OP Stack Components

The OP Stack is a common development stack for building L2 blockchain ecosystems, built by the Optimism Collective to power Optimism.

The OP Stack is best thought of as a collection of software components maintained by the Optimism Collective that either help to define new layers of the stack or fit in as modules within the stack.

Because the OP Stack is a work in progress, the different layers and modules are still evolving. This page sketches out the different conceptual layers of the stack as they exist today and introduces some of the modules that fit into those layers. This doesn't include all of the modules or layers that may exist in the future, but it gives a good overview of the major OP Stack components.

⚠ Please note that not all of the modules described on this page already exist in a production state — these are explicitly marked as either "in development "or "proposed ."

Major Components

Layers

Data Availability

The Data Availability Layer defines where the raw inputs to an OP Stack based chain are published. An OP Stack chain can use one or more Data Availability module to source its input data. Because an OP Stack chain is derived from the Data Availability Layer, the Data Availability module(s) used have a significant impact on the security model of a system. For example, if a certain piece of data can no longer be retrieved from the Data Availability Layer, it may not be possible to sync the chain.

Ethereum DA

Ethereum DA is currently the most widely used Data Availability module for the OP Stack. When using the Ethereum DA module, source data can be derived from any piece of information accessible on the Ethereum blockchain. This includes Ethereum calldata, events, and 4844 data blobs.

- Specifications(opens in a new tab)
- Source code(opens in a new tab)

Sequencing

The Sequencing Layer determines how user transactions on an OP Stack chain are collected and published to the Data Availability Layer module(s) in use. In the default Rollup configuration of the OP Stack, Sequencing is typically handled by a single dedicated Sequencer. Rules defined in the Derivation Layer generally restrict the Sequencer's ability to withhold transactions for more than a specific period of time. In the proposed future, Sequencing will be modular such that chains can easily select and change the mechanism that controls their current Sequencer.

Single Sequencer

The default Sequencer module for the OP Stack is the Single Sequencer module in which a dedicated actor is given the ability to act as the Sequencer. The Single Sequencer module allows a governance mechanism to determine who may act as the Sequencer at any given time.

Multiple Sequencer (proposed)

A simple modification to the Single Sequencer module is the Multiple Sequencer module in which the Sequencer at any given time is selected from a pre-defined set of possible actors. Individual OP Stack based chains would be able to determine the exact mechanism that defines the set of possible Sequencers and the mechanism that selects a Sequencer from the set.

Derivation

The Derivation Layer defines how the raw data in the Data Availability Layer is processed to form the processed inputs that are sent to the Execution Layer via the standardEthereum Engine API(opens in a new tab) . The Derivation Layer may also use the current system state, as defined by the Execution Layer, to inform the parsing of raw input data. The Derivation Layer can be modified to derive Engine API inputs from many different data sources. The Derivation Layer is typically tied closely to the Data Availability Layer because it must understand how to parse any raw input data.

Rollup

The Rollup module derives Engine API inputs from Ethereum block data, Sequencer transaction batches, Deposited transaction events, and more.

- Specifications(opens in a new tab)
- Source code(opens in a new tab)

Indexer (proposed)

The Indexer module is a proposed Derivation Layer module that would derive Engine API inputs when transactions are sent to, events are emitted by, or storage is modified in specific smart contracts on a Data Availability Layer module like Ethereum DA.

Execution

The Execution Layer defines the structure of state within an OP Stack system and defines the state transition function that mutates this state. State transitions are triggered when inputs are received from the Derivation Layer via the Engine API. The Execution Layer abstraction opens up the door to EVM modifications or different underlying VMs entirely.

EVM

The EVM is an Execution Layer module that uses the same state representation and state transition function as the Ethereum Virtual Machine. The EVM module in the Ethereum Rollup configuration of the OP Stack is alightly modified(opens in a new tab) version of the EVM that adds support for L2 transactions initiated on Ethereum and adds an extra L1 Data Fee to each transaction to account for the cost of publishing transactions to Ethereum.

- Specifications(opens in a new tab)
- (where it differs fromgeth(opens in a new tab)
-)
- Source code(opens in a new tab)

Settlement Layer

The Settlement Layer is a mechanism on external blockchains that establish aview of the state of an OP Stack chain on those external chains (including other OP Stack chains). For each OP Stack chain, there may be one or more Settlement mechanisms on one or more external chains. Settlement Layer mechanisms areread-only and allow parties external to the blockchain to make decisions based on the state of an OP Stack chain.

The term "Settlement Layer" has its origins in the fact that Settlement Layer mechanisms are often used to handle withdrawals of ETH and tokens out of a blockchain. This sort of withdrawal system first involves proving the state of the target blockchain to some third-party chain and then processing a withdrawal based on that state. The Settlement Layer, at its core, simply allows a third-party chain to become aware of the state of the target chain.

Once a transaction is published and finalized on the corresponding Data Availability layer, the transaction is also finalized on the OP Stack chain. Short of breaking the underlying Data Availability layer, it can no longer be modified or removed. It may not be accepted by the Settlement Layer yet because the Settlement Layer needs to be able to verify transactionresults , but the transaction itself is already immutable.

Attestation-based Fault Proof

An Attestation-based Fault Proof mechanism uses an optimistic protocol to establish a view of an OP Stack chain. In optimistic settlement mechanisms generally,Proposer entities can propose what they believe to be the current valid state of the OP Stack chain. If these proposals are not invalidated within a certain period of time (the "challenge period"), then the proposals are assumed by the mechanism to be correct. In the Attestation Proof mechanism in particular, a proposal can be invalidated if some threshold of pre-defined parties provide attestations to a valid state that is different than the state in the proposal. This places a trust assumption on the honesty of at least a threshold number of the pre-defined participants.

- Specifications(opens in a new tab)
- (calledwithdrawal transactions
-)
- Source code(opens in a new tab)

Fault Proof Optimistic Settlement (proposed)

A Fault Proof Optimistic Settlement mechanism is mostly identical to the Attestation-based Fault Proof mechanism used today but it replaces the MultiSig challenger with a permissionless fault proving process. A correctly constructed fault proof should be able to invalidate any incorrect proposals during the allocated challenge period. This places a trust assumption on

the correctness of the fault proof construction. At this time, work on the development of a Fault Proof mechanism is well underway.

Validity Proof Settlement (proposed)

A Validity Proof Settlement mechanism uses a mathematical proof to attest to the correctness of a proposed view. A proposed state will not be accepted without a valid proof. This places a trust assumption on the correctness of the validity proof construction.

Governance

The Governance Layer refers to the general set of tools and processes used to manage system configuration, upgrades, and design decisions. This is a relatively abstract layer that can contain a wide range of mechanisms on a target OP Stack chain and on third-party chains that impact many of the other layers of the OP Stack.

MultiSig Contracts

MultiSig Contracts are smart contracts that carry out actions when they receive a threshold of signatures from some pre-defined set of participants. These are often used to manage upgrades of components of an OP Stack based system. Currently, this is the mechanism used to manage upgrades of the bridge contracts on OP Mainnet. The security of a MultiSig Contract system depends on many different factors, including the number of participants, the threshold, and the safety procedures of each individual participant.

Governance Tokens

Governance Tokens are widely used to decentralize decision making. Although the exact functionality of a Governance Token varies on a case-by-case basis, the most common mechanisms allow token holders to vote on some subset of decisions that a project must make. Voting can either be carried out directly or via delegation.

Superchain Explainer Rollup Protocol Overview