Release Flow

Releasing the widget to IPFS involves several steps, including:

1. Creating a content-addressable archive (CAR file) containing the necessary files for the widget to function.
2. Uploading the CAR file to the IPFS network using an IPFS provider.
3. Pinning the uploaded CAR file to ensure its permanent availability on the IPFS network.

IPFS Pinning By default, IPFS nodes only keep data in their cache for a limited time. Then, the data is removed by an automatic garbage collection process. To ensure that content remains available on the IPFS network permanently, it must be pinned using its Content Identifier (CID). The power of GitHub Actions is used to complete and automate these and other required steps.

GitHub Actions workflow

GitHub Actions has already been used to build and deploy Lido applications, so it was decided to adapt them for IPFS releases.

IPFS release occurs as the next step after a regular application release. But only major or critical updates are released to IPFS due to the numerous actions required to make an IPFS release, and also the fact that each new release of a Lido app will produce a new CID and will be available at the new address, which is inconvenient for users willing to always use the latest version of an application.

The developed workflow allows automatic pinning of any Lido app that is technically ready to operate on IPFS. Additionally, pinning is not limited to a single provider but can be performed on multiple providers simultaneously. This approach aims to leverage decentralization, guaranteeing accessibility to IPFS content from multiple providers in case one of them fails. This setup also enables independent testing of the UI on various networks and environments.

The IPFS pinning and preparing ENS transactions in the workflow are facilitated by the Blumen package, developed in collaboration with Lido.

On every IPFS release, the content verification is carried out by both development and QA Lido contributors to ensure that there is no unexpected content added to the code during CI process. The verification relies on hash comparisons, and if you want, you can also perform it using the provided instructions .

After the verification, the IPFS release is initiated, which results in adding the obtained pinning information to the application releases page . The details about IPFS pinning (CID, IPFS providers, https gateway, source code archives) is attached to the release description.

Workflow steps

You can find the source code of the workflow on GitHub . The workflow performs further steps:

- Retrieving necessary secrets, tags, and the application's build files.
- Obtaining information about the date and the commit hash of the tag.
- Executing the Blumen
- script to perform IPFS pinning.
- Creating artifacts to be attached to the release.
- Searching for an existing release on GitHub based on the provided tag for modifying the description.
- Generating and modifying the existing description of the release.

Edit this page Previous About IPFS Next Security