How to customize your Orbit chain's deployment configuration

PUBLIC PREVIEW, MAINNET READY Orbit chains are nowMainnet ready ! Note that Orbit is still apublic preview capability - the Orbit product and its supporting documentation may change significantly as we capture feedback from readers like you.

To provide feedback, click theRequest an update button at the top of this documentjoin the Arbitrum Discord , or reach out to our team directly by completingthis form . When you visit theOrbit chain deployment portal to launch your Orbit chain, you'll be prompted to complete a form that looks like this:

Chain ID Chain name Challenge period (blocks) Stake token Base stake Owner This form will be prefilled with default values that usually don't need to be changed. However, there are some cases where you may want to customize the configuration values. This how-to explains how and when to modify these default values.

Let's briefly review each of the deployment configuration parameters, the rationale underlying the default values, and the tradeoffs that you should consider when modifying them.

Chain ID

Don't worry about this; it's inconsequential for devnets. In production scenarios (which aren't yet supported), you'll want to use a unique integer identifier that represents your chain's network on chain indexes likeChainlist.org .

Chain name

This name provides a way for people to distinguish your Orbit chain from other Orbit chains. You'll want to make this a name that you can easily remember, and that your users and developers will recognize.

Challenge period (blocks)

TheChallenge period (blocks) parameter determines the amount of time your chain's validators have to dispute - or "challenge" - the current state of the chain posted to your Orbit chain's base chain on L2 (Arbitrum Goerli or Sepolia for now; settlement to One and Nova mainnet chains isn't supported yet).

A longer challenge period means that your chain's nodes will have more time to dispute fraudulent states, but it also means that your chain's users will have to wait longer to withdraw their assets from your chain. This is one of the many tradeoffs that Orbit allows you to make when configuring your chain.

Note that the challenge period is measured in blocks on the underlying L1 chain, not the base (L2) chain. For example, if your Orbit chain settles to Arbitrum Goerli, the challenge period window would be the number ofChallenge period (blocks) multiplied by the L1 Goerli block time (~12 seconds).

Gas token

TheGas Token parameter specifies the token (ETH or an ERC-20 token) that is natively used for gas payments on the network. On Ethereum, Arbitrum One, and Arbitrum Nova the gas token is ETH. Orbit chains that are configured as AnyTrust chains can specify a different gas token as long as it falls within certain requirements.

The main requirement for custom gas tokens is that they are natively deployed on the parent chain. For example, if a team deploying an Orbit chain wants to use a specific ERC-20 as the gas token, that token must be deployed on the parent chain first (i.e. Arbitrum One or Nova). During chain deployment, that token is "natively bridged" and then properly configured as the native gas token on the new network.

There are other important considerations to keep in mind when deciding to use a custom gas token. Restrictions on the ERC-20 token include:

- In this version, only tokens with18
- decimals are permitted to be the native token.
- The token can't be rebasing or have a transfer fee.
- The token must only be transferrable via a call to the token address itself.
- The token must only be able to set allowance via a call to the token address itself.
- The token must not have a callback on transfer, and more generally a user must not be able to make a transfer to themselves revert.

It is worth reiterating that currently this feature is only supported onOrbit AnyTrust chains . Additionally, using a gas token other than ETH adds additional overhead when it comes to ensuring chains are funded properly when posting data to their parent chain.

Stake token

Your Orbit chain will be supported by at least one validator node. In order for your chain's validators to post assertions of the state of the chain on the base chain (L2), they're required to stake some value as a way to incentivize honest participation.

ThisStake token parameter specifies the token that your chain's validators must deposit into this contract when they stake. This is specified using the token's contract address on the L2 chain that your chain is settling to - Arbitrum Goerli or Arbitrum Sepolia - or0x0000000000000000000000000000000000000000 if you want to use ETH as the stake token.

Base stake

TheBase stake parameter specifies the amount of the stake token (ETH or an ERC-20 token) that your chain's validators must deposit in order to post assertions of the state of your Orbit chain on the base chain's rollup contracts. This is specified using a float value.

If yourBase stake is low, the barrier to participation will be low, but your chain will be more vulnerable to certain types of attacks.

For example, an Orbit chain with a base stake of 0.01 ETH could be halted by an adversary who can afford to deploy sacrificial validators that maliciously challenge every RBlock submitted to your Orbit chain's base chain.

The malicious challenges would result in slashed Orbit chain validators (one slashed validator per malicious challenge), but from the adversary's perspective, periodic slashing is just the price they have to pay to keep your chain offline.

A higher base stake incentivize honest participation by making it more expensive to launch these types of attacks. However, a higher base stake also translates to a higher barrier to entry for your chain's validators. This is another tradeoff to consider.

Owner

This account address is responsible for deploying, owning, and updating your Orbit chain's base contracts on its base chain.

In production scenarios, this is a high-stakes address that's often controlled by a DAO's governance protocol or multisig. For your Orbit devnet chain, think of this as a low-stakes administrative service account.

Note thatyou'll have to fund this address with enough ETH to cover the gas costs of deploying your core contracts to L2.

When deploying your Orbit chain, this address must be a standard Ethereum wallet address (precisely speaking, an EOA); it can't be a smart contract/wallet contract.

Additional configuration parameters

There are a number ofadditional parameters that are not presented in the deployment UI, but are still configurable for more advanced chain deployers. Edit this page Last updatedonMar 7, 2024 Previous How to run a full node for an Orbit chain Next How to customize your Orbit chain's behavior