Hyperlane Local Setup Guide: Sending Messages between Anvil Nodes

This guide walks you through sending interchain messages between two local Anvil nodes using the Hyperlane CLI.

Prerequisites

- Hyperlane CLI
- :
- Make sure you have the latest version of the Hyperlane CLI installed.
- npm
- install
- -g
- @hyperlane-xyz/cli
- Anvil (foundry)
- :
- Installed to run local chains. Install it via
- curl
- -L
- https://foundry.paradigm.xyz
- |
- bash
- Node.js
- (v18 or later)
- Deployer Wallet Private Key
- : You need a funded wallet for deploying contracts. This will be used as the HYP_KEY.
- export
- HYP_KEY
- =
- <
- YOUR_PRIVATE_KEY
-

Step-by-Step Guide

1. Environment Setup: Create a working directory for the Hyperlane configuration:

mkdir hyperlane-local-test &&

cd hyperlane-local-test

2. Start Two Distinct Anvil Nodes

We will run two Anvil nodes with unique chain IDs.

- On a first terminal, start the first Anvil node:
- anvil
- --port
- 8545
- --chain-id
- 31337
- --block-time
- 1
-
    - Runs onhttp://localhost:8545
-
    - .
-
    - Chain ID:31337
-
    - .
- In a new terminal, start the second Anvil node: :
- anvil
- --port
- 8546

- --chain-id
- 31338
- --block-time
- 1
-
    - Runs onhttp://localhost:8546
-
    - .
-
    - Chain ID:31338
-
    - .

3. Initialize the Hyperlane Registry

On a new terminal, use the Hyperlane CLI to create configurations for both Anvil nodes:

hyperlane registry init Follow the prompts to set upanvilnode1 . The CLI will ask you for the details of your chains including chainId and RPC URLs. Repeat the process foranvilnode2 .

This process will createmetadata.yaml files underHOME/.hyperlane/chains/anvilnode1 andHOME/.hyperlane/chains/anvilnode2 .

Example metadata:

- anvilnode1

chainId :

31337 displayName : Anvilnode1 domainId :

31337 isTestnet :

true name : anvilnode1 nativeToken : decimals :

18 name : ETH symbol : ETH protocol : ethereum rpcUrls : -

http : http : //localhost : 8545 * anvilnode2

chainId :

31338 displayName : Anvilnode2 domainId :

31338 isTestnet :

true name : anvilnode2 nativeToken : decimals :

18 name : ETH symbol : ETH protocol : ethereum rpcUrls : -

http : http : //localhost : 8546

4. Deploy Core Contracts

We'll configure and deploy Hyperlane core contracts.

tip You'll need the deployer wallet private key to deploy the core contracts. You can useexport HYP_KEY=''to set the private key as an environment variable. hyperlane core init The deployment configuration will be saved to./configs/core-config.yaml .

Next, deploy the core contracts:

hyperlane core deploy Follow the prompts to selectanvilnode1 . The CLI will deploy Mailbox, Interchain Security Modules (ISMs), and other required contracts. Repeat the process foranvilnode2 .

Once complete, you'll findaddresses.yaml inHOME/.hyperlane/chains/anvilnode1 andHOME/.hyperlane/chains/anvilnode2 , with the deployed contract addresses.

tip You should be able to see the messages of the contract deployments on your terminals running the local nodes.

5. Send a Test Message

Use the Hyperlane CLI to send a message fromanvilnode1 toanvilnode2 .

hyperlane send message --relay The CLI will prompt you to provide the origin chain (anvilnode1) and the destination chain (anvilnode2).

tip For local testing, the--relay flag automatically relays the message to the destination chain. After sending the message, check the following:

- Validator Logs: Look for entries indicating that signatures were generated and stored.
- Relayer Logs: Look for successful metadata retrieval and message processing.
- Anvil Logs: Ensure blocks were mined to process the transactions.

success You've sent a message between two local Anvil nodes using Hyperlane!

Troubleshooting

1. “Could not fetch metadata” warning:
2.
   - Reason:
3.
   - This occurs when the relayer cannot retrieve validator signatures required to process a message. Common causes:* The validator key lacks testnet funds.
4.
   -
     - The validator has not announced its storage locations.
5.
   - Solution:
6.
   -
     - Ensure the validators have announced their storage locations. Check validator logs for a message such as:Validator has announced signature storage location, locations: ["file:///tmp/hyperlane-validator-signatures-local"].
7.
   -
     - Verify that each validator has a unique signature storage path (--checkpointSyncer.path
8.
   -
     -) to prevent overwriting.
9.
   -
     - Confirm that the relayer has read access to the storage paths.
10. Messages time out:
11.
    - Reason:
12.
    - Anvil doesn't auto-mine blocks by default, causing validators or relayers to wait indefinitely for new blocks.
13.
    - Solution:
14.
    - Make sure to use the--block-time 1 flag
15.
    - when starting Anvil to auto-mine blocks every second.
16. Validator mismatch or misconfiguration:
17.
    - Reason:
18.
    - The ISM configuration on the destination chain does not match the validator key(s) used by the origin chain.
19.
    - Solution:
20.
    - Check that the ISM configuration includes the correct validator addresses. Validator logs can help identify the announced addresses. Edit this page Previous Deploying a Bridge UI for Hyperlane Warp RoutesNext Deploy Yield Routes