AnyTrust Protocol

AnyTrust is a variant ofArbitrum Nitro technology that lowers costs by accepting a mild trust assumption.

The Arbitrum protocol requires that all Arbitrum nodes, including validators (nodes that verify correctness of the chain and are prepared to stake on correct results), have access to the data of every L2Transaction in theArbitrum chain 's inbox. An Arbitrum rollup provides data access by posting the data (in batched, compressed form) on L1 Ethereum as calldata. The Ethereum gas to pay for this is the largest component of cost in Arbitrum.

AnyTrust relies instead on an external Data Availability Committee (hereafter, "the Committee") to store data and provide it on demand. The Committee hasN members, of which AnyTrust assumes at least two are honest. This means that ifN - 1 Committee members promise to provide access to some data, at least one of the promising parties must be honest. Since there are two honest members, and only one failed to make the promise, it follows that at least one of the promisers must be honest — and that honest member will provide data when it is needed to ensure the chain can properly function.

Keysets

A Keyset specifies the BLS public keys of Committee members and the number of signatures required for a Data Availability Certificate to be valid. Keysets make Committee membership changes possible and provide Committee members the ability to change their keys.

A Keyset contains:

- the number of Committee members, and
- for each Committee member, a BLS public key, and
- the number of Committee signatures required.

Keysets are identified by their hashes.

An L1 KeysetManager contract maintains a list of currently valid Keysets. The L2 chain's Owner can add or remove Keysets from this list. When a Keyset becomes valid, the KeysetManager contract emits an L1 Ethereum event containing the Keyset's hash and full contents. This allows the contents to be recovered later by anyone, given only the Keyset hash.

Although the API does not limit the number of Keysets that can be valid at the same time, normally only one Keyset will be valid.

Data Availability Certificates

A central concept in AnyTrust is the Data Availability Certificate (hereafter, a "DACert"). A DACert contains:

- the hash of a data block, and
- an expiration time, and
- proof that N-1 Committee members have signed the (hash, expiration time) pair, consisting of* the hash of the Keyset used in signing, and
-
    - a bitmap saying which Committee members signed, and
-
    - a BLS aggregated signature (over the BLS12-381 curve) proving that those parties signed.

Because of the2-of-N trust assumption, a DACert constitutes proof that the block's data (i.e., the preimage of the hash in the DACert) will be available from at least one honest Committee member, at least until the expiration time.

In ordinary (non-AnyTrust) Nitro, the ArbitrumSequencer posts data blocks on the L1 chain as calldata. The hashes of the data blocks are committed by the L1 Inbox contract, allowing the data to be reliably read by L2 code.

AnyTrust gives the sequencer two ways to post a data block on L1: it can post the full data as above, or it can post a DACert proving availability of the data. The L1 inbox contract will reject any DACert that uses an invalid Keyset; the other aspects of DACert validity are checked by L2 code.

The L2 code that reads data from the inbox reads a full-data block as in ordinary Nitro. If it sees a DACert instead, it checks the validity of the DACert, with reference to the Keyset specified by the DACert (which is known to be valid because the L1 Inbox verified that). The L2 code verifies that

- the number of signers is at least the number required by the Keyset, and
- the aggregated signature is valid for the claimed signers, and
- the expiration time is at least two weeks after the current L2 timestamp.

If the DACert is invalid, the L2 code discards the DACert and moves on to the next data block. If the DACert is valid, the L2

code reads the data block, which is guaranteed to be available because the DACert is valid.

Data Availability Servers

Committee members run Data Availability Server (DAS) software. The DAS exposes two APIs:

- The Sequencer API, which is meant to be called only by the Arbitrum chain's Sequencer, is a JSON-RPC interface allowing the Sequencer to submit data blocks to the DAS for storage. Deployments will typically block access to this API from callers other than the Sequencer.
- The REST API, which is meant to be available to the world, is a RESTful HTTP(S) based protocol that allows data blocks to be fetched by hash. This API is fully cacheable, and deployments may use a caching proxy or CDN to increase scale and protect against DoS attacks.

Only Committee members have reason to support the Sequencer API. We expect others to run the REST API, and that is helpful. (More on that below.)

The DAS software, based on configuration options, can store its data in local files, or in a Badger database, or on Amazon S3, or redundantly across multiple backing stores. The software also supports optional caching in memory (using Bigcache) or in a Redis instance.

Sequencer-Committee Interaction

When the Arbitrum sequencer produces a dataBatch that it wants to post using the Committee, it sends the batch's data, along with an expiration time (normally three weeks in the future) via RPC to all Committee members in parallel. Each Committee member stores the data in its backing store, indexed by the data's hash. Then the member signs the (hash, expiration time) pair using its BLS key, and returns the signature with a success indicator to the sequencer.

Once the Sequencer has collected enough signatures, it can aggregate the signatures and create a valid DACert for the (hash, expiration time) pair. The Sequencer then posts that DACert to the L1 inbox contract, making it available to the AnyTrust chain software at L2.

If the Sequencer fails to collect enough signatures within a few minutes, it will abandon the attempt to use the Committee, and will "fall back to rollup" by posting the full data directly to the L1 chain, as it would do in a non-AnyTrust chain. The L2 software can understand both data posting formats (via DACert or via full data) and will handle each one correctly. Edit this page Last updatedonJan 27, 2025 Previous Challenges: Interactive Fraud Proofs Next Gas and fees