Deploy a Warp Route

Create an interchain route for your token

A Warp Route is a type ofrouter application, requiring aHypERC20 orHypERC721 token contract to be deployed on each chain you wish to support.

info Check out theWarp Route reference page for more information on the interface and security implications of a Warp Route.

Prerequisites

- Hyperlane CLI
- A private key for contract transaction signing

1. Configuration

Warp Route deployment config

To deploy the route, you will need a Warp Route deployment config file. A valid config will specify:

- Which token, on which chain, is this Warp Route being created for?
- Optional:
- Hyperlane connection details including contract addresses for themailbox
- ,interchain gas
- , andinterchain security modules
- .
- Optional:
- The token standard - fungible tokens using ERC20 or NFTs using ERC721. Defaults to ERC20.

The easiest way to create one is with the CLI's config command:

hyperlane config create warp tip If your config looks correct, you can now skip toStep 2: Deployment . Or see below for details on how to define your config manually.

Base

Your config must have exactly onebase entry. Here you will configure details about the token for which you are creating a Warp Route.

- chainName
- : Set this equal to the chain on which your token exists
- type
- :* Set this toTokenType.collateral
-
    - to create a basic Warp Route for an ERC20/ERC721 token.
-
    - Set this toTokenType.collateralVault
-
    - to create a yield-bearing Warp Route for an ERC20 token that deposits into an ERC4626 vault.
-
    - Set this toTokenType.native
-
    - to create a Warp Route for a native token (e.g. ether)
- address:
-
    - If usingTokenType.collateral
-
    - , the address of the ERC20/ERC721 contract for which to create a route
-
    - If usingTokenType.collateralVault
-
    - , the address of the ERC4626 vault to deposit collateral into
- isNft:
- If usingTokenType.collateral
- for an ERC721 contract, set totrue
- .

Synthetics

Your config must have at least onesynthetics entry. Here you will configure details about the remote chains supported by your Warp Route.

- chainName:
- Set this equal to the chain on which you want a wrapped version of your token

Optional fields

You may specify the following optional values in yourbase andsynthetics entries. If no values are provided, the deployer will attempt to pull these values from elsewhere. In the case of metadata (symbol, name decimals), it will query the contract. For addresses (mailbox, ISM) it will check the core deployment artifacts, either yours (if provided) or theSDK's .

- symbol:
- The token symbol
- name:
- The token name
- decimals:
- The number of decimal places for the token
- mailbox:
- The address of the themailbox
- contract to use to send and receive messages
- interchainSecurityModule:
- The address of aninterchain security modules
- to verify interchain messages

Example

For a minimal Warp deployment config example using local anvil chains, seewarp-route-deployment.yaml .

Chain config

The Warp Route deployment will require basic information about any chains it will be interacting with. If the target chains include any not already in the Hyperlane SDK, you must specify chain metadata for it.

To see what chains are in the SDK, run the following command:

hyperlane chains list To create a chain metadata config for any other chains, run the following command:

hyperlane config create chain Or you can define the config manually. See theChainMetadata type for its schema. A minimal example can be found inanvil-chains.yaml .

2. Deployment

Run the following command to initiate the Warp Route deployment:

hyperlane deploy warp Once finished, the CLI will create two new JSON artifact files:warp-deployment andwarp-config . The first contains just addresses for the newly deployed Warp router contracts. The second is a config file for the Warp UI, which you can use for the next optional step.

3. Testing

You can initiate a test transfer of a single wei with the following command:

hyperlane send transfer If everything goes well, you should see the output similar to the following:

Waiting for message delivery on destination chain Waiting for message delivery on destination chain Waiting for message delivery on destination chain Message delivered on destination chain! Confirmed balance increase Warp test transfer complete

4. (Optional) Warp UI

The Warp UI is a DApp template for interacting with Warp Routes. See theWarp UI docs for details on how to set it up.Edit this page Previous Send your first interchain messageNext Deploy a UI for your Warp Route