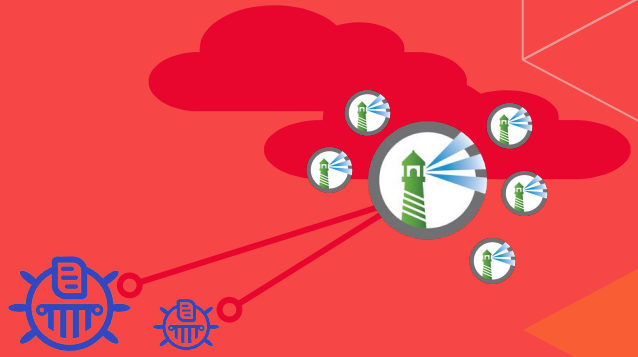
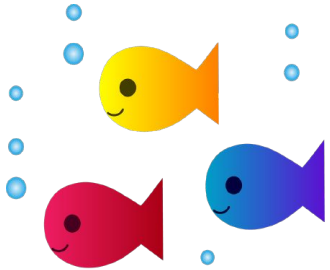


From ChartMuseum to Harbor

Helm Summit
Amsterdam, September 2019

Josh Dolitsky
>> bloodorange.io
@jdolitsky

👋 I’m Josh Dolitsky



- Founder / Engineer - **bloodorange.io**
- Creator / Core Maintainer - **ChartMuseum**
- Org / Core Maintainer - **Helm**
- Here to tell you a story and predict the future

Who likes lunch?

bloodorange.io

Proud sponsor of your lunch today!

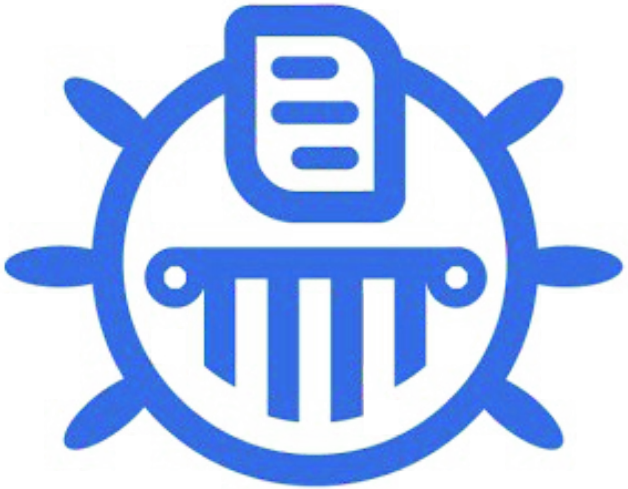
(We hope that it tastes good and meets all dietary requirements!)

Today I shall make a prediction

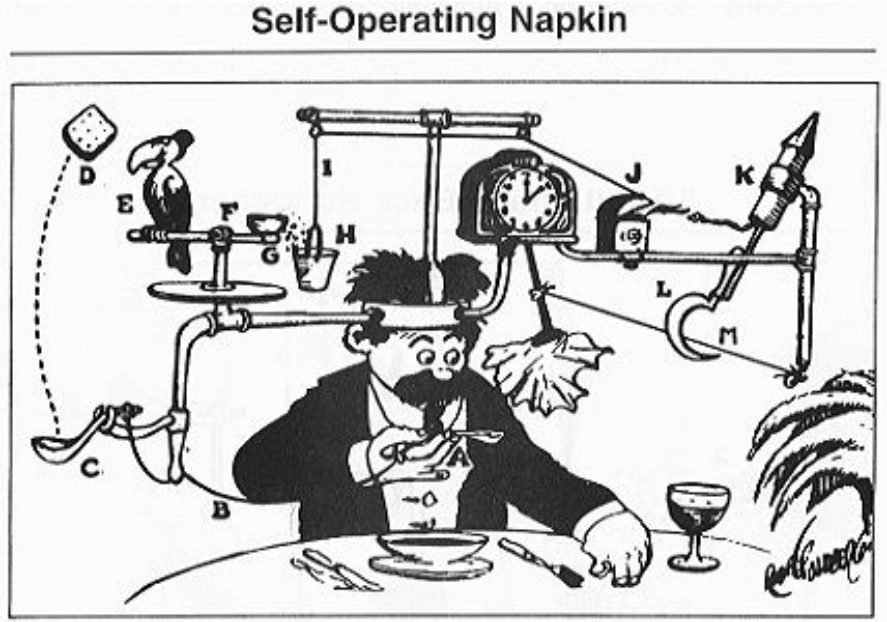
Not too bold of a prediction, I don't think..

By 2021, **OCI Distribution** will become the standard for sharing/storing things across all cloud-native tools (things such as Helm charts)

The title of this talk is...



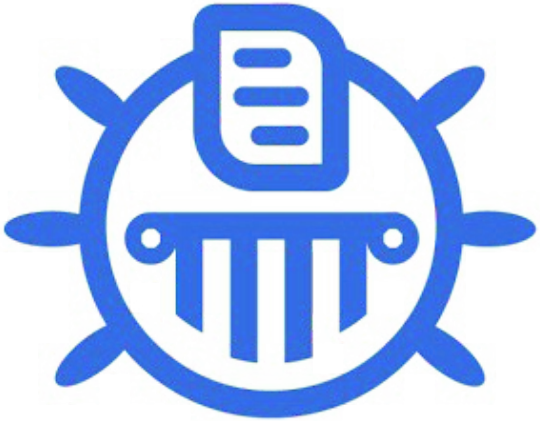
But really it should be...



OPEN CONTAINER INITIATIVE

Let’s get back to that in a moment.

ChartMuseum, you said?



The ChartMuseum Project

- Web server written in Go, started in 2017 (for fun)
- Donated to Helm as an official subproject
- Simple, easy way to run a chart repository
- Pluggable storage options: S3, Azure, Google, etc.
- Powers lots of companies' internal chart services

CHARTMUSEUM

github.com/helm/chartmuseum

“The chart repository system is insufficient”

Chart repos fall short (pt. 1)

March 2017, Quay App Registry & Helm plugin introduced as an alternative to using chart repos

- Led by Antoine Legrand & Jimmy Zelinskie
- API layer on top of a container registry (Quay.io)
- Features such as customizable release channels

Quay App Registry

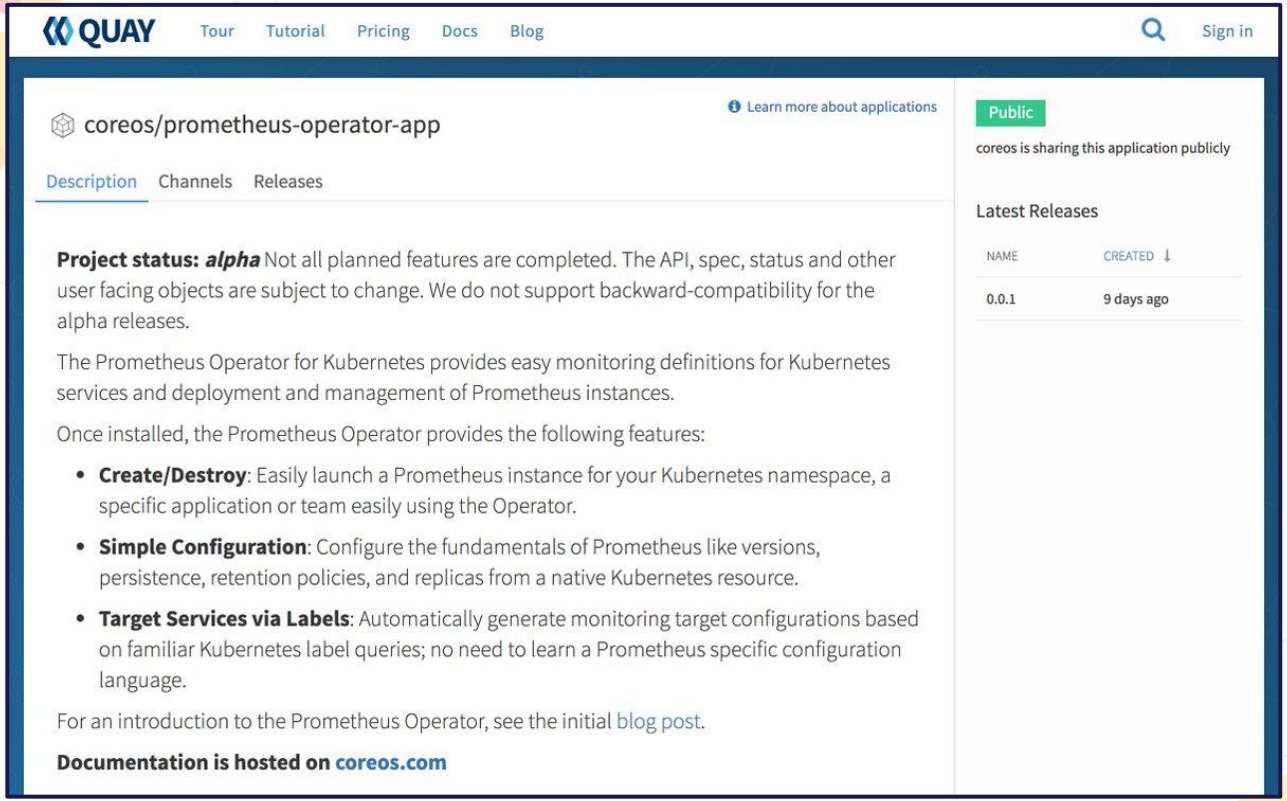


Chart repos fall short (pt. 2)

February 2018, Ankush Chadha gives a talk at Helm Summit on index.yaml scalability concerns

- Later that year, Matt Farina submits an in-depth proposal for index partitioning in Helm 3

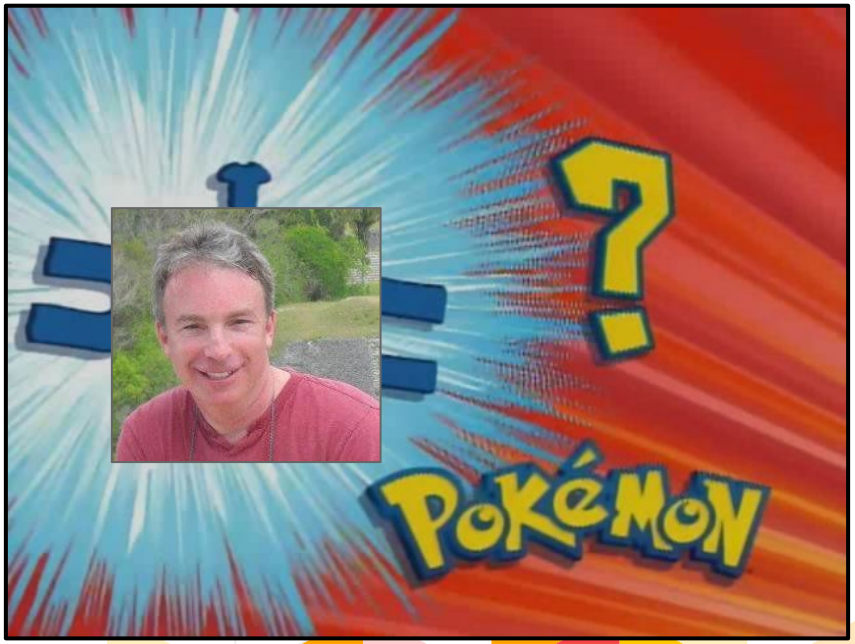
Chart repos fall short (pt. 3)

Late 2018, I begin hosting weekly meetings to plan improvements for Helm 3 repo system

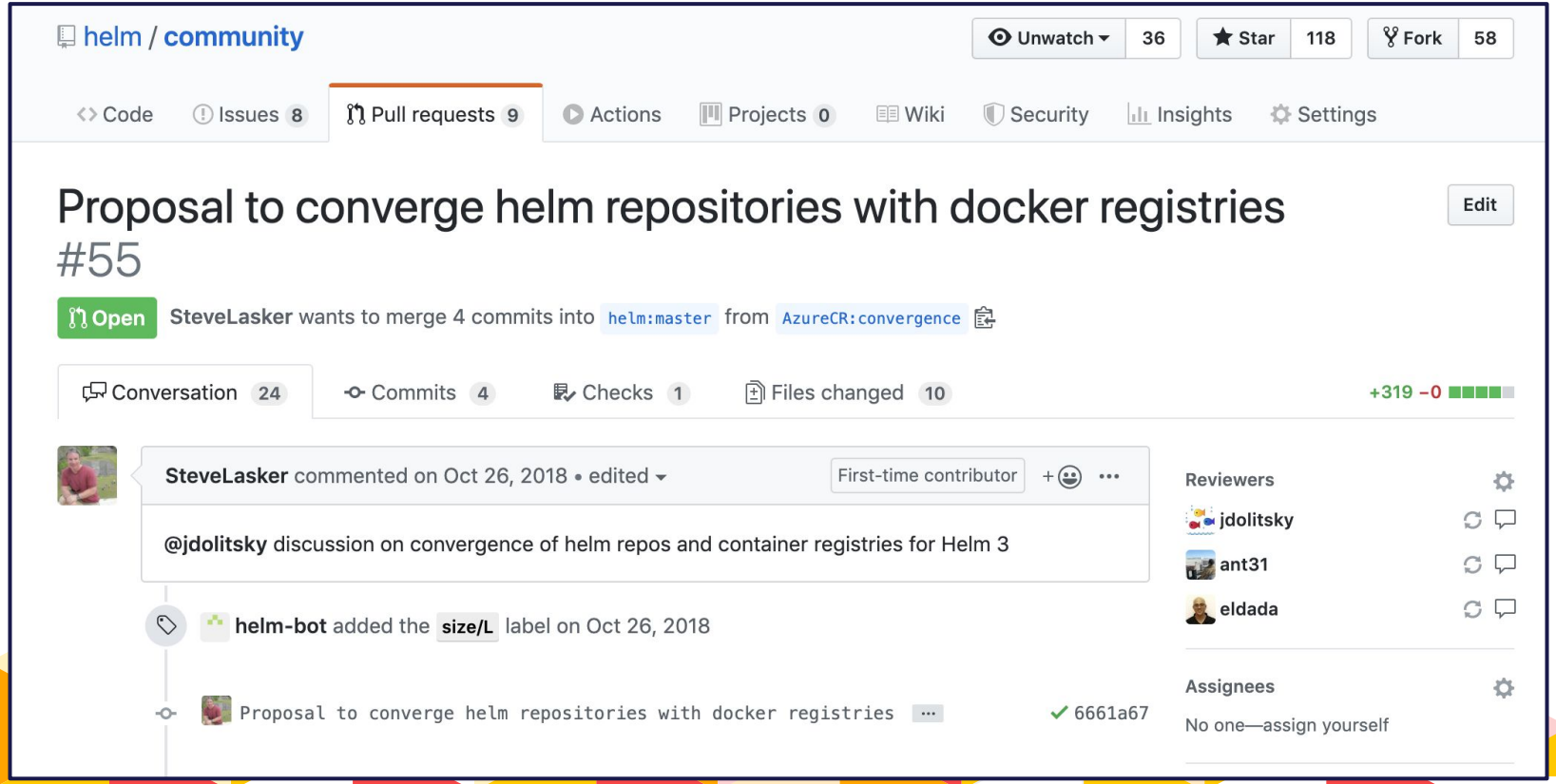
- How do we enable fine-grained authorization?
- How do we enable chart uploads?
- How do we “helm login” ?
- How do we “helm push” ?

And then...

a wild Steve Lasker appears!



Steve: Helm should work like Docker!



This all led to “seemingly interminable registry discussions”...

Then - just in time for KubeCon NA 2018...

Helm team decides that someway, somehow, the next generation of chart distribution will be powered by OCI

- Conversations moved to OCI call/mailing list
- Still leaves open many, many questions
- “What’s going to happen to the museum?”
- Also.. OC-what???

OCI = Open Container Initiative


OPEN CONTAINER INITIATIVE
AN OPEN GOVERNANCE STRUCTURE FOR THE EXPRESS PURPOSE
OF CREATING OPEN INDUSTRY STANDARDS AROUND CONTAINER
FORMATS AND RUNTIME

Super Brief History of OCI (pt. 1)

2013

Docker is released

2014

rkt is released (CoreOS)

2015

OCI launched as *the* open container runtime+image spec with support from Docker, CoreOS, Google, RedHat, etc.

Super Brief History of OCI (pt. 2)

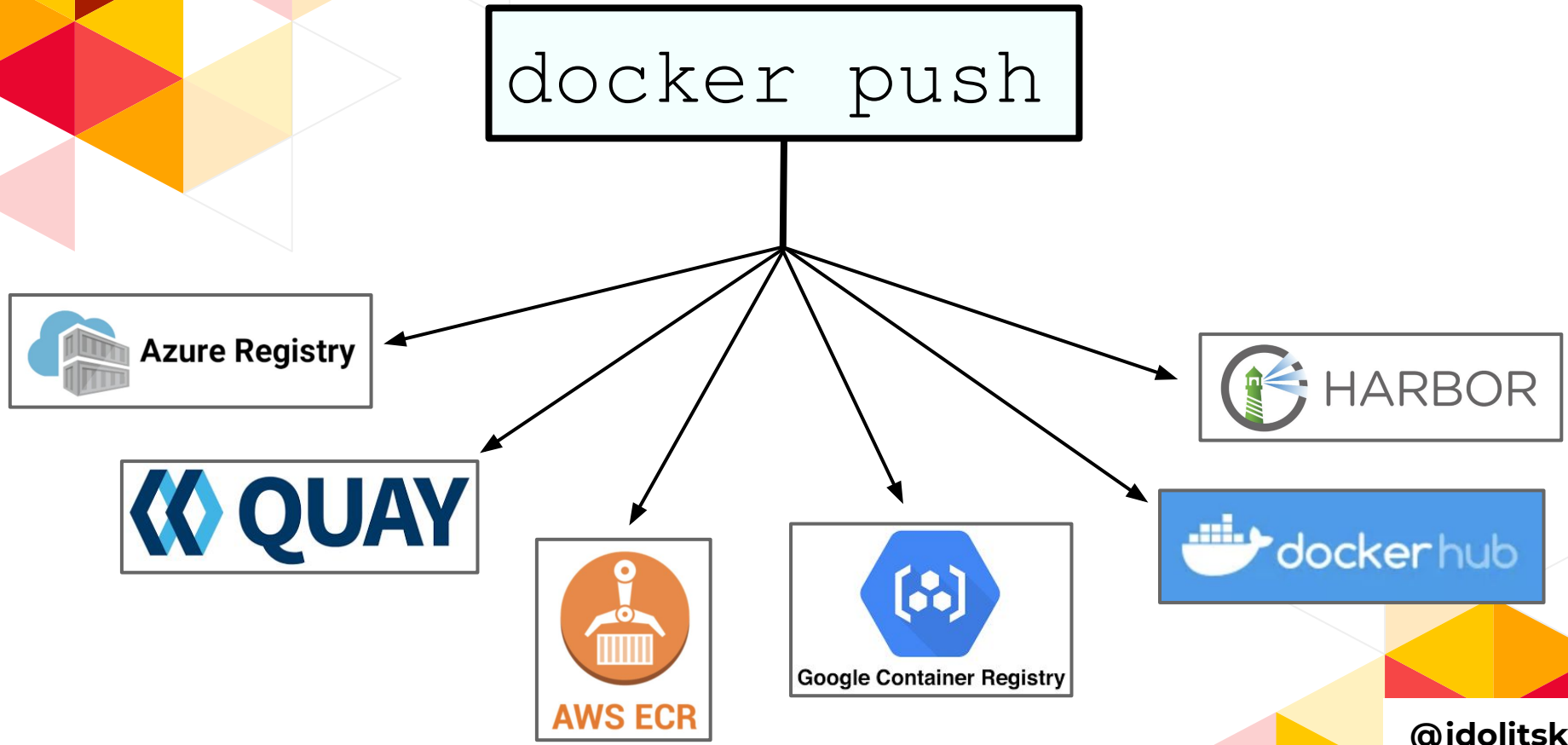
Fast-forward a bit...

2018

OCI adds **distribution spec** ?

The OCI Distribution Specification

- Announced in April 2018
- A standard for container image distribution
- Based on the Docker Registry HTTP API (v2)
- Supports the pushing and pulling of container images


docker push
Azure Registry
QUAY
AWS ECR
Google Container Registry
HARBOR
dockerhub

docker push

This is the OCI Distribution Spec!!

```
POST /v2/<name>/blobs/uploads/
```

Leveraging the OCI Distribution Spec for other things (such as Helm charts)

OCI = Open *Containers* Initiative - but the distribution spec provides value as a generalized API for storage

- So now what?

Enter the ORAS project:

OCI Registry As Storage

The ORAS project: a toolset for multi-artifact OCI registry support in Go

- Microsoft engineer Shiwei Zhang creates the first version of ORAS based on existing work he and his team has done for storing Helm charts in Azure Container Registry
- I help ensure that the library works with Helm
- Allows pushing of custom artifacts to an OCI registry, and specifying different mediatypes

github.com/deislabs/oras

Helm 3 uses the ORAS library - so you can now do things like this:

```
$ export HELM_EXPERIMENTAL_OCI=1
```


```
$ helm registry login mysite.io
$ helm chart pull mysite.io/somechart:1.5.0
$ helm chart save mychart/ mysite.io/mychart:2.0.1
$ helm chart push mysite.io/mychart:2.0.1
```

For more info on Helm 3 registry support:

v3.helm.sh/docs/topics/registries

Also - join the workshop at 14:35 today!

@jdolitsky

So now we can push Helm charts, but where do we push them to??

That’s a bit of a tricky question…

Support for new artifact types are limited on public hubs/registries due to reasons such as UX, security, etc.

Two (2) primary open source options to deploy our own registry:

1.
2.

Two (2) primary open source options to deploy our own registry:

1. **Docker Distribution**
2.

Docker Distribution: The Docker toolset to pack, ship, store, and deliver content

- Also known as “Docker Registry”
- Adheres to the OCI spec (because the spec was based upon it!)
- Several config options, storage backends, etc.
- Relatively easy to get started using Docker
- `$ docker run --rm -p 5000:5000 registry`

github.com/docker/distribution

Built on top of Docker Distribution:

Bundle Bar

A registry for lightweight, cloud-native artifacts

bundle.bar

bundle.bar

Welcome to

Bundle Bar!

Bundle Bar is a free service that hosts new types of lightweight, cloud-native artifacts for use by tools leveraging OCI for package distribution.

Community-provided, public artifacts are made available for download. Please see below for a complete list of artifacts hosted by type.

Supported Artifact Types

Helm Charts

Helm 3 provides support for downloading chart packages from OCI-based registries. For more info on how to use these features, please see this doc.

Here's an example of how to install one of the charts below using Helm 3:

```
export HELM_EXPERIMENTAL_OCI=1
helm chart pull bundle.bar/helm/hub/bitnami/nginx:4.3.3
helm chart export bundle.bar/helm/hub/bitnami/nginx:4.3.3
helm install myrelease nginx/
```

The following chart packages have been imported from Helm Hub (880 total):

```
bundle.bar/helm/hub/agones/agones:0.12.0
bundle.bar/helm/hub/appdynamics/cluster-agent:0.1.1
bundle.bar/helm/hub/appdynamics/machine-agent:0.2.3
bundle.bar/helm/hub/appscode/csi-vault:0.2.0
bundle.bar/helm/hub/appscode/g2:0.3.2
```

@jdolitsky

But what about..

- Vulnerability scanning?
- Content trust?
- SSO integration and RBAC?
- Replication?
- User interface?

Two (2) primary open source options to deploy our own registry:

1. Docker Distribution
2. **Harbor!!**

Harbor: an open source cloud native registry that stores, signs, and scans container images for vulnerabilities

- Originally a VMware project, now part of CNCF
- Adds extra features on top of Docker Distribution
- Nice graphical UI to manage images, permissions
- OpenID Connect (OIDC) and LDAP/AD support
- Collection of tools such as Notary, Claire, etc.
- Actually already supports charts via ChartMuseum

github.com/goharbor/harbor

Deploying Harbor with Helm 2

Easy right?

```
$ helm repo add harbor https://helm.goharbor.io
$ helm install --name=harbor harbor/harbor
```

github.com/goharbor/harbor

Deploying Harbor with Helm 2

```
> kubectl get pods [redacted]
NAME                                          READY   STATUS    RESTARTS   AGE
harbor-harbor-chartmuseum-bf97b6df6-dwktq     1/1     Running   0          109s
harbor-harbor-clair-5bd8f76d9c-q9sm4          1/1     Running   0          109s
harbor-harbor-core-855fc68cdd-7rgrn           1/1     Running   0          108s
harbor-harbor-database-0                      1/1     Running   0          107s
harbor-harbor-jobservice-6c58df48c5-vtp7f     1/1     Running   0          108s
harbor-harbor-notary-server-6f6bb7874-ptcnz   1/1     Running   1          108s
harbor-harbor-notary-signer-59988f796c-2qg42  1/1     Running   1          108s
harbor-harbor-portal-64cff84747-fvnbq         1/1     Running   0          108s
harbor-harbor-redis-0                         1/1     Running   0          107s
harbor-harbor-registry-56cd7b9c79-rpjks       2/2     Running   0          108s
```

github.com/goharbor/harbor

@jdolitsky

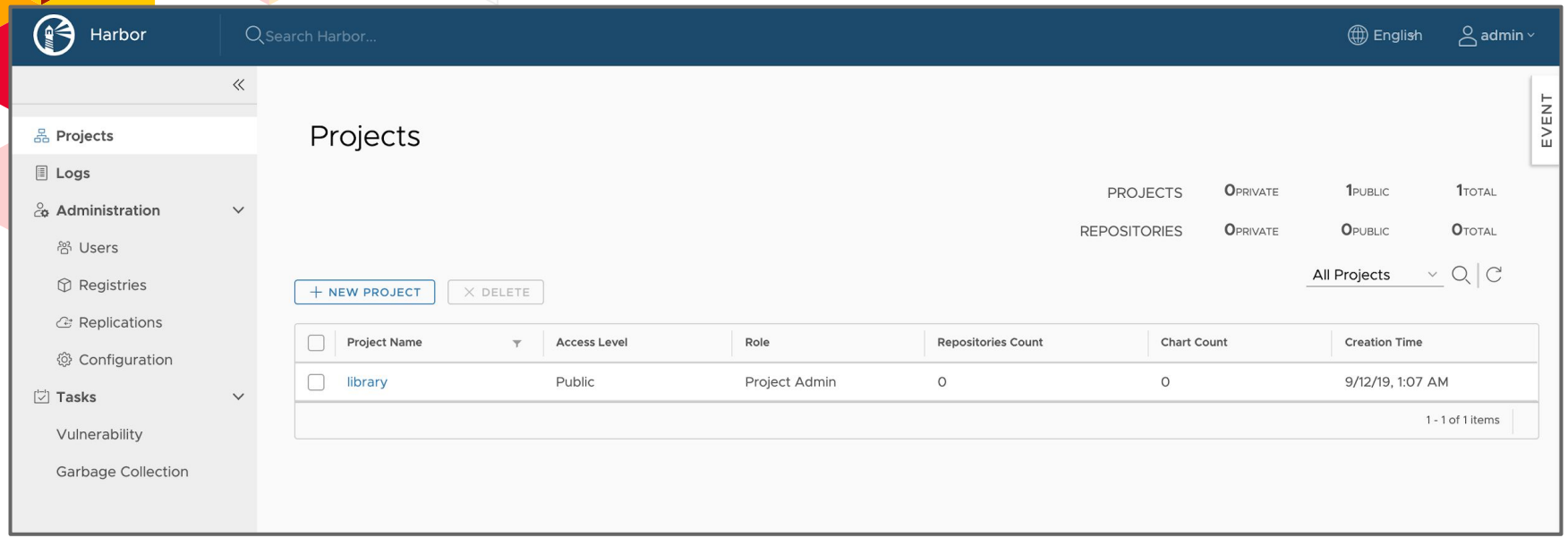
Accessing the Harbor UI

```
$ POD_NAME="$(kubectl get pods \
    | grep core | awk '{print $1}')"

$ kubectl port-forward $POD_NAME 8080:8080
```

Go to http://localhost:8080 in your browser,
and enter default admin credentials:

Harbor UI


Harbor
Search Harbor...
English
admin
Projects
Logs
Administration
Users
Registries
Replications
Configuration
Tasks
Vulnerability
Garbage Collection
Projects
EVENT
PROJECTS 0 PRIVATE 1 PUBLIC 1 TOTAL
REPOSITORIES 0 PRIVATE 0 PUBLIC 0 TOTAL
NEW PROJECT
DELETE
All Projects
Project Name
Access Level
Role
Repositories Count
Chart Count
Creation Time
library
Public
Project Admin
0
0
9/12/19, 1:07 AM
1 - 1 of 1 items

Tweaking Harbor to accept Helm charts

```yaml
# helm-support.yaml
core:
 image:
   repository: bloodorangeio/harbor-core
   tag: master
```

OCI Artifacts support is still in-progress in Harbor, so we use a fork with a few small workarounds

Tweaking Harbor to accept Helm charts

```yaml
# helm-support.yaml
core:
 image:
   repository: bloodorangeio/harbor-core
   tag: master

notary:
  enabled: false
externalURL: https://core.harbor.bundle.bar
expose:
 ingress:
   hosts:
     core: core.harbor.bundle.bar
   annotations:
     kubernetes.io/ingress.class: nginx
     kubernetes.io/tls-acme: "true"
     ingress.kubernetes.io/proxy-body-size: "1m"
```

Disable notary (for now), enable ingress, request TLS cert via cert-manager, allow 1MB uploads

Tweaking Harbor to accept Helm charts

```
$ helm upgrade --install \
    harbor harbor/harbor \
    -f helm-support.yaml
```

Pushing charts to Harbor with Helm 3

```
[jdolitsky⎈uno:~/hs] $ helm fetch harbor/harbor
[jdolitsky⎈uno:~/hs] $ ls *.tgz
harbor-1.1.2.tgz
[jdolitsky⎈uno:~/hs] $ export HELM_EXPERIMENTAL_OCI=1
[jdolitsky⎈uno:~/hs] $ helm chart save harbor-1.1.2.tgz helmsummit.bundle.bar/library/harbor
ref:     helmsummit.bundle.bar/library/harbor:1.1.2
digest:  0f8c0650d55f5e00d11d7462381c340454a3b9e517e15a0187011dc305690541
size:    28.1 KiB
name:    harbor
version: 1.1.2
1.1.2: saved
[jdolitsky⎈uno:~/hs] $ helm registry login -u admin -p Harbor12345 helmsummit.bundle.bar
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
Login succeeded
[jdolitsky⎈uno:~/hs] $ helm chart push helmsummit.bundle.bar/library/harbor:1.1.2
The push refers to repository [helmsummit.bundle.bar/library/harbor]
ref:     helmsummit.bundle.bar/library/harbor:1.1.2
digest:  0f8c0650d55f5e00d11d7462381c340454a3b9e517e15a0187011dc305690541
size:    28.1 KiB
name:    harbor
version: 1.1.2
1.1.2: pushed to remote (1 layer, 28.1 KiB total)
[jdolitsky⎈uno:~/hs] $
```

So where does Helm stand now?

- We must evolve the experimental Helm 3 OCI work to integrate with the rest of Helm UX. For example:
  - `$ helm install mysite.io/mychart:2.0.1`

- We must follow/engage in the conversation going on in the new “OCI Artifacts” project:
  - **github.com/opencontainers/artifacts**

Most importantly, we need YOU!

- Use it
- Break it
- Fix it
- Live it

Sources

- **"Quay: Introducing an Application Registry for Kubernetes"** - Antoine Legrand, 2017-03-23
https://coreos.com/blog/quay-application-registry-for-kubernetes.html

- **"Helm: Understanding Charts API vs App Registry"** - Matt Farina, 2017-09-25
https://codeengineered.com/blog/2017/helm-charts-api-vs-app-registry/

- **"The Hitchhiker's Guide to the Container Galaxy"** - Itay Shakury, 2018-02-06
https://blog.itaysk.com/2018/02/06/the-hitchhickers-guide-to-the-container-galaxy

- **"Cloud Native Artifact Registries evolve from Docker Container Registries"** - Steve Lasker, 2019-01-25
https://stevelasker.blog/2019/01/25/cloud-native-artifact-stores-evolve-from-container-registries/

- **"Distributing with Distribution: Upcoming Changes to Helm Chart Repositories"** - Matt Fisher, 2019-01-25
https://blog.bacongobbler.com/post/2019-01-25-distributing-with-distribution/index.html

- **"OCI Artifacts"** - Jimmy Zelinskie, 2019-08-23
https://medium.com/@jzelinskie/oci-artifacts-f5f3b4ef2889

Thank you!

bloodorange.io