an implementation based on schemes like the DOS Network [8], [9]. It uses VRF to randomly select nodes for tasks and has no data filtering module. We compared our scheme with the baseline and with schemes that remove some of the sub-algorithms (node selection with reputation weighting and data filtering). The parameters of the experiment are shown in the Table II.

To ensure anonymous security in node selection, our scheme requires all participants to participate in the judgment at the beginning of the task, which increases the computational complexity. In the case of off-chain aggregation, compared with the baseline, our scheme mainly increases the priority of $O(Nlog(N))$ for node selection and the computational complexity of $O(t)$ for sliding time windows in data filtering. However, compared with the long communication time in heterogeneous networks, we believe that the computation time is negligible.

TABLE II
Experimental Parameter Settings

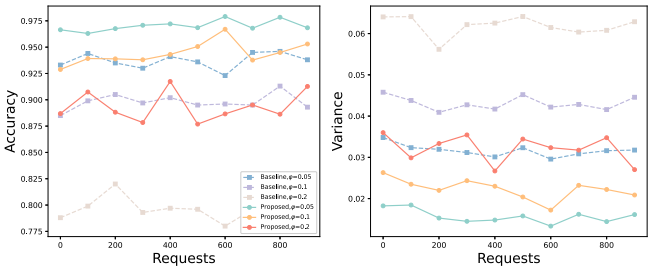
Parameter	Meaning	Value
N	Total number of oracle nodes	100
δ	Proportion of malicious nodes	0.1
t	Number of nodes selected per round	10
w	Length of time window	1s
α	The preference weight of reputation	0.5
ζ	The minimum threshold number of aggregation	1

A. Quality of Service

To demonstrate the performance advantage of our proposed approach over the baseline, we analyze the accuracy, variance, and time of the data obtained by our approach, the baseline, and the approach with some sub-algorithms removed (reputation-weighted node selection and data filtering), as shown in Fig. 6. Variance is often used to measure the degree of deviation of a random variable from its expected value. Therefore, we use the variance of the returned data to reflect the consistency of the data, with lower variance indicating higher consistency.

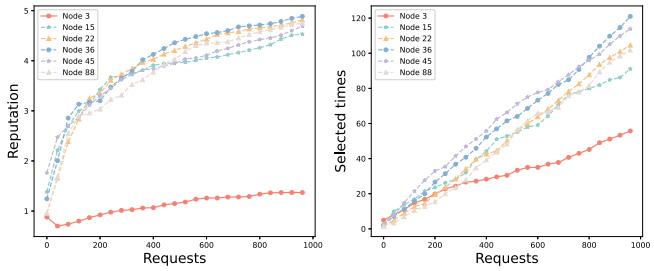
Fig. 6(a) shows the accuracy of data obtained by different approaches. It can be observed that our approach has an accuracy of approximately 4% higher than the baseline. Moreover, we find that the data is polarized, and the approach with reputation-weighted node selection has a significant improvement over the one without. The impact of the data filtering module on accuracy is not significant, but this does not mean that our data filtering module is meaningless. We will analyze it in the following.

Fig. 6(b) shows the variance of data obtained by different approaches. It can be found that the variance of our approach is much lower than that of the baseline. Moreover, our approach still has a lower variance than the conventional approach even with some sub-algorithms (node selection and data filtering) removed. This indicates that both of our proposed sub-algorithms can effectively improve data quality, but they work in different mechanisms. The data filtering algorithm filters more concentrated data by sliding time windows, while the reputation-weighted node selection algorithm filters out malicious nodes to reduce error data and improve data quality. They can work together to provide higher-quality data for oracles.



(a) Accuracy of data (b) Variance of data

Fig. 7. The accuracy and variance of the data obtained by each scheme when there are different proportions φ of [5%, 10%, 20%] of the malicious nodes.



(a) Reputation of node (b) Selection time of node

Fig. 8. Change of reputation and selection times of randomly selected 6 nodes.

Fig. 6(c) shows the time of data obtained by different approaches. Both our approach and the approach with some sub-algorithms removed (node selection and data filtering) can effectively reduce the response time of oracle tasks. The node selection algorithm considers the node response time in the selection process and selects nodes with faster response for the task to participate in the task. At the same time, the filtering algorithm also eliminates the slow response results of a small number of users and encourages nodes to respond quickly.

The three experiments depicted in Fig. 6 illustrate the performance superiority of our approach compared to the baseline. This suggests that our designed reputation and data filtering algorithms are effective and operate in complementary directions. Specifically, the reputation algorithm notably enhances data accuracy, whereas the data filtering algorithm more effectively reduces data acquisition time.

B. Security

To demonstrate the security of our proposed approach, we conduct 1000 oracle tasks with φ set to [5%, 10%, 20%], respectively. As shown in Fig. 7, we analyze the differences between our proposed approach and the baseline in terms of data accuracy and variance.

Fig. 7(a) depicts the accuracy of data acquisition for different approaches. It can be observed that our approach consistently outperforms the baseline in terms of accuracy, even under the same φ. As the φ increases, the accuracy of data acquisition decreases, but our approach shows a slower decline. This indicates that our approach can provide high-accuracy data in oracle networks with varying proportions of malicious nodes, and the advantage becomes more pronounced as the number of malicious nodes increases.

Fig. 7(b) shows the variance of data acquisition for different approaches. Similar to the accuracy, our approach consistently maintains lower data variance than the baseline. As the φ increases, the variance of data acquisition for all approaches increases, but our approach exhibits a significantly slower growth rate. Even when 20% of the nodes are malicious, our approach can still maintain a similar level of variance as the baseline when only 5% of the nodes are malicious.

The two experiments in Fig. 7 demonstrate that our proposed approach can provide trustworthy high-quality off-chain real-time data for blockchain, even as the number of malicious nodes in the oracle network increases.

C. Reputation Update and Node Selection

To demonstrate the effectiveness of the reputation mechanism and node selection algorithm proposed in this paper, we randomly select six oracle nodes (Node3 is a malicious node) to observe their reputation value changes and the number of nodes selected.

Fig. 8(a) shows the changes in the reputation values of the nodes. It can be observed that, except for the malicious node (Node3), the reputation values of the other nodes improved as they honestly participated in the tasks. The reputation values increase rapidly in the initial stages and then gradually level off as the number of iterations increases, which is consistent with our goal of avoiding the Matthew effect. For malicious nodes, the rate of increase in reputation values is much slower compared to honest nodes.

Fig. 8(b) presents the frequency of node selections. Similar to the reputation update pattern, the malicious node is rarely selected due to its lower reputation value. We can observe that the malicious node still has some participation opportunities since we ensure that $1 \leq R_i$ in our simulation. However, in a real production environment, we recommend imposing stricter constraints on malicious nodes to ensure the security and credibility of the oracle network.

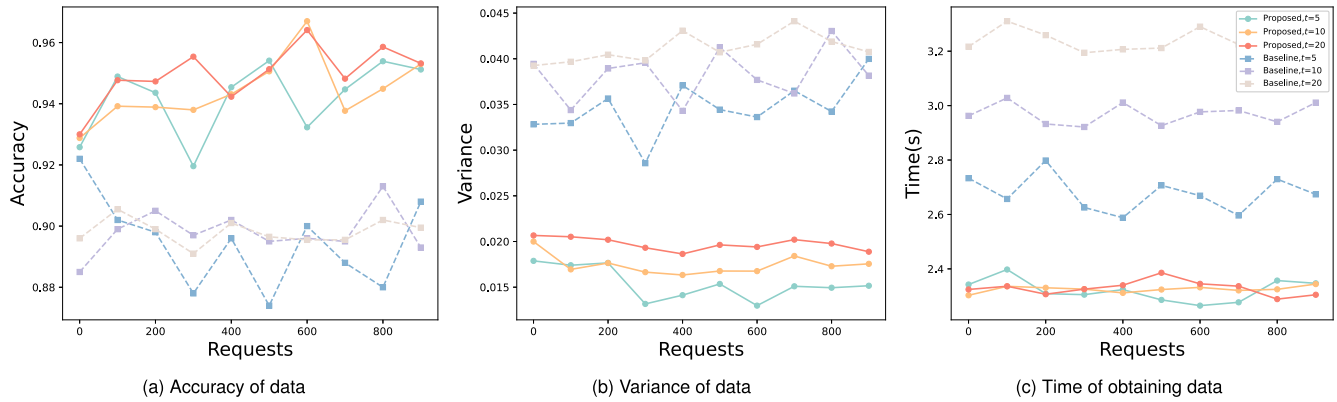
D. Scalability

To explore the scalability of this scheme, we adjust the number of selected nodes t and analyze the impact of t increase on the performance of the oracle.

Fig. 9(a) shows the relationship between the node's data accuracy and t. We can observe that the impact of t on accuracy is not significant.

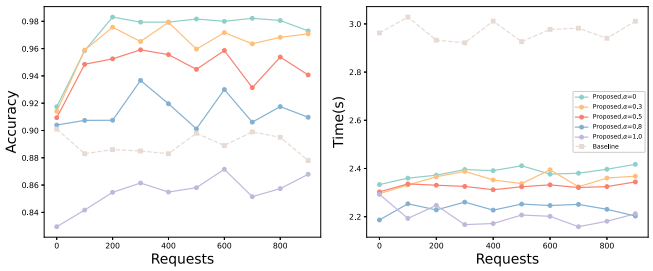
Fig. 9(b) shows the relationship between the node's data variance and t. In contrast to accuracy, t has a significant impact on the data variance. The smaller the value of t, the smaller the variance of the obtained data, which indicates higher data quality. However, since t is the threshold for returning data, a smaller t may result in lower security. Therefore, t should be set based on the actual scenario's requirements.

Fig. 9(c) shows the relationship between the node's data acquisition time and t. We can see that as t changes, the time taken for baseline data acquisition increases rapidly, while our proposed approach can maintain a relatively stable time. We have also analyzed the impact of our node selection and data filtering process on data acquisition time. It is because our



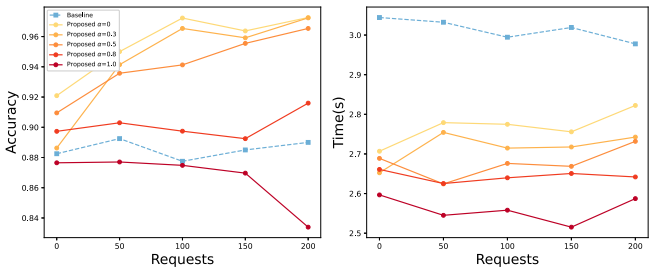
(a) Accuracy of data (b) Variance of data (c) Time of obtaining data

Fig. 9. The relationship between the number of selected nodes t and the accuracy, variance, and time of data acquisition.



(a) Accuracy of data (b) Time of obtaining data

Fig. 10. The relationship between the hyperparameter α and the accuracy and time of the obtained data.



(a) Accuracy of data (b) Time of obtaining data

Fig. 11. Availability on a real blockchain.

node selection algorithm selects nodes that can obtain data more quickly, and our data filtering algorithm filters data that is more concentrated in time.

The above experiments show that the proposed scheme has good scalability. As the number of selected nodes t increases, the accuracy and response time of the obtained data can remain relatively stable.

E. Robustness

Fig. 10(a) shows the relationship between the data accuracy of the node and the hyperparameter α. We can see that α has a significant impact on data accuracy and a smaller α leads to higher data accuracy. This is because α controls the balance between data accuracy and reputation computation time, and a smaller α increases the probability of selecting nodes with high data accuracy. However, we also find that when α is set to 1.0, which means that the reputation calculation only considers the time to obtain data rather than the data accuracy, the data accuracy of our scheme is worse than the baseline.

Fig. 10(b) presents the relationship between the node's data acquisition time and α. As α increases, the time taken by our scheme to acquire data decreases gradually. Regardless of the changes in α, our scheme can achieve a lower response time than the baseline. This is consistent with our original design intention and previous verification, as the data filtering algorithm can reduce the data acquisition time.

Finally, to verify the availability of the proposed scheme on the real blockchain, we use Truffle Suite[3] to generate an oracle network composed of 100 active oracle nodes (with Ethereum Wallet) and deploy smart contracts on the local Ethereum blockchain. We perform the same experiment as Fig. 10, and the results are shown in Fig. 11. The results are similar to Fig. 10, which shows that the proposed scheme is still available in real blockchain scenarios.

VII. Security Analysis

In this section, we present a security analysis of the proposed scheme.

In this paper, we primarily discuss three security issues faced by oracle nodes: Sybil attack, Targeted attacks, and the scenario where malicious nodes return false data. Regarding the situation of malicious nodes, we analyze it in Section VI-B. For the remaining Sybil attack and Targeted attacks, we analyze their security from the perspectives of cryptographic security and economic security.

A. Sybil Attack

The Sybil attack refers to the practice of an organization or individual creating or using multiple accounts (false identities) in an attempt to manipulate or control a P2P network system [6]. In distributed oracle systems, Sybil attack often involves a single person running multiple oracle nodes in an attempt to disrupt the normal operation of the service or gain more centralized power.

[3]It's an Ethereum smart contract development framework that makes it easier for developers to build, test, and deploy smart contracts. https://trufflesuite.com/.

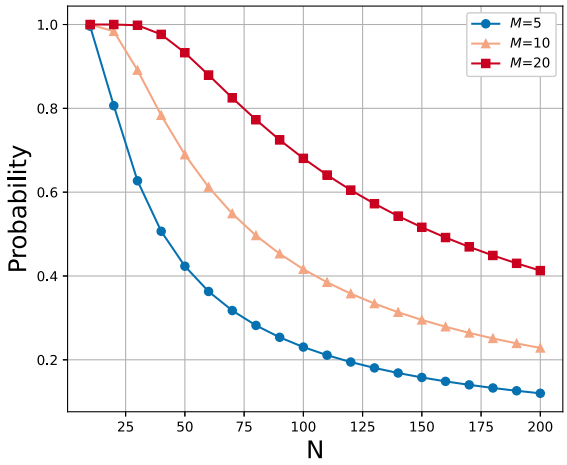


Fig. 12. The influence of M and N on the probability of malicious nodes being selected.

a) Definitions and Assumptions: Assuming an oracle network consisting of N nodes, where each node must pledge S of tokens to participate in the network. Furthermore, the total amount of tokens controlled by attackers X_m is less than the total token supply in the network X. Considering the extensive number of nodes N in IIoT environments, the number of malicious nodes attackers can create is $M = \frac{X_m}{S}$, significantly smaller than the total number of network nodes N, satisfying $N \gg M$. In each round, the system randomly selects R nodes to participate in tasks, with each node's probability of being selected given by $\{w_1, \ldots, w_N\}$.

b) Analysis: Under the assumption of nodes being selected completely at random, where the probability of all nodes being selected is equal $\{w_1 = w_2 = \cdots = w_N\}$, this event can be simplified to a classical probability model. In this scenario, the probability P_r of at least one malicious node being selected can be expressed as:

$$P_r = 1 - \frac{\binom{N-M}{R}}{\binom{N}{R}} \tag{6}$$

Among them,

$$\binom{N-M}{R} = \frac{(N-M)!}{R!(N-M-R)!} \tag{7}$$

$$\binom{N}{R} = \frac{N!}{R!(N-R)!} \tag{8}$$

Therefore, the probability P_r can be further expressed as:

$$P_r = 1 - \frac{(N-M)!(N-R)!}{N!(N-M-R)!} \tag{9}$$

As shown in Fig. 12, by fixing R at 5 and varying M within [5, 10, 20], as the total number of network nodes N increases, we observe a gradual decrease in the probability P_r of malicious nodes being selected, and the smaller the value of M, the faster this probability decreases. It indicates that the extensive number of nodes N in IIoT, along with setting the token stake quantity S, effectively reduces the probability of selecting malicious nodes, thereby enhancing network security.

Furthermore, considering the integration of a node reputation mechanism, the probability w_i of malicious nodes being selected to participate in tasks is lower than the average selection probability $\frac{\sum_{i=1}^{N} w_i}{N}$. Consequently, the actual probability P_r of malicious nodes being selected will be lower than the theoretical value mentioned earlier. This further underscores the effectiveness of token staking and reputation-based node selection strategies in reducing the risk of Sybil attack.

B. Targeted Attacks

Targeted attacks are not a specific type of attack, but refer to the continuous attack launched by attackers against a specific target [7]. Malicious users may attempt to disrupt the normal operation of the oracle service by launching Targeted attacks on the nodes in the distributed oracle.

a) Definitions and Assumptions: Assuming a total number of network nodes as N, with a committee consisting of t nodes operating in the off-chain. Suppose attackers can control K nodes to achieve the following objectives: 1) causing network service disruption; and 2) tampering with system data. Meanwhile, it is assumed that the cryptographic schemes based on VRF, threshold signatures, and data interaction protocols [19] rely on the discrete logarithm problem, which is computationally hard under current computational capabilities.

b) Analysis: If attackers attempt to disrupt the off-chain network using methods like Distributed Denial of Service (DDoS) to prevent it from reaching the threshold t and cease service, they would need to control at least $N - t + 1$ nodes. Therefore, as long as the number of nodes controlled by attackers $K < N - t + 1$, the system can still operate normally. In the case of off-chain aggregation, referring to the security features of Raft [31], the system can still operate normally when $K < \lfloor \frac{t}{2} \rfloor + 1$.

If an attacker attempts to tamper with the system data in the case of on-chain aggregated, due to the unpredictability of the verifiable random number ξ_r in $g_{q,i} = Sign(q \| \xi_r \| sk_i)$, as well as the collision resistance and private key secrecy of the signature algorithm, the attacker cannot directly target specific nodes and can only randomly select K nodes for attack. The probability P_r of selecting malicious nodes is referenced from Eq. (9). However, even if the attacker tampers with the data through controlled nodes, according to the data interaction protocol [19], any tampered data cannot pass the verification process of the proof $p_{q,i}$ by the smart contract. For off-chain aggregation, the security of the system depends on the consensus protocol adopted by the committee. Replacing Raft with a Byzantine fault-tolerant algorithm can improve security, but efficiency may be compromised.

VIII. Conclusion

This paper proposes a secure and trustworthy oracle scheme to address the challenge of difficult interaction between blockchains and external data in IIoT. We design a novel node selection algorithm based on VRF and reputation mechanism to anonymously select high-quality nodes while improving the quality of off-chain real-time data. Moreover, we propose a sliding window-based data filtering algorithm to improve data consistency and data acquisition efficiency. Through security analysis and simulation experiments, we verify that

the proposed scheme can improve system service quality while maintaining security. In summary, this study provides reliable IIoT data for blockchain applications, facilitating trusted interaction between blockchain and IIoT.

The current study has not delved deeply into aspects such as data validation and user incentives. In our future work, we will focus on the following improvements: 1) Designing a universal method for node anomaly detection to reduce the cost of designing specific detection methods for each task; 2) Exploring more reasonable incentive mechanisms to enhance the participation enthusiasm of various users in the oracle network; 3) Designing corresponding privacy protection methods to safeguard the privacy of IoT devices and their data during transmission and usage processes, etc.

REFERENCES

[1] A. Hazra, M. Adhikari, T. Amgoth, and S. N. Srirama, "A comprehensive survey on interoperability for IIoT: Taxonomy, standards, and future directions," *ACM Comput. Surveys*, vol. 55, no. 1, pp. 1–35, 2021.

[2] R. Huo et al., "A comprehensive survey on blockchain in Industrial Internet of Things: Motivations, research progresses, and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 88–122, 1st Quart., 2022.

[3] A. Pasdar, Y. C. Lee, and Z. Dong, "Connect API with blockchain: A survey on blockchain oracle implementation," *ACM Comput. Surveys*, vol. 55, no. 10, pp. 1–39, 2022.

[4] S. N. Steve Ellis and A. Juels. "Chainlink a decentralized oracle network." 2017. [Online]. Available: https://chain.link

[5] M. Taghavi, J. Bentahar, H. Otrok, and K. Bakhtiyari, "A reinforcement learning model for the reliability of blockchain oracles," *Expert Syst. Appl.*, vol. 214, Mar. 2023, Art. no. 119160.

[6] J. R. Douceur, "The sybil attack," in *Proc. Int. Workshop Peer-to-Peer Syst.*, 2002, pp. 251–260.

[7] G. Sun, M. Dai, J. Sun, and H. Yu, "Voting-based decentralized consensus design for improving the efficiency and security of consortium blockchain," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6257–6272, Apr. 2021.

[8] (DOS Netw., Singapore). *A Decentralized Oracle Service Network to Boost Blockchain Usability with Real World Data and Computation Power*, 2019. [Online]. Available: https://dos.network

[9] Y. Lin et al., "A novel architecture combining oracle with decentralized learning for IIoT," *IEEE Internet Things J.*, vol. 10, no. 5, pp. 3774–3785, Mar. 2023.

[10] J. Peterson, J. Krug, M. Zoltu, A. K. Williams, and S. Alexander, "Augur: A decentralized oracle and prediction market platform," 2015, *arXiv:1501.01042*.

[11] J. Adler, R. Berryhill, A. Veneris, Z. Poulos, N. Veira, and A. Kastania, "Astraea: A decentralized blockchain oracle," in *Proc. IEEE Int. Conf. Internet Things (IThings) Green Comput. Commun. (GreenCom) Cyber, Phys. Soc. Comput. (CPSCom) Smart Data (SmartData)*, 2018, pp. 1145–1152.

[12] Y. Cai, G. Fragkos, E. E. Tsiropoulou, and A. Veneris, "A truth-inducing sybil resistant decentralized blockchain oracle," in *Proc. 2nd Conf. Blockchain Res. Appl. Innovat. Netw. Services (BRAINS)*, 2020, pp. 128–135.

[13] Y. Cai, N. Irtija, E. E. Tsiropoulou, and A. Veneris, "Truthful decentralized blockchain oracles," *Int. J. Netw. Manage.*, vol. 32, no. 2, 2022, Art. no. e2179.

[14] E. C. Galariotis, W. Rong, and S. I. Spyrou, "Herding on fundamental information: A comparative study," *J. Bank. Finan.*, vol. 50, pp. 589–598, Jan. 2015.

[15] P. Jauernig, A.-R. Sadeghi, and E. Stapf, "Trusted execution environments: Properties, applications, and challenges," *IEEE Security Privacy*, vol. 18, no. 2, pp. 56–60, Apr. 2020.

[16] F. Zhang, E. Cecchetti, K. Croman, A. Juels, and E. Shi, "Town crier: An authenticated data feed for smart contracts," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 270–282.

[17] S. Woo, J. Song, and S. Park, "A distributed oracle using Intel SGX for blockchain-based IoT applications," *Sensors*, vol. 20, no. 9, p. 2725, 2020.

[18] C. Liu et al., "Extending on-chain trust to off-chain–trustworthy blockchain data collection using trusted execution environment (TEE)," *IEEE Trans. Comput.*, vol. 71, no. 12, pp. 3268–3280, Dec. 2022.

[19] F. Zhang, D. Maram, H. Malvai, S. Goldfeder, and A. Juels, "Deco: Liberating Web data using decentralized oracles for TLS," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2020, pp. 1919–1938.

[20] T. Manoj, K. Makkithaya, and V. Narendra, "A trusted IoT data sharing and secure oracle based access for agricultural production risk management," *Comput. Electron. Agricult.*, vol. 204, Jan. 2023, Art. no. 107544.

[21] S. Vári-Kakas, O. Poszet, A. M. Pater, E. V. Moisi, and A. Vári-Kakas, "Issues related to the use of blockchains in IoT applications," in *Proc. 16th Int. Conf. Eng. Modern Elect. Syst. (EMES)*, 2021, pp. 1–4.

[22] Y. Du, J. Li, L. Shi, Z. Wang, T. Wang, and Z. Han, "A novel oracle-aided industrial IoT blockchain: Architecture, challenges, and potential solutions," *IEEE Netw.*, vol. 37, no. 3, pp. 8–15, Jun. 2023.

[23] L.-e. Wang, Y. Bai, Q. Jiang, V. C. Leung, W. Cai, and X. Li, "Beh-raft-chain: A behavior-based fast blockchain protocol for complex networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1154–1166, Jun. 2021.

[24] J. Yu, D. Kozhaya, J. Decouchant, and P. Esteves-Verissimo, "Repucoin: Your reputation is your power," *IEEE Trans. Comput.*, vol. 68, no. 8, pp. 1225–1237, Aug. 2019.

[25] R. K. Merton, "The Matthew effect in science: The reward and communication systems of science are considered," *Science*, vol. 159, no. 3810, pp. 56–63, 1968.

[26] N. Goel, C. Van Schreven, A. Filos-Ratsikas, and B. Faltings, "Infochain: A decentralized, trustless and transparent oracle on blockchain," in *Proc. 29th Int. Conf. Int. Joint Conf. Artif. Intell.*, 2021, pp. 4604–4610.

[27] S. Micali, M. Rabin, and S. Vadhan, "Verifiable random functions," in *Proc. 40th Annu. Symp. Found. Comput. Sci. (Cat. No. 99CB37039)*, 1999, pp. 120–130.

[28] L. Chen, Q. Fu, Y. Mu, L. Zeng, F. Rezaeibagha, and M.-S. Hwang, "Blockchain-based random auditor committee for integrity verification," *Future Gener. Comput. Syst.*, vol. 131, pp. 183–193, Jun. 2022.

[29] D. Karger et al., "Web caching with consistent hashing," *Comput. Netw.*, vol. 31, nos. 11–16, pp. 1203–1213, 1999.

[30] C. Fraleigh, F. Tobagi, and C. Diot, "Provisioning IP backbone networks to support latency sensitive traffic," in *Proc. IEEE INFOCOM 22nd Annu. Joint Conf. IEEE Comput. Commun. Soc. (IEEE Cat. No. 03CH37428)*, 2003, pp. 375–385.

[31] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *Proc. USENIX Annu. Tech. Conf. (USENIX ATC 14)*, 2014, pp. 305–319.

Peng Liu received the Ph.D. degree from Beihang University, China, in 2017. He began his academic career as an Assistant Professor with Guangxi Normal University in 2007, subsequently attaining the rank of a full professor in 2022. His current research interests are focused on federated learning and blockchain.

Youquan Xian received the B.E. degree from Beibu Gulf University in 2021. He is currently pursuing the master's degree with Guangxi Normal University. His research interests include blockchain, edge computing, and federated learning.

Chuanjian Yao received the B.E. degree from the Guilin University of Electronic Technology in 2020, and the master's degree from Guangxi Normal University in 2023. His research interests include blockchain, edge computing, and deep reinforcement learning.

Peng Wang received the master's degree from the Guilin University of Technology in 2018. He is currently pursuing the Doctoral degree with Guangxi Normal University. His research interests include blockchain, data fusion, and data security.

Li-e Wang received the M.S. degree from Hunan University, Changsha, in 2007, and the Ph.D. degree from Guangxi Normal University, Guilin, China, in 2022, where she is currently a Professor with the School of Computer Science and Engineering. Her research interests mainly include data privacy, recommendation, distributed system security, and machine learning. She has authored more than 40 refereed papers in these areas. She has served as a reviewer for several high impact research journals and ACM/IEEE flagship conferences.

Xianxian Li received the Ph.D. degree from Beihang University in 2002. He worked as a Professor with Beihang University from 2003 to 2010. He currently serves as a Professor with Guangxi Normal University, Guilin, China. His research interests include information security and machine learning.