Analysis of on-chain oracles

Blockchain
Miquel Martinez
Xami Miquel Bosch
Pol Collado
Universitat Pompeu Fabra de Barcelona
19/6/2023

1 - Abstract

In this project we will do an exhaustive analysis of on chain oracles focusing on the security evaluation, scalability, performance and validation of data. Blockchain is getting more and more important nowadays, but it is also thanks to oracles as they have a very important paper. They provide external and real world data to the smart contracts, but traditional oracles present a big problem of security which on chain ones proposes a solution because those are executed directly on the block chain.

Mainly we'll try to evaluate in depth security of on-chain oracles analyzing common vulnerabilities and best practices of mitigation for those. Also as we said we will study the scalability and performance that they give to us the chance to work with enormous volume of data. Moreover data validation will be explored to know more about how they guarantee veracity and precision of data.

This project seeks to contribute to existing knowledge of on-chain oracles and supply valuable information about security, scalability, performance and validation of data which we think are some topics we believe that must be very clear when working with them. Also we will present some interesting use cases of these decentralized oracles.

2 - Introduction

Blockchain technology has transformed some real world sectors since it offers a decentralized infrastructure and security for the execution of smart contracts. Decentralized oracle networks enable the creation of hybrid smart contracts, where on-chain code and off-chain infrastructure are combined to support advanced decentralized applications that react to real world events and interoperate with traditional systems.

Oracles as we said perform a crucial paper as they act as intermediaries between smart contracts and external data. Those can provide information about prices, sport events, and much more. Oracles are responsible for compiling, verifying and giving that data to smart contracts.

For example, we will think about a sports event bet. We have Bob and Alice who bet 20€ on the outcome of a football match. As the result of this bet we have the 40€ retained by a smart contract. Then when the game ends the smart contract needs to know whether to release the funds to Alice or Bob so for this we require an oracle mechanism to fetch accurate math outcomes off-chain and deliver it to the block chain in a secure and reliable manner.

Also we know that traditional oracles have been highly used but as they are centralized they may have serious security flaws and are subject to malicious attacks. Furthermore the scalability of it can also be limited since as we now they have the need to trust on a centralized entity. As we said, on-chain oracles come to us to solve this big problem.

In the next sections of this project we will do an exhaustive analysis of on-chain oracles mainly on security, scalability, performance and data validation as we said before. We will do

it through comparisons and examples that will help us to contribute to the knowledge on this area and also to provide valuable information about this topic.

3 - Our contribution

As shown in the sections before we are mainly focusing on 3 different aspects of blockchain oracles so this is what we are looking for in this project in each of these three aspects.

In terms of security we analyze most common vulnerabilities that can affect on-chain oracles as it can be the spot price manipulation but also how to mitigate these vulnerabilities with different common practices as it can be the use of secure smart contracts, verification mechanisms and encryption techniques. Another thing that is important to have a look is how on-chain oracles can protect themselves.

Related to the scalability and performance we will compare different implementations types of on-chain oracles, analyzing the capacity that they have to manage big volumes of data and transactions but without losing too much performance which is something that we always look for in everything related with technology. Moreover we look for different optimizations that we can do to our code that increase the scalability of these oracles.

For the validation of data we will also investigate how on-chain oracles manage these, as is one of the most important aspects because they have different behaviors as they do not depend on an entity. We will see an amount of methods used to guarantee data veracity as it can be the reputation system or advanced cryptography.

Finally but not least we also want to present practical examples to evaluate these oracles and to finish understanding how they work using the different theoretical techniques that we have presented during this project. These practical proves allow us to gain a deeper understanding of the strengths and limitations on-chain oracles have in terms of the three focused points.

4 - Related Work

In this section, we will review some of the previous work related to on-chain oracles. These can be useful to us to know more about on-chain oracles. For doing this we are going to have a look at different related studies.

In a study on security and privacy researchers explored the challenges and solutions in these fields protection on a decentralized network, including blockchain and oracle systems [19]. This study highlights the importance of adopting new security and privacy protection solutions to address the increasing interdependence of IT solutions and the growing amount of data in decentralized systems [19].

In a paper on provably secure security-enhanced timed-release encryption in the random oracle model, researchers proposed a new encryption based on fixed and variable random numbers together as the time servers private key to generate trapdoors [20]. This scheme reduces the computational cost 10.8% compared to the most efficient solution in the random

oracle model while increasing the storage space [20]. This can be applied to enhance the security of on-chain oracles.

In a study titled "Blockchain Oracle Design Patterns," researchers conducted a comprehensive survey of blockchain oracles, focusing on their design, security, and performance [21]. They classified the blockchain oracle techniques into two major groups such as voting-based strategies and reputation-based ones. This study can provide valuable insights into the best practices for optimizing the use of on-chain oracles in smart contracts.

In a study titled "Towards a middleware design for efficient blockchain oracles selection," researchers proposed a middleware design to select the best data providers available in the Chainlink oracle network based on individual job's service requirements [22]. They modeled the oracle selection problem as a decision optimization problem and proposed two heuristic algorithms to approximate a solution to the modeled problem. This study can provide insights into optimizing the use of on-chain oracles in smart contracts.

In terms of scalability of decentralized oracles, researchers proposed a new oracle architecture based on a combination of on-chain and off-chain data sources to see how the scalability improves and the reduction of the data costs [18]. This architecture uses decentralized network nodes where each node is responsible for a specific data source. By using this combination the architecture can handle large-scale applications and complex computations while reducing the computational load of the oracle network [18].

5 - Background, Terminology, and Definitions

For the best understanding of this project it is essential to understand some key concepts and also some literature background. This section is done to create a solid base of the knowledge of the topic.

5.1 - Background

The power of Bitcoin lies not only in its decentralized features but also in its programmability. Just by using some scripts and without needing the intervention of third parties. However, because of Vitalik Buterin and the introduction of the Ethereum virtual machine with smart contracts, blockchains became more developer-friendly. It is necessary to ensure that all the required data for smart contracts are publicly verifiable and auditable by all nodes. Without real data from the external world, the range of possible automated contracts would have been extremely limited. Therefore, a means to deliver extrinsic data to the blockchain was needed to broaden the use of smart contracts. This method is called an oracle. Oracles are an entire ecosystem that permits the collection and transfer of external data to decentralized applications. This ecosystem is divided into 3 parts [2].

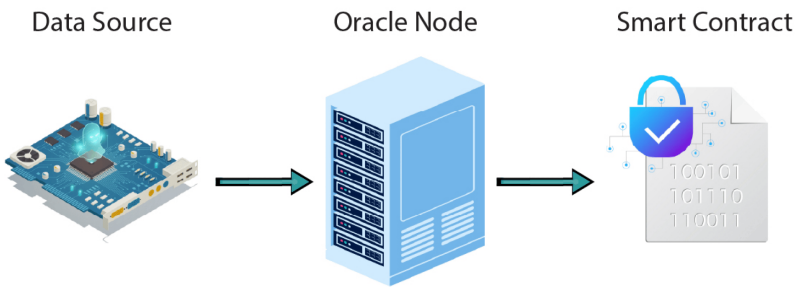

Figure 1. Oracle ecosystem

Data source: This is simply the source where the data is collected and stored. This can be data from a sensor, an API or an event for example [2].
Communication Channel: We call it “node” and it collects data from the source and it provides it to the smart contract for the later use [2].
Smart contract: This contains the code that establishes how collected data can be managed [2].

As we said before oracles play a crucial role on blockchain technology both in applications and smart contracts as they provide external data to the chain. They are prepared for doing different tasks as:

- Listening - Permanent listening to blockchain networks in order to find out every type of off-chain data request.
- Validate - Create a cryptographic prove of the oracle activity using different data
- Notify - Sign and notify a transaction on the blockchain for the later use of the data on smart contracts.
- Processing - They do some data calculations using different personalized types of data

If we want them doing all this stuff they must operate inside and outside the blockchain at the same time. On one hand the on-chain component consists of establishing connection with some blockchain, transmitting data, sending proves, extracting data from blockchains and sometimes doing some data calculations on the blockchain. On the other hand, the off-chain component consists of processing petitions, storing data and sending data of the blockchain to external systems.

Traditional oracles have been used a lot, as we have seen they have a lot of functionalities and they are really important for using off-chain data on our smart contracts but they present some security problems as they depend on a centralized entity. For this reason on-chain oracles appear as a solution to this problem as they are executed directly on the blockchain [4][5].

5.2 - Terminology and definitions

Now we will define some relevant concepts providing examples in order to be more easily to know them better:

- Oracles: Blockchain and smart contracts cannot access data directly from the outside of their network. To know where the data are and what contents are provided, a smart contract needs to access information from the real world in the form of electronic data also referred to as Oracles. Oracles send and verify real-world

occurrences and submit the related information to smart contracts, triggering state changes on the blockchain. In general, Oracles are like bridges that can connect our smart contracts with real world data so they can be much more useful in our routine as they can be autonomous. For example , imagine a smart contract that needs to know if it has rained enough to trigger a farm insurance payout. Oracle will provide data about the amount of rain on the desired location [1][3].

- Blockchain: Blockchain can be viewed as a distributed and globally shared database. It is a public ledger of transactions and digital events that have been executed and shared among participating parties. It can also be considered as a log file split into many time-stamped blocks. Each block contains the hash of the previous block, which guarantees that no previous block or the associated contents have been changed without being noticed, therefore blockchain can be regarded as a trustworthy platform for providing correctness and availability [1].
- Smart contracts: A smart contract ensures minimizing the occurrence of malicious intervention, the appearance of accidental exceptions and the transaction cost of the intermediaries. in general a smart contract is a replicated state machine that receives inputs, executes logic operations on the states according to the input data and then provides an output depending on its function. They are scripts stored on the blockchain, allowing us to have general-purpose computation occurring within the chain. It is powerful enough to implement some business logic through the contract beyond “Bob paying money to Alice”. These contracts are executed on the blockchain and they allow automatized transactions and deals based on conditions without needing intermediaries. An essential feature of a smart contract is that it is always deterministic, the same input should always give the same output. If it was non-deterministic, different nodes would reach different results so this would prevent the blockchain network from reaching a consensus [1].
- Security: Security is really important when talking about on-chain oracles. New system security holes may be created if the integration of the Oracles and the blockchain network has not been carefully conducted. Let's consider a scenario where a smart contract does transactions based on the price of some cryptocurrency. An on-chain oracle must guarantee that the given price is correct and therefore not manipulated. To achieve this, some cryptography techniques are used to verify the authenticity and the integrity of data. A common way to ensure that security is to use homomorphic encryption techniques.
- Homomorphic Encryption Algorithms: Homomorphic encryption is a class of encryption algorithms in which certain operations can be directly carried out on ciphertexts to generate the corresponding encrypted results After decrypting the encrypted result, it will match the result of the same operations performed on the corresponding plaintexts. For example, m denotes a plaintext message and $[m]$ denotes the corresponding ciphertext. As a result, $[m] = E_{pk}(m)$ means m is encrypted with a public key pk, and m $D_{sk}([m])$ means [m] can be decrypted with the secret key sk. This is highly used in the field of data security and privacy preservation applications [1].
- Scalability: Scalability is something important on on-chain oracles. Let's assume that a smart contract needs access to take some data from an IoT sensor in real time for executing a desired action specified on the smart contract. If the oracle is not scalable he can not manage with big volumes of data efficiently which causes it to have significant delays.

- Performance: When we talk about on-chain oracle performance we are referring to the capacity that they have to make a response quickly to data requests. For example a smart contract needs to check a flight status to trigger a travel insurance payment, this will require an on-chain oracle with high performance. This implies that the oracle must be able to obtain and provide real time status about the flight so the smart contract works correctly and how we want.
- Gas: Gas is a measure of resource consumption on the blockchain used to calculate fees for executing trades and smart contracts. Every operation or instruction executed on the blockchain consumes a certain amount of gas and it requires that users are required to pay a fee in gas to execute these operations. Also smart contracts require gas for its execution. Each operation and each line has a cost in terms of gas.

6 - Main Description / Analysis

6.1 - On-chain oracles security

Security is really important when it comes to on-chain oracles because they can be easily attacked or manipulated. This could lead to the data they provide being compromised and not trustworthy if they are not implemented properly [8].

One way to make on-chain oracles more secure is by using machine learning (ML) techniques to better detect and find threats. For example, a security provider can use ML to figure out the best times to use another service or oracle, like VirusTotal, to improve their ability to detect problems and keep their customers safe [8]. However, using ML to make security oracles more affordable can also create new security issues. One problem is that ML can make it easier for attackers to find ways to attack the system. To deal with these issues, it's important to find ways to make the ML process more secure [8].

As most common security threats to on-chain oracles we have:

- Data manipulation: Attackers may attempt to manipulate data provided by the oracle in order to give incorrect information that later will feed smart contracts. The result of this attack will produce incorrect outcomes that can have a high risk such as incorrect payments [11].
- Sybil attacks: On a sybil attack the attacker tries to create multiple accounts, fake identities, to gain a huge control over the network. Thanks to that the attacker will control fake nodes to manipulate the data provided to smart contracts [11].
- Denial of service (DoS) attacks: DoS attacks consist of sending as many requests as possible to a system or network in order to collapse it. This attack prevents our oracle from providing the data to smart contracts so we will have a contract failure. These attacks are mainly done by a botnet installed on a lot of systems around the world [11].
- Replay attacks: An attacker tries to retransmit previous valid data in order to trick the system and accept it as a new request. This could involve serious problems if old data is sent to the smart contract and that is executed with this information [11].

If we want to mitigate as many threats as possible we need to implement robust security measures such as we said at the beginning of this point using ML techniques which are really used and rigorous data validation [11][12][13].

To protect against attacks that try to manipulate the data, on-chain oracles can use a few different strategies:

- Get data from many different sources instead of just one, so they're not relying on a single source that might be compromised [9].
- Use agreement systems where the majority of the network nodes have to agree on the data being provided. This makes it harder for attackers to manipulate the data [9].
- Use special techniques involving codes and signatures to make sure the data from on-chain oracles is genuine and hasn't been tampered with [10].
- Keeping our software up to date will help us to minimize the chances that attackers have to exploit the vulnerabilities [14].
- Implementing access control is also a good technique as data is really sensitive and there exists some systems that can help us to restrict the access to that data [14].
- Monitoring the activity can also help to prevent certain attacks such as the DoS attack that we said before. If you notice something strange on the request you can immediately take preventive measures [14].
- Using encrypted communication channels can help to protect data from being intercepted and manipulated during the transmission [10].

6.2 - On-chain oracles scalability and performance

When doing a task, performance is crucial and we always look for the maximum possible. For this reason there exists several methods to improve the performance of on-chain oracles on our smart contracts. Some of these methods include off-chain reporting, oracle computations and optimizing the use of on.chain and off-chain data. Here are some methods to improve performance:

- Off-chain Reporting: OCR is a technique that allows oracles to communicate off-chain, enabling them to aggregate data at zero gas costs using a distributed peer to peer network. This reduces on-chain network congestion and latency, ensuring data is rapidly provided to smart contracts even during blockchain network congestion and extreme market volatility [23]. Chainlink, a decentralized network of oracles, has implemented OCR to increase the amount of real-world data available to decentralized finance applications and other blockchain-based systems [23].
- Oracle computation: Oracle computation is a type of off-chain computation that uses decentralized oracle networks to perform any computation while remaining anchored to blockchains to create trust-minimization guarantee [24]. This approach falls between centralized Web 2.0 computation and decentralized blockchain computation, achieving more performance and feature richness than blockchains while being more tamper-resistant and transparent than Web 2.0 systems [24]. Chainlink offers oracle computation services that extend the capabilities of smart contract execution by increasing scalability, cost-efficiency, and privacy, as well as granting access to new features like order fairness, verifiable randomness, off-chain aggregation, and transaction automation [24].

- Optimizing On-Chain and Off-Chain data:Balancing the use of on-chain and off-chain data can help improve the performance and scalability of smart contracts. On-chain data is slow and reliable, while off-chain data allows for faster processing and storage[25]. By moving some computations and storage off the chain, you can increase the scalability and performance of your smart contracts [25].

On the other hand, as we can imagine these on-chain oracles optimizations have some limitations. Some of these limitations include:

- Latency and throughput: These oracles often struggle to maintain low latency and high throughput, especially when there is a high number of users and transactions [26]. This can lead to delays in data delivery and loss of performance.
- Cost and resource constraints: Decentralized oracles can be expensive to operate and maintain due to the resources required for data aggregation, validation, and transmission [27]. This can limit the scalability of decentralized oracles, as the costs and resources required to support a growing number of users and transactions may become prohibitive.
- Interoperability and integration: Decentralized oracles need to be compatible with various blockchain platforms and smart contract languages, which can be challenging due to the differences in protocols, standards, and data formats [28]. Ensuring seamless integration and interoperability between decentralized oracles and different blockchain platforms can be a complex and resource-intensive task
- Network congestion: As the number of users and transactions on a blockchain network increases, network congestion can become a significant issue, leading to delays in data delivery and reduced performance of decentralized oracles [26]. This can impact the overall efficiency of smart contracts and decentralized applications that rely on timely and accurate data from decentralized oracles.

6.3 - On-chain oracles data validation

Data validation is essential for getting accurate and reliable information which comes from our oracle. Now let's see some validation ideas used:

- If we are talking about DeFi oracles, data validation is crucial in order to ensure that data provided to our smart contract is reliable. In a study on DeFi oracles researchers proposed a set of metrics for designing trustworthy DeFi oracles and a potential trust architecture for building future trustworthy oracles [15].
- Tara is a trusted blockchain oracle based on a decentralized Trusted Execution Environment network that provides authenticated data to our smart contracts. For providing this authenticated data, Tara uses a mechanism based on Proof-of-Availability (PoA). This prototype was implemented on Ethereum with Software Guard eXtensions and evaluated with different measures including gas costs, off.chain execution time and throughputs [16].
- Paired-question decentralized oracle protocol is a new protocol designed to extract true answers from the public. When using the oracle, a user asks opposite questions in pairs, and users who vote on the answers have a chance to get rewards. This process encourages everyone to honestly report their findings and creates a situation where telling the truth is strongly encouraged [17].

Also if we are using a decentralized oracle there are some points to have a look at because it is important to know the advantages that we have compared to centralized oracles.

- As we said before, the security of decentralized oracles is better and less vulnerable to targeted attacks and single points of failure. This is because to provide the data that we want they use the network of nodes, so on, decentralized oracles can ensure that the majority of nodes agree on the data that has been provided. For this reason it is more difficult for the attacker to manipulate the data.
- As it is not centralized, these oracles can provide data from different sources making it more reliable. This helps us in a way to get better accuracy when executing our smart contracts.

6.4 - On-chain oracles use cases

Developers are using smart contracts with oracles in order to build more advanced decentralized applications across a wider range of blockchain use cases. As applications are decentralized there are an infinite number of possibilities but here are some of the most uses of these oracles.

Decentralized Finance (DeFi)
A significant segment of the decentralized finance (DeFi) landscape heavily relies on oracles for obtaining financial information pertaining to various assets and markets. To illustrate, decentralized money markets utilize price oracles to ascertain users' borrowing potential and assess whether their positions lack sufficient collateral, thereby making them susceptible to liquidation. Also, platforms facilitating synthetic assets depend on price oracles to establish the value of tokens in relation to real-world assets, while automated market makers (AMMs) rely on price oracles to concentrate liquidity at the prevailing market price, thereby enhancing capital efficiency [3][7].

Dynamic NFTs and Gaming
Oracles have another use case out of the financial world too such as dynamic NFTs which are Non-Fungible Tokens that can change in appearance or value based on different events like time of the day or weather. On-chain gaming applications also use verifiable randomness to create more engaging and unpredictable gameplay experiences like the appearance of random loot boxes or randomized matchmaking during a tournament [3].

Insurance
Input oracles are used to verify the occurrence of insurable events during claims processing, opening up access to physical sensors, web APIs and legal data. Output oracles can also provide insurance smart contracts with a way to make payouts on claims using other blockchain or traditional payment networks [3].

Sustainability
Hybrid smart contracts are advancing a lot by creating better incentives to participate in green practices through different verification techniques around the true impact of green initiatives. In this section Oracles provide smart contracts the desired data through sensors, satellite images, advanced ML computations, which allow them to dispense rewards to people practicing some green actions like reforestation [3].

Enterprise
Cross-chain oracles provide a secure technology solution for businesses, acting as a bridge between their internal systems and various blockchain networks. By utilizing cross-chain oracles, enterprises can seamlessly connect their backend systems to any blockchain, enabling them to interact with and transfer data to different blockchain networks. This integration empowers businesses to efficiently deploy assets and information across multiple chains and collaborate with other parties using the same oracle network. Consequently, institutions can easily join popular blockchains that are in demand among their partners, swiftly offer smart contract services desired by their users, and eliminate the need to invest significant time and resources in integrating with each individual blockchain separately [3].

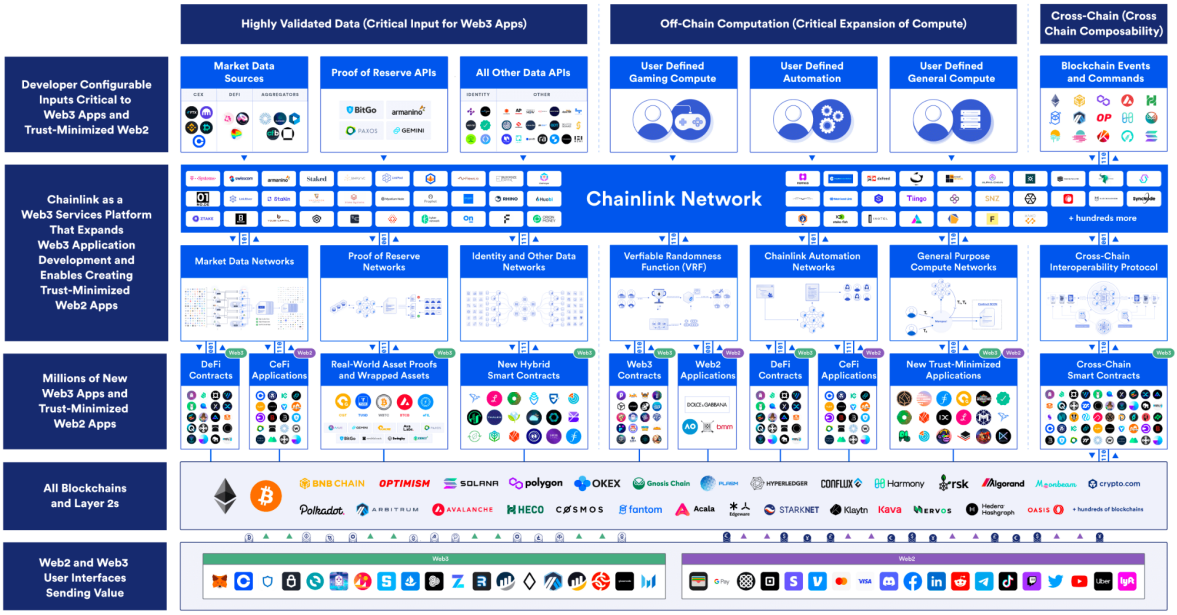

Figure 2. Chainlinks's growing collection of decentralized oracle services

7 - Implementation Description

In this part we will do a simple example of how to apply an oracle in code. For this implementation we will use the Ethereum platform and the programming language of Solidity for our smart contracts.

7.1 Voting Smart Contract

We have made a decentralized voting system smart contract which uses an on-chain oracle in order to validate the emitted votes by participants. This contract also keeps track of votes cast.

This smart contract consists of the following functions:
vote(uint256 option, bytes memory signature)
This function allows participants to emit votes for a specific option. They must provide the vote and also a cryptographic signature generated with their personal key for the

authentication of the vote. Then the contract makes a call to the oracle to validate the authenticity of the vote with the function **validateVote()** of the contract.

getVoteCount(uint256 option)
This function returns the number of votes for a specific option. Also you can use this function to have a look at the on time results.

isVoteValid(uint256 option)
This function verifies if a specific option has been validated by the oracle which means that counts as a valid vote. It can be used for participants to know the validity of the registered votes.

7.2 - On-Chain Oracle implementation

For the implementation of the oracle we have created a smart contract called Oracle. This contract acts as an intermediary between the voting smart contract and the validity and verification functions.

This Oracle contract contains the following functions:

validateVote(uint256 option, bytes memory signature)
This function is called to validate the authenticity of the vote. It receives the option number desired by the participant and its cryptographic signature.

In our example of code the function **validateVote()** always returns '**true'** without doing any validation. This function has to be personalized according to your authentication requisites and security.

7.3 - Testing and validation

It is crucial to do exhaustive tests to guarantee the correct functionality and security of our implementation. Also as we have mentioned during the project, it is mandatory to consider data validation and resist possible attacks. This includes in our example the simulation of different voting scenarios, load tests for evaluating the performance of the system and the implementation of different security measures as it can be the use of trusted crypto libraries and proper gas management.

7.4 - Scalability and performance considerations

In every implementation as ours, it is important to consider scalability and associated costs with the use of the oracle. Oracle calls and votes validity can generate a significant gas cost and it also can affect the efficiency and costs of transactions to the blockchain network.
To deal with these challenges we can explore optimization techniques such as storing relevant data in the blockchain for a less frequency of calls to the oracle. Also other techniques as shown on previous sections of this project can be used.

8 - Conclusion

In conclusion, on-chain oracles hold significant potential to revolutionize decentralized applications landscape as they do not need a third entity. However, several challenges must be addressed in order to realize their full potential. Some key areas to understand are the ones we provided on these project but there are some more that we would need to have a look at.
Continued research and collaboration among academia, industry and the blockchain community are vital for the advancement of on-chain oracles. If we collaborate and we solve some new challenges we could integrate this in some sectors because as we saw they have the potential to play a big role in the future.

References:

1. Chen, Y., Wu, J., Hsieh, Y., & Hsueh, C. (2020). An Oracle-Based On-Chain Privacy. *Computers*, *9*(3), 69. https://doi.org/10.3390/computers9030069
2. Caldarelli, G. (2022). Overview of Blockchain Oracle Research. *Future Internet*, *14*(6), 175. https://doi.org/10.3390/fi14060175
3. *What Is an Oracle in Blockchain? » Explained | Chainlink*. (s. f.). https://chain.link/education/blockchain-oracles
4. Diligence, C. (s. f.). *Oracle Manipulation - Ethereum Smart Contract Best Practices*. https://consensys.github.io/smart-contract-best-practices/attacks/oracle-manipulation/
5. De Chainlink En Español, C. (2022, 6 enero). ¿En qué consiste el problema del oráculo en las blockchains? *Medium*. https://medium.com/chainlink-community/en-qu%C3%A9-consiste-el-problema-del-or%C3%A1culo-en-las-blockchains-1b18d4f3007c
6. S. Ellis, A. Juels and S. Nazarov, ChainLink: A decentralized oracle network, vol. 11, pp. 2018, Mar. 2017, [online] Available: https://link.smartcontract.com/whitepaper.
7. *Oracles and Blockchain: DeFi Oracles Examined | Gemini*. (s. f.). Gemini. https://www.gemini.com/cryptopedia/crypto-oracle-blockchain-overview
8. Preuveneers, D. (2022). *Applying Machine Learning to use security oracles: a case study in virus and malware detection*. https://www.semanticscholar.org/paper/Applying-Machine-Learning-to-use-security-oracles%3A-Preuveneers-Lavens/249fe0654dff7b3c747e24adc4e47b1ee36e3337
9. Aspembitova, A. T., & Bentley, M. A. (2022). Oracles in Decentralized Finance: Attack Costs, Profits and Mitigation Measures. *Entropy*, *25*(1), 60. https://doi.org/10.3390/e25010060
10. *Human Verification*. (s. f.). https://www.semanticscholar.org/paper/A-power-resource-dispatching-framework-with-a-in-of-Liu-Zheng/ef21079afa31467731d25b977fcec393cb692c6d
11. U, D. (2021). *A Secure File Transfer over Virtual Machine Instances using Hybrid Encryption Technique*. https://www.semanticscholar.org/paper/A-Secure-File-Transfer-over-Virtual-Machine-using-Dhanush-Hulikatti/8d7948a34d1cead935de20dae8fca23a8b123403
12. Akram, M. (2022). *Fog-based low latency and lightweight authentication protocol for vehicular communication*. https://www.semanticscholar.org/paper/Fog-based-low-latency-and-lightweight-protocol-for-Akram-Mian/d6658fb80b75c2059a32f9ac0ae471444d2580c0
13. Kolhe, G. (2021). *Securing Hardware via Dynamic Obfuscation Utilizing Reconfigurable Interconnect and Logic Blocks*. https://www.semanticscholar.org/paper/Securing-Hardware-via-Dynamic-Obfuscation-Utilizing-Kolhe-Salehi/f4dd658a9cf70e8623951aec2d5ddb19f3cece4a
14. Desai, P. D. (2021). *Best Practices for Securing Financial Data and PII in Public Cloud*.

https://www.semanticscholar.org/paper/Best-Practices-for-Securing-Financial-Data-and-PII-Desai-Hamid/98e4f08b7127fa6782ec9e431e33ec3bf95bcf9d

15. Zhao, Y. (2022, 7 enero). *Towards Trustworthy DeFi Oracles: Past,Present and Future*. arXiv.org. https://arxiv.org/abs/2201.02358
16. Chen, L. (2021). *Tora: A Trusted Blockchain Oracle Based on a Decentralized TEE Network*. https://www.semanticscholar.org/paper/Tora%3A-A-Trusted-Blockchain-Oracle-Based-on-a-TEE-Chen-Yuan/f2769fbf7c669961dc8859ec7657463121b1c69d
17. Merlini, M. (2019). *On Public Decentralized Ledger Oracles via a Paired-Question Protocol*. https://www.semanticscholar.org/paper/On-Public-Decentralized-Ledger-Oracles-via-a-Merlini-Veira/63d53a6e811ba996ce78035cb4a576a143339092
18. Chen, J. (2023). *Oscms: A Decentralized Open Source Coordination Management System Using a Novel Triple-Blockchain Architecture*. https://www.semanticscholar.org/paper/Oscms%3A-A-Decentralized-Open-Source-Coordination-a-Chen-Zhao/fe5d44086f57f61d9a24d9b62353344461a8b8d3
19. Cheng, X. (2021). *IEEE Access Special Section Editorial: Security and Privacy in Emerging Decentralized Communication Environments*. https://www.semanticscholar.org/paper/IEEE-Access-Special-Section-Editorial%3A-Security-and-Cheng-Liu/70f5e7fc9fbd7a73c6461b45525944c3c7139f00
20. Yuan, K. (2021). *Provably Secure Security-Enhanced Timed-Release Encryption in the Random Oracle Model*. https://www.semanticscholar.org/paper/Provably-Secure-Security-Enhanced-Timed-Release-in-Yuan-Wang/59f0a9eade1162759adacb8c75d4e7f65fb9c985
21. Pasdar, A. (2021, 17 junio). *Blockchain Oracle Design Patterns*. arXiv.org. https://arxiv.org/abs/2106.09349
22. Goswami, S. (2022). *Towards a middleware design for efficient blockchain oracles selection*. https://www.semanticscholar.org/paper/Towards-a-middleware-design-for-efficient-oracles-Goswami-Danish/aa89105f5e4b6519958e2cb3ef7f2bbb8adaeea3
23. Chainlink. (2022). Chainlink Achieves Major Scalability Upgrade With Mainnet Launch of Off-Chain Reporting (OCR). *Chainlink Blog*. https://blog.chain.link/off-chain-reporting-live-on-mainnet/
24. Chainlink. (s. f.). Oracle Computation: Expanding the Purpose of Oracles to Data Delivery and Off-Chain Computation. *chain.link*. https://chain.link/education-hub/oracle-computation
25. *On-chain or Off-chain Business Logic | A Blockchain Challenge*. (s. f.). https://www.kaleido.io/blockchain-blog/on-or-off-chain-business-logic
26. Chainlink. (s. f.-a). Blockchain Scalability: Execution, Storage, and Consensus. *chain.link*. https://chain.link/education-hub/blockchain-scalability
27. Agaev, V. (s. f.). Decentralized Oracles: Problematic by design. *www.linkedin.com*. https://www.linkedin.com/pulse/decentralized-oracles-problematic-design-vladimir-agaev/

28. Madrian, R. (2022, 30 abril). The problem facing decentralized oracles in Web 3.0 applications and their solutions. *Cryptopolitan*. https://www.cryptopolitan.com/problems-facing-decentralized-oracles-in-web-3-0/