Journal of Data Acquisition and Processing
ISSN: 1004-9037
https://sjcjycl.cn/
DOI: 10.5281/zenodo.776766

RELIABILITY ENHANCEMENT MODEL FOR BLOCKCHAIN DATABASE SERVICES

Eesha Mishra
Research Scholar, Dept. of Computer Science & Engineering, Maharishi University of Information Technology, Lucknow, INDIA, Eeshamishra786@gmail.com

Santosh Kumar
Associate Professor, School of Computing Science and Engineering, Galgotias University, Greater Noida, India, Sant7783@hotmail.com

Abstract

Blockchain is a new technology that is enabling more and more vital economic systems. Blockchain oracles are agents that retrieve data from the outside world because the execution environment of blockchain is cut off from it. Oracles are off-chain components that could be sources of failure in entire blockchain-based systems, despite the fact that blockchain is considered to be extremely dependable. It is still unknown whether blockchain oracles are reliable. In this study, we suggest a paradigm for evaluating and classifying the commercially available blockchain oracle mechanisms now in use. Fault Tree Study is the method we utilise for architecture analysis and re-liability modelling of blockchain oracle systems. Weak links that affect the overall dependability of a blockchain-based system can be found by calculating the reliability of oracles methods.

Keywords: Blockchain, Blockchain Database, Reliability, Reliability Analysis

I. INTRODUCTION

A decentralised, unchangeable digital record known as a blockchain stores transaction information among a vast network of nodes. By offering a platform for decentralising trust for data, blockchain has the potential to upend established business models and infrastructure in numerous areas [1-3]. With the help of so-called "smart contracts," most notably in Ethereum1, blockchains can also offer decentralised trust for general computation[1].

Blockchain exchanges and platforms have come under more attacks as a result of their growing importance [4]. Since doing so requires the least amount of work, attacks frequently target a system's weakest link [5]. A total of hundreds of millions of dollars have been lost as a result of these attacks. We anticipate that as blockchain use spreads, it will be employed in both commercially and safety-critical contexts, such as IoT systems and pharmaceutical supply chains. Blockchain-based solutions need to be trustworthy.

In order to include data in the isolated execution environment of a blockchain, a blockchain oracle is a method that collects data from the outside world. The reliability characteristics of blockchains do not apply to blockchain oracles because they are often off-chain components. Due to the special qualities of blockchain, blockchain oracles are required to connect blockchains with the outside world. Because they have restricted access or are temporary sensor data, certain types of external data are intrinsically unable to be independently evaluated by several distant parties. Oracles add this type of data to a blockchain as transactions. The total reliability of a blockchain-based system, however, may be impacted because oracles are likely to be less reliable than blockchain platforms relies on an oracle. To evaluate the total dependability of blockchain-based systems, the reliability of oracle methods must be examined.

In this study, we found current blockchain platforms with industry oracle methods, evaluated them, and compared their dependability strategies. We suggest using fault tree analysis to evaluate the dependability of oracle methods (FTA). The method starts by drawing activity diagrams that represent user requests on external data. These activity diagrams are taken from white papers outlining the oracle mechanisms and converted into an analytical tool called a Fault Tree Diagram (FTD). We determine the oracle mechanisms' reliability and pinpoint any weak points that reduce the overall dependability of blockchain-based systems. Failure potential common causes are examined.

II. BACKGROUND

2.1 Database

Many blockchain-based applications require communication with other external systems. Therefore, the status of those external systems might affect how blockchain transactions are validated. Blockchain oracles deliver the necessary data to the blockchain from other systems, including for usage in smart contracts. Oracles come in five different flavours. 2 Normally, smart contracts can only access data that has already been stored on the blockchain and cannot use additional data in order to retain the deterministic validation of blocks. By capturing external data in transactions on the blockchain and using oracles, for example, communication with the outside world is made feasible.

2.2 Reliability and architecture analysis

Software and system engineers may objectively evaluate hardware, software, and systems in terms of failure probability thanks to reliability analysis [6]. Analysis of potential threats to assets that are economically and strategically essential requires reliability. One of the most used techniques for reliability analysis is FTA. Architecture analysis is the process of determining if a design is suitable for the task at hand. Typically, architecture analysis is used to assess software qualitatively and to weigh the pros and cons of various design choices.

Blockchain platforms have previously been the subject of reliability research, but not the specific parts of blockchain-based systems. Availability, reliability, and integrity are qualities

that contribute to dependability and security [7]. A prior investigation of the availability of Bitcoin and Ethereum [8] discovered that write availability for transactions is poor compared to read availability. This study also discovered that network reordering has an impact on a block's commit time and that it is highly variable.

Wan et al. looked into the characteristics of defects in open source blockchain projects and came up with 10 different categories of bugs [9]. Their findings demonstrate that semantic flaws are the most prevalent runtime bug category, and a similar pattern can be seen in the frequency distribution of bug kinds among blockchain projects. Performance defects require the most time to repair on average, whereas security bugs take the least time.

To anticipate the latency of blockchain-based systems, Yasaweerasinghelage et al. developed an architecture modelling and simulation approach [10]. The presented model can be used to assess design choices made for blockchain-based systems, including the amount of confirmation blocks, which might affect write latency and write availability

2.3 Overall architecture

Figure 2.3.1. depicts a generic overall architecture of various types of oracle mechanisms based on our investigation of existing oracle solutions. A generic oracle mechanism starts with a *requester* creating a *smart contract* (1 in Figure 2.3.1) specifying the

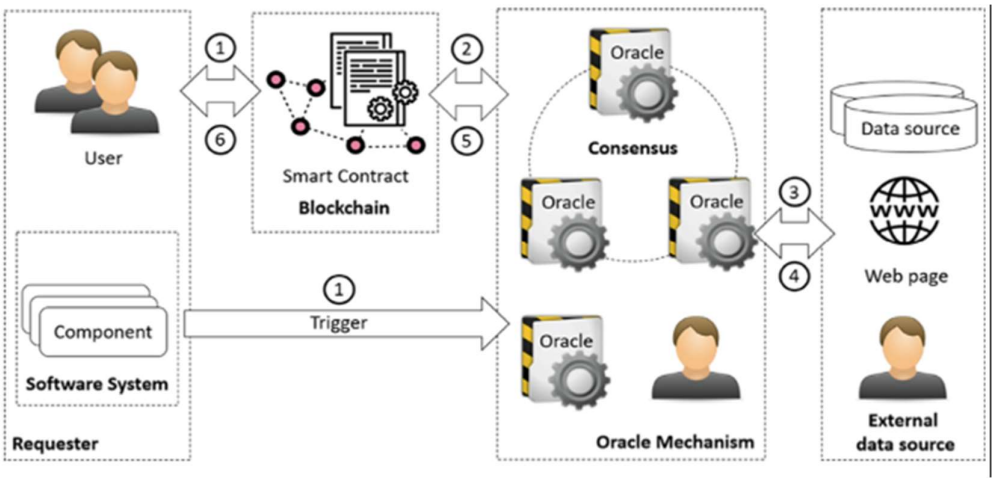

Figure 2.3.1. Blockchain Oracle Database Mechanism Architecture Diagram

Table 2.3.1: Summarized Database Mechanisms

Platform	Database	Consensus	Reliability Features	Compatible Platform	Data Sources	Time interval	Types of Database
Provable	Single	N/A	TLS-Notary Proof	Bitcoin, Ethereum, Corda	Single	Fast	Provable contract
TownCrier	Single	N/A	Intel SGX	Ethereum	Single	Fast	TownCrier Contract
Corda	Single	N/A	Intel SGX	Corda	Single	Fast	Corda code
MS Bletchey	Multiple	N/A	Secure Container, Intel SGX	Azure, AWS, Google	Multiple	Fast	Off-chain code
Chain Link	Multiple	N-of-M multigeniture	Off-chain aggregation, reputation	Bitcoin, Ethereum, Hyperledger	Multiple	Slow	Reporter
Augur	Multiple	Voting	Reputation, Dispute Window, Fork	Ethereum	Multiple	Slow	Voter
Gnosis	Multiple	Voting	Ultimate database & Centralized database	Ethereum	Multiple	Slow	Voter

data necessary to deploy the contract on the blockchain and for it to execute. A user or a software system component can make the request. In some circumstances, the requester might directly trigger the oracle (1 Trigger in Fig. 2.3.1) to add a value to a blockchain that would later be used.

A smart contract's needs can be automatically recognised by centralised oracles (2 in Fig. 2.3. 1). Multiple redundant oracles that offer the same functionality to examine the external state make up a distributed oracle. With a blockchain account, some oracles can manually enter oracle data and sign transactions.

An autonomous oracle will communicate with its external data source, such as a physical sensor or web service, once it has been implemented, to gather the necessary data (3 and 4 in Fig. 2.3.1). When a requester executes a smart contract (5 in Figure 2.3.1), they will be able to see the data that was injected into the blockchain by Oracles (6 in Figure 1).

2.4 Using an oracle database introduces some issues:

• Oracles give the decentralised blockchain system a dependable third party. All the parties involved in the pertinent transactions must have faith in it.

• Unlike oracle data, which is based on an external state, blockchain transactions are immutable.

2.5 Comparison of representative oracle database solutions

We've chosen seven blockchain platforms with oracle mechanisms that are currently in use. Their properties are summarized in Table 2.3.1. Oracle service providers include Provable, TownCrier, and ChainLink, while leaders in the prediction market Augur and Gnosis make use of the potential of blockchain oracles. Provable, The next section discusses this table's columns.

2.5.1 Consensus

Multiple oracles reach a consensus to produce a final answer that is then posted to the blockchain. External data is fetched right into the blockchain for a single oracle mechanism. Different consensus protocols are employed by various platforms to choose the outcome for many distributed oracles.. Multiple oracles in ChainLink utilise a K-out-of-M threshold signature to agree on the answer that will be accepted. To accept a value as the answer, for instance, a 3-out-of-5 signature system requires at least three or more oracles out of five oracles to sign on the same value. Gnosis and Augur, where the oracles are human, both use voting. Anyone with a blockchain account who disagrees with the answer that an oracle gave back to the blockchain can contest the result by reporting another value as a tentative answer by staking their token..

2.5.2. Reliability features

The specific qualities utilised by blockchain systems to ensure the dependability of their oracles are reliability features. For the purpose of obtaining data from external data sources, Provable introduces TLS-Notary3. Data from an HTTPS secure site is supported by a cryptographic proof thanks to TLS-Notary. Intel Software Guard Extensions (SGX) are used by TownCrier and Corda for hardware attestation in order to prevent unwanted access outside of the SGX context. In order to connect Ethereum and HTTPS-enabled websites, TownCrier was created. In Town- Crier, a datagram is a brief piece of data (such as market quotes) sent to the blockchain. For little data and continuously changing data, Corda uses commands. Cryplets from Microsoft Bletchley function similarly to the datagrams of TownCrier. To guarantee a trusted connection for data to be submitted to the smart contract, it makes use of Intel SGX hardware attestation. Multiple oracles can retrieve data from various data sources using ChainLink. Both Augur and Gnosis are marketplaces for predictions. Any user on Augur has the option to dispute an oracle's claimed response within a certain time frame.

On-chain oracle, centralised oracle, and ultimate oracle are the three different oracle configurations found in Gnosis. If any user disputes the reported value, 100 ETH is needed to

activate the ultimate oracle. Gnosis uses ETH because it was created on the Ethereum platform. According to Coinmarketcap, one ETH is worth $408 USD as of August 2018 [4].

2.5.3 Data source(s)

The information required by the requester is gathered by oracles by calling the appropriate data source(s). A data source could be a physical sensor, a component of a system, a static web page, or even human interaction. Even though there are numerous data sources, some information, like the owner of a property, may ultimately originate from a single authorised source. For instance, the local city council's asset registry may be the only source of information regarding city assets that can be trusted.

2.5.4 Time interval

Oracles call the relevant data source to acquire the information the requester has requested (s). A physical sensor, a system component, a static web page, or even human interaction can all serve as data sources. Despite the fact that there are many sources of data, some data, such as the identity of a property's owner, might ultimately come from a single authorised source. For instance, the sole reliable source of information about city assets may be the asset registry maintained by the local city council.

2.5.5 Type of oracles

While MS Bletchley employs many oracles, Provable, TownCrier, and Corda use a single oracle to retrieve data from an external source. Humans are used by ChainLink, Augur, and Gnosis as oracles to provide information to the requester.

3. RELIABILITY ANALYSIS FRAMEWORK

3.1. Approach overview

Figure 5.1.1 depicts our suggested framework's general layout. To comprehend the features and characteristics of the oracle systems, white papers, official technical articles, and official blogs of the chosen blockchain platforms were researched. Table 2.3.1 provides a comparison of the characteristics of seven chosen oracle methods.

Manually created UML activity diagrams (ADs) were created for each of the chosen oracle mechanisms. Another researcher double-checked every generated AD to make sure it accurately reflected the system's reasoning. Any disagreements were spoken through and settled. In Figure 5.1.2, examples of the ADs are displayed.

The ADs were subsequently changed into FTDs. The rules from [11] were applied to convert the ADs. A success tree diagram was created from an AD, and it was then inverted to create an FTD. The AD can only be utilised to create a portion of the FTD because not all probable flaws would be modelled in an AD, as discovered by Tiwari et al. [11]. As a result, to create a

complete FTD, all the system's recognised possible faults were manually mapped to the newly created FTD. Potential flaws were gleaned through pertinent articles, github, and forum posts. sets of minimal cuts (MCS) were subsequently created from the FTDs to determine the accuracy of each oracle method. The shortest set of events necessary for the top event to occur is known as a minimal cut set.

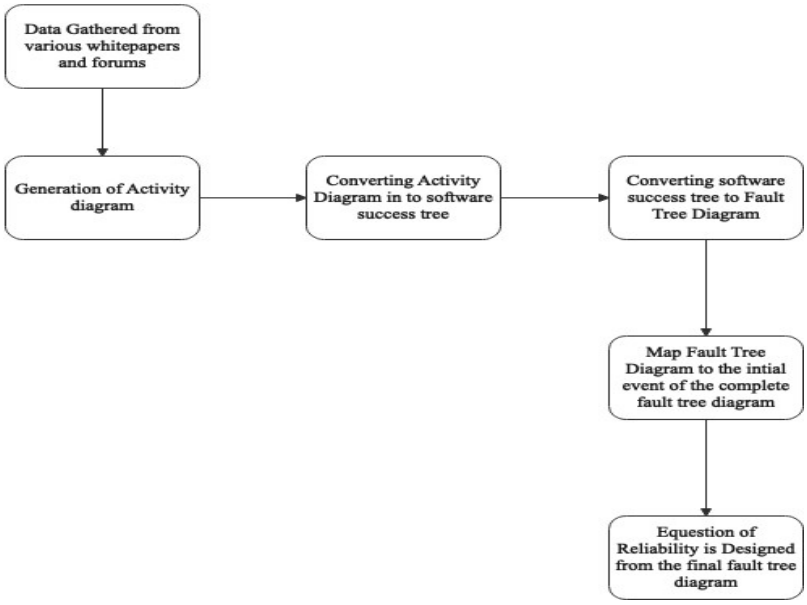

Figure 3.1.1: Overall Block diagram of reliability for blockchain database

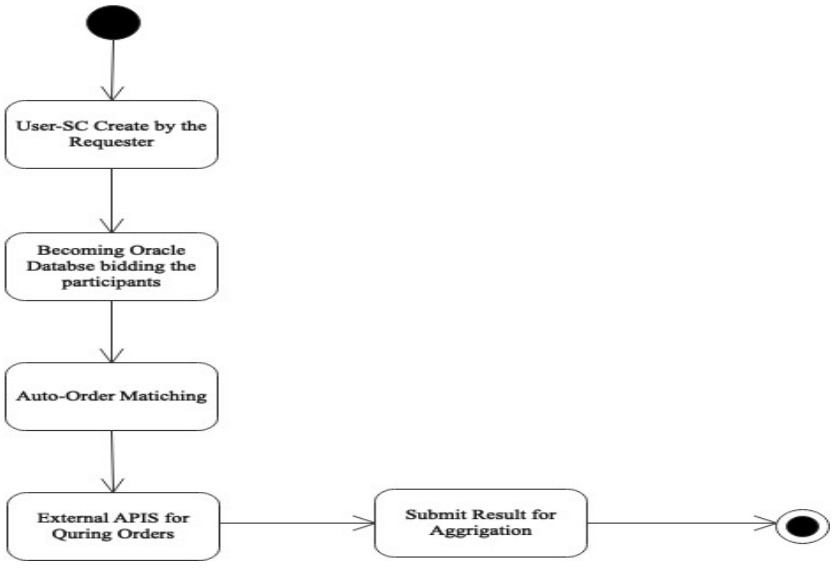

Figure 3.1.2: Activity diagram for ChainLink

Table 3.1.1: Low Level Errors

Error Type		Examples	Failure Rate
Human error	Smart contract error	Greedy Contract	0.003678
		Prodigal Contract	
		Suicidal Contract	
	Server error	Server Hacked	0.001-0.005 (avg-0.003)[7]
		Infrastructure error	
	Simple Task	Chose invalid Data source	0.0005
	Routine Task	Client side error	0.06
	Hard Task	Reporter not reporting	0.1-0.25
		Update new features	

Both qualitative and quantitative analysis were conducted on the FTD. Common causes of failure are identified through qualitative analysis. For quantitative analysis, error rates of the events were used to calculate the reliability of the oracles systems. The error rates of all the events were identified from existing publications. Any unavailable error rate for an event in the FTD was substituted by similar traditional hardware component failure rates. All the events in the FTD were categorized into groups in Table 5.1.1 to determine the applicable error rate for similar events. Sensitivity analysis was also conducted to analyze the impact of the number of oracles on the reliability of the oracle(s) mechanism.

3.2. MODEL GENERATION

3.2.1. Activity diagram generation

ADs are used to depict a single operation's logic, a use case, or the progression of business logic. An AD can be used to visualise how a system behaves dynamically through various activity flows, such as concurrent and parallel activities.

The flow of oracle procedures from a request to ask for external information via the oracle retrieving information off-chain and reporting the information back to the blockchain are represented here by an AD. To enable comparable examination of the oracle processes, the level of detail in the modelling of these processes was maintained across all oracle systems. Compared to decentralised oracle systems, centralised oracle systems have a simpler AD. Fig.5.1.2 displays the ChainLink AD that was developed.

3.2.2. Success tree diagram generation

The development of a Success Tree Diagram (STD) from the AD is the next stage in the method. A STD is a tree diagram that is employed to assess and pinpoint the components

necessary for the expected success. The STD outlines the requirements and components needed to accomplish the top event using a combination of different logic gates and fundamental events. In order to analyse reliability and risk, an STD can be transformed into an FTD. We can see from the AD in Fig. 3 that the ChainLink oracle needs to complete five tasks in order to successfully query external data. We were able to convert Fig. 3 into an STD with five events needed to access the outside data.

3.2.3. Fault tree diagram (FTD)

An FTD is the complement of a STD. FTDs are commonly used in engineering to analyze the potential risks to safety and economically-critical assets. They are also widely used in identifying risks of a software or system. FTDs are generally directed acyclic graphs, where a component's faults are modeled at the leaves of the graph. Logic gates represent how the faults propagate.

The events on the FTD are the complement from the STD, and gates are also swapped, e.g. an AND gate from the STD is changed into an OR gate in the FTD. A limitation of generating a tree diagram from an AD is that it might not cover all lower-level elements identified from forums, gitHub and publications [4-5] that contribute to the top event. These lower-level elements are mapped onto the generated FTD to form the complete FTD.

3.3. RELIABILITY ANALYSIS

Reliability analysis is an essential part of designing, constructing and operating economically-critical technical systems. Various methods and models have been created to support systematic analysis of the reliability and risk of a system and FTDs are one of the most commonly used models. A FTD generated in the previous step can be used to conduct both qualitative and quantitative analysis for reliability of blockchain oracle system. A cut set (CS) is a unique set of events obtained from a FTD that is sufficient to cause the top event to happen. It provides a mechanism for probability calculations and also reveals the critical links in the system design [12]. A minimum CS (MCS) is the CS with minimum number of events that can cause the occurrence of the top event [12].

By using the complete ChainLink FTD in 3.1.2. as an example, we can form the MCS equation (Eq.) for the FTD as below: $T = P1+P2+P3+P4+P5+K/N[P6+P7+P8+P9+P10]$. ChainLink contains 10 single-component minimum cut sets. Any of the lower events from *P1* to *P10* is sufficient to cause the top event *T* to occur. Lower events from *P6* to *P10* are bounded by K-out-of-N gate, hence we applied the reliability Eq. for K-out-of-N to lower events *P6* to *P10*.

To calculate reliability of blockchain oracles using FTD, we show the reliability Eq. for a system and also reliability Eq. for K-out-of-N system.

$$R = \sum_{k}^{n} \frac{n!}{(1)_k\, k!(n-k)!} (e^{-\lambda t})^k (1 - e^{-\lambda t})^{n-k}$$

Eq. 1 is the Equation for *K-out-of-N system*

$$RChainLinkoracle = RP1 \times RP2 \times RPn \ldots \times RP10 \quad (2)$$

Eq. 2 below is obtained from the MCS for the ChainLink FTD By substituting system reliability equation($R = e^{-\lambda t}$) and Eq. 1 into Eq. 2, the complete Eq. can be formed:

$$RP1...P5 = e^{-\lambda P1t} \times e^{-\lambda Pnt} \ldots \times e^{-\lambda P5t}$$

$$RP6...P10 = {}^{\square n}\, n! \; (e^{-\lambda P6t})^{k}(1-e^{-\lambda P6t})^{n-k} \times \ldots {}_{k}\, k!(n-k)!$$

$$\times {}^{\square n}\, n!_{k}\, k!(n-k)! \; (e^{-\lambda P10t})^{k}(1-e^{-\lambda P10t})^{n-k}$$

Failure rates of the components are substituted into the Eq. to calculate the overall reliability of the oracle mechanism. Blockchain platforms and oracle services are too recent to have sufficient historical data for us to empirically calculate the failure rate of the components. Hence, to demonstrate the approach in this paper we have used failure rates reported in the literature for traditional software components. We also draw on some prior research on failure rates of smart contract. The values input to our model are tabulated in Table 3.1.1.

There are three types of major errors, which are *smart contract error, server error*, and *human error*. All issues related to data sources, such as server hacks and infrastructure errors, are grouped under the server failure category[7]. Human-related error can be categorized into three different types [14], including simplest possible task with failure rate of 0.0005, routine task with care needed with failure rate of 0.06 and complicated non-routine task with the failure rate of 0.1–0.25.

IV. RESULT & DISCUSSION

The values of reliability of different oracles are shown in Table 4.1. Based on the values, Augur has the highest reliability, follow by MS Bletchley. Both TownCrier and Corda with the same reliability. The two platforms with lowest calculated reliability involve humans. ChainLink designed its oracle solution with the need for humans to compete to become an oracle. Human error has the highest failure rate, hence contributes to the lower calculated value of ChainLink reliability. Augur also uses human oracles, but uses both a designated reporter and open reporters. This reduces the risk of not having anyone act as reporter, because there is the choice to choose a trusted oracle from a pre-defined list.

Table 4.1: Reliability of Database

Platform	Reliability
Augur	0.9928
Ms Bletcgely	0.9861
TownChair	0.9840
Corda	0.9840
Provable	0.9810
ChainLink	0.9297
Gnosis	0.8837

Augur allows disputations of the reported value by staking 2 times the amount of the no-show bond (amount of REP used in initial market creation). Gnosis also offers the same dispute windows but a fixed amount of 100 ETH is required in order to dispute the reported value. This discourages spurious disputes due to the high stake required. Gnosis network's participants might be reluctant to dispute an incorrect reported value because of the required high stake and thus reliability of Gnosis becomes lower than Augur. Provable, Town Crier, Corda and MS Bletchley have a similar oracle mechanism. They all utilize trusted hardware to directly fetch information from an external trusted execution environment (TEE).

Provable offers TLS-Notary proof to validate the actual data submitted by an URL, but this serves as a single point of failure. TownCrier and Corda have a similar setup while MS Bletchley has multiple oracles that fetch data from multiple sources, resulting in higher calculated reliability compared with the other single oracle mechanisms. Augur, Gnosis, MS Bletchley and ChainLink propose to have more than one oracle. Multiple oracles prevent single points of failure. Augur and Gnosis allow voting on correct answers by any participating node, and disputes on tentative answer. Any participant in a market has the ability to act as an oracle as long as they stake a result. MS Bletchley and ChainLink also have multiple oracles. ChainLink implements a K-out-of-N scheme, where the final consensus on the result will only be reached if a pre-determined K-out-of-N oracles agree on the reported value.

4.1. Sensitivity analysis

We especially performed sensitivity analysis on the K-out-of-N oracles approach, as depicted in Fig. 5. The sensitivity analysis of N (blue curve) demonstrates that the greater the N value, the lower the probability of failure and the higher the reliability of the oracle. This is done by setting K as a constant to 1. We also performed a K sensitivity analysis, in which the number of N was fixed at 5, and the number of K was changed. The outcome (green curve) demonstrates that the dependability increases with decreasing K. The more oracles needed for proper operation increases as K increases. The entire system will fail if K incorrect oracles are used. A N/2 + 1 oracle sensitivity analysis was then completed. The plan outlined in ChainLink's white paper is this one. The outcome (red curve) demonstrates that the oracles' services are more reliable the greater the number N. Because an even number of oracles could

lead to a deadlock if half of them voted for one branch and the other half for a different branch, N can only be an odd number. The reliability of the services would rise with more oracles reporting, but there might be a higher cost to cover for all the oracles to operate.

4.2 Reliability patterns

Our research revealed that different system fault-tolerance design patterns are shared by all oracle mechanism designs. A strategy to have a minimum of two nodes delivering services simultaneously is the active redundancy pattern [15]. The active-active redundancy pattern is equivalent to the ChainLink and MS Bletchley oracle configurations. Both blockchain networks feature numerous active oracles that simultaneously fetch external data. While ChainLink utilises human redundant oracles, MS Bletchley uses automated redundant oracles. The risk of a single point of failure is lowered by using numerous active oracles to fetch values into the blockchain. Multiple oracle data collection increases the accuracy of the final approved value. The oracles are permitted to obtain the required information from numerous external data sources, improving the likelihood of getting accurate external data.

There are certain time and money-related drawbacks to the active redundancy reliability pattern. To use numerous oracles to retrieve data and submit it to a blockchain, data requesters could have to pay extra. A longer time period is also necessary for all oracles to fetch values and aggregate the received values into a final value due to the requirement to aggregate collected data from various oracles and sources.

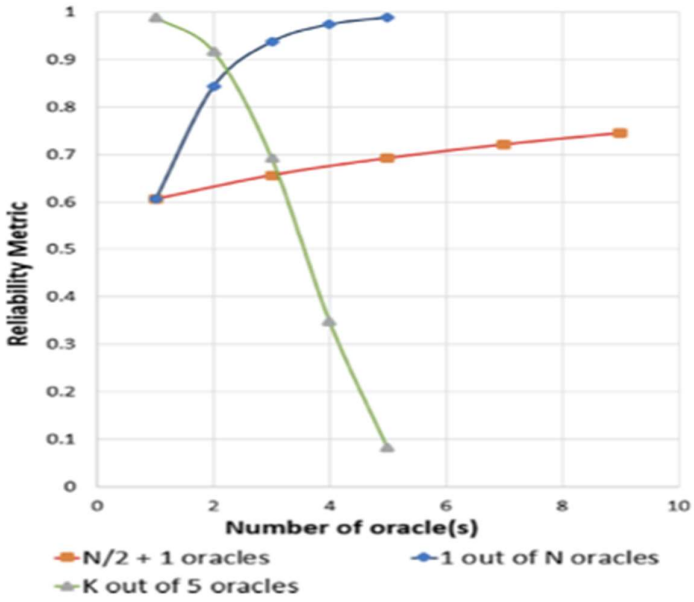

Figure 4.2. Sensitivity Analysis of K-Out-N Oracle Database on Chainlink

MS Bletchley has a higher reliability because ChainLink oracles require human intervention. Participants in the network would need to bid to become an oracle. If no one is interested, the

process will stuck after data is requested. Hence, the incentive scheme plays a very vital role to ensure involvement and interest from participants in the ChainLink network.

A minimum of two nodes are required for the active-passive redundant design, with one serving as a standby server. Only in the event of an active server failure will the standby server take over [15]. The setups of Augur and Gnosis resemble the active-passive redundancy pattern. Because there is only one oracle reporting the value in Augur and Gnosis, the first reporting oracle serves as the active server. If the other network participants believe the reported value to be inaccurate, they have the option to contest the tentative value. In a way, the disputants serve as a passive server. Gnosis dependability is lower than Augur reliability since users must pay 100 ETH to contest a reported value in Gnosis, making Augur reliability better oracle. Due to the huge stakes involved, this economically dissuades players from correcting any misreported value. With Intel SGX technology, TownCrier and Corda leverage the input guard design pattern to contain faults. The input guard design pattern prevents an error from spreading into the guarded component from the outside [16].

7.2. Common causes of failure

All oracles employ techniques to reduce the possibility that inaccurate information will be used in the blockchain and lead to improper execution of smart contracts. The data source(s) can, nevertheless, always be a common reason for failure. In the situation depicted by the box with a dashed line in Figure 4, common reasons of failure could impact all three data sources. Take into account the request for a Randwick, Sydney, weather forecast for a specific day. A weather forecast will be sought out from sources on the internet by several oracles, who will then provide it to the requester. If the Bureau of Meteorology predicts the weather for Randwick and all sources get their forecast information from the Bureau, faulty sensors or reports would affect all the sources. From an operational standpoint, the oracle's result would still be valid, but the accuracy of the data they provided would no longer be guaranteed. Therefore, information gathered from several web services may only be based on one original data source from the real world or from another web service.

CONCLUDING REMARKS

Blockchain oracles are a crucial component in expanding the capability of blockchain. We have selected, reviewed and characterized seven active blockchain platforms with oracle mechanisms according to their reliability features, consensus, and other characteristics. Our framework can serve as a reference on deciding suitable oracle mechanisms to fulfil different requirements. To assess reliability of blockchain-based systems using oracles, we tailored existing approaches for reliability calculations using FTDs to model and evaluate the reliability of the selected oracle platforms. We showed the calculation of reliability using this approach for the selected oracles based on representative failure rates of the components in the oracle mechanisms. The result shows that decentralized oracles are more reliable than centralized oracle mechanisms, and human-related faults are the main factor affecting the overall reliability of oracle mechanisms.

REFERENCES

[1]. Lo SK, Xu X, Chiam YK, Lu Q. Evaluating suitability of applying blockchain. In: ICECCS. IEEE; 2017. p. 158–61.

[2]. Almadhoun R, Kadadha M, Alhemeiri M, Alshehhi M, Salah K. A user authentication scheme of IoT devices using blockchain-enabled fog nodes. In: 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA). IEEE; 2018. p. 1–8.

[3]. Hasan HR, Salah K. Combating deepfake videos using blockchain and smart contracts. IEEE Access 2019;7:41596–606.

[4]. Iqbal M, Matulevicˇius R. Blockchain-based application security risks: asystematic literature review. In: Proper HA, Stirna J, editors. Advanced information systems engineering workshops. Cham: Springer International Publishing; 2019. p. 176–88.

[5]. Anderson R, Moore T. Information security economics–and beyond. In: Annual international cryptology conference. Springer; 2007. p. 68–91.

[6]. Ohba M. Software reliability analysis models. IBM J Res Dev 1984;28(4):428–43.

[7]. Avizienis A, Laprie J-C, Randell B, Landwehr C. Basic concepts and taxonomy of dependable and secure computing. IEEE Trans Depend Secure Comput 2004;1(1):11–33.

[8]. Weber I, Gramoli V, Ponomarev A, Staples M, Holz R, Tran AB, Rimba P. On availability for blockchain-based systems. In: 2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS). IEEE; 2017. p. 64–73.

[9]. Wan Z, Lo D, Xia X, Cai L. Bug characteristics in blockchain systems: alarge-scale empirical study. In: MSR; 2017. p. 413–24.

[10]. Yasaweerasinghelage R, Staples M, Weber I. Predicting latency of blockchain-based systems using architectural modelling and simulation. In: ICSA 2017. IEEE; 2017. p. 253–6.

[11]. Tiwari S, Gupta A. An approach to generate safety validation test cases from UML activity diagram. In: APSEC; 2013. p. 189–98.

[12]. Dhillon BS. Engineering reliability: new techniques and applications. Technical Report; 1981.

[13]. Nikolić I, Kolluri A, Sergey I, Saxena P, Hobor A. Finding the greedy, prodigal, and suicidal contracts at scale. In: Proceedings of the 34th annual computer security applications conference. In: ACSAC '18. New York, NY, USA: ACM; 2018. p. 653–63.

[14]. Smith DJ. Reliability, maintainability and risk: practical methods for engineers. Butterworth-Heinemann; 2017.

[15]. Saridakis T. A system of patterns for fault tolerance.. In: EuroPLoP; 2002. p. 535–82.