# `iOracle`: Hybrid Indexing the Real-Time Block Data of Bitcoin on Lightweight Devices

iOracle Team

## Abstract

*We introduce `iOracle`, a decentralized Bitcoin block indexing tool that enables lightweight devices to perform blockchain indexing without synchronizing the full chain history. `iOracle` employs a 1-of-N trust consensus mechanism for data aggregation, ensuring that the system can settle disputes when at least one honest node is operational. Additionally, `iOracle` aims to establish a million-node class computation network, serving as a co-processing layer for Bitcoin.*

## 1. Introduction

### 1.1. The Evolution of Bitcoin's Utility

Since its creation, Bitcoin has primarily functioned as a digital payment tool, utilizing its underlying blockchain technology to facilitate secure and decentralized financial transactions worldwide. Traditionally, the main appeal of Bitcoin lies in its ability to operate as an alternative monetary system, independent of central bank oversight. However, the scope of Bitcoin's utility is experiencing a significant shift, broadening from strictly financial transactions to include a wider range of applications.

### 1.2. Challenges of Current Indexing Methods

The introduction of protocols like Ordinals has significantly expanded Bitcoin's functionality, enabling a host of innovative applications. These protocols permit the embedding of small data pieces directly onto the Bitcoin blockchain, transforming it into a versatile platform capable of supporting not only payments but also the storage and transfer of digital artifacts such as inscriptions and rune-like tokens. These inscriptions vary from simple messages to complex scripts and digital collectibles, all permanently recorded on the blockchain.

This diversification in usage has naturally led to an uptick in transactions involving these new features, such as trades and new inscriptions, driving the demand for more sophisticated indexing solutions. However, the ecosystem predominantly relies on centralized indexing and relay services to access Bitcoin events, placing services in both Centralized Exchanges (CEX) and Decentralized Finance (DeFi) within the Bitcoin space at risk of single points of failure and susceptibility to double-spending attacks. Current indexing methods, like those used by platforms such as Unisat, are largely centralized. Although these systems offer the benefits of rapid data retrieval and easy setup—requiring synchronization with a full

| | Centralized Indexer | Full-chain Index Oracle | Fully ZK Oracle | DePIN ZK Oracle |
|---|---|---|---|---|
| **Trust Model** | Single point trust | BFT | 1-of-N trust | 1-of-N trust |
| **Data Credibility** | Reputation-based guarantee | Tokenomics-based guarantee | Mathematics-based guarantee | Gaming- & Mathematics-based guarantee |
| **Trust Level** | Heavy | Medium | Trustless | Trustless |
| **Efficiency** | Seconds | 11-50 min | Minutes or seconds | Minutes or seconds (Need simulation) |
| **Query Cost** | $$$$ | $$$$$$ | $$$$ | $$$ |
| **Challenges** | Database hacks, DDoS, data asymmetry (double spending attack) | Costly and still very centralized | Uncertain ZK proof generation time and cost | Number of nodes for security and efficiency |
| **Example** | UniSat | Bitcoin Oracle | ORA | iOracle |

Table 1. Comparison of Indexing Models

Bitcoin node—they introduce substantial risks. Centralized indexes demand complete trust in the authority managing them, leading to potential security breaches, including attacks on central nodes and exposure to single points of failure.

Moreover, full-chain blockchain data indexing is necessary for most use cases, which entails processing raw events and reprocessing all data upon reload. For Bitcoin, syncing nearly 500GB of historical data represents a significant barrier for indexing nodes wishing to participate in a decentralized network, thus impeding the overall security of the ecosystem.

### 1.3. A Decentralized Indexing Solution

To address these deficiencies, a decentralized indexing solution is proposed, incorporating advanced cryptographic techniques such as zero-knowledge proofs. This innovative approach ensures that users do not need to trust a single central entity, thereby enhancing security while maintaining efficiency in Bitcoin block indexing. This system, known as the `iOracle` solution, marks a pivotal step forward in the evolution of blockchain technologies.

`iOracle` introduces an ”one-block-only” indexing solution leveraging zero-knowledge proof, where iNodes are only required to index and process data from the latest Bitcoin block and leaving data aggregation consensus to a verifiable ZK-processor follow 1-of-N trust model. In `iOracle`, data outputs are demonstrated with subsequent zero-knowledge proof of aggregation. The ZK-processor can sync historical states with iNodes for essential computations such as token transfer verification and client-side validation.

To mitigate these deficiencies, a decentralized indexing solution is proposed, integrating advanced cryptographic techniques like zero-knowledge proofs. This innovative approach eliminates the need to trust a single central authority, thus enhancing security while maintaining efficiency in Bitcoin block indexing. This system, known as the `iOracle` solution, represents a significant advancement in blockchain technology.

`iOracle` introduces a ’one-block-only’ indexing approach that utilizes zero-knowledge proofs. Under this system, iNodes are only required to index and process data from the most recent Bitcoin block, leaving the task of data aggregation consensus to a verifiable ZK-processor that follows a 1-of-N trust model. In

iOracle, data outputs are validated through subsequent zero-knowledge proofs of aggregation. The ZK-processor can synchronize historical states with iNodes for critical computations, such as token transfer verification and client-side validation, ensuring a robust and secure indexing framework.

## 2. Related Work

In the rapidly evolving field of blockchain technology, the efficiency and security of data access methods have become paramount, especially as these systems scale and diversify in application. Recent advancements have led to a proliferation of indexing solutions that aim to enhance the operability and reliability of blockchain networks, particularly for Bitcoin. This section of the paper reviews related works that explore various indexing models, highlighting their distinct approaches and the contexts in which they have been developed.

The traditional centralized indexers, such as those provided by 'unisat.io', have long been the backbone of blockchain data retrieval. They rely on a single-point trust model which, while efficient, suffers from significant vulnerabilities including database hacks and DDoS attacks. Such challenges underscore the inherent risks associated with centralized systems, which include data asymmetry and the potential for double-spending attacks.

In response to these vulnerabilities, several decentralized solutions have been proposed to distribute trust among multiple parties, thereby enhancing security and resilience. The Full-chain Index Oracle and the Fully zk Oracle, represented by platforms like 'bitcoin-oracle.network' and 'hyperoracle.io', respectively, utilize consensus mechanisms like Byzantine Fault Tolerance (BFT) and 1-of-N trust models. These systems not only address the single-point failure issue but also introduce novel economic and mathematical guarantees to bolster data credibility.

The DePIN zk Oracle, accessible through platforms such as 'otrack.xyz', represents a further innovation in this field by integrating zero-knowledge proofs to achieve a trustless environment. This approach promises high security and efficiency but also faces challenges, particularly regarding the uncertain times and costs associated with zk-proof generation, as well as the scalability issues tied to the number of participating nodes.

Each of these models offers unique benefits and faces distinct challenges, reflecting the diverse needs and priorities within the blockchain community. By examining these different approaches, this paper aims to provide a comprehensive overview of the current landscape of blockchain indexing technologies, setting the stage for a discussion on future developments and potential improvements in decentralized data indexing.

### 2.1. Zero-Knowledge Proof

The zero-knowledge proving system in ZK-processor is the final piece of the security puzzle. One of the primary factors enabling iOracle to achieve cost-effectiveness and speed superiority over other ZK coprocessors is that the ZKP circuit in iOracle is deterministic so that the time and cost required for proof generation can remain constant, allowing ample space for intricate computations and processing time.

Agnostic to arbitrary program logics, iOracle's ZKP system is designed to specifically validate the honest data aggregation and node election for every deploying program.

## 3. Protocol

Unlike The-Graph, iNodes in this system focus solely on indexing the latest block of a blockchain. This approach eliminates the need for iNodes to synchronize the entire blockchain history. Instead, they only process the most recent block because the updates to the data feed are managed by the zk-processor. This processor also provides state synchronization for iNodes, enabling more advanced computations such as the verification of inscription token transactions in Bitcoin.

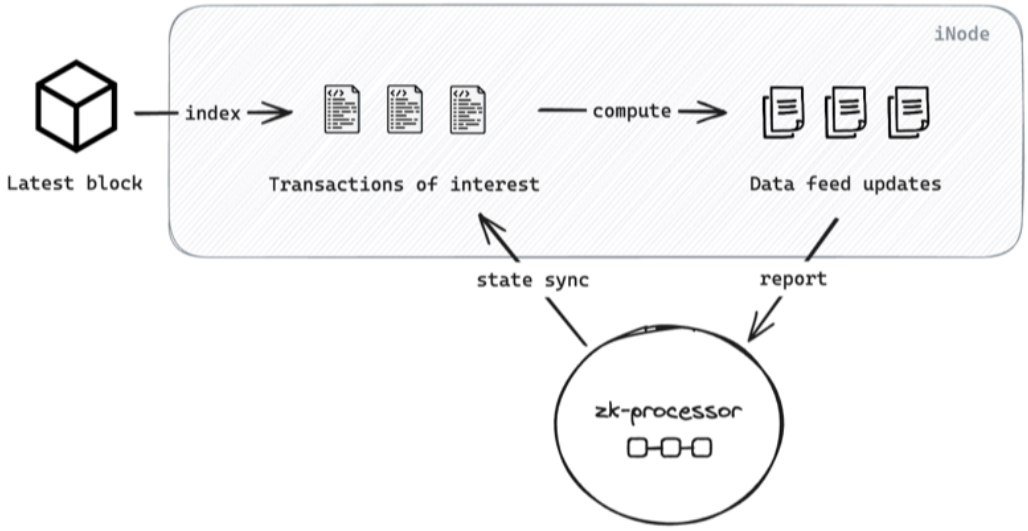


Figure 1. One-block-only indexing

### 3.1. System Architecture

### 3.2. Network Model

**`iNode`** refers to nodes with limited computational capacity and network bandwidth, such as mobile devices, routers, and Raspberry Pi systems. These nodes receive only the latest block data from full nodes each time.
**`ZK-Processor`** is a node that distributes indexing tasks to connected `iNode`, collects updated states from these `iNode`, and generates corresponding proofs related to these states. The `ZK-Processor` also handles all functions associated with the ZK circuits.
**`Reputation Curator`** manages the reputation of `iNode` based on their performance track records, which is deployed on an external blockchain with smart contract functionalities, e.g. Ethereum, Solana, etc. It randomly samples a collection of active `iNode` as reported by the ZK-Processors.
**Gateway Node** serves as an HTTP, WebSocket, or JSON RPC endpoint for decentralized application (dApp) clients to query `iOracle`.
**Query Node** is a node that participates in handling queries.
**dApp Client** is a frontend application developed by third parties that queries `iOracle`.

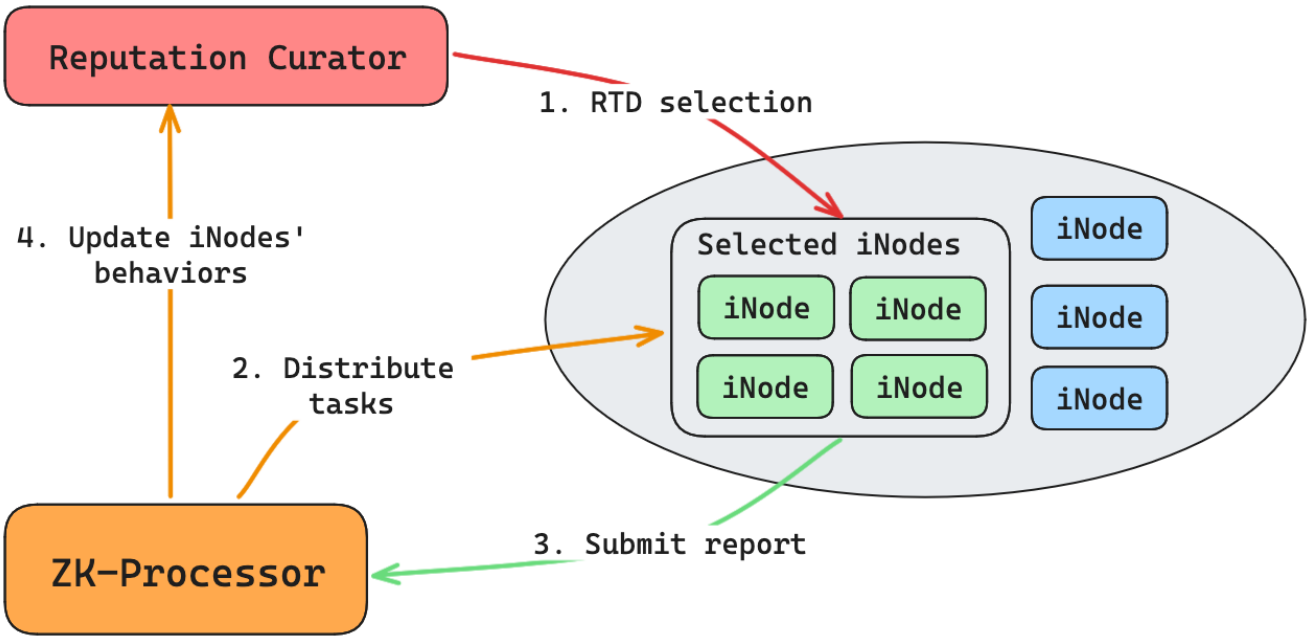


Figure 2. `iOracle` system architecture

### 3.3. Threat Model

**`iNode`** may be dishonest; they might not apply the correct transition function. However, we assume that at least one honest edge node is selected during the sampling process. This node is assumed to be online, willing to generate and distribute fraud proofs, and is not subject to an eclipse attack.
**`ZK-Processor`** is considered honest because all functions executed on it are implemented in ZK-circuits.
**States and Consensus** States created by adversarial `iNode` are invalid. The security model does not rely on the assumption that the majority of nodes are honest.

### 3.4. Protocol Requirements

In order to support a new class of data-intensive decentralized applications (dApps), a Decentralized Query Protocol must fulfill the following requirements:
**Metering**: Clients should have the capability to conveniently pay for every query processed by the network, thereby reducing counterparty risk for both clients and nodes.
**Predictable Performance**: Clients should expect and pay for predictable performance for queries executed against specific data sources.
**Data Availability**: Clients should be empowered to guarantee and financially support the availability of essential data required for executing queries against designated data sources.
**Price Efficiency**: The protocol should enable the creation of efficient and competitive marketplaces, allowing clients to acquire queries, performance, and data availability with ease and effectiveness.
**Incentive Alignment**: The protocol should align incentives among dApp clients, `iNode`, and `ZK-Processor` to foster network growth and generate positive network effects.

### 3.5. Protocol Parameters

**Report State Structure** $S_i$

| Name | Remark |
|---|---|
| **dataRoot**$_i$ | The root of the Merkle tree of the transactions included in the block |
| **stateTree**$_i$ | The Merkle tree of the hashed intermediate states |
| **state**$_i$ | The state until block $i$ |
| **edgeRoot**$_i$ | The root of the Merkle tree of participating `iNode` in this report |

Table 2. Parameters

### 3.6. Decentralized Reputation Construction

The network relies on a decentralized reputation to foster Sybil-resistant task distribution. Upholding the principle of decentralization, `iOracle` allows `iNode` to join our network permissionlessly. Furthermore, we leverages a sophisticated decentralized `Reputation Curator` to diligently monitor and evaluate the performance of every node within the network. Each `iNode` is associated with a reputation score that reflects its historical performance. The `Reputation Curator` employs a reputation-based `iNode` election algorithm to allocate computation tasks, the `iNode` earns rewards for honestly executing computation tasks distributed by the `ZK-Processor`.

We categorize the behaviors of `iNode` as follows, and the `iNode` behaviors verified by zero-knowledge proofs:

**Positive behavior**: The `iNode` honestly executes the distributed tasks and submit the computation results to `ZK-Processor`.

**Negative behavior**: 1) The `iNode` tries to submit fault computation results to deceive `ZK-Processor`. 2) The `iNode` fails to submit computation results due be network problems (e.g. being offline)

**`iNode` Reputation Initialization**: New nodes entering the network are assigned an initial reputation score, which increases with each honest report submission. `iNodes` that accumulate higher reputation scores are more likely to be selected for task execution. This decentralized reputation system promotes network performance through positive incentives and bolsters network security. Entities attempting to carry out Sybil attacks face significant challenges due to the prevalence of high-reputation nodes within the network.

**`iNode` Reputation Tracking**: Each `iNode`'s public key is registered in the `Reputation Curator`. An `iNode`'s reputation increases with each honest state submission, thereby enhancing its mining efficiency. However, if an `iNode` submits a single dishonest report, its reputation will experience a significant decrease, thereby providing resistance to DDoS attacks.

**Reputation-based Task Distribution Algorithm (RTD)**: During each task distribution cycle, `ZK-Processor` employs a deterministic algorithm to select a specific number of nodes to form a shard. This selection process takes into account the reputation of all available nodes, which is recorded in the `Reputation Curator` beforehand.

The probability $\Pr(r)$ of selecting a node with reputation $r$ is given by the formula:

$$\Pr(r) = \frac{r^p}{\sum_{r_i} r_i^p} \propto r^p$$

where $r_i$ represents the reputation of each node and $p$ is a parameter that adjusts the influence of reputation on the selection probability.

### 3.7. Zero-Knowledge Proof of Aggregation

Each final data output is accompanied by a Zero-Knowledge Proof (ZKP) verifying all processes executed in the`ZK-Processor`, corresponding to the original task carried out by the `iNode` network.
This verification essentially confirms three key aspects:

- The output has a designated number of signatures from a group of iNodes registered in the reputation smart contract.
- All signatures validate the signing to the same output and its corresponding task script.
- The ZK-processor pre-selected this group of iNodes based on the Reputation-based Task Distribution (RTD) algorithm.

Here, we assume `iOracle` relies on an external blockchain to host the `Reputation Curator`.

### 3.8. Protocol Implementation

In this section, we detailed introduce the implementation of our protocol, and the main functionalities of each component is detailed in Figure 3, Figure 4 and Figure 5.

1. An `iNode` sends its heartbeat to `Reputation Curator` to tell `Reputation Curator` that it is willing to join the indexing tasks in the following $t$ rounds.
2. The `Reputation Curator` randomly samples $N(N \leq M)$ `iNode` from given alive `iNode` and publishes them to the `ZK-Processor`, i.e. $\{e_1, e_2, \ldots, e_N, i, t\}$ where $i$ is target indexing block $i$, and $R$ is the number of attempts. Meanwhile, the `Reputation Curator` also publish the Merkled Tree of the alive `iNode` submitted by `ZK-Processor`.
3. `ZK-Processor` distributes indexing task to selected `iNode` $e_i$ with payload $(S_{i-1}, P, i)$, where $P$ is the specification of indexing task.
4. The `iNode` receives the payload from `ZK-Processor` and downloads block $i$ from a full node it connects to. The `iNode` applies the transition function defined according to the given protocol or program $P$. Once finalizing the computation, the `iNode` uploads report to the `ZK-Processor`, which is $\{e- > (S_i, \text{dataRoot}_i, \text{stateRoot}_i)\}$
5. The `ZK-Processor` collects $X$ replicas of reports from `iNode` within a given $\delta$ timeframe. It checks the consistency of collected replicas via $\text{stateRoot}_i$. If not the same, `ZK-Processor` repeats Step 2. If all of them are the same, it generates and publishes a ZK proof $ZKP_i$ for updated states. The `Reputation Curator` updates the reputation of `iNode` which contributed to this report generation.
6. If the `ZK-Processor` is successful to collect $X$ replicas of reports after $R$ attempts, it starts to generate proofs and updates misbehaved `iNode` to `Reputation Curator`.
7. If the `ZK-Processor` fails to collect $X$ replicas of reports after $R$ attempts, the protocol stops.

**1-of-N Trust:** `iOracle` employs a hybrid consensus mechanism that strikes a balance between low cost and high security. In `iOracle`, aggregation is managed by a ZK-processor using a 1-of-N trust model. If an inconsistency is detected in the reports provided by any nodes, the system initiates arbitration. During

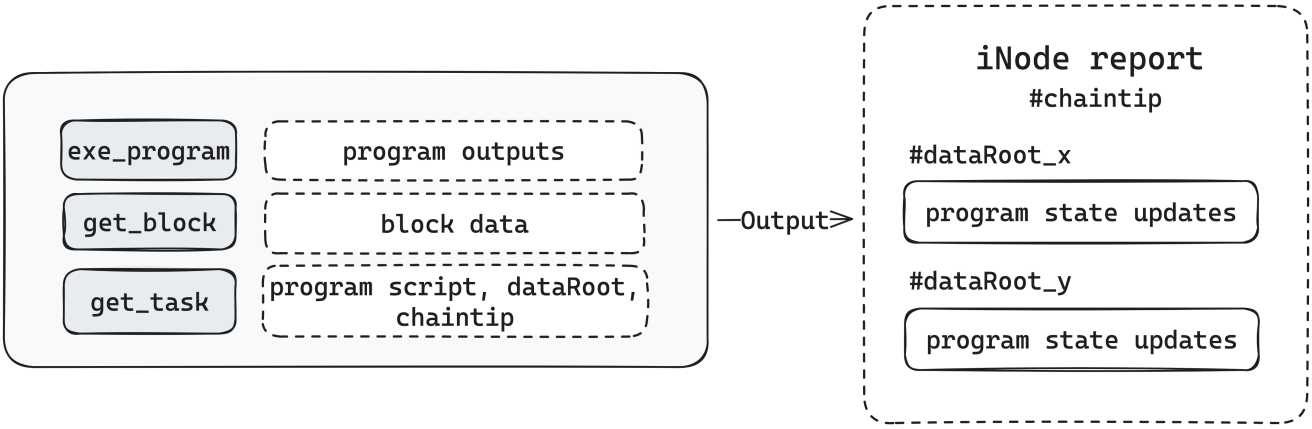


Figure 3. iNode

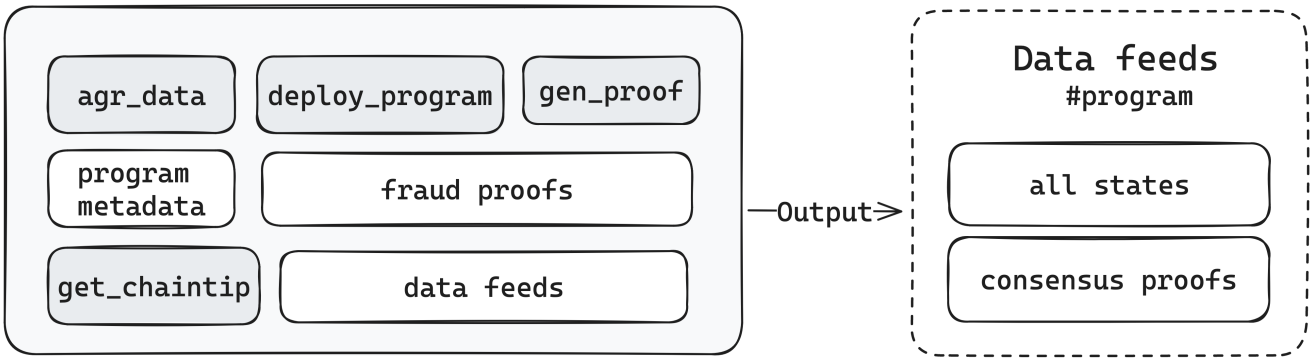


Figure 4. ZK-Processor

arbitration, another shard of nodes is elected and the search for consistency continues until consensus is reached.

### 3.9. Sharding

### 3.10. Scalability

## 4. Token Economics

TBD

## 5. Conclusion

iOracle introduces an innovative framework that enhances blockchain functionality and security through advanced cryptographic methods and decentralized mechanisms. The hybrid consensus model, which incorporates a 1-of-N trust system, ensures that iOracle maintains both cost-efficiency and robust security. This model adeptly handles inconsistencies through a systematic arbitration process, demonstrating the system's capability to adapt and sustain integrity even when discrepancies arise. By leveraging Zero-Knowledge Proofs for process verification and employing a reputation-based task distribution, iOracle not only secures data transactions but also fosters a trustless environment where

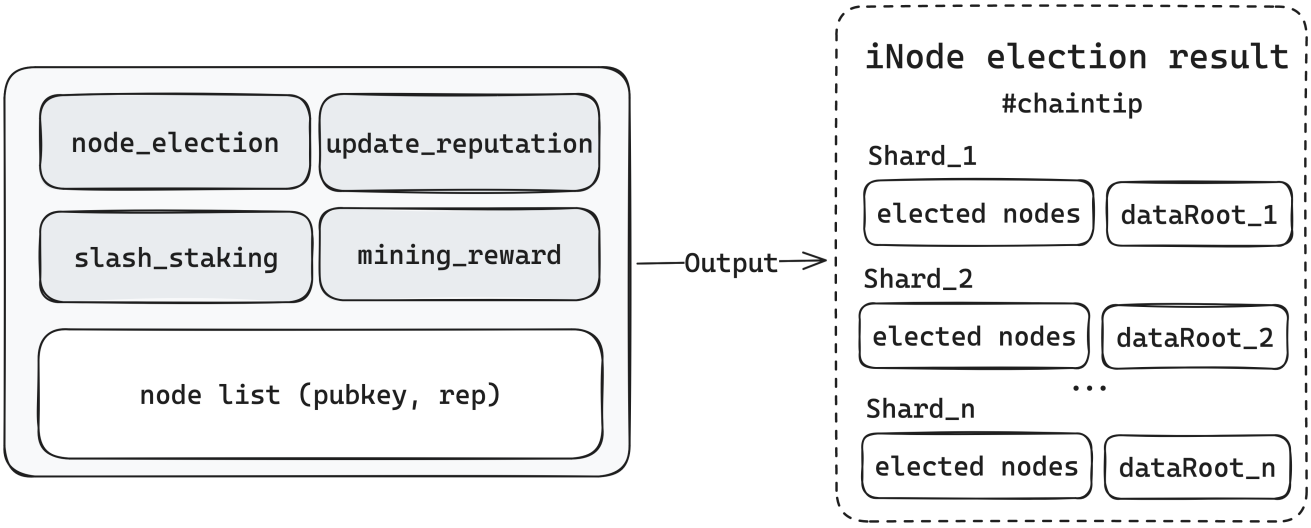


Figure 5. `Reputation Curator`

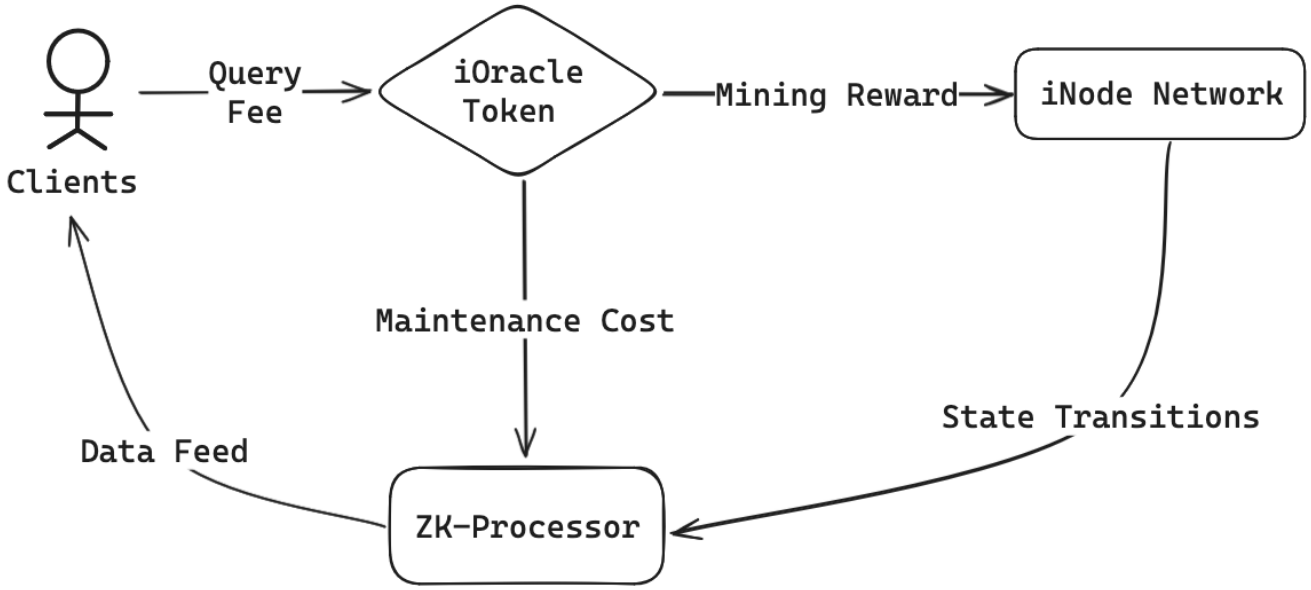


Figure 6. Tokenomics

nodes are incentivized to operate honestly. As such, `iOracle` stands as a significant advancement in blockchain technology, offering a scalable, secure, and efficient solution for decentralized applications.