

Tekton Hub + Catalog

SIG Interoperability May 14, 2020

Christie Wilson (Google) + Vincent Demeester (Red Hat)

Tekton:

Industry-standard,
cloud-native CI/CD
platform components
and ecosystem.

Tekton:

Industry-standard,
cloud-native CI/CD
platform components
and ecosystem.

Tekton Ecosystem: Catalog + Hub

Tekton's Reusability Model

User Personas



Releng team:

Responsible for how
software is
delivered/deployed

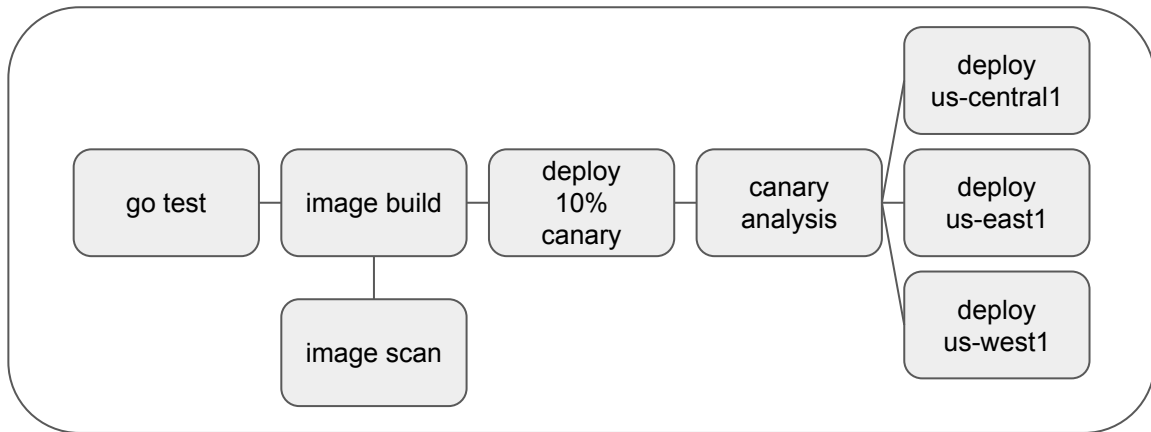


Appdev team:

Responsible for
creating the
software

Scenario

- Releng team supporting 8 teams
- Each team deploys and manages 1-3 services
- They want every team to:
 - Use more or less the same Pipeline
 - Not have to write their own Tasks for testing, building, etc.
 - Always include mandatory steps (e.g. image scanning)



Appdev team



Appdev team



Appdev team



Appdev team



Appdev team



Appdev team



Appdev team

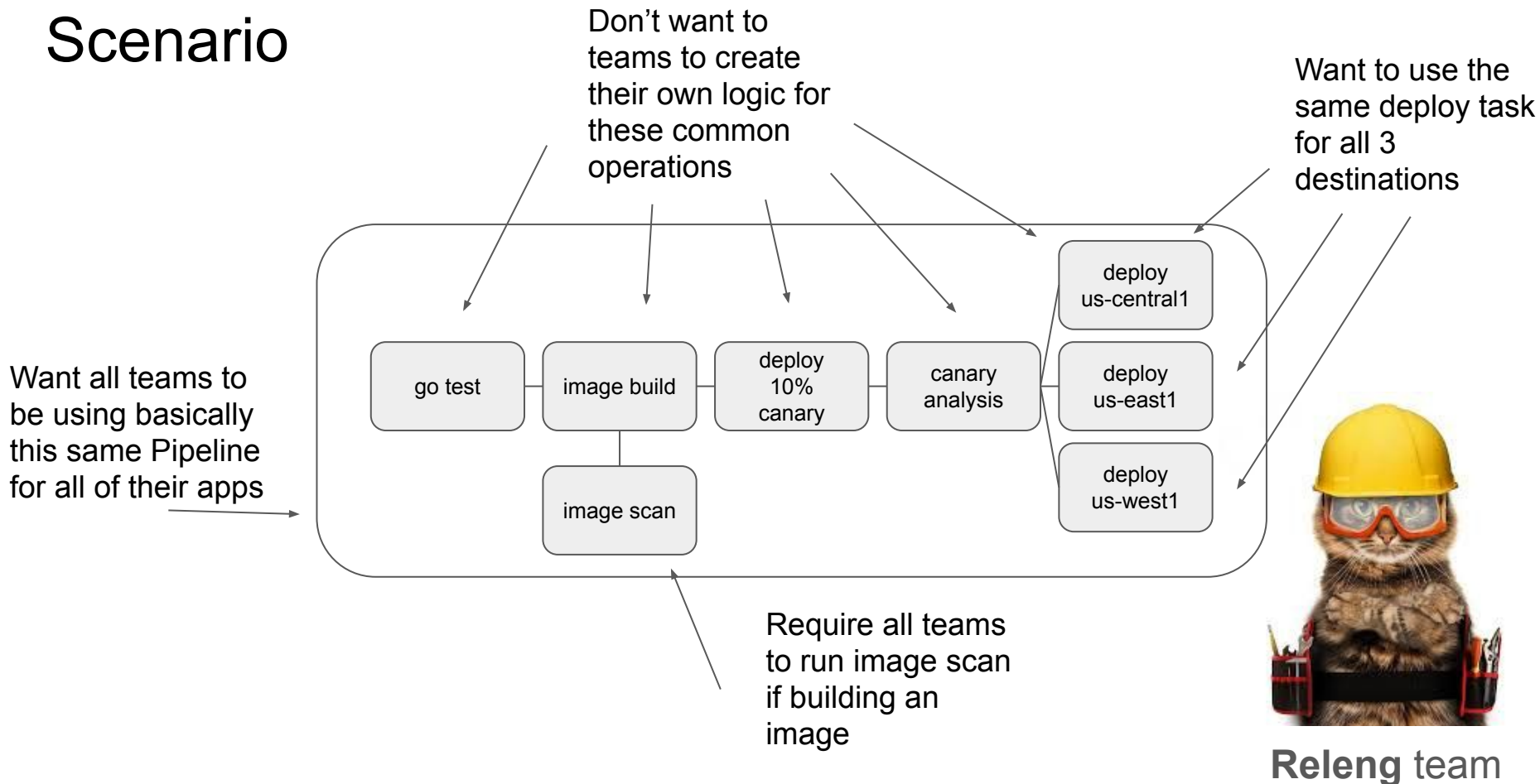


Appdev team



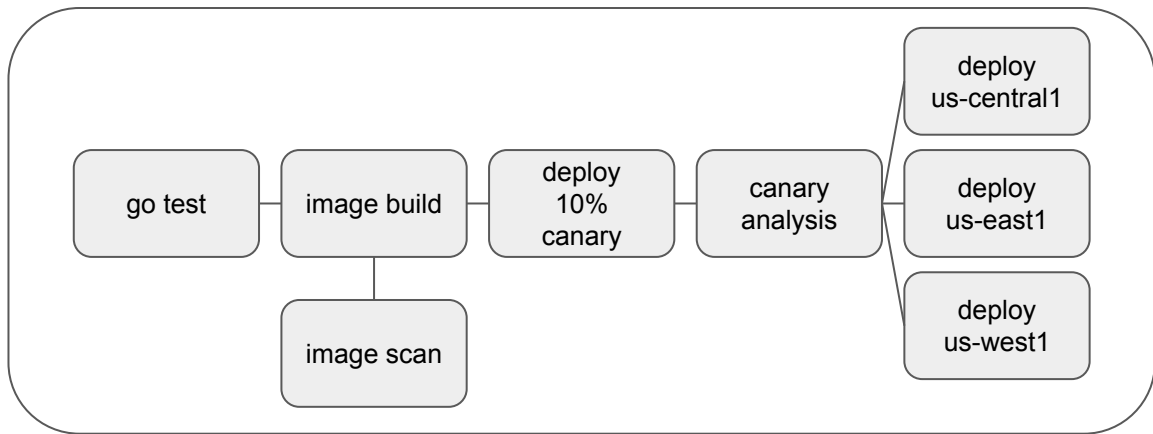
Releng team

Scenario

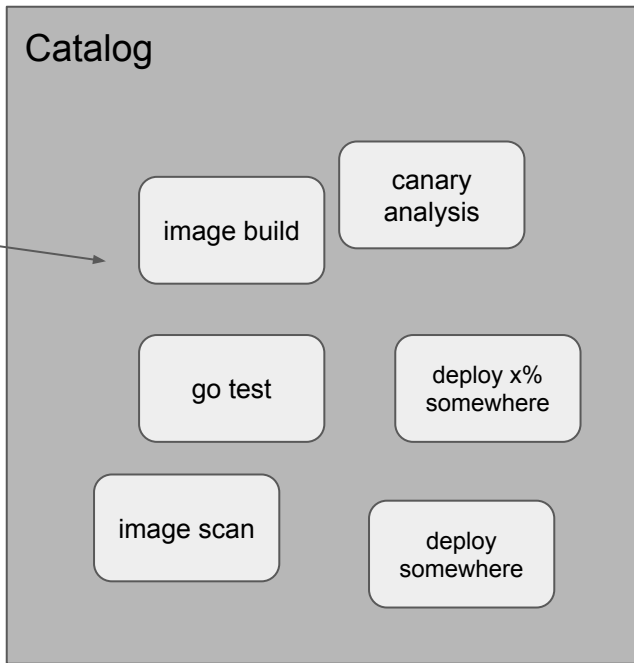


Tasks

- Pluggable, reusable **Tasks** assembled into **Pipelines**
- Tasks can be defined once and reused by multiple Pipelines, different teams



Releng
team writes
these



Appdev team
uses these

Simple Task

```
=== task.yaml ===
```

```
apiVersion: tekton.dev/v1beta1
```

```
kind: Task
```

```
metadata:
```

```
  name: go-test
```

It's the
go-test Task!




```
spec:
```

```
  steps:
```

```
  - image: golang:1.14
```

```
    script: go test ./...
```

Does what
it says on
the tin



```
$ kubectl apply -f task.yaml
```

```
$ tkn task start go-test
```

Run it with tkn



```
... streams logs from Pod ...
```

Parameterized Task (inputs)

```
apiVersion: tekton.dev/v1beta1
```

```
kind: Task
```

```
metadata:
```


```
  name: go-test
```

```
spec:
```

```
  params:
```

```
    - name: pkg  
      default: ./...
```


This parameter
lets the Task be
used for multiple
packages



```
  steps:
```

```
    - image: golang:1.14  
      script: go test $(params.pkg)
```

This is how
we use the
parameter



Parameterized Task (outputs) example

```
apiVersion: tekton.dev/v1beta1
```

```
kind: Task
```

```
metadata:
```

```
  name: kaniko
```

```
spec:
```


```
  results:
```

```
    - name: image
```

```
  steps:
```

```
    - image: gcr.io/kaniko-project/executor:latest
```

This Task declares
it provides an
output that other
Tasks can
consume



Pipeline

kind: Pipeline

metadata:

name: build-and-deploy

spec:

tasks:

- name: image-build

taskRef: kaniko

- name: deploy

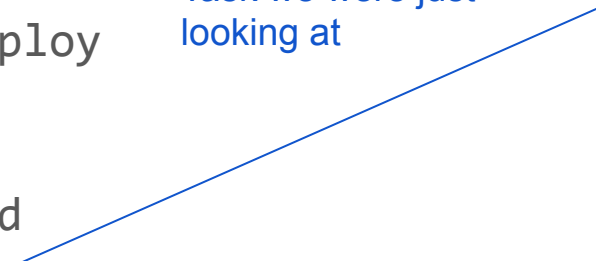
taskRef: kubectl-set-image

params:

- name: image

value: \$(tasks.image-build.results.image)

The Pipeline is
using the kaniko
Task we were just
looking at



apiVersion: tekton.dev/v1beta1

kind: Task

metadata:

name: kaniko

spec:

results:

- name: image

The deploy Task can
use the image result
from the Kaniko Task



Bundling Tasks in an OCI image

```
kind: Pipeline
```

```
...
```

```
spec:
```

```
  tasks:
```

```
    - name: test
```

```
      taskRef:
```

```
        image: gcr.io/project/catalog:v1.2.3
```

```
        name: go-test
```

```
    ...
```

```
...
```

What if we could
reference Tasks outside
the cluster?

Here we use oci image
versioning, can even
digests

Tekton Catalog



Releng
team writes
these

Catalog

image build

canary
analysis

go test

deploy x%
somewhere

image scan

deploy
somewhere

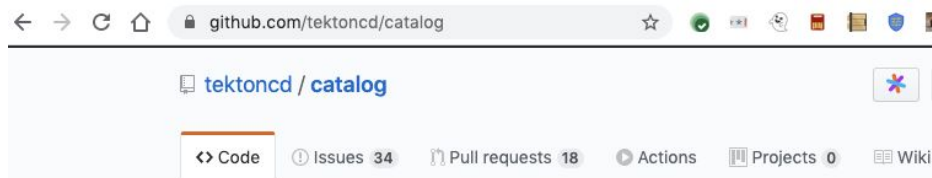


Appdev team
uses these

ansible-tower-cli
argocd
azure-cli
buildah
buildkit-daemonless
buildkit
buildpacks
conftest
gcloud
gcs
git
github

gke-deploy
golang
jib-gradle
jib-maven
kaniko
kn
knctl
kubecfg-creator
kubectl
kubeval
mail
makisu

maven
openshift-client-python
openshift-client
openshift-provision
openwhisk
pullrequest
replace-tokens
s2i
skopeo
slackmessage
telegrammessage
terraform-cli
tkn



2020 Tekton Catalog Roadmap

1. Component versioning (OCI)
2. Tekton Pipelines compatibility
3. Increased integration testing (dogfooding)
4. Tiers: e.g. community, verified, official
5. Hub!

Tekton Hub

Discover official and community Tasks* for creating pipelines

** long-term it will be “any resource” (pipeline, trigger-template, ...)*



Welcome to Tekton Hub

Discover, search and share reusable Tasks and Pipelines

Sort Name ▾

Refine By: ×

Kind

- ☐ Task
- ☐ Pipeline

Support Tier

- ☐ Official
- ☐ Verified
- ☐ Community

Categories

- ☐ Build Tools
- ☐ CLI
- ☐ Cloud
- ☐ Deploy
- ☐ Image Build



★ 5.0

buildah

v0.2

Buildah task builds source into a container image and then pushes it to a container registry.

Updated 2 days ago

image-build



★ 3.0

gcs-create-bucket v0.1

A Task that creates a new GCS bucket.

Updated 2 days ago

cloud



★ 1.0

gcs-delete-bucket v0.1

A Task that deletes a GCS bucket.

Updated 2 days ago

cloud



★ 0.0

gcs-download v0.1

A Task that fetches files or directories from a GCS bucket and puts them on a Workspace.

Updated 2 days ago

cloud



★ 0.0



★ 5.0



★ 0.0



★ 0.0

2020 Tekton Hub Roadmap

1. Published (hub.tekton.dev)
2. Multiple catalog support
 - a. Officials (tektoncd/catalog, tektoncd/community*)
 - b. Vendors (openshift/*, vmware/*, google/*)
 - c. Community (nixos/*, vdemeester/*, bobcatfish/*, ...)
3. API integrations
 - a. Tekton : “tkn catalog search”, dashboard, ...
 - b. Other consumers 🙋

Do you work on a
CI/CD system?
Consider supporting
the Tekton catalog!

Tekton Hub + Catalog

Join our working group!

Working group Thursdays
10:30am-11am PST

github.com/tektoncd/community/blob/master/working-groups.md#catalog-and-hub