Querying APIs to Generate Custom Oracles - Devcon Hacker House 2024

Oracles are essential instruments built into blockchains that indicate off-chain data on-chain. In our project, we created a method for any user to deploy an oracle on Eigenlayer by providing the API.

The Intersection of AVS and Custom Oracles

By using Web Assembly System Interface (WASI), Actively Validated Services (AVS) bring a new level of sophistication to how custom oracles validate data, processing complex services off-chain. Through AVS, custom oracles undergo a validation mechanism that allows operators to verify the integrity of potential on-chain data. Oracles can deliver high-fidelity and reliable data feeds to dApps without an obligation to create and run their independent validator networks. Moreover, AVS supports creating decentralized oracle networks, which will integrate real-world data into blockchain ecosystems. The combination of AVS and authorized oracles patches mitigates vulnerabilities by making the connection between oracle services and smart contracts without complicating their communication process. This makes them much more secure for broader use in a chain-based environment.

Advantages of AVS Integration

- Advanced-Data Validation Off-chain

: AVS can perform smart checks and calculations on off-chain data fetched from third-party APIs before finally submitting to the chain.

- Enhance Efficiency

: Offload complex processing from the blockchain to an AVS, optimizing resource use and lowering on-chain gas costs.

- Enhanced Security

: AVS can prevent wrong or malicious data from entering the system and thus corput the smart contracts with false information.

- Enhanced Flexibility

: Oracles can be customized to support different types of data formats and interfaces with widely disparate APIs for a variety of use cases.

Network Participants

Role

Description

Restakers

Earn an additional yield by using EigenLayer to stake ETH and help secure these oracles, in turn enhancing the reliability of the system.

Operators

Operate off-chain infrastructure that pulls and processes external data through AVS. Operators must validate data using the provided requirements and submit these confirmed values to the on-chain oracle.

Developers

Developers configure the custom oracle parameters (e.g., API endpoints, data processing logic) and design the validation rules implemented by AVS.

End Users

Decentralized applications and the users of those applications need accurate, tested data to execute functions from financial transactions to predictions.

Oracle Creation Flow

[

982×996 52.5 KB

](https://canada1.discourse-cdn.com/flex028/uploads/eigenlayer/original/2X/0/0b527fe7d5c9813388d4564e14b7e340cc154989.png)

Purpose

The incorporation of Actively Validated Services (AVS) within a custom oracle enhances the quality and trustiness of off-chain data on-chain. AVS supports off-chain validation and processing for indisputable facts, giving smart contracts concrete data that passes the test of trust, improving security and efficiency for decentralized applications. Restakers, operators, developers and end-users all benefit from this system working together seamlessly to create a stronger and more flexible blockchain data ecosystem.

Future enhancements include enabling the execution of arbitrary WASM-compiled code off-chain, further expanding the capabilities of AVS-powered oracles. This would allow developers to implement highly specialized data processing logic and complex validation methodologies, ensuring the data on-chain remains both dynamic and trustworthy for a wide range of blockchain applications.

Feedback

In order to deploy AVS, we had to access the Layer testnet but while implementing curl commands with the RPC links, we encountered a 504 error, entailing that our request could not be completed due to the lack of a timely response to our server. This was due to one of the Layer repos having outdated urls for the curl commands.

Another roadblock we encountered was while we were trying to access the faucet for funds to deploy on testnet. We used the Layer docs to implement the Faucet API function code but upon using it, we found out that the API in the Layer SDK does not have funds and that the only way to access faucet funds was through a game on a telegram bot. The docs were misleading for this reason.

We also had an issue passing multiple environment variables using the CLI when deploying a WASI component. There is not any specific documentation regarding this exact syntax.

Overall, through our developer experience, we enjoyed building with Layer and integrating AVS into our project. We do believe that our experience and the experiences of other developers would be improved if the repos and docs had consistent links and functions that would allow us to access the testnet easier. Additionally, if the telegram bot is the preferred method to access faucet funds, we believe that it would helpful if the information for that method was provided on the docs.

Authors

Tanay Appannagari, Ravi Riley, Rohan Patra, Jameson Crate

Links

GitHub

Website