KubeCon | CloudNativeCon

North America 2023

KubeCon | CloudNativeCon
North America 2023

Flux 2.0: Amplifying Gitops with OCI & Cosign

Sanskar Jaiswal, Flux & Flagger maintainer, Weaveworks
Kingdon Barrett, Flux maintainer, Weaveworks

flux
@KubeCon

Flux: Highlights

Flux is a set of continuous and progressive delivery solutions for Kubernetes that are open and extensible.

- CNCF Graduated Project
- Integrations with Terraform, AWS Cloud Formation
- Free and open source UI through Weave GitOps
- Trusted by companies like GitLab, Orange, Deutsche Telekom

flux
@KubeCon

Azure Amazon EKS VMware Tanzu orange™

Flux has a GA release!

APIs:

- *GitOps:*
    - GitRepository, Kustomization, Receiver (v1)
- *Helm:*
    - HelmRepository, HelmChart (v1beta2)
    - HelmRelease (v2beta1)
- *OCI:*
    - OCIRepository (v1beta2)
- *Image Automation:*
    - ImageRepository, ImagePolicy, ImageUpdateAutomation (v1beta2)
- *Notifications:*
    - Provider, Alerts (v1beta2)

Controllers:

- source-controller: v1.1.x
- kustomize-controller: v1.1.x
- notification-controller: v1.1.x
- helm-controller: v0.36.x
- image-automation-controller: v0.36.x
- image-reflector-controller: v0.30.x

Installation:

- flux2 (cli): v2.1.x (bootstrap only)
- terraform-provider-flux: v1.1.x

Flux: Non GA components

APIs:

- *GitOps:*
  - GitRepository, Kustomization, Receiver (v1)
- *Helm:*
  - HelmRepository, HelmChart (v1beta2)
  - HelmRelease (v2beta1)
- *OCI:*
  - OCIRepository (v1beta2)
- *Image Automation:*
  - ImageRepository, ImagePolicy, ImageUpdateAutomation (v1beta2)
- *Notifications:*
  - Provider, Alerts (v1beta2)

Controllers:

- source-controller: v1.1.x
- kustomize-controller: v1.1.x
- notification-controller: v1.1.x
- helm-controller: v0.36.x
- image-automation-controller: v0.36.x
- image-reflector-controller: v0.30.x

Installation:

- flux2 (cli): v2.1.x
- terraform-provider-flux: v1.1.x

Flux: Non GA components

APIs:

- *GitOps:*
  - GitRepository, Kustomization, Receiver (v1)
- *Helm:*
  - HelmRepository, HelmChart (v1beta2)
  - HelmRelease (v2beta1)
- *OCI:*
  - OCIRepository (v1beta2)
- *Image Automation:*
  - ImageRepository, ImagePolicy, ImageUpdateAutomation (v1beta2)
- *Notifications:*
  - Provider, Alerts (v1beta2)

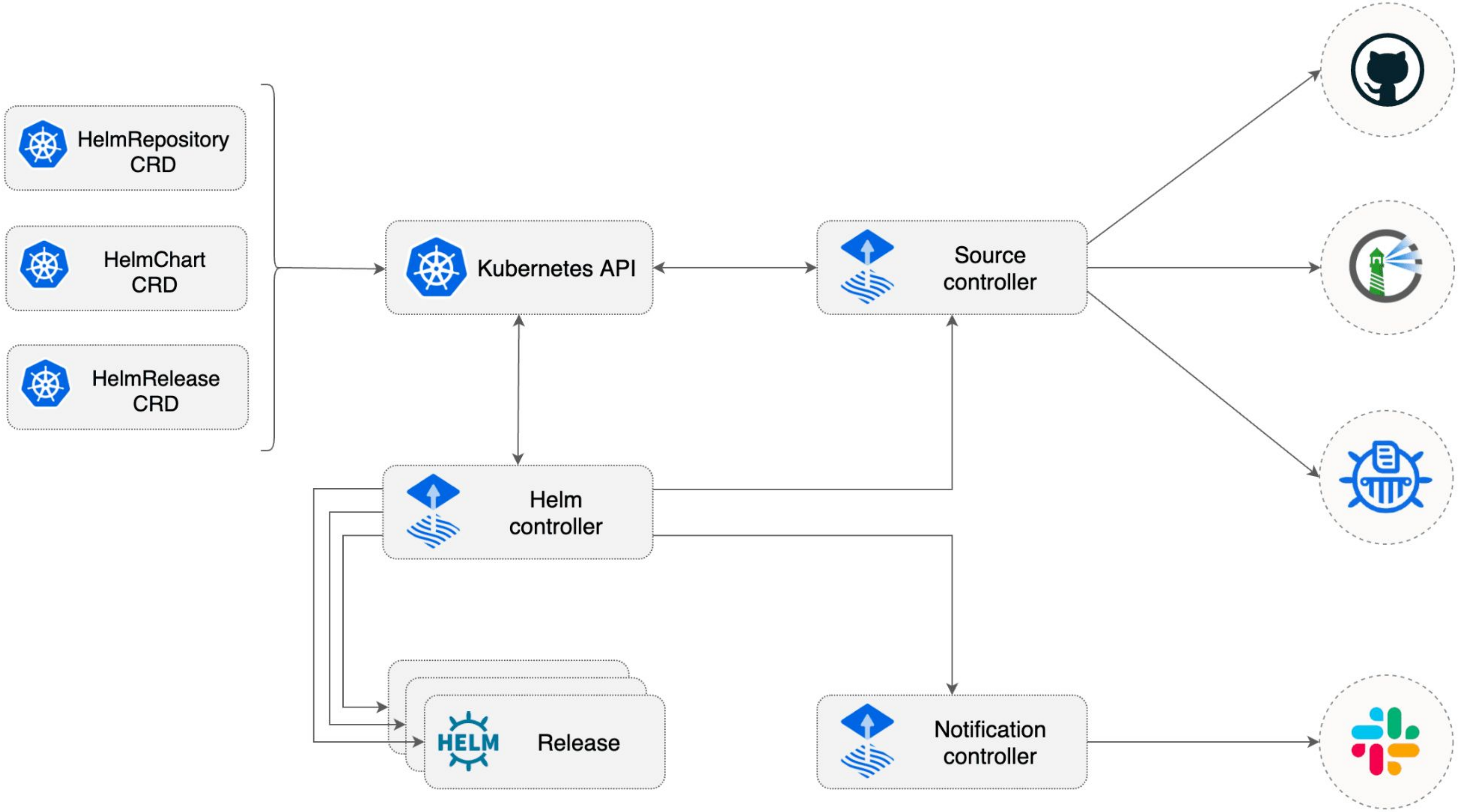
Controllers:

- source-controller: v1.1.x
- kustomize-controller: v1.1.x
- notification-controller: v1.1.x
- helm-controller: v0.36.x
- image-automation-controller: v0.36.x
- image-reflector-controller: v0.30.x

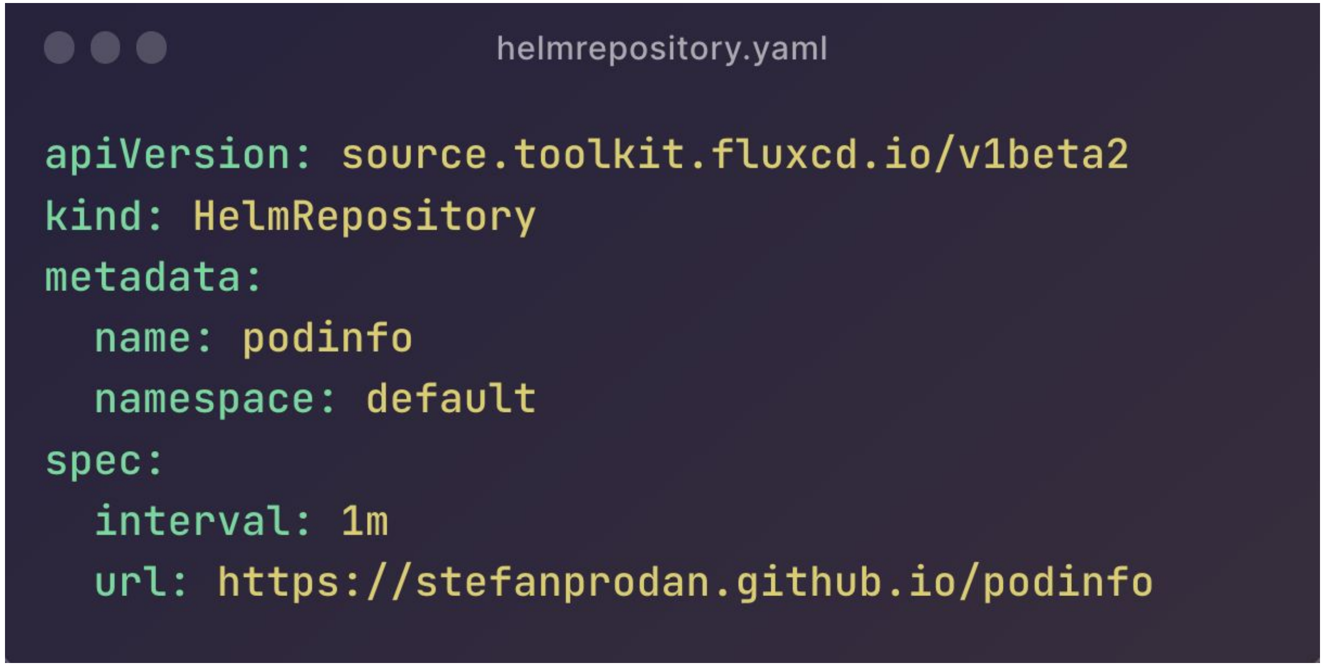
Installation:

- flux2 (cli): v2.1.x
- terraform-provider-flux: v1.1.x

Helm with Flux

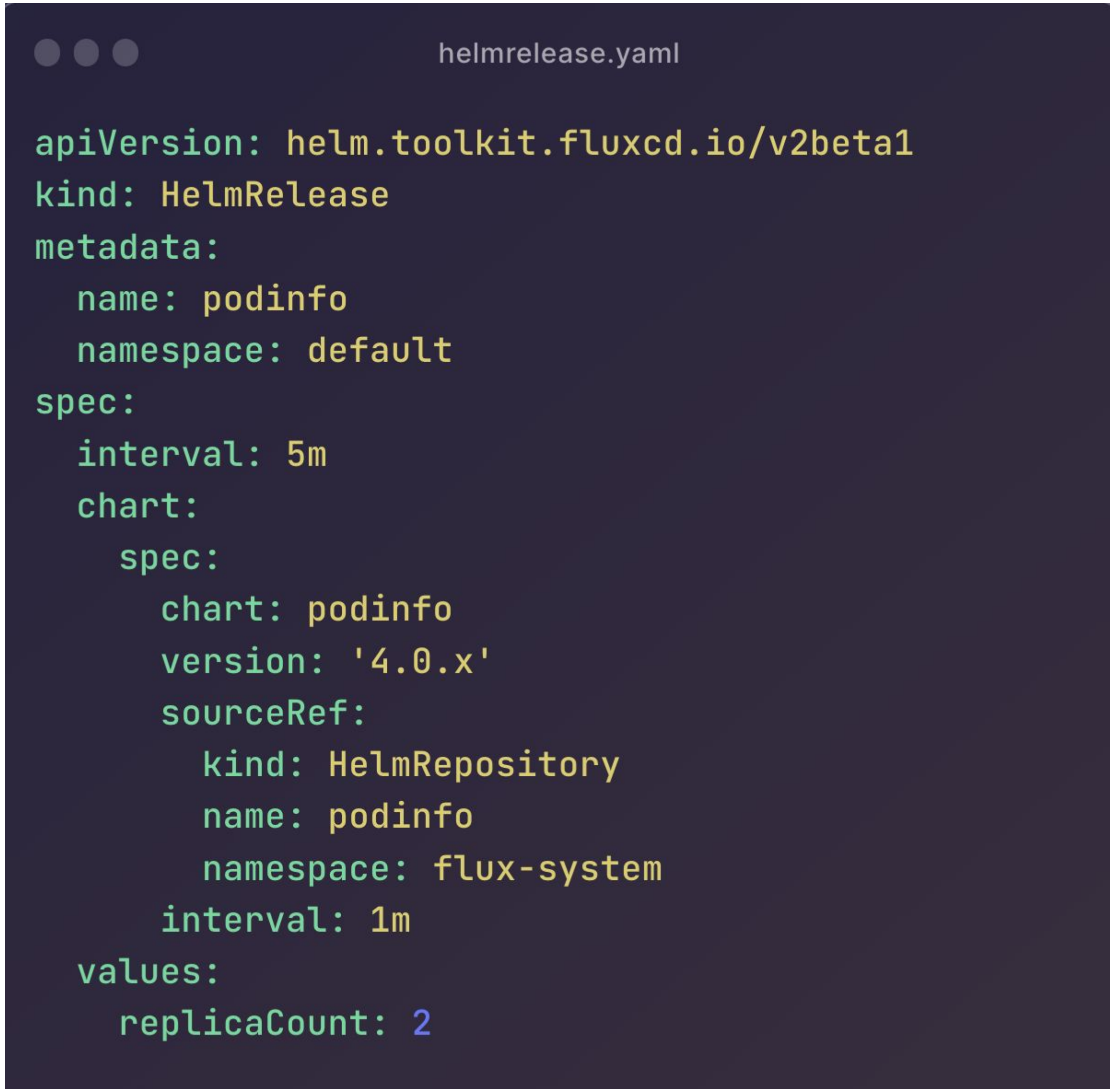


Helm with Flux



helmrepository.yaml

```yaml
apiVersion: source.toolkit.fluxcd.io/v1beta2
kind: HelmRepository
metadata:
  name: podinfo
  namespace: default
spec:
  interval: 1m
  url: https://stefanprodan.github.io/podinfo
```



helmrelease.yaml

```yaml
apiVersion: helm.toolkit.fluxcd.io/v2beta1
kind: HelmRelease
metadata:
  name: podinfo
  namespace: default
spec:
  interval: 5m
  chart:
    spec:
      chart: podinfo
      version: '4.0.x'
      sourceRef:
        kind: HelmRepository
        name: podinfo
        namespace: flux-system
      interval: 1m
  values:
    replicaCount: 2
```

The Problem

index.yaml does not scale:

- YAML parsing is slow: https://github.com/mattfarina/yamljsonperftest
  - Load the entire file into memory and then look up the exact entry
- No way to filter unwanted entries
  - Contains undesired metadata
  - Contains unwanted charts
- Verification requires PGP and provenance files

https://github.com/bitnami/charts/issues/10539

> After some investigation, it seems the root cause is related to CloudFront reaching some limits due to the volume of traffic when serving the index.yaml.
This index.yaml contains all the Bitnami Helm charts history (around 15300 entries), producing a pretty fat 14MB file. Given the size of the file and the volume of traffic, thousands of terabytes of download traffic per month are being generated.

What is the solution?

No index.yaml

Open Container Initiative

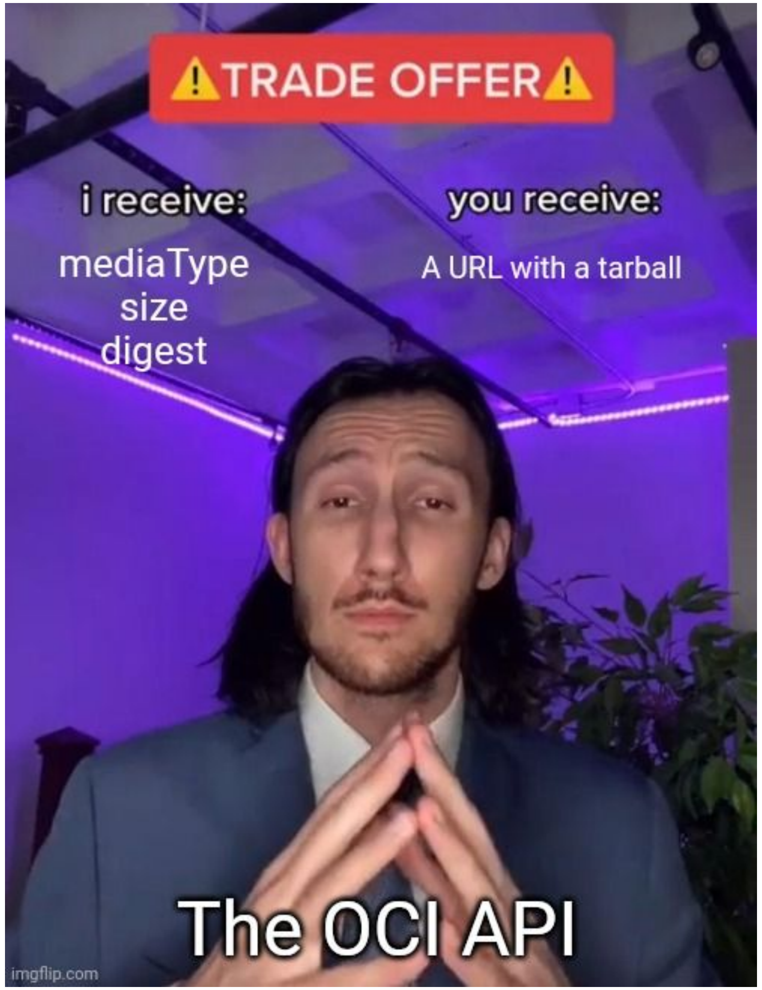
What is the Open Container Initiative?

The Open Container Initiative (OCI) is an open governance structure that defines the specifications and standards for container technologies (artifacts, distribution and runtimes).

What is the OCI Distribution spec?

The OCI Distribution Specification defines an API protocol to facilitate and standardize the distribution of content.
The specification is designed to be agnostic of content types.

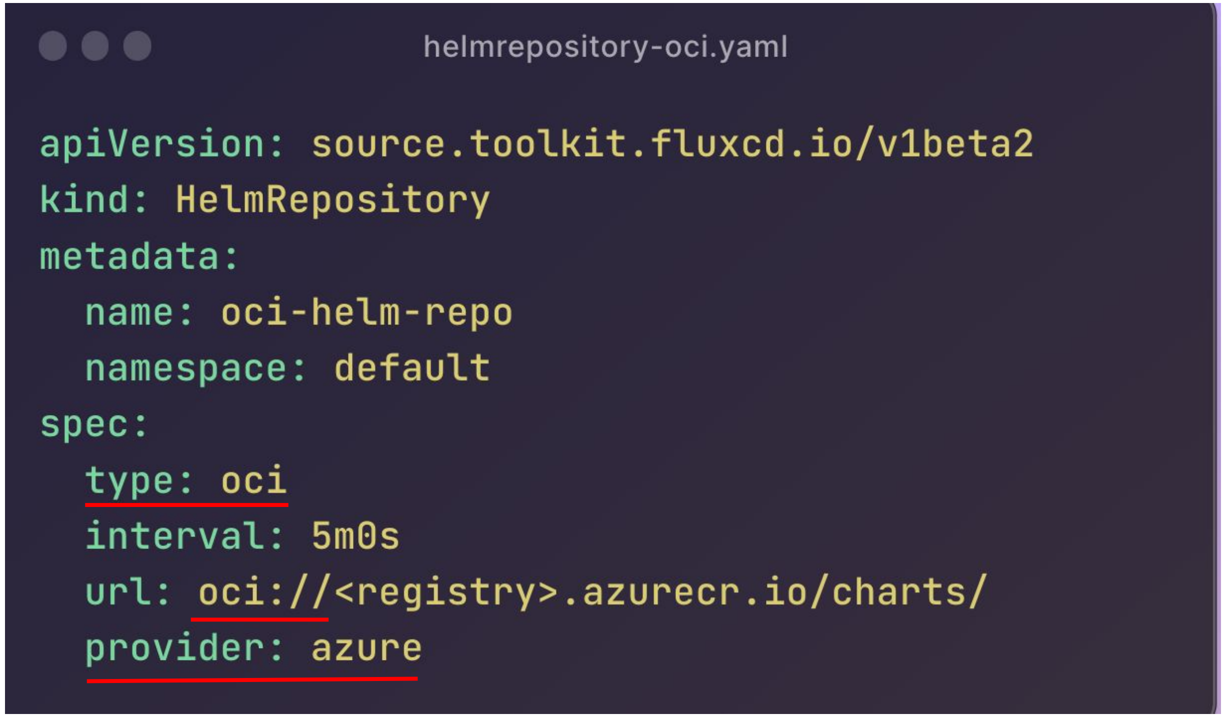
Essentially



(stolen from @lorenc_dan)

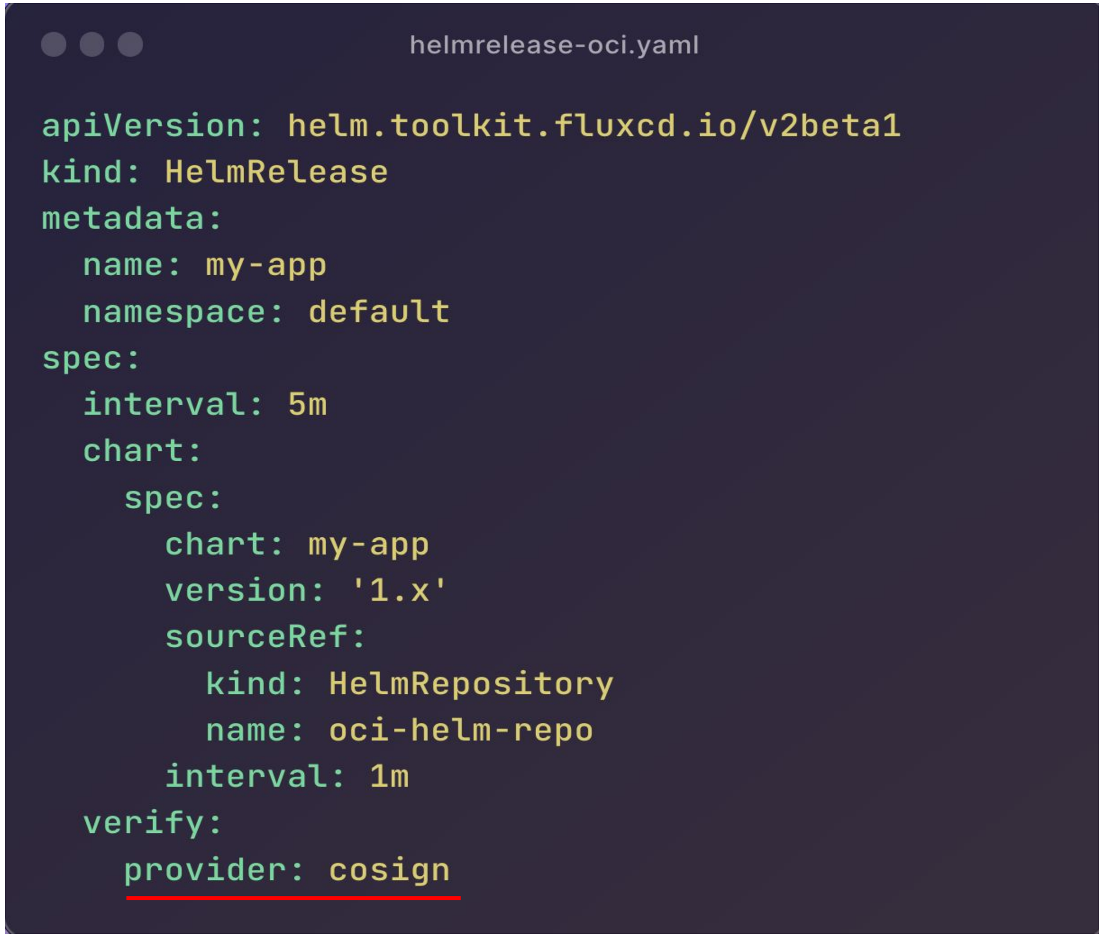
Helm + OCI

```
$ helm package my-app/        # -> my-app-1.0.0.tgz <- metadata via Chart.yaml
$ helm push my-app-1.0.0.tgz oci://<registry>.azurecr.io/charts
$ cosign sign oci://<registry>.azurecr.io/charts/my-app@<digest>
```



helmrepository-oci.yaml

```yaml
apiVersion: source.toolkit.fluxcd.io/v1beta2
kind: HelmRepository
metadata:
  name: oci-helm-repo
  namespace: default
spec:
  type: oci
  interval: 5m0s
  url: oci://<registry>.azurecr.io/charts/
  provider: azure
```



helmrelease-oci.yaml

```yaml
apiVersion: helm.toolkit.fluxcd.io/v2beta1
kind: HelmRelease
metadata:
  name: my-app
  namespace: default
spec:
  interval: 5m
  chart:
    spec:
      chart: my-app
      version: '1.x'
      sourceRef:
        kind: HelmRepository
        name: oci-helm-repo
      interval: 1m
  verify:
    provider: cosign
```

Helm + OCI Benefits

- Apps, images and signatures in one place
- Passwordless authentication
- Keyless integrity verification
- No index.yaml:
  - less CPU and RAM -> less 💰 spent
  - less ingress traffic -> less 💰 spent
  - no network bottlenecks due to size

Workload Identity with Flux

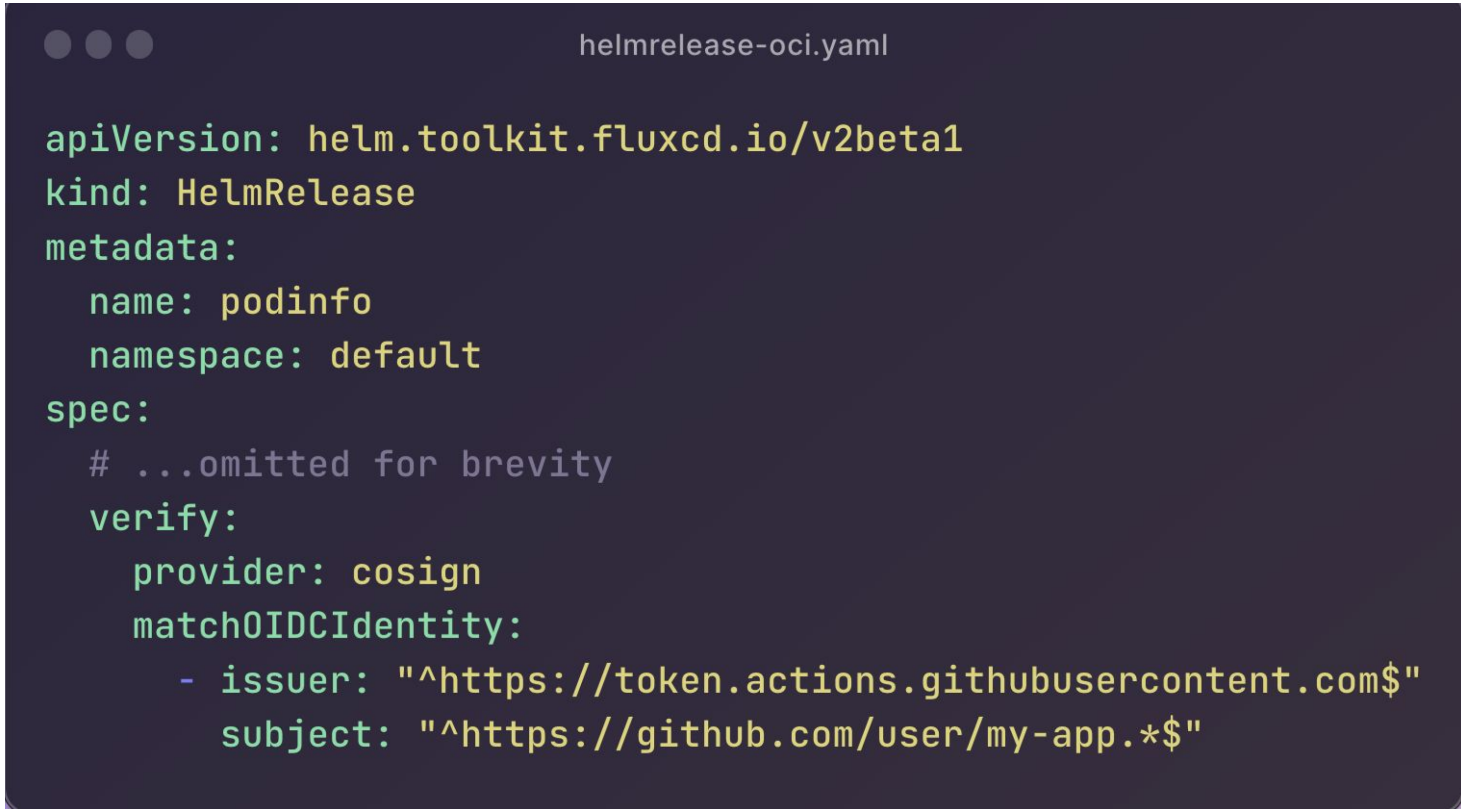
Cloud IAM Role bound to Pod/Node which can be used to access the registry.

Benefits:

- No Secrets with username/password required
- Integrates seamlessly with Azure, AWS and GCP
- Native to Kubernetes (uses ServiceAccounts)

Integrity verification with Flux

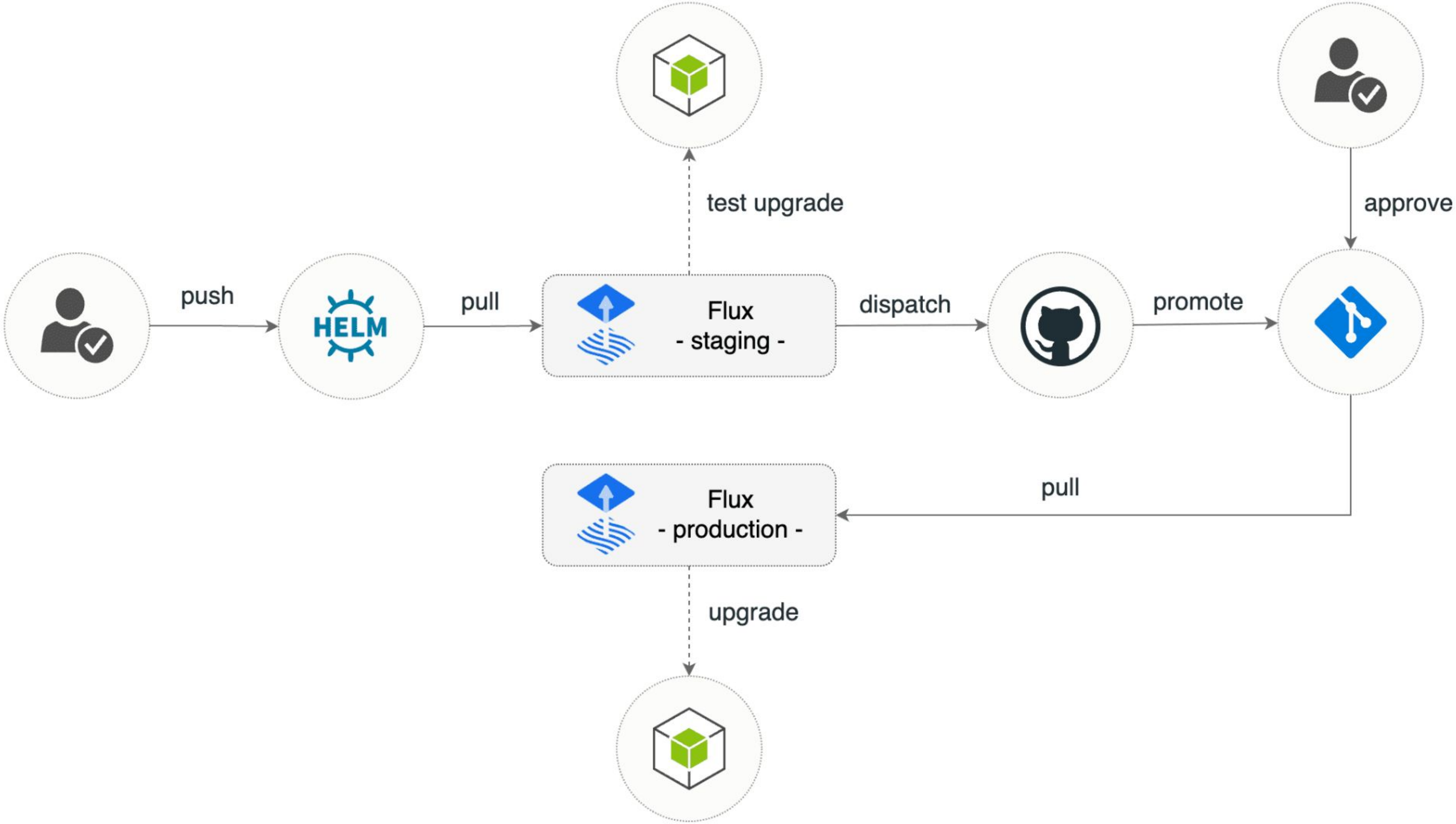
- Support for Cosign keyless verification, no need for private/public key pairs
- Bound to OIDC identity
- Support for OIDC identity matching



helmrelease-oci.yaml

```yaml
apiVersion: helm.toolkit.fluxcd.io/v2beta1
kind: HelmRelease
metadata:
  name: podinfo
  namespace: default
spec:
  # ...omitted for brevity
  verify:
    provider: cosign
    matchOIDCIdentity:
      - issuer: "^https://token.actions.githubusercontent.com$"
        subject: "^https://github.com/user/my-app.*$"
```

Promoting Helm releases



DEMO TIME 🎉

Promoting Helm releases

- fluxcd.io / flux / use-cases / gh-actions-helm-promotion /
  Est'd. Jun 2022 - (5mo before OCI Helm Cosign verification landed)

Pro:

Helm Test can extra test above & beyond Health Checking ✅
Battle hardened Pull Request workflow on GitHub ⚠️
Extensible for any platforms that support webhooks (AZ DevOps, + etc.) 🎉

Promoting Helm releases

- `fluxcd.io / flux / use-cases / gh-actions-helm-promotion /`
  Est’d. Jun 2022 - (5mo before OCI Helm Cosign verification landed)

Pro:

Helm Test can extra test above & beyond Health Checking ✅
Battle hardened Pull Request workflow on GitHub ⚠️
Extensible for any platforms that support webhooks (AZ DevOps, + etc.) 🎉

Con:

You have to use Helm - implies we build Helm Charts 😟
Good reasons not to use Helm everywhere in 2023 📝

Promoting Helmless OCI releases

- `fluxcd.io / flux / use-cases / gh-actions-ocirepo-promotion /`
  Coming Soon (Nov 2023) - Not yet, as of KubeCon press deadlines

How:

`OCIRepository` resource acts as cache layer for Git releases ✅
Set `Kustomization.spec.wait` to enable ✅
The same workflow, only with OCIRepository 🎉
Skip writing Helm charts, still take advantage of provenance 💌

Promoting Helmless OCI releases

- fluxcd.io / flux / use-cases / gh-actions-ocirepo-promotion /
  Coming Soon (Nov 2023) - Not yet, as of KubeCon press deadlines

How:

OCIRepository resource acts as cache layer for Git releases ✅
Set Kustomization.spec.wait to enable ✅
The same workflow, only with OCIRepository 🎉
Skip writing Helm charts, still take advantage of provenance 💌

However:

Unclear how to get the same effect as Helm Test ⁉️
Not yet all paths documented, you still have to paint your own 📝

TODO:

Add more documentation focused on OCI in Q4 2023, Q1 2024 ❤️

Statelessly promoting with `spec.semver`

- Use `semver` and promote every release artifact

  Use tags of `OCIRepository`, `HelmRelease`, or `GitRepository` ✅
  App/Project git repos already use `semver` ✅
  Able to scale with smaller team, iterate fast in staging envs 🏃

Statelessly promoting with `spec.semver`

- Use `semver` and promote every release artifact

  Use tags of `OCIRepository`, `HelmRelease`, or `GitRepository` ✅
  App/Project git repos already use `semver` ✅
  Able to scale with smaller team, iterate fast in staging envs 🏃

For Production - (No! **Don’t**):

Now we can skip staging, deliver releases straight to Prod 😤
Promotion of new releases is faster than ever 🔥
Adds sig. risk, when any release is bad Prod will feel it 🌶️
(To be clear, this is only appropriate for staging envs*) ℹ️

*(Or it could be made safe for production through the use of **Flagger canaries**!)
But that’s another whole talk, and now we’re running out of time!

HELM Advice for Users

what do **users & vendors** need to understand about the fact that

index.yaml does not scale:

- Use OCI for helm everywhere you can

Advice for Users

what do users & vendors need to understand about the fact that

index.yaml does not scale:

- Use OCI for helm everywhere you can
- Vendors: Should we abandon our legacy Helm Repositories?
    - That’s up to you!

      From Flux perspective, there is no good reason for index.yaml

Advice for Users

what do users & vendors need to understand about the fact that

index.yaml does not scale:

- Use OCI for helm everywhere you can
- Vendors: Should we abandon our legacy Helm Repositories?
  - That’s up to you!
- End users: **Do we need to support OCI?**
  - **Yes! ♻️⚡🔌 Be Environmentally Friendly**

Advice for Users

so what does it really mean **for me**

if
index.yaml **does not scale**?

- Users: what to do if chart vendor does not support OCI?
  - Maybe ask them if they plan to support it
  - (Or, just don’t worry about it! Just use legacy Helm repos, it’s fine)

Advice for Users

so what if index.yaml does not scale:

- Just use OCI for helm everywhere you can

- Users: what to do if chart vendor does not support OCI? (don't worry about it!)
    - The performance gains are negligible unless you publish many charts with many versions that are upgraded/published frequently or w/ v. long history
    - No need to ask them to switch, unless there is a performance issue

- We can follow the github.com/fluxcd/flux2-monitoring-example
NEW Flux Monitoring Guide: fluxcd.io/flux/monitoring/ – and its examples
to find out if there is a major performance impact to your Flux installation!

Sanskar Jaiswal

GitHub: @aryan9600
Twitter: @aryan9600_

Thanks again from flux @KubeCon

Please scan the QR Code above to leave feedback on this session

Kingdon Barrett

GitHub: @KingdonB
Fediverse: yebyen@hachyderm.io