Shapeshift DAO Release Process

Purpose

The purpose of this document is to outline a process that standardizes a mechanism for the DAO to release code to production frequently and with checks in place to ensure the quality of those releases. We expect this process to change as we learn more and grow in our proficiencies as a DAO in releasing software. Our ultimate vision is to move towards full CI/CD with more automated testing and less manual validation.

Roles

1. Release Manager (a code owner from the Engineering Workstream)
    a. Responsible for the end to end release process overall. See process below.
2. Product Lead (someone from the Product workstream)
    a. Responsible for testing of any new features or feature flags that have been enabled for this release. Will work closely with the release manager to determine if the release candidate will move to production.
3. Operations Lead (someone from the Operations workstream)
    a. Responsible for testing and confirming that expected fixes are as intended. The Operations Lead is more focused on testing defects that have been mitigated rather than new features, but will work with Product Lead as needed on new features. Assists with general regression testing.

Cadence

Twice weekly, Tuesday and Thursdays @ 7PM UTC

Process

1. 24-36 hours before the scheduled release review each of the following repositories for changes since the last release.
    a. Web or its dependencies (lib, etc)
    b. Foxfarm
    c. Unchained
2. For each repository with changes create a release branch from the current develop branch. This branch should be named in the following format with XX.YY representing the major and minor versions `release/XX.YY.ZZ`.
3. Open a Pull Request on Github for this branch with the following naming convention `YYYY-DD-MM release/XX.YY.ZZ`. YYYY-MM-DD should be the release date, not the current date.
4. Create a thread in the #operations-publicchat discord channel with the following format. `YYYY-MM-DD repo-name release/XX.YY.ZZ`. For example `2022-01-18 web release/1.55.21`

5. In the thread create a summary of the release including new customer facing functionality or bug fixes that can help product and operations to verify functionality, ping the following workstreams by mentioning them in the discord thread and identify each of the people filling the above Roles by discord handle. Please also alert any engineer with code shipping to production in the channel.
    a. Operations
    b. Engineering
    c. Product
6. If blocking issues are discovered in the release branch the person who has found the issue should add it to github and comment on the Pull Request with a link to the issue. This will serve as the release checklist to inform the go / no-go decision on the final merge to production.
7. The Release Manager will be responsible for coordinating any fixes. If they are able to resolve the issue themselves they may, or if not they will pull in the needed engineering resources to do so. Ultimately, they may also decide to push back the release until the next window if the issues are too severe or risky given the timeframe.
8. Once all blocking issues have been resolved and Operations Lead and Product Lead have signed off, the Release Manager will merge the branch to main and monitor the deployment. Once the deployment has been completed they will notify the release discord thread for final confirmation from Product and Operations once in production.
9. Operations Lead monitors customer channels and any applicable performance metrics / smoke tests over the next 24 hours and alerts the Release manager if any issues arise that are suspected to be from the release. Our default response is to roll back if the release is expected to have broken any critical functionality.

Hotfixes

1. Hotfixes can be made out of band from our scheduled releases as dictated by the severity of the issue being mitigated.
2. We still recommend following steps 3 - 9 above in these cases. With a hotfix branch being cut from main and then merged back into main and develop once completed.

Feature Releases in Web:

Shapeshift DAO will have four distinct environments that are a part of the testing and release flow: 1) development 2) release branch 3) alpha and 4) production. LaunchDarkly, pending a security review, will be used to control feature flags in all sub-production environments.

1. **Development** - This deployment will track the develop branch of the web repository. Feature flags can be controlled via LaunchDarkly.

2. **Release** - This deployment will track the latest release branch that has been created by the Release Manager. Feature flags can be controlled via LaunchDarkly for Alpha testing but once moved to production, LaunchDarkly will no longer be able to be used.
3. **Alpha** - The environment served at alpha.shapeshift.com. This build will be identical to the production build and deployed alongside with production. The only difference between the two will be the ability to use LaunchDarkly for feature flagging.
4. **Production** - The environment served at app.shapeshift.com. Identical to the Alpha build, but with no ability to control feature flags via LaunchDarkly.