What Is the Blockchain Oracle Problem?

August 27, 2020 Chainlink

The blockchain oracle problem is one of the most important barriers to overcome if smart contracts on networks like Ethereum are to achieve mass adoption across a wide variety of markets and use cases.

Smart contracts running on blockchains offer immense potential to redefine the way independent entities engage in contractual agreements and exchange value. Operating separately from the smart contract economy is the much larger non-blockchain digital economy, made up of all the Internet-connected devices computing online. A byproduct of digital infrastructure is an ever-expanding reservoir of data and APIs, which provide insights into how the world works; e.g., Internet search results presenting popular discussion topics in society or IoT sensors showcasing common traffic patterns.

Blockchain-based smart contracts and traditional data and API economies have immense potential to combine into hybrid smart contracts and form the future architecture of data-driven automation, but the question is how do these two worlds connect? This encompasses the crux of the "oracle problem" and will be the focus of this article.

The article is broken down into five key sections:

- The oracle problem
- The job of an oracle
- Why blockchains like Ethereum don't offer native oracle solutions
- The security risks of centralized oracles
- Chainlink, the standard for secure and reliable decentralized oracle networks

The Oracle Problem

The oracle problem revolves around a very simple limitation—blockchains cannot pull in data from or push data out to any external system as built-in functionality. As such, blockchains are isolated networks, akin to a computer with no Internet connection. The isolation of a blockchain is the precise property that makes it extremely secure and reliable, as the network only needs to form consensus on a very basic set of binary (true/false) questions using data already stored inside of its ledger. These questions include: did the public key holder sign the transaction with their corresponding private key, does the public address have enough funds to cover its transaction, and is the type of transaction valid within the particular smart contract? The very narrow focus of blockchain consensus is why smart contracts are referred to as being deterministic—they execute exactly as written with a much higher degree of certainty than traditional systems.

However, for smart contracts to realize upwards of 90% of their potential use cases, they must connect to the outside world. For example, financial smart contracts need market information to determine settlements, insurance smart contracts need IoT and web data to make decisions on policy payouts, trade finance contracts need trade documents and digital signatures to know when to release payments, and many smart contracts want to settle in fiat currency on a traditional payment network. None of this information is inherently generated within the blockchain, nor are these traditional services directly accessible.

Bridging the connection between the blockchain (on-chain) and the outside world (off-chain) requires an additional and separate piece of infrastructure known as an oracle.

What Is a Blockchain Oracle?

A blockchain oracle is a secure piece of middleware that facilitates communication between blockchains and any off-chain system, including data providers, web APIs, enterprise backends, cloud providers, IoT devices, e-signatures, payment systems, other blockchains, and more. Oracles take on several key functions:

Listen – monitor the blockchain network to check for any incoming user or smart contract requests for off-chain data.

Extract – fetch data from one or multiple external systems such as off-chain APIs hosted on third-party web servers.

Format – format data retrieved from external APIs into a blockchain readable format (input) and/or making blockchain data compatible with an external API (output).

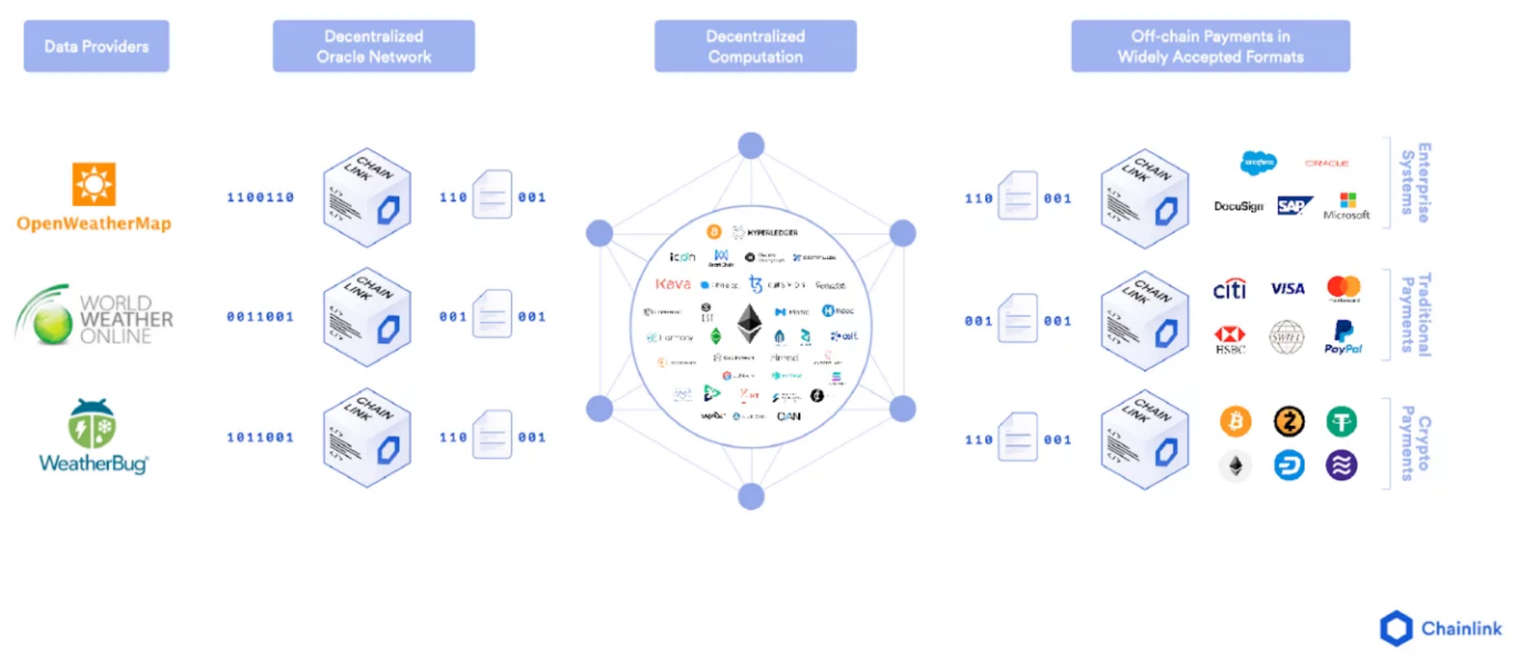
Validate – generate a cryptographic proof attesting to the performance of an oracle service using any combination of data signing, blockchain transaction signing, TLS signatures, Trusted Execution Environment (TEE) attestations, or zero-knowledge proofs.

Compute – perform some type of secure off-chain computation for the smart contract, such as calculating a median from multiple oracle submissions or generating a verifiable random number for a gaming application.

Broadcast – sign and broadcast a transaction on the blockchain in order to send data and any corresponding proof on-chain for consumption by the smart contract.

Output (optional) – send data to an external system upon the execution of a smart contract, such as relaying payment instructions to a traditional payment network or triggering actions from a cyber-physical system.

Carrying out the functions above requires that the oracle system operate both on and off the blockchain simultaneously. The on-chain component is for establishing a blockchain connection (to listen for requests), broadcasting data, sending proofs, extracting blockchain data, and potentially performing computation on the blockchain. The off-chain component is for processing requests, retrieving and formatting external data, sending blockchain data to external systems, and performing off-chain computation for greater scalability, privacy, security, and various other smart contract enhancements.



Why Blockchains Can't Solve the Oracle Problem

Blockchains are highly secure and reliable because of a few specific design principles. As described above, a blockchain only needs to form consensus on very basic binary questions using data already stored on its own ledger. The blockchain's ledger is deemed to be true because it leverages decentralization to redundantly validate every piece of data using all the nodes in the network. It also uses decentralization to maintain the integrity of its consensus algorithm (PoW, PoS, etc.), ensuring the protocol rules only change when a substantial portion of the network signs off on it (e.g., 51%). These properties provide strong guarantees of computational and data storage determinism, especially in highly decentralized and Sybil-resistant networks.

However, blockchains are not well suited to answer questions that delve into the realm of subjectivity or require external data that is not easily accessible to every node in the network. For example, a simple question like 'What is the market price of Bitcoin?' or 'What is the weather in New York?' can elicit a wide range of different answers that may vary depending on what data source they use and when they request data from the source. The question then becomes, what is the correct answer and how can it be verified as true?

Introducing subjectivity at the base layer of the blockchain opens up Pandora's box to a whole host of security, reliability, and governance concerns, putting at risk the very value proposition blockchains aim to provide.

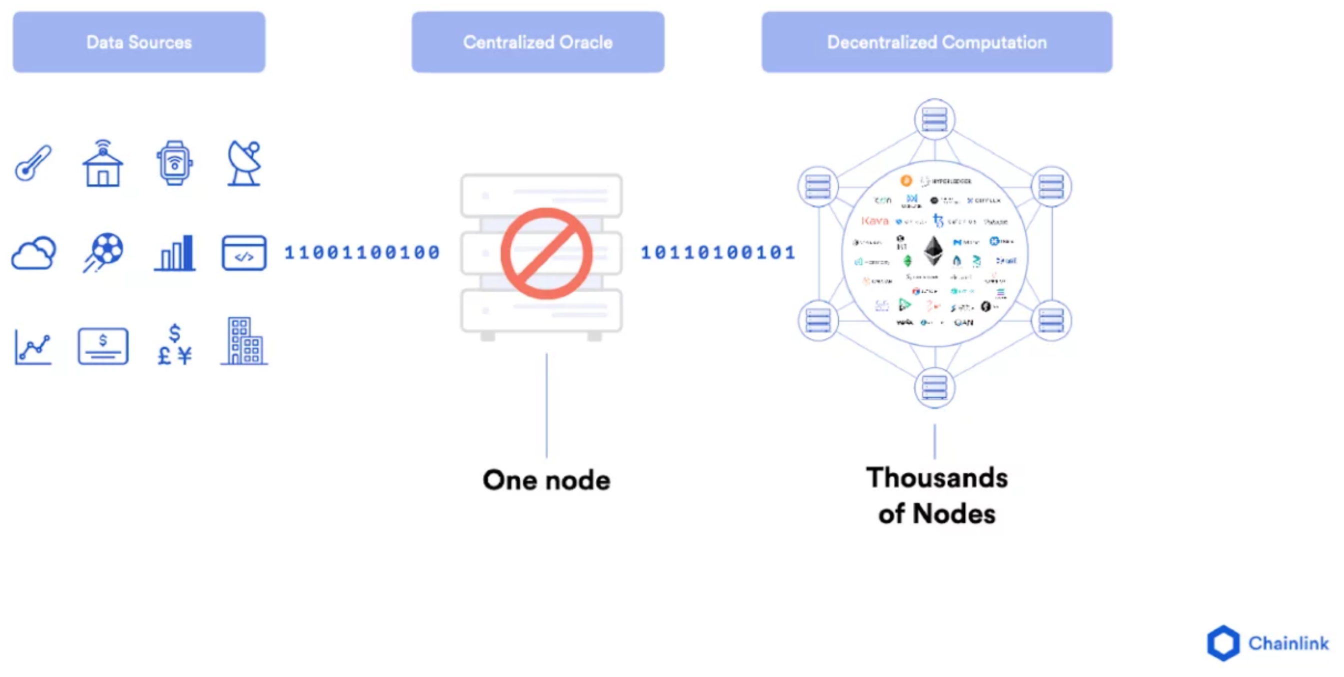
One major concern is how to ensure external data inputted into the blockchain is high quality? Even a basic data request for the price of Bitcoin is quite challenging because simply looking at a website or a single exchange may not be as accurate or reliable as a paid API subscription to a professional data aggregator that has decades of experience filtering data and creating market coverage and is financially incentivized to maintain high-quality services. It's extremely difficult to manage and enforce quality for off-chain data submitted by blockchain nodes since anyone can pseudo-anonymously run a node and submit answers, even if they are not willing to buy a subscription to a high-quality off-chain API. If data quality was enforced, the blockchain would then have a lower upper bound on decentralization since the costs of running nodes would increase for every new oracle job on the network, affecting the security of all other applications running on the particular blockchain.

Another major concern is scalability. Every time a new data source needs to be added to the network or an existing data aggregation method must be adjusted, it requires massive social governance coordination to get every node in the network to agree and upgrade their software. The addition of governance overhead leads to increased friction, slower development of core blockchain features (such as PoS and sharding), and major limits on oracle innovation. Ultimately, the more complexity there is at the base layer of the blockchain, the more attack surface and risk to all applications that run on it. Even applications not using oracles or not involved with confrontational data requests will get caught in the crossfire and potentially get disrupted if the entire chain comes to a halt because of an oracle issue.

It's for these reasons and many more that oracles are not integrated into the base layer of any major blockchain, but instead operate as separate networks. This ensures that blockchains have a lower attack surface and retain their determinism by maintaining a singular focus on consensus, while oracles have the required flexibility needed to generate determinism from a complex and subjective off-chain world without creating dependencies and limitations that put at risk all other applications.

Centralized Blockchain Oracles Introduce Major Risks

The entire point of a smart contract is to achieve determinism through technological enforcement of the contract's terms as opposed to probabilistic execution carried out by human enforcement. To achieve this end, the blockchain cannot have any single point of failure—a feature that must extend to the oracle if determinism of smart contracts is to be maintained end-to-end. Why have a multi-million dollar contract function as a smart contract on a fully decentralized blockchain if a single centralized oracle can control the inputs that determine the contract's outcome?



A centralized oracle is a central point of failure in the smart contract

Whether it's the development team running the oracle themselves or a centralized third-party service, both scenarios give excessive power to a single entity to influence the contract via control of the oracle. While the centralized oracle operator may operate with the best intentions, they are still subject to all the common centralized problems of today like downtime, DDOS attacks, hacks, and accidental incompetence, all of which put users' funds at risk.

Even the noblest centralized entities can come under pressure once the value of the contract scales, opening them up to bribes, intimidation, and regulatory pressure, which only require one person involved in the operation to go rogue. This model is not scalable and doesn't fit with the idea of decentralized infrastructure being a key driver of secure and reliable automation.

In order to overcome these shortcomings, oracles need to create the same security and reliability guarantees of a blockchain, although in a different manner given their many differences.

Chainlink: The Standard for Secure and Reliable Oracles

In order to bring determinism to the oracle layer, Chainlink has developed a network of decentralized oracle networks (DONs), with each DON involving a combination of multiple security techniques needed to service a particular use case.

Open-source – being an open-source technology allows the wider blockchain community to independently verify the security and reliability of Chainlink's source code and functions, as well as contribute to its improvement.

External Adapters – allowing nodes to securely store API keys and manage account logins enables smart contracts to retrieve data from any external system and API, including those that are password/credential protected.

Decentralization – employing decentralization at the node and data source level ensures no one node or data source is a single point of failure, providing users strong guarantees that data will be available, delivered on time, and resistant to manipulation.

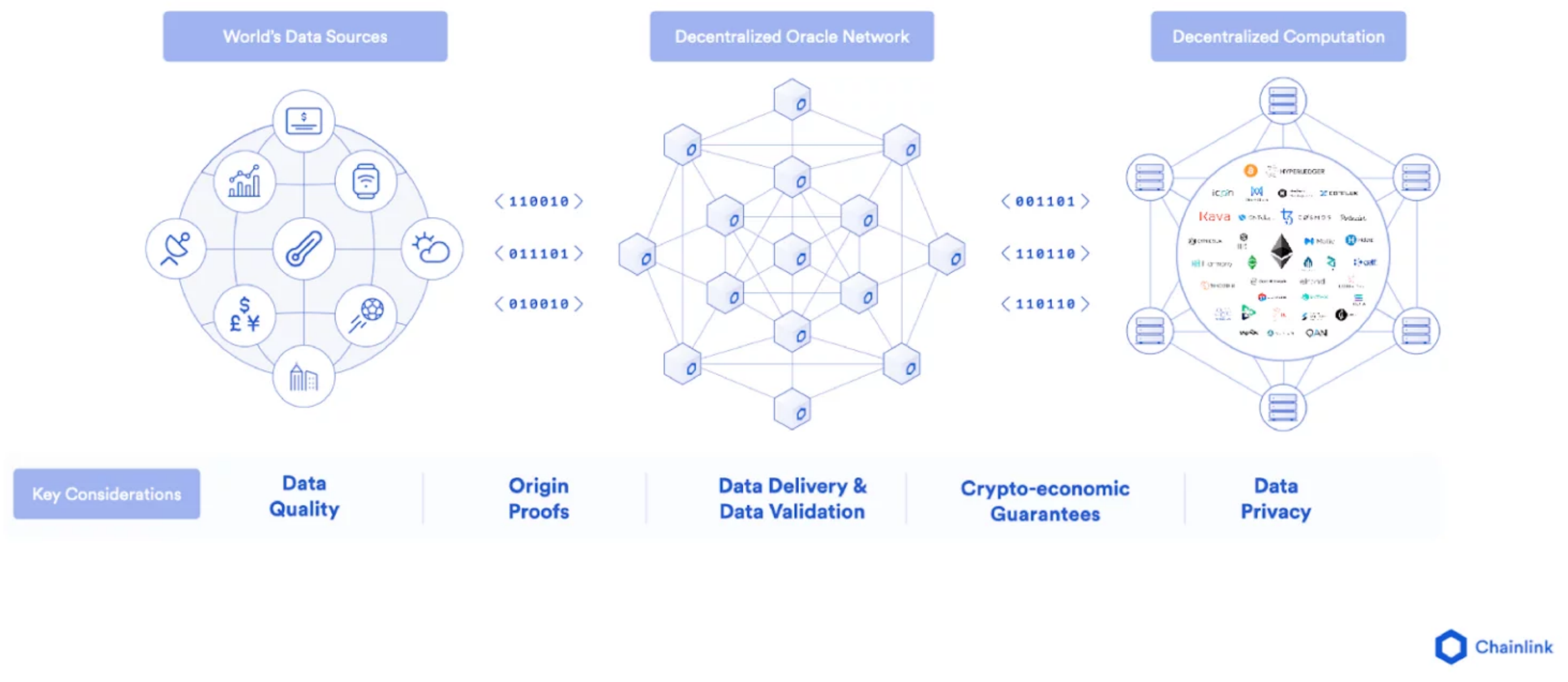
Data Signing – having nodes cryptographically sign the data they provide to smart contracts allows users to identify which nodes sent data and look at their past history to determine their performance quality.

Service Agreements – using binding on-chain agreements between the requesting smart contract and the oracle provider that outline the terms of the oracle service and penalties/rewards for performance provides users with enforceable guarantees on the quality of their off-chain data requests.

Reputation Systems – feeding signed on-chain data into reputation systems allows users to make informed decisions about which nodes are good and which nodes are not based on a variety of metrics like successful jobs performed, list of clients served, average response time, etc.

Certification Services – enabling nodes to increase their security and reliability by obtaining any number of certifications can provide users additional guarantees like KYC, geographic location of the node, security reviews of their infrastructure, and more.

Advanced Cryptography and Hardware – providing flexibility for more advanced cryptography (like zero-knowledge proofs) and hardware (such as trusted execution environments) enables oracles to perform additional functions like proving the origin of data (e.g. specific data came from a specific server), keeping data confidential, performing off-chain computation, and more.



Decentralized Oracle Networks (DONs) allow smart contracts to securely connect with external data and systems.

These are just some of the many features offered by Chainlink that provide users with a whole set of guarantees to ensure a highly secure and reliable oracle mechanism. By building out these key features on Chainlink, smart contracts on any blockchain can now access off-chain data without sacrificing its core value of determinism, providing a solid foundation from which to build out the future of data-driven automation.

Follow us on Twitter to get notified of upcoming article releases, join our Telegram or Reddit for general news on Chainlink, or take part in the technical discussion on our Discord. Also, check out other, more advanced articles such as: