

The background of the slide features a detailed painting of a three-masted sailing ship on a choppy sea under a blue sky with white clouds. The ship's wooden hull, rigging, and sails are clearly visible. A prominent wooden helm is positioned in the lower-left foreground, angled towards the right.

What the HELM !?

A HELM INTRODUCTION OR WHY EVERYONE SHOULD BE USING HELM

Haggai Philip Zagury | DevOps Architect & Group Lead Tikal Knowledge LTD



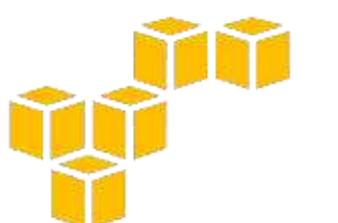
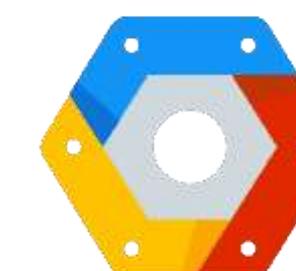
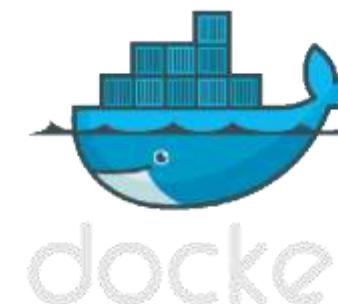
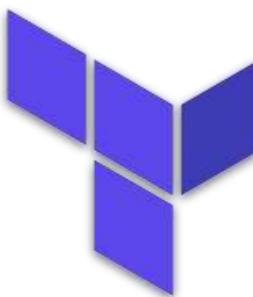
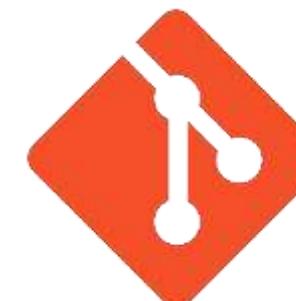
FullStack Developers Israel





HAGGAI PHILIP ZAGURY - DEVOPS ARCHITECT AND GROUP TECH LEAD

- ▶ My *open thinking* and *open techniques* ideology is driven by *Open Source technologies* and the collaborative manner defining my M.O.
- ▶ My solution driven approach is strongly based on hands-on and deep understanding of Operating Systems, Applications stacks and Software languages, Networking, Cloud in general and today more and more *Cloud Native* solutions.





[Get Helm](#)

[Blog](#)

[Docs](#)



The package manager for Kubernetes

Helm is the best way to find, share, and use software built for Kubernetes.



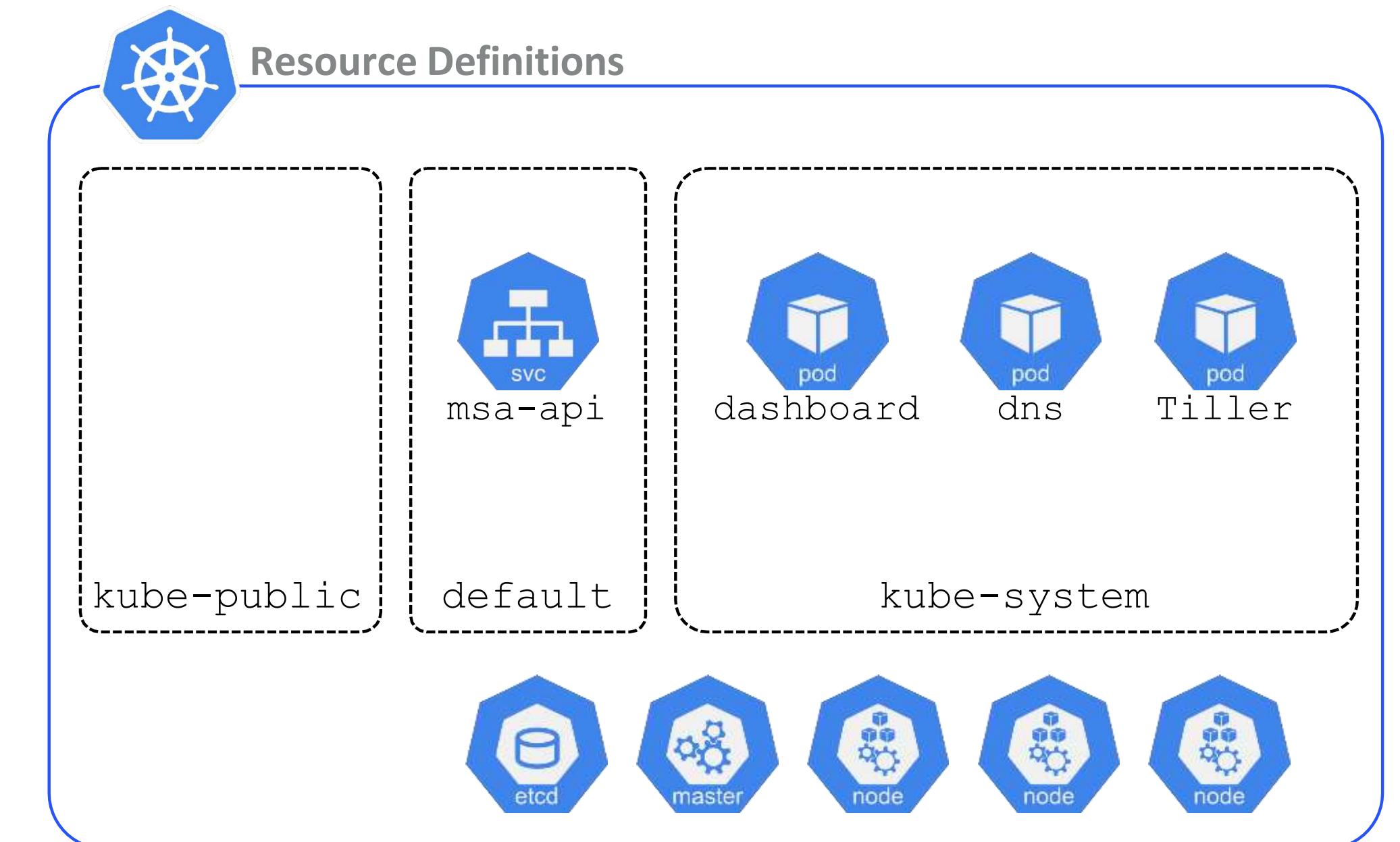
AGENDA

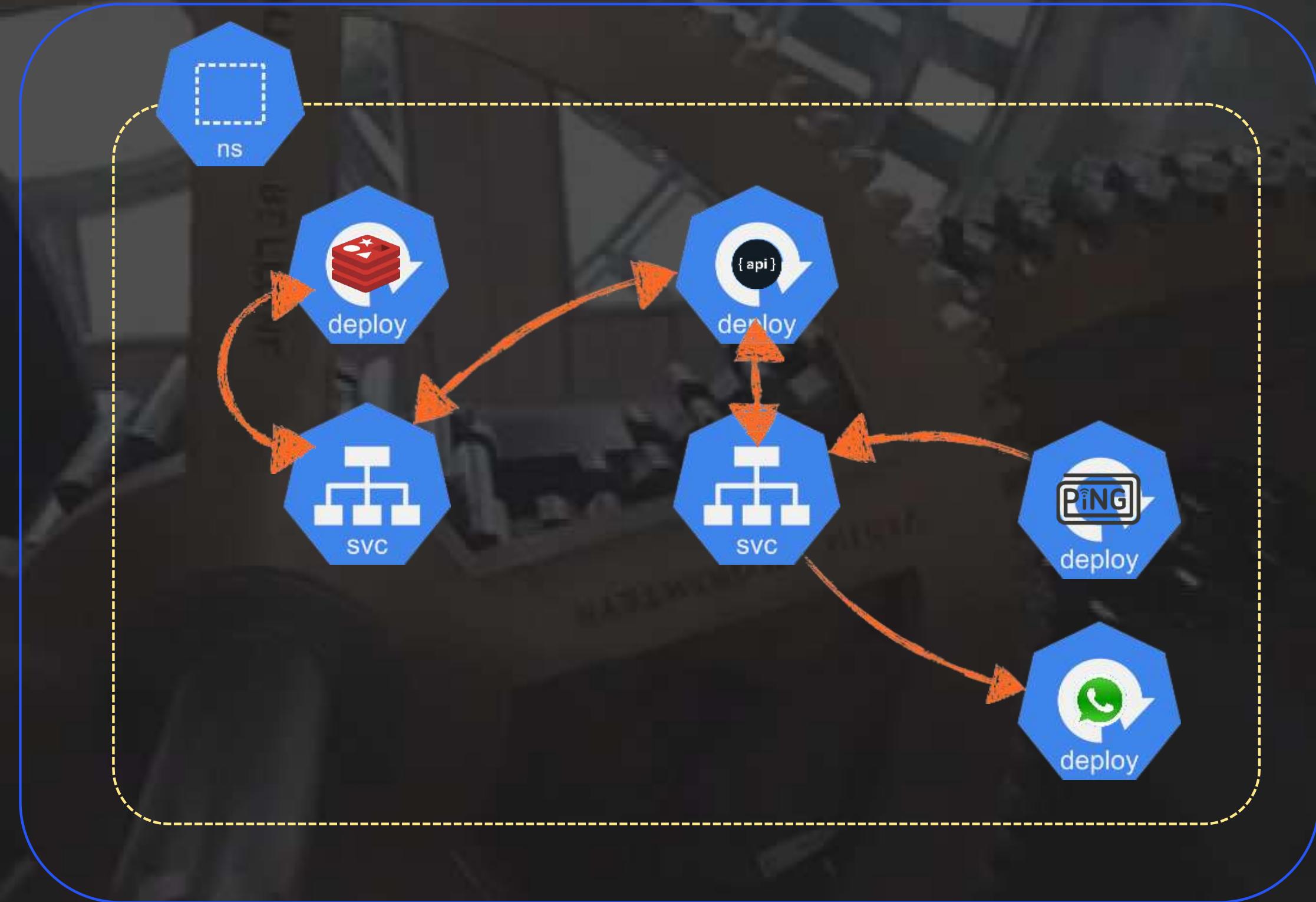
- ▶ Introduction to what helm is
 - ▶ Life before HELM
 - ▶ Intro - sample app
 - ▶ demo
 - ▶ Life with HELM
 - ▶ HELM Architecture
 - ▶ Real life examples



ASSUMPTIONS | PERQUISITES

- ▶ You know what kubernetes is:
 - ▶ Namespaces [[link](#)]
 - ▶ Labels & Selectors [[link](#)]
 - ▶ Deployments [[link](#)]
 - ▶ Pods [[link](#)]
 - ▶ Services [[link](#)]





LIFE BEFORE HELM



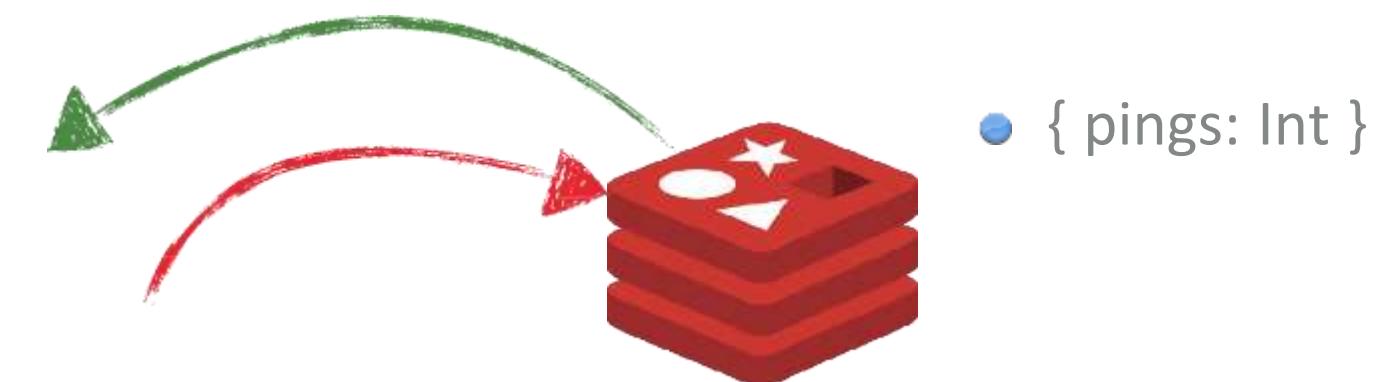


LAND

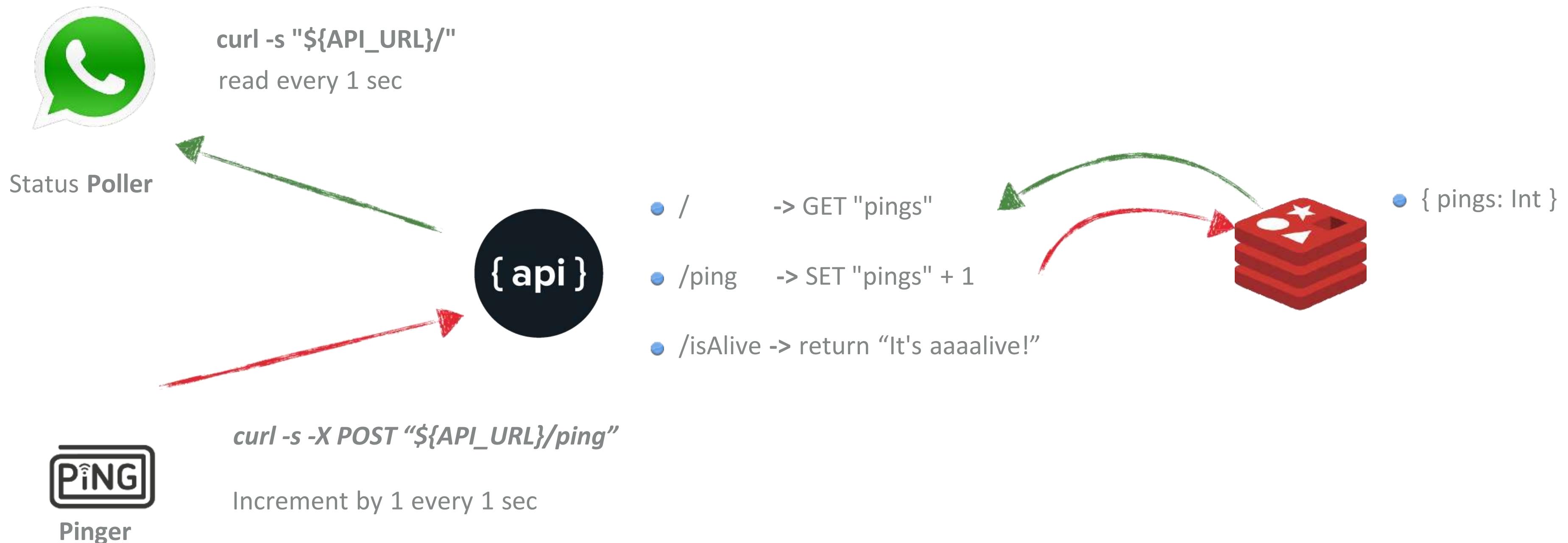
OUR “COMPLICATED” MICRO SERVICE APPLICATION



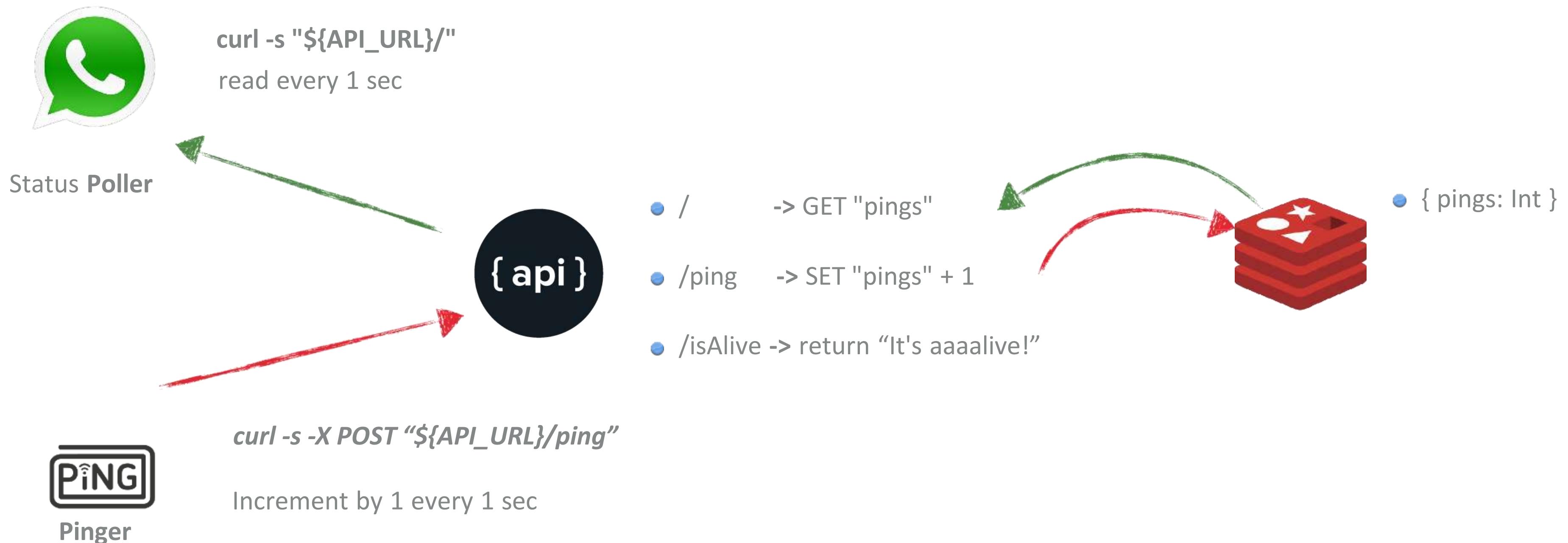
- / → GET "pings"
- /ping → SET "pings" + 1
- /isAlive → return "It's aaaalive!"



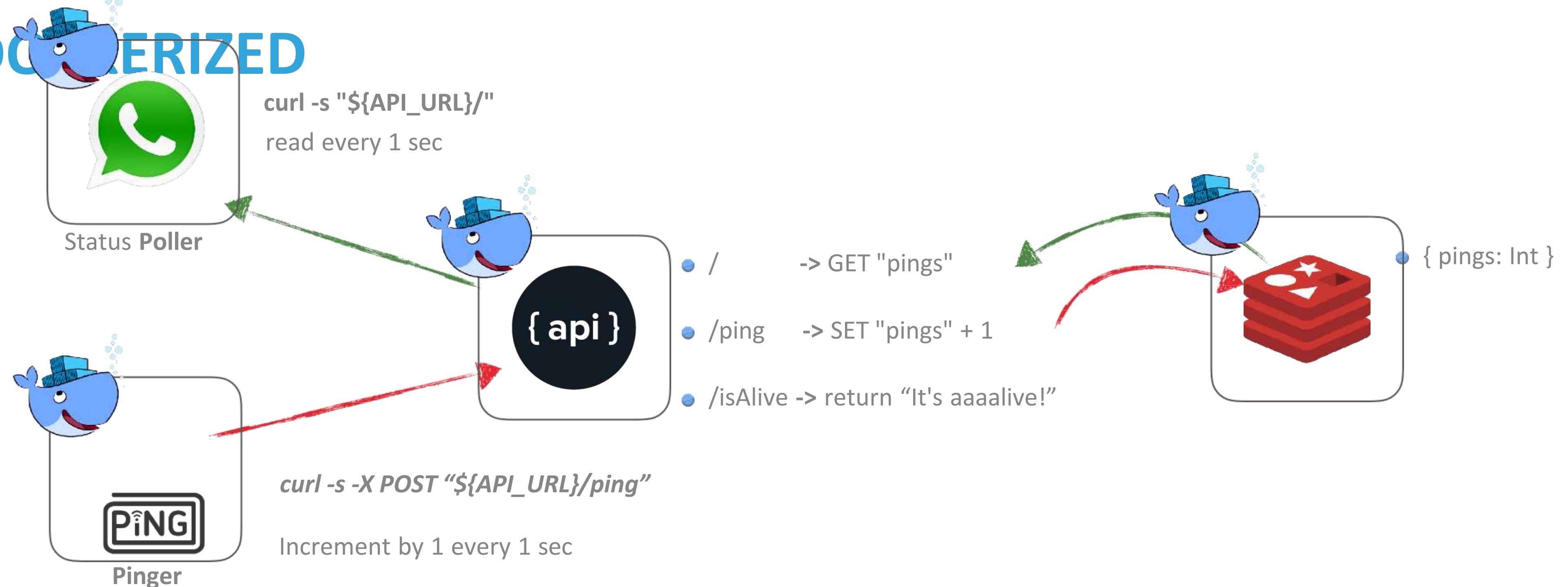
OUR “COMPLICATED” MICRO SERVICE APPLICATION



OUR “COMPLICATED” MICRO SERVICE APPLICATION



OUR “COMPLICATED” MICRO SERVICE APPLICATION - Dockerized



GET THE DEMO CODE

- ▶ <https://github.com/shelleg/msa-demo-app>

Branch: master ▾

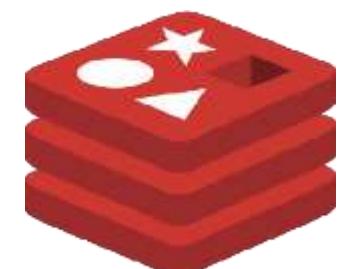
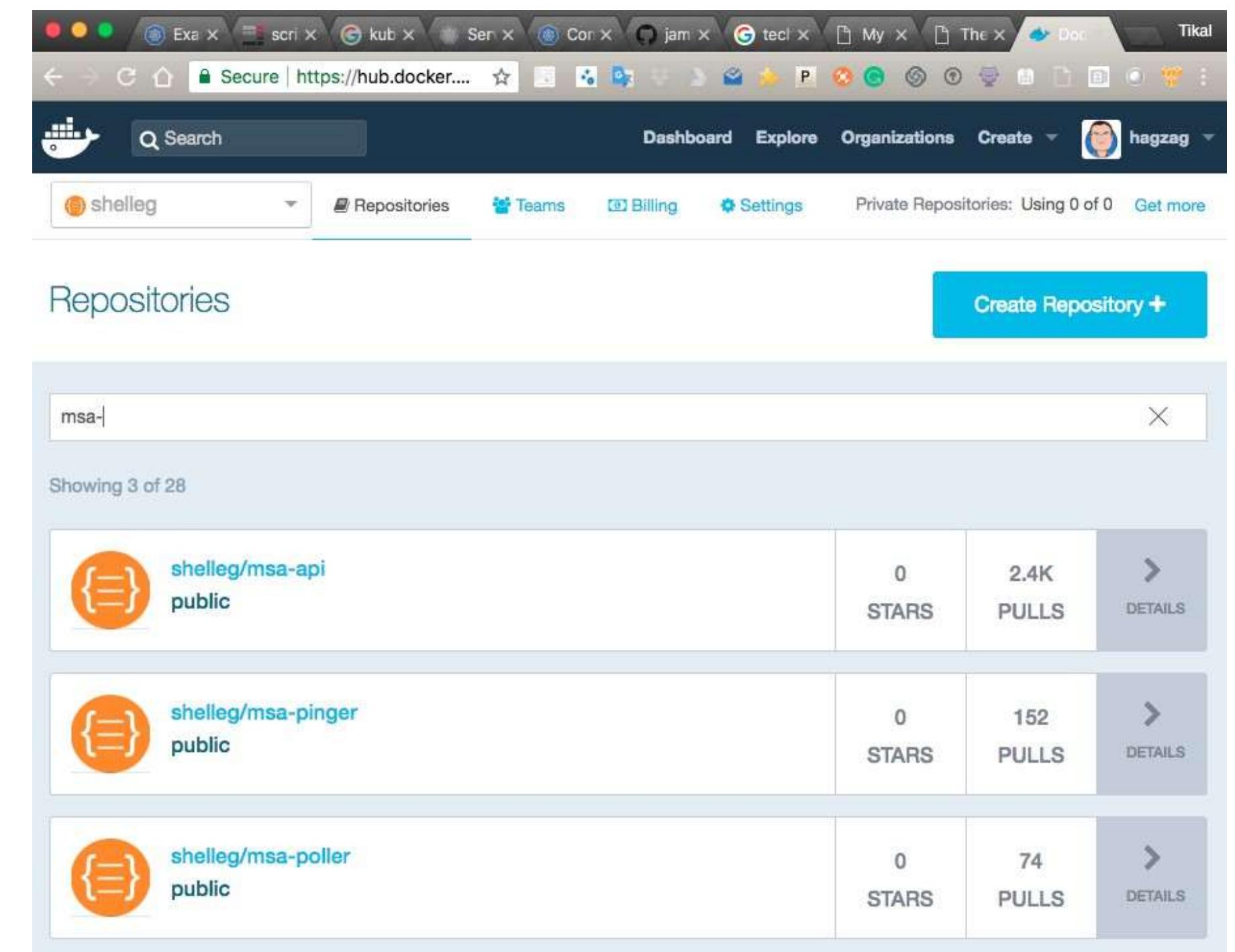
hagzag Initial Commit

msa-api	Initial Commit
msa-pinger	Initial Commit
msa-poller	Initial Commit
README.md	Initial Commit
docker-compose.yml	Initial Commit



PLANNING OUR DEPLOYMENT

- ▶ Each micro service has a build process which pushes the container to a docker registry.
- ▶ Docker registry is available to the k8s cluster during deployment (with or without authentication depending on the deployment)



DEPLOYMENT SEQUENCE

DEPLOYMENT SEQUENCE



► *redis* deployment + service for redis

DEPLOYMENT SEQUENCE



▶ *redis* deployment + service for redis



▶ **msa-api** deployment + service for the *msa-api*

DEPLOYMENT SEQUENCE



▶ *redis* deployment + service for redis



▶ **msa-api** deployment + service for the *msa-api*



▶ **msa-pinger** deployment

DEPLOYMENT SEQUENCE



▶ *redis* deployment + service for redis



▶ **msa-api** deployment + service for the *msa-api*



▶ *msa-pinger* deployment



▶ *msa-poller* deployment

DEPLOYMENT SEQUENCE



▶ *redis* deployment + service for redis



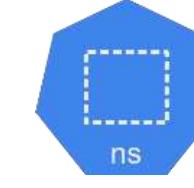
▶ **msa-api** deployment + service for the *msa-api*



▶ *msa-pinger* deployment



▶ *msa-poller* deployment



▶ *msa-demo* namespace

DEPLOYMENT SEQUENCE



- ▶ *redis* deployment + service for redis



- ▶ **msa-api** deployment + service for the *msa-api*



- ▶ *msa-pinger* deployment



- ▶ *msa-poller* deployment



- ▶ *msa-demo* namespace



- ▶ *persistent volume* for redis data

DEPLOYMENT SEQUENCE



- ▶ *redis* deployment + service for redis



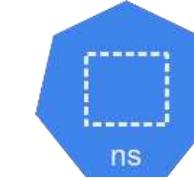
- ▶ **msa-api** deployment + service for the *msa-api*



- ▶ *msa-pinger* deployment



- ▶ *msa-poller* deployment



- ▶ *msa-demo* namespace



- ▶ *persistent volume* for redis data



- ▶ *configmaps / secrets* for all components

KUBERNETES “NATIVE” WAY



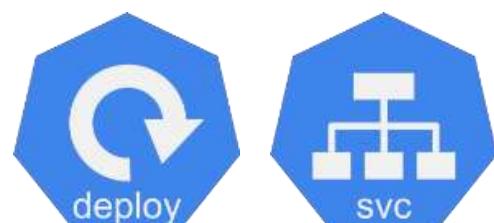
- ▶ Create a namespace:

`kubectl create namespace msa-demo --dry-run -o yaml > msa-demo-ns.yml`



- ▶ Create a deployment + service for redis:

`kubectl run redis --image=redis --port=6379 --expose --dry-run -o yaml > redis.yml`



- ▶ Create a deployment + service for msa-api:

`kubectl run msa-api --image=shelleg/msa-api:config --port=8080 --expose --image-pull-policy=Always --dry-run -o yaml > msa-api.yml`

KUBERNETES “NATIVE” WAY



- ▶ Create a deployment for msa-pinger:

```
kubectl run msa-pinger --image=shelleg/msa-pinger:latest --env="API_URL=msa-api: 8080" --  
env="DEBUG=true" --dry-run -o yaml > msa-pinger.yml
```



- ▶ Create a deployment for msa-poller:

```
kubectl run msa-poller --image=shelleg/msa-poller:latest --env="API_URL=msa-api:8080"  
--dry-run -o yaml > msa-poller.yml
```

A close-up photograph of a person's hands typing on a white laptop keyboard. The hands are positioned in a standard QWERTY layout, with fingers pressing the keys. The background is slightly blurred.

HANDS ON



FullStack Developers Israel



WE NOW HAVE A BUNCH OF FILES DESCRIBING OUR APPLICATION

- | -- msa-api.yml
| -- msa-demo-ns.yml
| -- msa-pinger.yml
| -- msa-polller.yml
`-- redis.yml

0 directories, 5 files



```
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  name: msa-api
spec:
  ports:
    - port: 8080
      protocol: TCP
      targetPort: 8080
  selector:
    run: msa-api
status:
  loadBalancer: {}
```



```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    run: msa-api
    name: msa-api
spec:
  replicas: 1
  selector:
    matchLabels:
      run: msa-api
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        run: msa-api
    spec:
      containers:
        - image: shelleg/msa-api:config
          imagePullPolicy: Always
```



SO WHY DO WE NEED HELM ?!



- ▶ We want to better control “what goes where” e.g ports
- ▶ Container image versions ?! (is the only change between releases)
- ▶ Manage Dependencies
- ▶ Update components individually updating things like image versions e.g *msa-api:latest* -> *msa-api:config*
- ▶ Write once run on any cluster
- ▶ Reproducible & Shareable resource definition specs -
TEMPLATES



```
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  name: msa-api
spec:
  ports:
    - port: {{ variable }}
      protocol: TCP
      targetPort: {{ variable }}
  selector:
    run: msa-api
status:
  loadBalancer: {}
```

TEMPLATES



- ▶ Not a foreign concept ... Jinja / erb / jsonnet

```
{% for key, svc in services.items() %}  
upstream {{ key }} {  
  least_conn;  
  {% for server in svc.delegate %}  
    server {{ server }}:{{ svc.ports[0].split(':')[0] }} weight=10  
  max_fails=3 fail_timeout=30s;  
  {% endfor %}  
}  
{% endfor %}
```



```
<% @sudoers_users.each do |user| -%>  
<%= user %> ALL=(ALL) <%= "NOPASSWD:" if @passwordless %>ALL  
<% end -%>  
  
# Members of the sysadmin group may gain root privileges  
%sysadmin      ALL=(ALL) <%= "NOPASSWD:" if @passwordless %>ALL
```



```
// A function that returns an object.  
local Person(name='Alice') = {  
  name: name,  
  welcome: 'Hello ' + name + '!',  
};  
{  
  person1: Person(),  
  person2: Person('Bob'),  
}
```

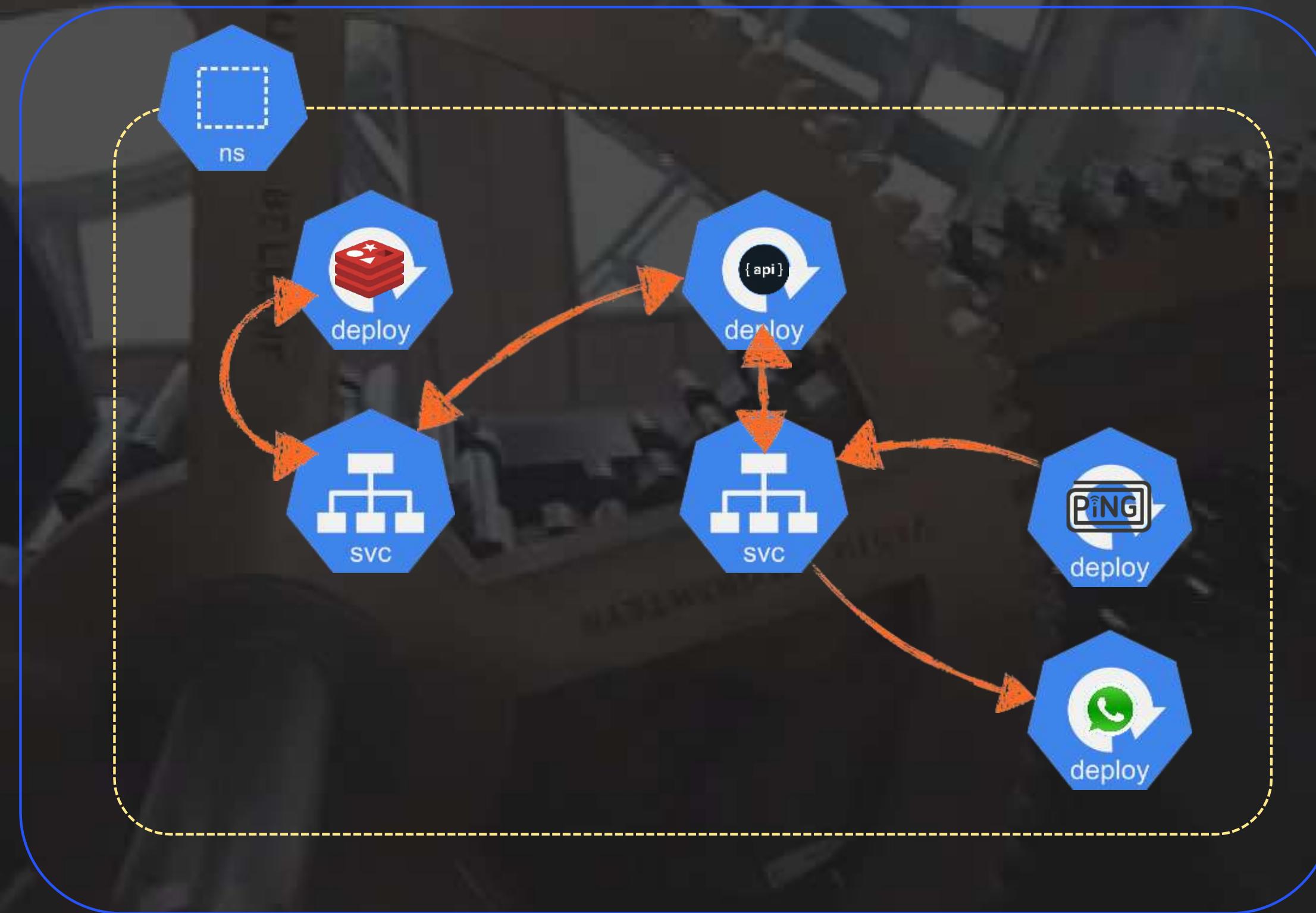


**BEEN THERE
DONE THAT**

SO WHY DO WE NEED HELM ?!

- ▶ Kubernetes native discovery mechanism is focused around the **Service** / deployment
- ▶ Helm Charts are **Application** focused





LIFE WITH HELM

IN A NUT SHELL -> HELM MANAGES CHARTS, RELEASES & VERSIONS



- .
- |--- Chart.yaml
- |--- charts
- |--- templates
- | |--- NOTES.txt
- | |--- _helpers.tpl
- | |--- deployment.yaml
- | `--- service.yaml
- `--- values.yaml

2 directories, 6 files

- ▶ Describe your deployment + service / any other kubernetes Resource Definition

```
apiVersion: v1
description: My Cool Chart
engine: gotpl
name: myCoolChart
version: 0.1.0
```

IN A NUT SHELL -> HELM MANAGES CHARTS



.

```
|--- Chart.yaml  
|--- charts  
|--- templates  
|   |--- NOTES.txt  
|   |--- _helpers.tpl  
|   |--- deployment.yaml  
`--- service.yaml  
`--- values.yaml
```

2 directories, 6 files

- ▶ Use *values.yaml*

```
image:  
  repository: shelleg/msa-api  
  tag: config  
  pullPolicy: IfNotPresent  
  
service:  
  type: ClusterIP  
  port: 8080  
...
```

- ▶ A rich templating system for making charts generic yet highly customisable

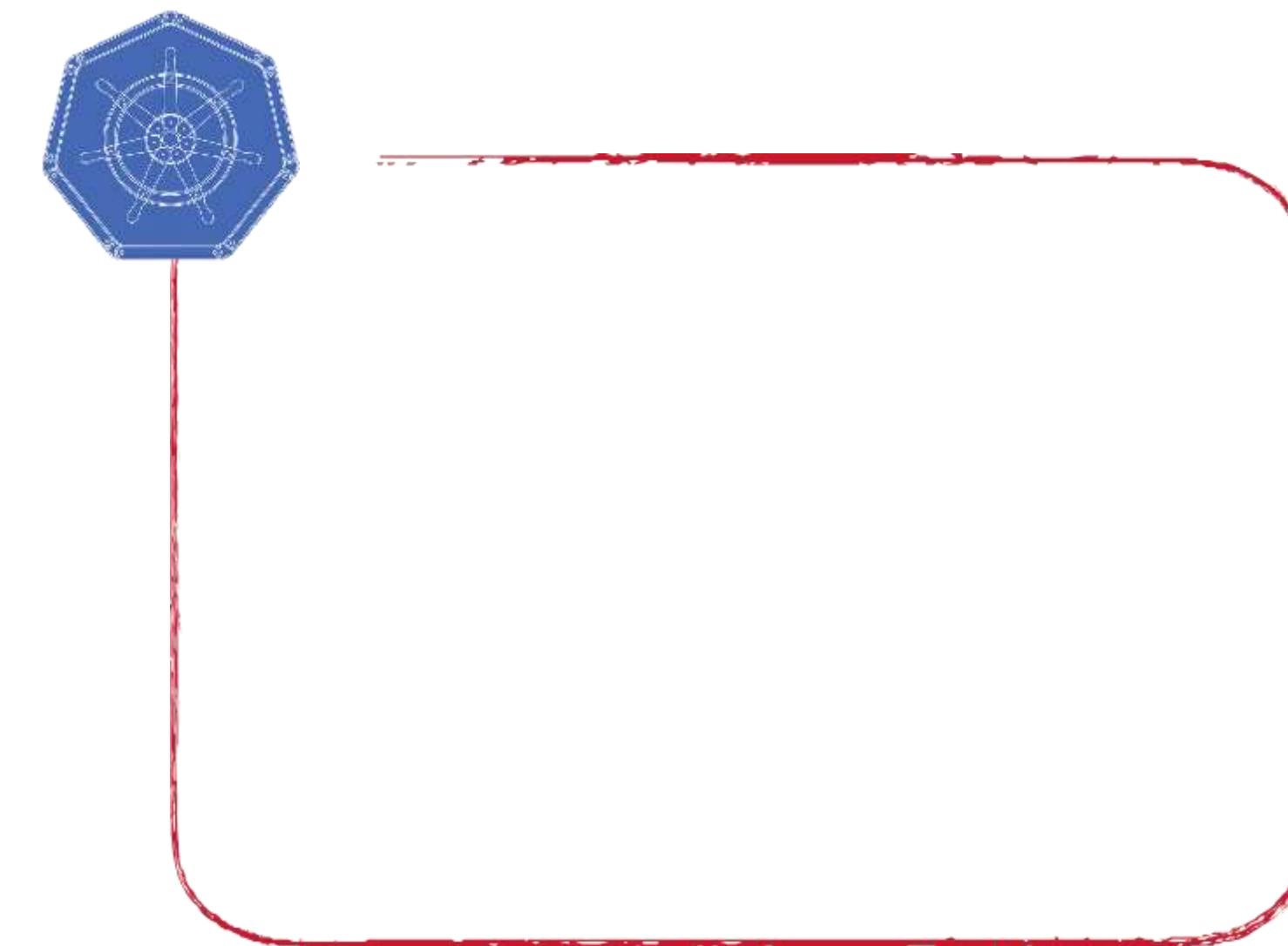
HOW DOES WORK ?

The Helm logo consists of the word "HELM" in a bold, sans-serif font, enclosed within a stylized blue compass rose icon.

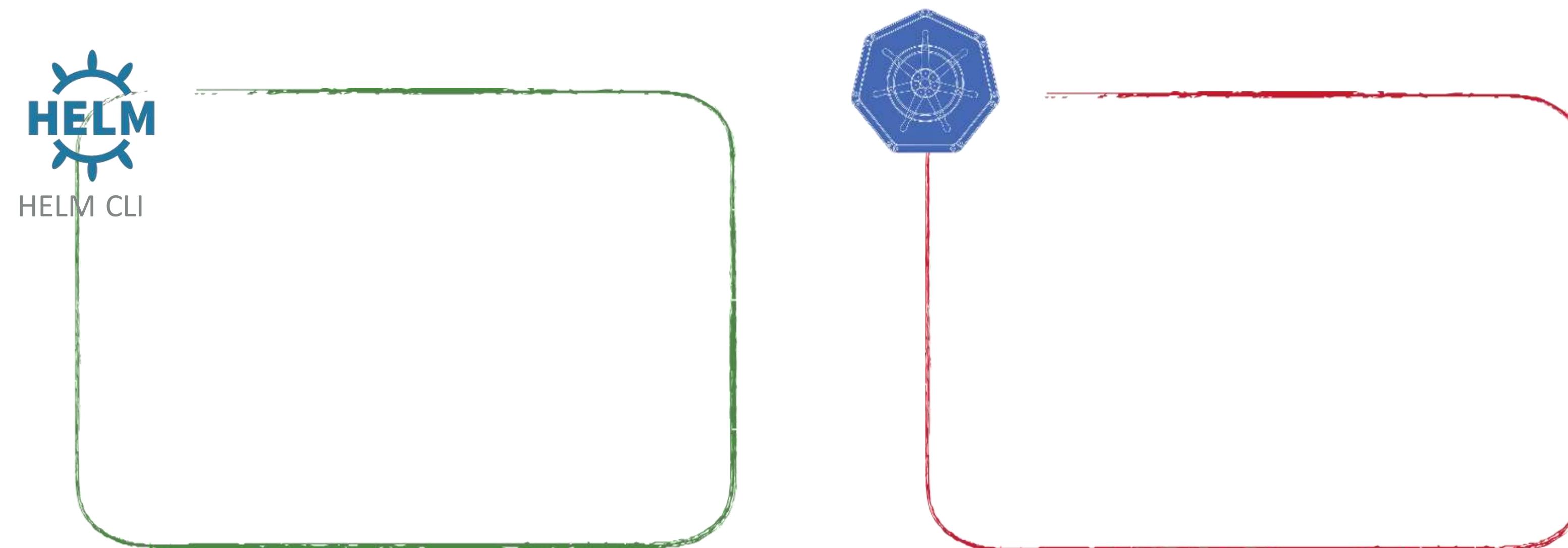
FullStack Developers Israel



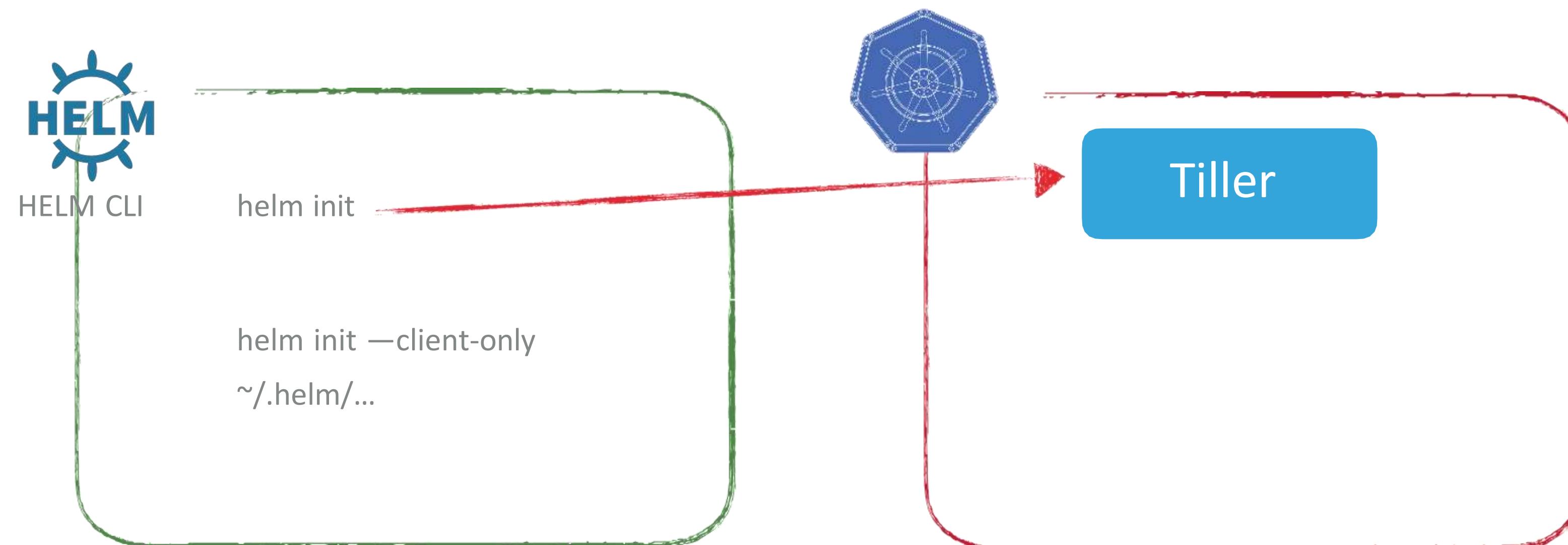
HELM ARCHITECTURE - A KUBERNETES CLUSTER



HELM ARCHITECTURE - A HELMSMAN WORKSTATION



HELM ARCHITECTURE - HELM INIT



HELM CLIENT



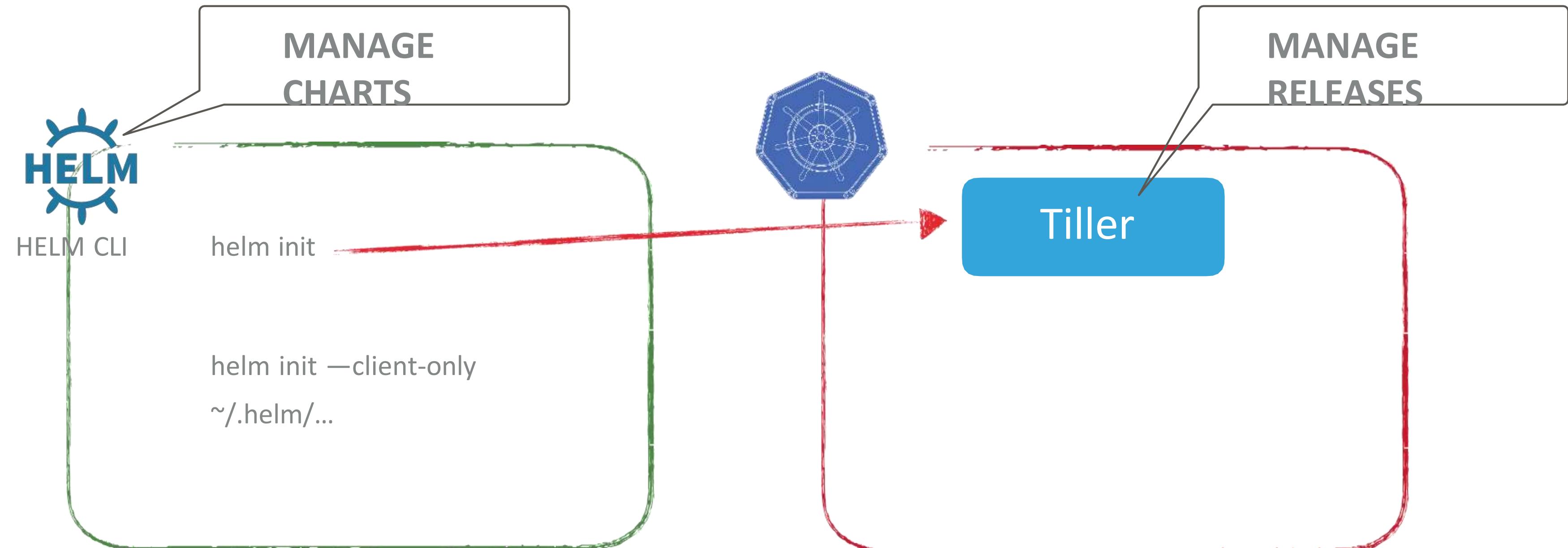
- ▶ Local chart development
- ▶ Managing repositories
- ▶ Interacting with the Tiller server
 - ▶ Sending charts to be installed
 - ▶ Asking for information about releases
 - ▶ Requesting upgrading or uninstalling of existing releases
- ▶ It can be used to serve your local charts via http ...

TILLER - “HELM SERVER”

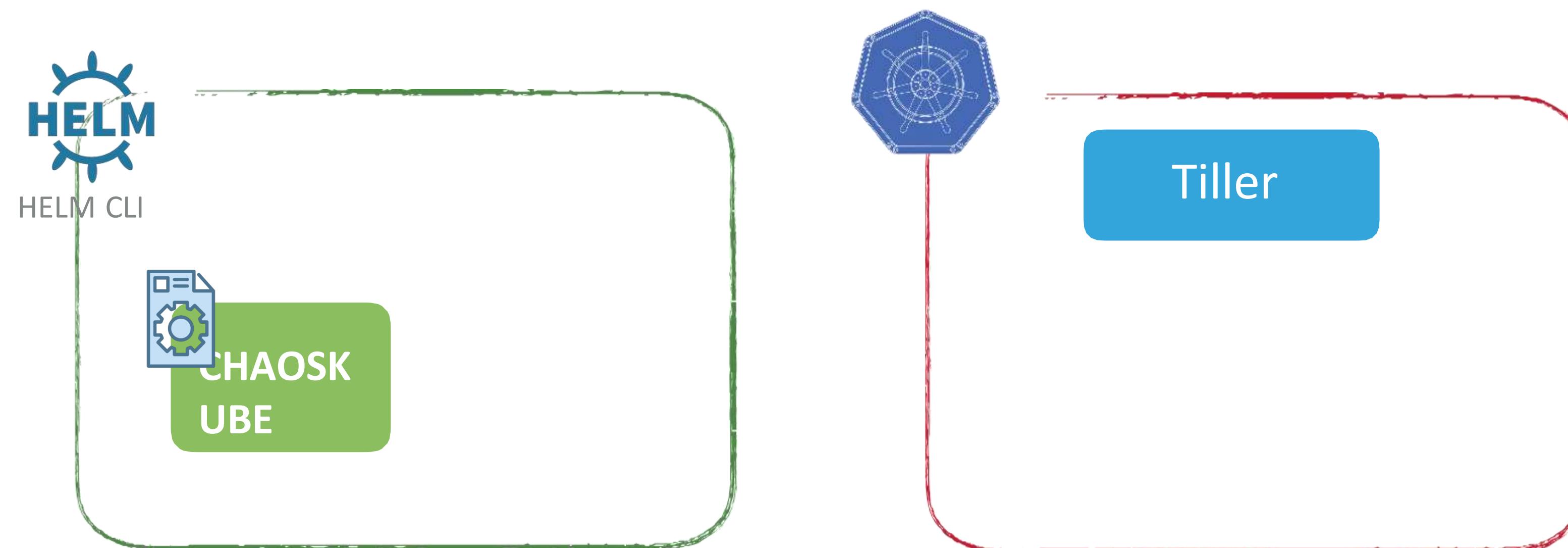


- ▶ Listening for incoming requests from the Helm client
- ▶ Combining a chart and configuration to build a release
- ▶ Installing charts into Kubernetes, and then tracking the subsequent release
- ▶ Upgrading and uninstalling charts by interacting with Kubernetes

HELM ARCHITECTURE - HELM INIT



HELM ARCHITECTURE - INSTALL AN EXISTING CHART





USE EXISTING CHARTS



FullStack Developers Israel



CHAOSKUBE



- ▶ chaoskube is a “*chaos-monkey lite*” it basically takes down pod based on a schedule to test you resilience (and there are some tweaks via configuration)

<https://github.com/linki/chaoskube>

chaoskube

build passing coverage 85% release v0.10.0 container ready godoc reference

chaoskube periodically kills random pods in your Kubernetes cluster.

Why

Test how your system behaves under arbitrary pod failures.

Example

Running it will kill a pod in any namespace every 10 minutes by default.

```
$ chaoskube
INFO[0000] starting up
INFO[0000] connecting to cluster
INFO[0000] setting pod filter
dryRun=true interval=10m0s version=v0.9.0
serverVersion=v1.9.3+coreos.0 master="https://kube.you.me"
annotations= labels= namespaces=
```

HELM ARCHITECTURE - WORK WITH REPOS



HELM REPO



helm repo list

NAME URL

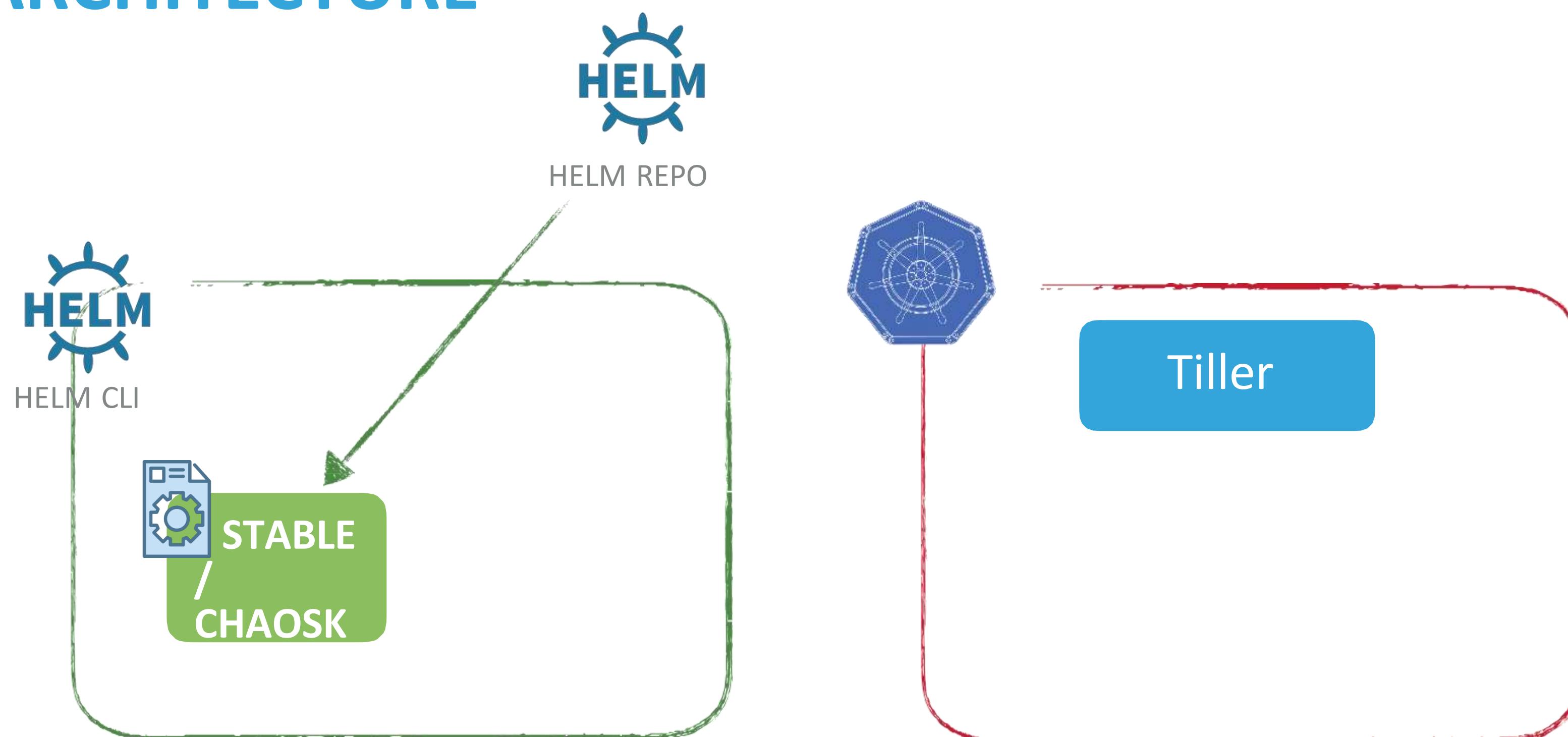
stable <https://kubernetes-charts.storage.googleapis.com>

local <http://127.0.0.1:8879/charts>



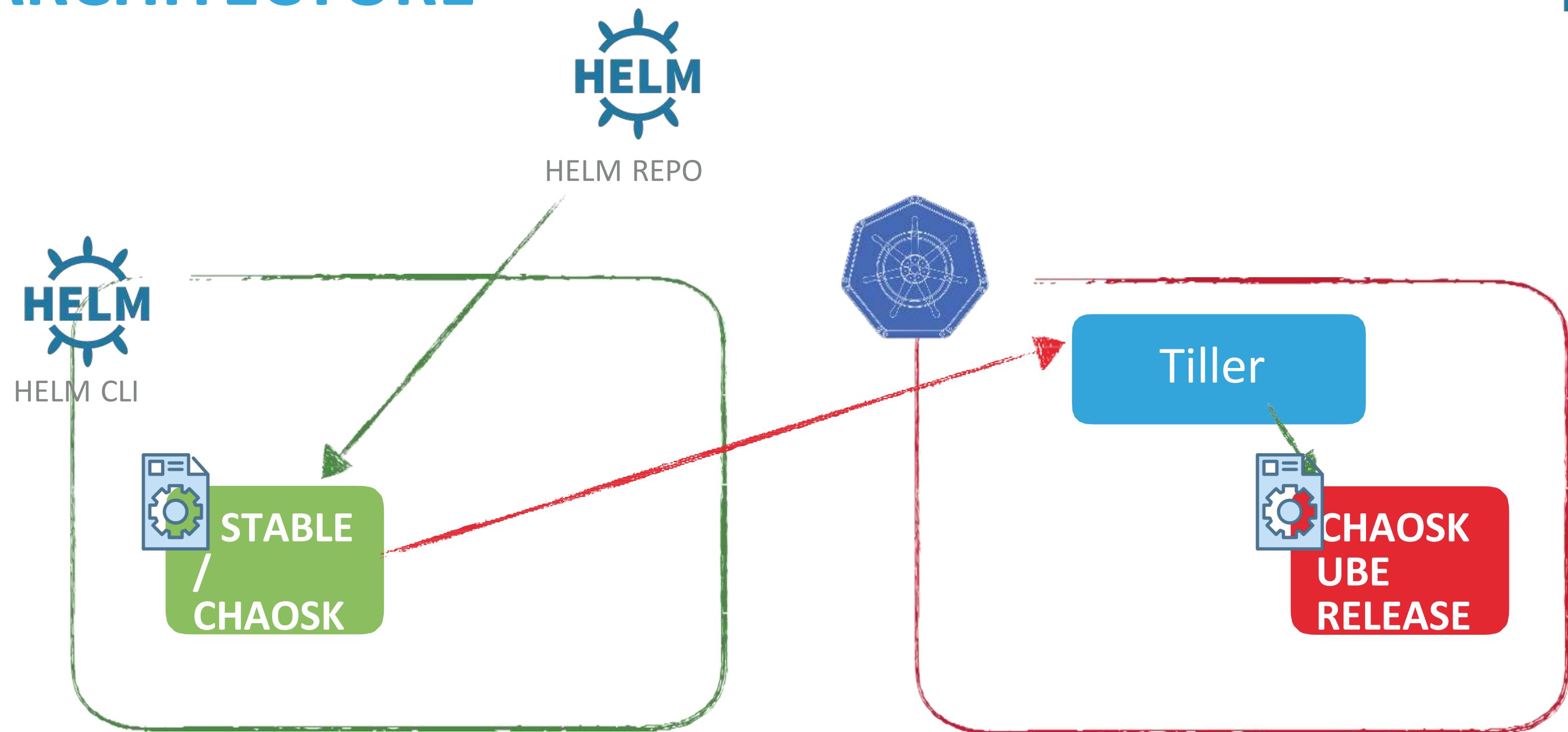
Tiller

HELM ARCHITECTURE



helm install stable/chaoskube

HELM ARCHITECTURE



CHARTS STRUCTURE



- - | -- Chart.yaml
 - | -- charts
 - | -- templates
 - | -- NOTES.txt
 - | -- _helpers.tpl
 - | -- deployment.yaml
 - | -- ingress.yaml
 - \-- service.yaml
 - \-- values.yaml

2 directories, 7 files



Chart.yaml

```
apiVersion: v1
appVersion: 0.10.0
description: Chaoskube periodically kills random pods in your cluster.
engine: gotpl
home: https://github.com/linki/chaoskube
maintainers:
- email: linki+kubernetes.io@posteo.de
  name: linki
name: chaoskube
sources:
- https://github.com/linki/chaoskube
version: 0.10.0
```

Chart and image version
{not always identical ...}

CHARTS STRUCTURE



```
| -- Chart.yaml  
| -- OWNERS  
| -- README.md  
| -- templates  
|   | -- NOTES.txt  
|   | -- _helpers.tpl  
|   | -- clusterrole.yaml  
|   | -- clusterrolebinding.yaml  
|   | -- deployment.yaml  
|   | -- role.yaml  
|   | -- rolebinding.yaml  
|   `-- serviceaccount.yaml  

```

1 directory, 12 files



- ▶ /templates for your **resource definitions** such as:

- ▶ Service
- ▶ Deployment
- ▶ configMap
- ▶ Secret
- ▶ Daemonset
- ▶ Pod
- ▶ Ingress
- ▶ Job
- ▶ persistentvolume
- ▶ persistentvolumeclaim
- ▶ Service account



CHARTS STRUCTURE

```
| -- Chart.yaml  
| -- OWNERS  
| -- README.md  
| -- templates  
|   | -- NOTES.txt  
|   | -- _helpers.tpl  
|   | -- clusterrole.yaml  
|   | -- clusterrolebinding.yaml  
|   | -- deployment.yaml  
|   | -- role.yaml  
|   | -- rolebinding.yaml  
|   `-- serviceaccount.yaml  
`-- values.yaml
```

1 directory, 12 files

values.yaml

```
# container name  
name: chaoskube  
  
# docker image  
image: quay.io/linki/chaoskube  
  
# docker image tag  
imageTag: v0.10.0  
  
# number of replicas to run  
replicas: 1  
  
# interval between pod terminations  
interval: 10m  
  
# label selector to filter pods by, e.g. app=foo,stage!=prod  
labels:  
  
# annotation selector to filter pods by, e.g.  
chaos.alpha.kubernetes.io/enabled=true  
annotations:  
...
```



CHARTS STRUCTURE



```

| -- Chart.yaml
| -- OWNERS
| -- README.md
| -- templates
|   | -- NOTES.txt
|   | -- _helpers.tpl
|   | -- clusterrole.yaml
|   | -- clusterrolebinding.yaml
|   | -- deployment.yaml
|   | -- role.yaml
|   | -- rolebinding.yaml
|   `-- serviceaccount.yaml
`-- values.yaml

```

1 directory, 12 files

serviceaccount.yaml

```

{{- if .Values.rbac.create -}}
apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    {{ include "labels.standard" . | indent 4 }}
    name: {{ printf "%s-%s" .Release.Name .Values.name }}
{{- end -}}

```

Control Structures:

{ render based on boolean }

values.yaml

```

# container name
name: chaoskube

```

CHARTS STRUCTURE

- | -- Chart.yaml
- | -- charts
- | -- templates
 - | -- NOTES.txt
 - | -- _helpers.tpl
 - | -- deployment.yaml
 - | -- ingress.yaml
 - | -- service.yaml
- \-- values.yaml

2 directories, 7 files

_helpers.tpl

```
/*
Expand the name of the chart.
*/
{{- define "chaoskube.name" -}}
{{- default .Chart.Name .Values.nameOverride | trunc 63 | trimSuffix "-" -}}
{{- end -}}


/*
Create a default fully qualified app name.
We truncate at 63 chars because some Kubernetes
name fields are limited to this (by the DNS
naming spec).
If release name contains chart name it will be
used as a full name.
*/
{{- define "chaoskube.fullname" -}}
{{- if .Values.fullnameOverride -}}
{{- .Values.fullnameOverride | trunc 63 |
trimSuffix "-" -}}
{{- else -}}
...
```

Custom functions +
set default based on
Environment



CHARTS STRUCTURE



- ▶ A *HELM standard way* of showing information the end user can use in order to access the application.

NOTES.txt

```
chaoskube is running and will kill arbitrary pods every
{{ .Values.interval }}.
```

You can follow the logs to see what chaoskube does:

```
POD=$(kubectl -n {{ .Release.Namespace }} get pods -
l='release={{ template "chaoskube.fullname" . }}' --
output=jsonpath='{{.items[0].metadata.name}}
')
```

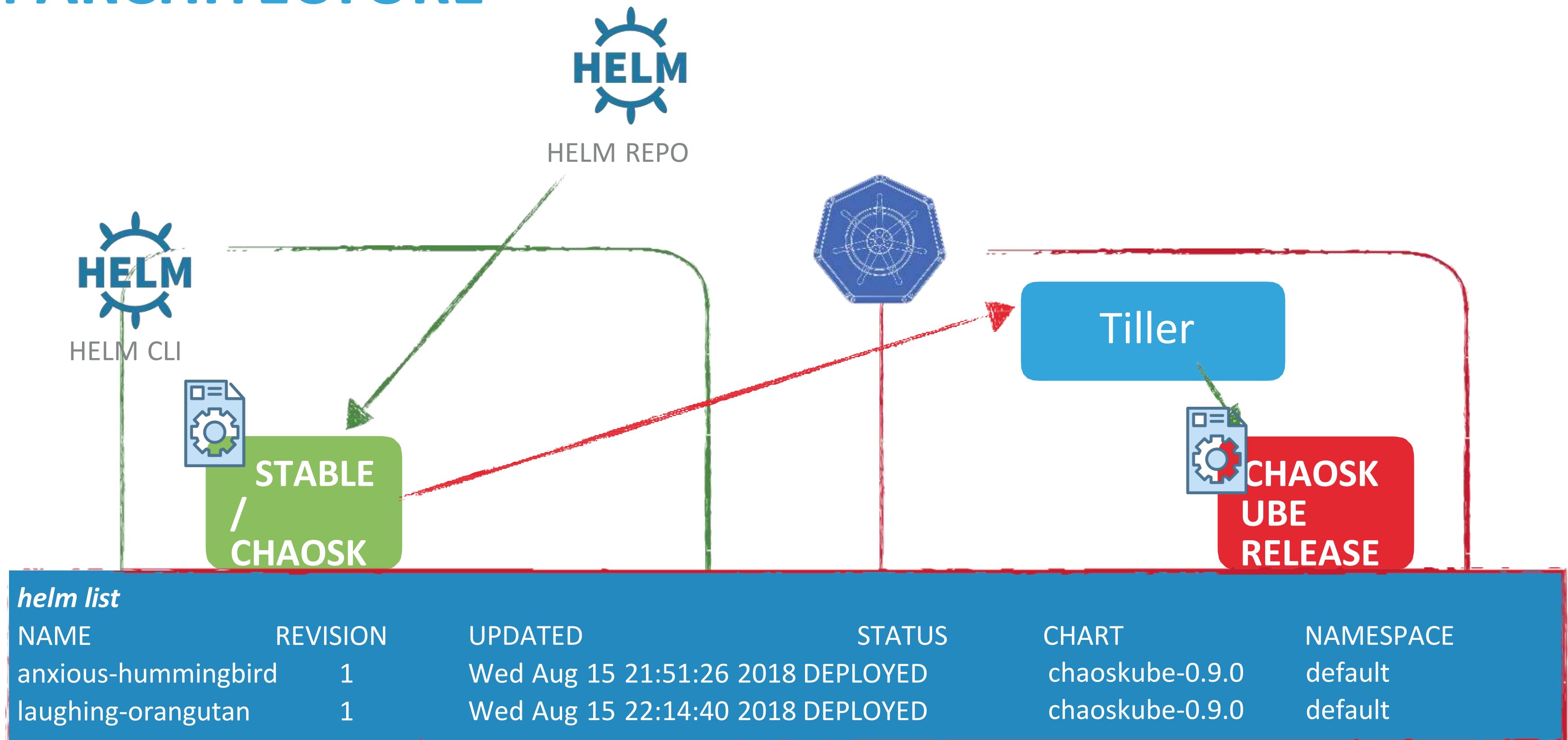
```
kubectl -n {{ .Release.Namespace }} logs -f $POD
```

```
{{ if .Values.dryRun }}
```

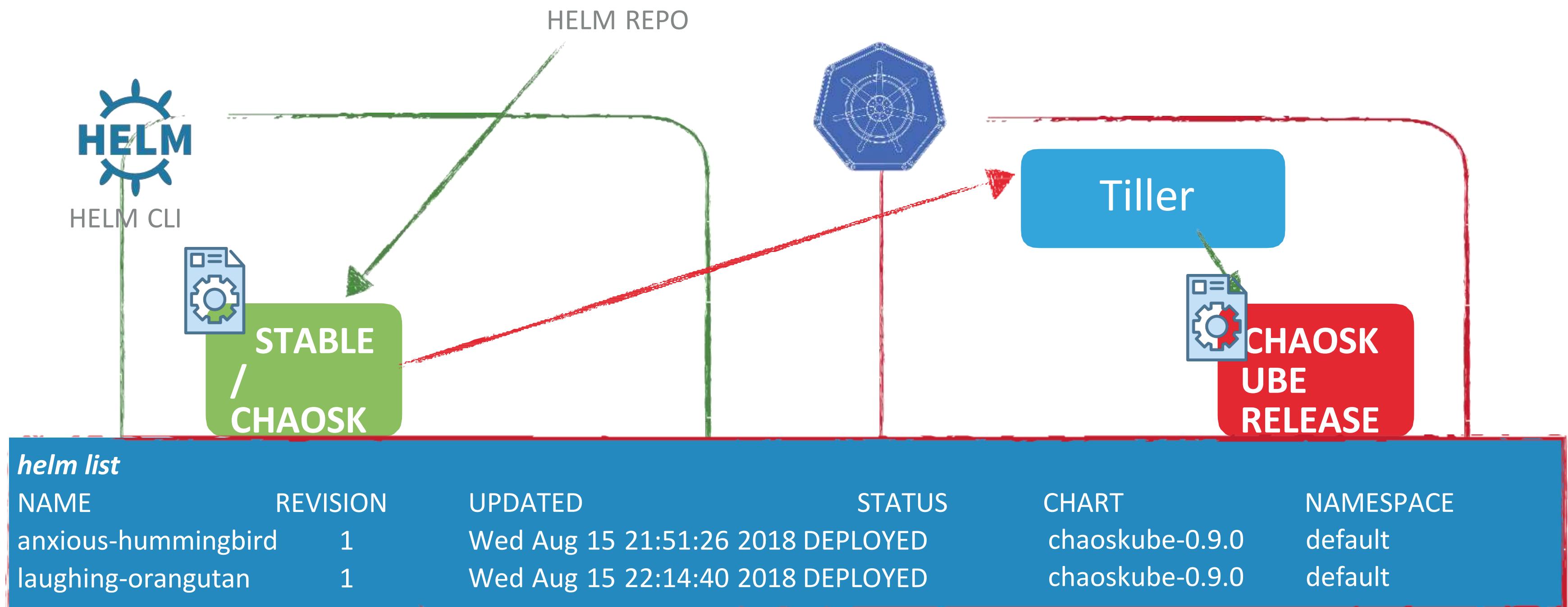
You are running in dry-run mode. No pod is actually terminated.

```
{{ end -}}
```

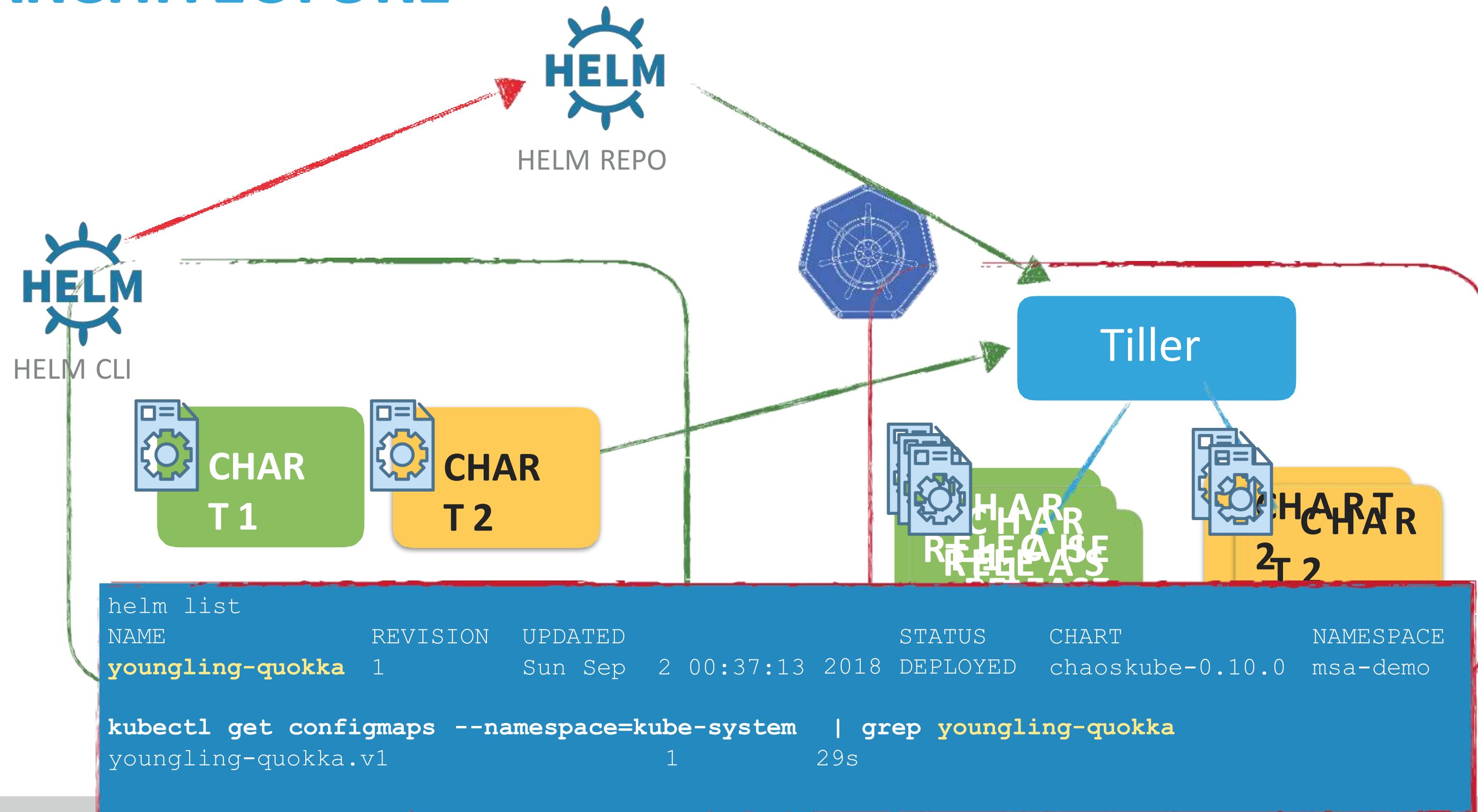
HELM ARCHITECTURE



HELM ARCHITECTURE - MANY REPOSITORY OPTIONS



HELM ARCHITECTURE





“CHARTIFY” (&
REUSE)...

OUR “COMPLICATED” MICRO SERVICE APPLICATION - Dockerized

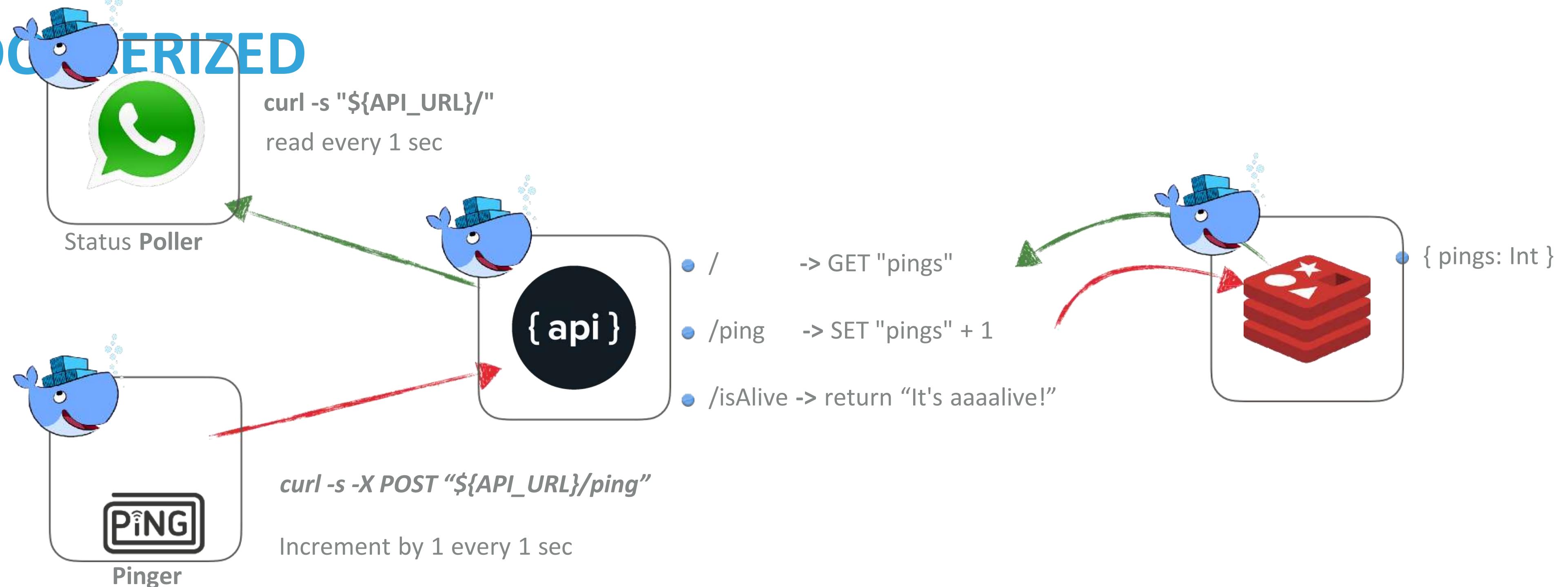


CHART PER COMPONENT

```
redis
|--- Chart.yaml
|--- charts
|--- templates
|   |--- NOTES.txt
|   |--- _helpers.tpl
|   |--- deployment.yaml
|   |--- ingress.yaml
|   `--- service.yaml
`--- values.yaml

api
|--- Chart.yaml
|--- charts
|--- templates
|   |--- NOTES.txt
|   |--- _helpers.tpl
|   |--- deployment.yaml
|   |--- ingress.yaml
|   `--- service.yaml
`--- values.yaml
```

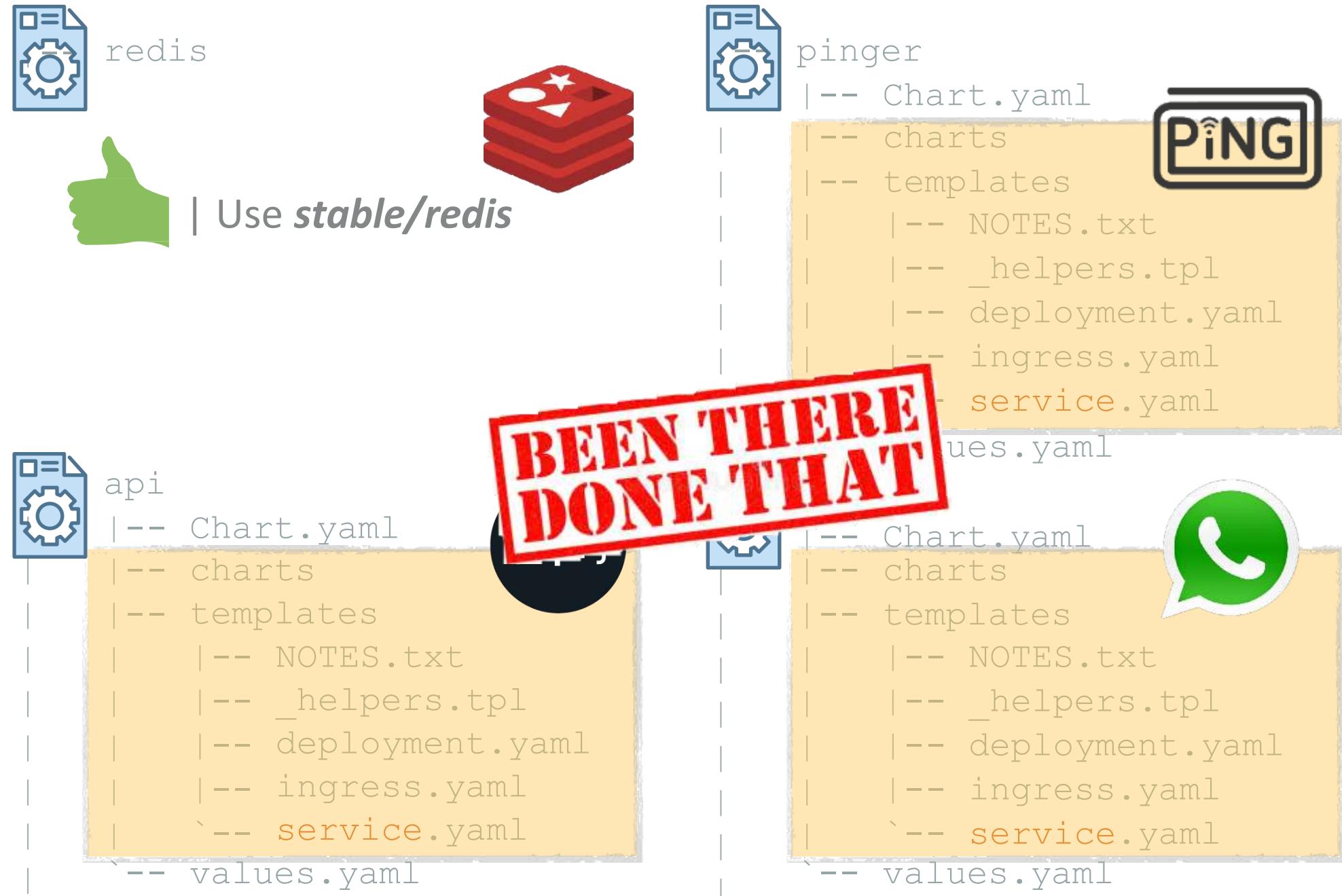


```
pinger
|--- Chart.yaml
|--- charts
|--- templates
|   |--- NOTES.txt
|   |--- _helpers.tpl
|   |--- deployment.yaml
|   |--- ingress.yaml
|   `--- service.yaml
`--- values.yaml

poller
|--- Chart.yaml
|--- charts
|--- templates
|   |--- NOTES.txt
|   |--- _helpers.tpl
|   |--- deployment.yaml
|   |--- ingress.yaml
|   `--- service.yaml
`--- values.yaml
```



CHART PER COMPONENT



“MSA” CHART



```
.
```

- | -- Chart.yaml
- | -- charts
- | -- templates
 - | -- NOTES.txt
 - | -- _helpers.tpl
 - | -- **api-deployment.yaml**
 - | -- **api-service.yaml**
 - | -- deployment.yaml
 - | -- **pinger-deployment.yaml**
 - | -- **poller-deployment.yaml**
 - | -- service.yaml
- values.yaml



VALUES



values.yaml

```
api:
  name: api
  replicaCount: 1
  image:
    repository: shelleg/msa-api
    tag: config
    pullPolicy: Always
    containerPort: 8080
  service:
    enabled: true
    type: ClusterIP
    port: 8080
  resources: {}
  nodeSelector: {}
  tolerations: []
  affinity: {}

pinger:
  name: pinger
  replicaCount: 1
  image:
    repository: shelleg/msa-pinger
    tag: latest
    pullPolicy: Always
  service:
    enabled: false
  resources: {}
  nodeSelector: {}
  tolerations: []
  affinity: {}

poller:
  name: poller
  replicaCount: 1
  image:
    repository: shelleg/msa-poller
    tag: latest
    pullPolicy: Always
  service:
    enabled: false
  resources: {}
  nodeSelector: {}
  tolerations: []
  affinity: {}
```



| This works ;) but ...

TEMPLATES



api-deployment.yaml

```
apiVersion: apps/v1beta2
kind: Deployment
metadata:
  name: {{ .Values.api.name }}
  labels:
    app: {{ .Values.api.name }}
    chart: {{ template "msa.chart" . }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
spec:
  replicas: {{ .Values.api.replicaCount }}
  selector:
    matchLabels:
      app: {{ .Values.api.name }}
      release: {{ .Release.Name }}
  template:
    metadata:
      labels:
        app: {{ .Values.api.name }}
        release: {{ .Release.Name }}
```

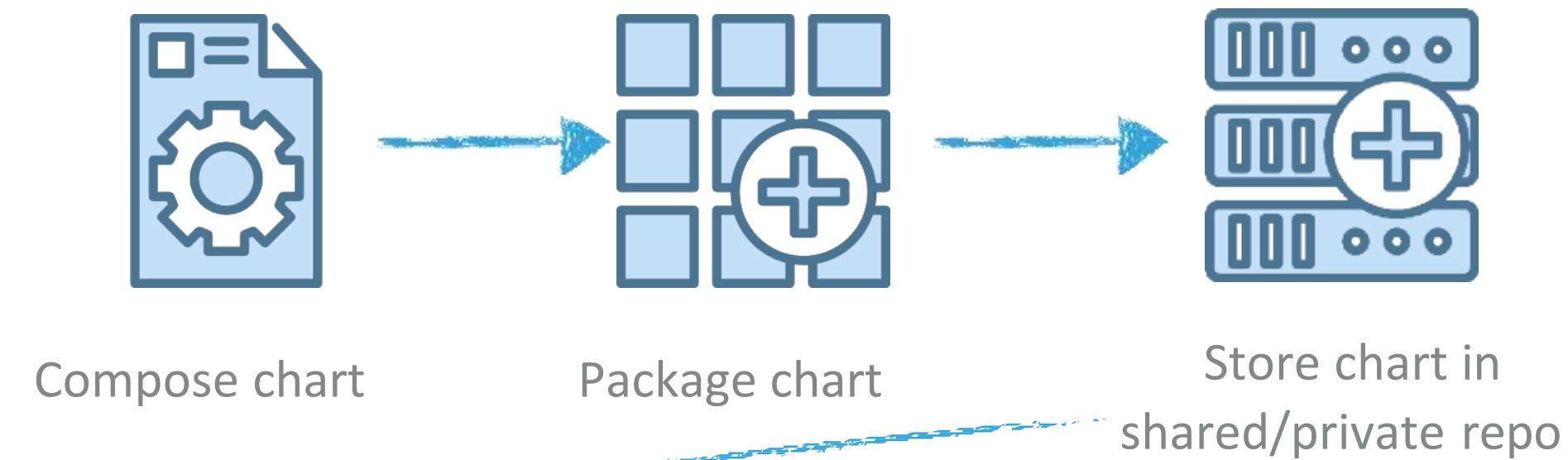


api-service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: {{ .Values.api.name }}
  labels:
    app: {{ .Values.api.name }}
    chart: {{ template "msa.chart" . }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
spec:
  type: {{ .Values.api.service.type }}
  ports:
  - port: {{ .Values.api.service.port }}
    targetPort: http
    protocol: TCP
    name: http
  selector:
    app: {{ .Values.api.name }}
    release: {{ .Release.Name }}
```



INTRODUCTION TO HELM



- | helm install myrepo/msa —set nameOverride: api —set service.enabled=true —name api
- | helm install myrepo/msa —set nameOverride: pinger —name pinger
- | helm install myrepo/msa —set nameOverride: poller —name poller

HELPERS & VALUES



- ▶ nameOverride: [e.g. api | pinger | poller]

HELPERS & VALUES



- ▶ nameOverride: [api | pinger | poller]

helpers.yaml

```
{%- define "chart.name" -%}
{%- default .Chart.Name .Values.nameOverride | trunc 63 | trimSuffix "-" -%}
{%- end -%}
```

HELPERS & VALUES

- ▶ nameOverride: [api | pinger | poller]



```
{- define "chart.name" -}  
{{- default .Chart.Name .Values.nameOverride | trunc 63 | trimSuffix "-" -}}  
{- end -}
```

_helpers.y
aml



| Make the chart reusable

HELPERS & VALUES



- ▶ nameOverride: [api | pinger | poller]
- ▶ service.enabled: true

```
{%- define "chart.name" -%}
{{- default .Chart.Name .Values.nameOverride | trunc 63 | trimSuffix "-" -}}
{{- end -}}
```

_helpers.y
aml

```
{%- if .Values.service.enabled %}
{{- end -}}
```

service.y
aml



| Make the chart reusable

HELPERS & VALUES



- ▶ nameOverride: [api | pinger | poller]
- ▶ service.enabled: true

```
{- define "chart.name" -}  
{- default .Chart.Name .Values.nameOverride | trunc 63 | trimSuffix "-" -}  
{- end -}
```

_helpers.y
aml



| Make the chart reusable

```
{- if .Values.service.enabled -}  
{- end -}
```

service.y
aml



| Make entire RD optional

TEMPLATES



This works ;)
but ...

api-

```
deployment.yaml

apiVersion: apps/v1beta2
kind: Deployment
  name: {{ .Values.api.name }}
  metadata:
    labels:
      app: {{ .Values.api.name }}
      chart: {{ template "msa.chart" . }}
      release: {{ .Release.Name }}
      heritage: {{ .Release.Service }}
  spec:
    replicas: {{ .Values.api.replicaCount }}
    selector:
      matchLabels:
        app: {{ .Values.api.name }}
        release: {{ .Release.Name }}
    template:
      metadata:
        labels:
          app: {{ .Values.api.name }}
          release: {{ .Release.Name }}
```



This is D.R.Y !

deployment.yaml

```
apiVersion: apps/v1beta2
kind: Deployment
metadata:
  name: {{ .Values.name }}
  labels:
    app: {{ .Values.name }}
    chart: {{ template "msa.chart" . }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
spec:
  replicas: {{ .Values.replicaCount }}
  selector:
    matchLabels:
      app: {{ .Values.name }}
      release: {{ .Release.Name }}
  template:
    metadata:
      labels:
        app: {{ .Values.name }}
        release: {{ .Release.Name }}
```

TEMPLATES

api-

service.yaml



This works ;)
but ...

```
apiVersion: v1
kind: Service
  name: {{ .Values.api.name }}
metadata:
  labels:
    app: {{ .Values.api.name }}
    chart: {{ template "msa.chart" . }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
spec:
  type: {{ .Values.api.service.type }}
  ports:
    - port: {{ .Values.api.service.port }}
      targetPort: http
      protocol: TCP
      name: http
  selector:
    app: {{ .Values.api.name }}
    release: {{ .Release.Name }}
```



This is D.R.Y !

service.yaml

```
{-{ if .Values.service.enabled -}}

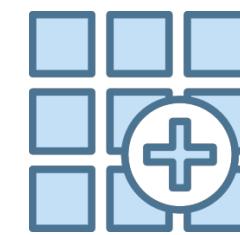
apiVersion: v1
kind: Service
metadata:
  name: {{ .Values.api.name }}
  labels:
    app: {{ .Values.api.name }}
    chart: {{ template "msa.chart" . }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
spec:
  type: {{ .Values.api.service.type }}
  ports:
    - port: {{ .Values.api.service.port }}
      targetPort: http
      protocol: TCP
      name: http
...

{%- end -%}
```

“MSA” CHART



```
.  
| -- Chart.yaml  
| -- charts  
| -- templates  
|   | -- NOTES.txt  
|   | -- _helpers.tpl  
|   | -- deployment.yaml  
|   | -- service.yaml  
`-- values.yaml
```

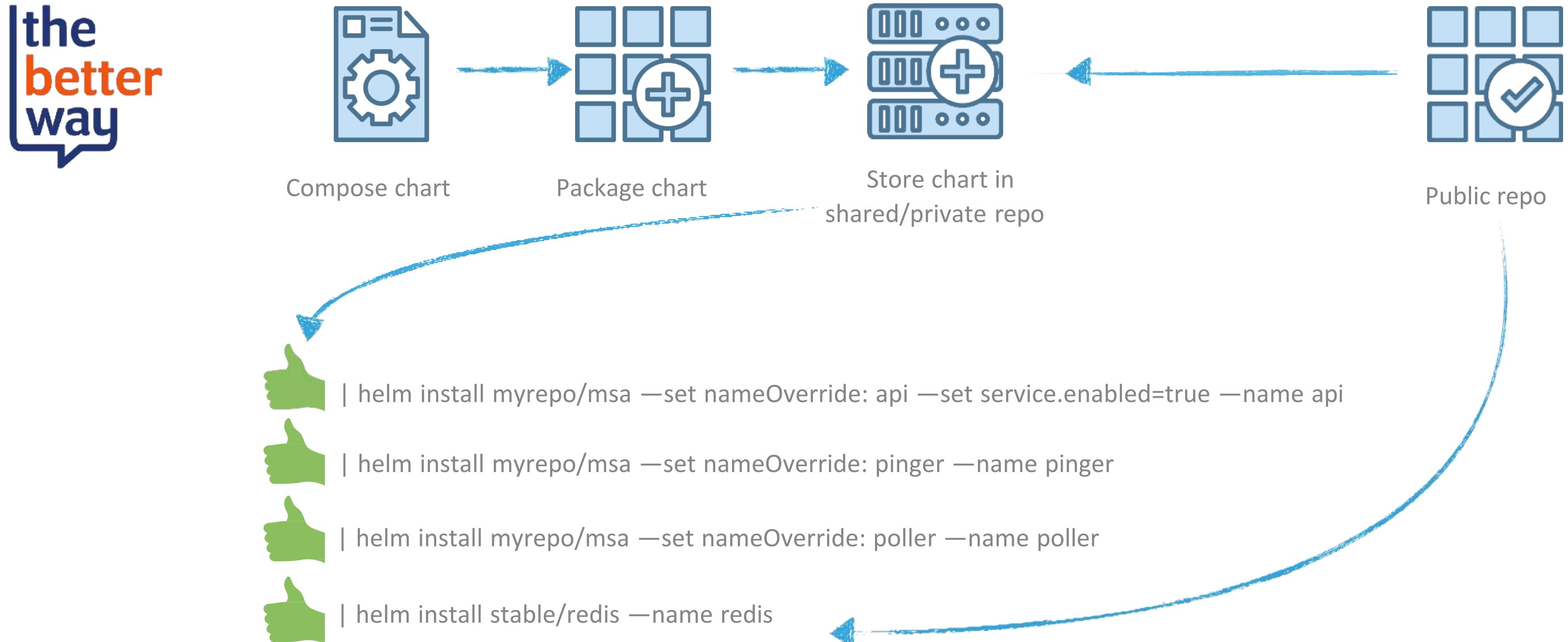


my-chart-repo/msa-0.1.0.gz

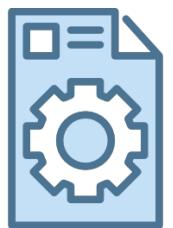
```
apiVersion: v1  
appVersion: "1.0"  
description: msg chart for Kubernetes  
name: api  
version: 0.1.0
```



INTRODUCTION TO HELM



MANAGING DEPENDENCIES - A.K.A UMBRELLA CHART



```
.  
| -- Chart.yaml  
| -- charts  
| -- templates  
|   | -- NOTES.txt  
|   | -- _helpers.tpl  
|   | -- dependencies.yaml  
`-- values.yaml
```



```
dependencies:  
  - name: msa  
    repository: file://../msa  
    version: 0.1.0  
  - name: redis  
    repository: stable/redis  
    version: x.x.x
```





HANDS ON



FullStack Developers Israel



HOW TO USE THIS LAB ?!



Informational -> you don't have / need to do it ;)



Get your hands dirty -> follow the instructions



Tikal Knowledge



MINIKUBE

- ▶ Minikube supports Kubernetes features such as:
 - ▶ DNS
 - ▶ NodePorts
 - ▶ ConfigMaps and Secrets
 - ▶ Dashboards
 - ▶ Container Runtime: Docker, rkt and CRI-O
 - ▶ Enabling CNI (Container Network Interface)
 - ▶ Ingress



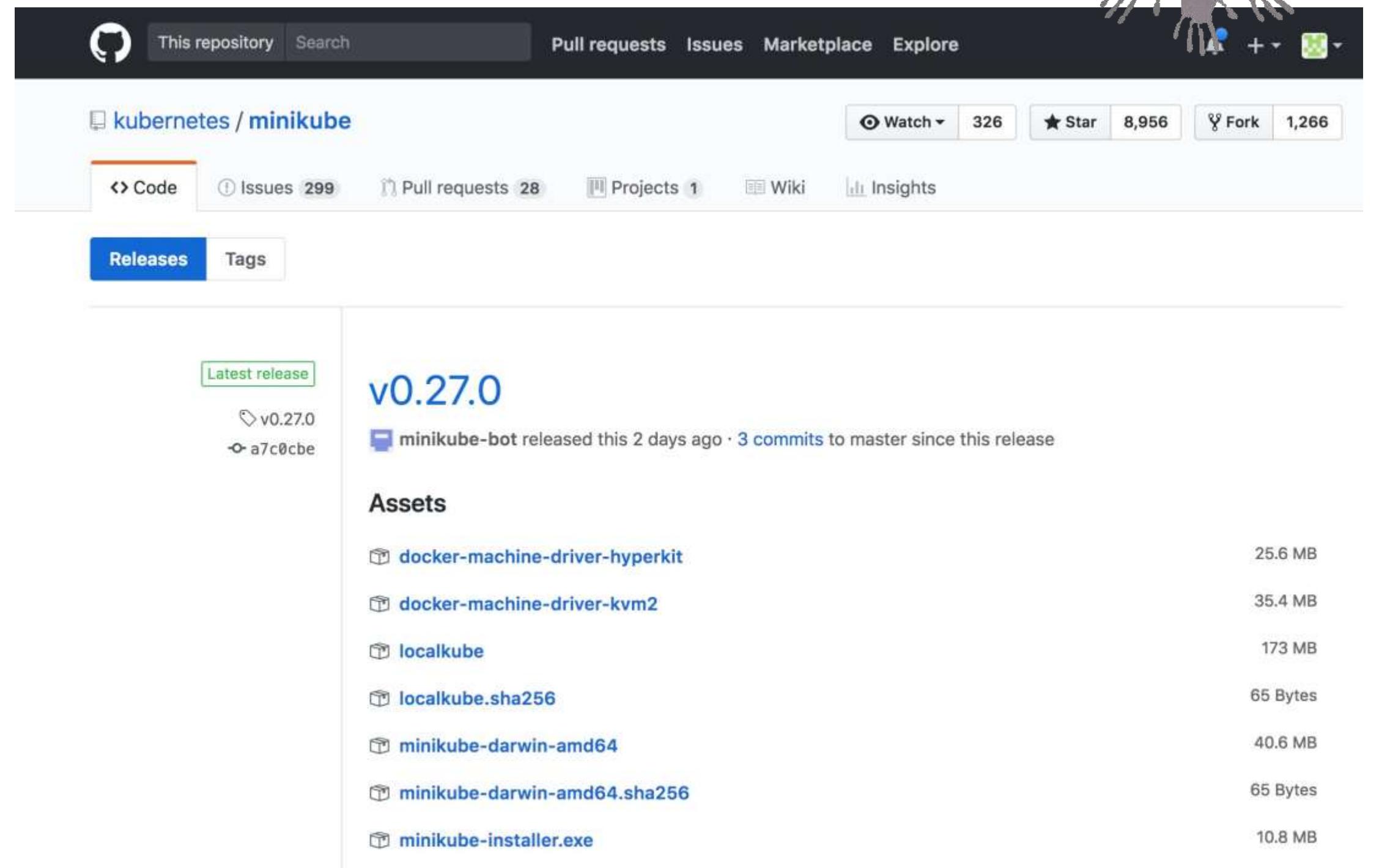
WHY

- ▶ Good for POC, QuickStarts and Demo's
- ▶ Gives a taste of Kubernetes and how to work with it, without going through a complex setup



INSTALLING MINIKUBE

- ▶ Install *kubectl* [link](#)
- ▶ Install *minikube* [link](#)

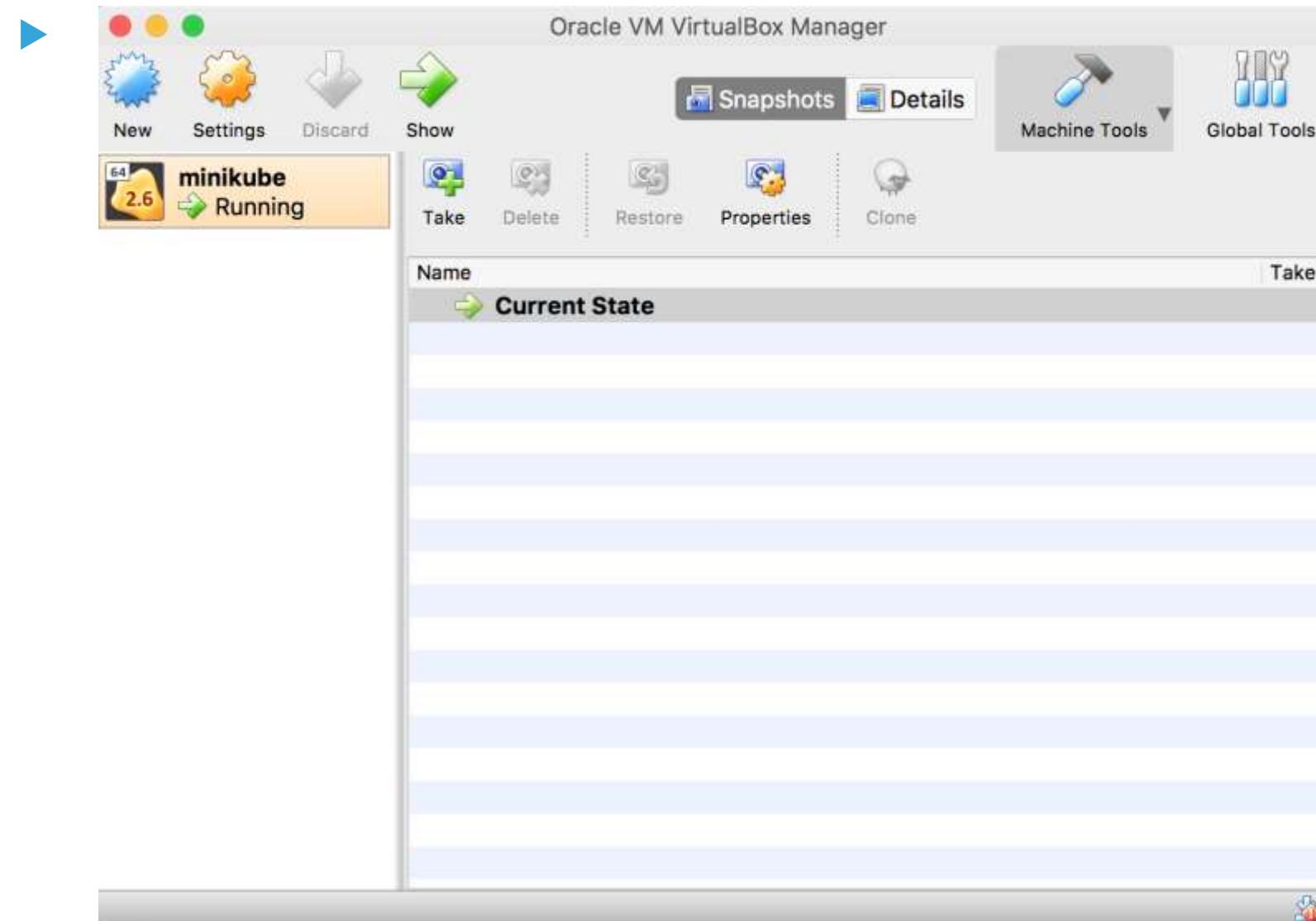


The image shows a screenshot of a GitHub repository page for the 'kubernetes/minikube' repository. The page has a dark header with the GitHub logo, navigation links for 'This repository', 'Search', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. On the right, there are buttons for 'Watch' (326), 'Star' (8,956), 'Fork' (1,266), and a clone icon. Below the header, the repository name 'kubernetes / minikube' is displayed, along with tabs for 'Code', 'Issues 299', 'Pull requests 28', 'Projects 1', 'Wiki', and 'Insights'. A 'Releases' tab is selected, showing the 'Latest release' 'v0.27.0'. The release was made by 'minikube-bot' 2 days ago with 3 commits to master. The assets section lists several files: 'docker-machine-driver-hyperkit' (25.6 MB), 'docker-machine-driver-kvm2' (35.4 MB), 'localkube' (173 MB), 'localkube.sha256' (65 Bytes), 'minikube-darwin-amd64' (40.6 MB), 'minikube-darwin-amd64.sha256' (65 Bytes), and 'minikube-installer.exe' (10.8 MB).



MINIKUBE START

► *minikube start*



20:31 \$ minikube start
Starting local Kubernetes v1.9.0 cluster...
Starting VM...
Getting VM IP address...
Moving files into cluster...
Setting up certs...
Connecting to cluster...
Setting up kubeconfig...
Starting cluster components...
Kubectl is now configured to use the cluster.
Loading cached images from config file.



MACOSX VIA HOMEBREW

```
brew update  
brew install kubernetes-cli kubernetes-helm  
brew cask install minikube
```



The screenshot shows the GitHub repository for `kubernetes/minikube`. The repository has 8,956 stars and 1,266 forks. The `Releases` tab is selected, showing the `v0.27.0` release. This release was made 2 days ago by `minikube-bot` with 3 commits to master. The release page lists several assets:

Asset	Size
<code>docker-machine-driver-hyperkit</code>	25.6 MB
<code>docker-machine-driver-kvm2</code>	35.4 MB
<code>localkube</code>	173 MB
<code>localkube.sha256</code>	65 Bytes
<code>minikube-darwin-amd64</code>	40.6 MB
<code>minikube-darwin-amd64.sha256</code>	65 Bytes
<code>minikube-installer.exe</code>	10.8 MB



UBUNTU/DEBIAN



```
sudo apt-get update && sudo apt-get install -y apt-transport-https  
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -  
echo "deb http://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee -a /etc/apt/  
sources.list.d/kubernetes.list  
sudo apt-get update  
sudo apt-get install -y kubectl  
  
curl https://raw.githubusercontent.com/kubernetes/helm/master/scripts/get | bash  
curl -Lo minikube https://storage.googleapis.com/minikube/releases/v0.30.0/minikube-linux-amd64 && chmod +x minikube && sudo cp minikube /usr/local/bin/ && rm minikube
```



CENTOS/RHEL/FEDORA

```
cat <<EOF > /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86\_64
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
EOF
```

```
yum install -y kubectl
```

```
curl https://raw.githubusercontent.com/kubernetes/helm/master/scripts/get | bash
curl -Lo minikube https://storage.googleapis.com/minikube/releases/v0.30.0/minikube-linux-amd64 && chmod +x minikube && sudo cp minikube /usr/local/bin/ && rm minikube
```



WINDOWS

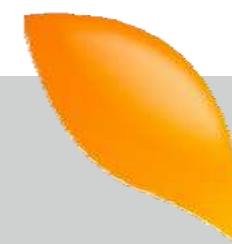
Use the [Chocolaty](#) package manager:

```
choco install kubernetes-cli kubernetes-helm
```

Download Minikube from: <https://github.com/kubernetes/minikube/releases>

Download the minikube-windows-amd64.exe file,
rename it to minikube.exe and add it to your path.





INSTALL HELM ON KUBERNETES

```
helm init --service-account helm --dry-run --debug > helm-init.yml
kubectl create -f helm-rbac-cluster-wide.yml kubectl
create -f helm-init.yml
```



```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: helm
  namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1beta1 kind:
ClusterRoleBinding
metadata:
  name: helm
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
  - kind: ServiceAccount
    name: helm
    namespace: kube-system
```

CONFIGURE HELM REPOSITORIES



helm repo add stable <https://kubernetes-charts.storage.googleapis.com>

helm / charts

Watch ▾ 243 Star 5,186 Fork 4,607

Code Issues 233 Pull requests 344 Insights

Branch: master charts / stable / Create new file Upload files Find file History

okgolove and k8s-ci-robot [stable/drone], changed repository name in NOTES (#8628) ... Latest commit fc4a83c 2 hours ago

..

acs-engine-autoscaler	Enrich deploy. template for acs-engine-autoscaler (#5662)	5 months ago
aerospike	[stable/aerospike] Add cmd and args options to Aerospike config (#3856)	8 months ago
anchore-engine	add brady todhunter as review/maintainer to anchore-engine (#8514)	5 days ago
apm-server	[stable/apm-server] Elastic APM Server (#6058)	4 months ago
ark	[stable/ark] Quote configuration parameter backupStorageProvider.conf...	8 days ago
artifactory-ha	Deprecate JFrog charts (moved to https://github.com/jfrog/charts) (#7627)	a month ago
artifactory	Deprecate JFrog charts (moved to https://github.com/jfrog/charts) (#7627)	a month ago
auditbeat	upgrade auditbeat (#8277)	12 days ago
aws-cluster-autoscaler	Typo fix in aws-cluster-autoscaler/README.md (#4297)	7 months ago

INSTALL HELM-S3 PLUGIN

helm plugin install <https://github.com/hypnoglow/helm-s3.git>



- > If you are planning on pushing to your s3 based helm repo
- > get your own repo and setup the required permissions.
- > In our case you will be using it for read / download purposes which is already public / open source.

ADD THE MSA-CHARTS S3 HELM REPOSITORY



Type a prefix and press Enter to search. Press ESC to clear.

Upload Create folder Actions EU (Ireland) ▾

Name	Last modified	Size	Storage class
index.yaml	Oct 18, 2018 11:00:19 AM GMT+0300	4.0 KB	Standard
msa-api-0.1.0.tgz	Oct 14, 2018 10:38:10 AM GMT+0300	2.6 KB	Standard
msa-api-0.1.10.tgz	Oct 17, 2018 11:52:21 AM GMT+0300	2.3 KB	Standard
msa-api-0.1.13.tgz	Oct 17, 2018 3:43:16 PM GMT+0300	2.3 KB	Standard
msa-api-0.1.14.tgz	Oct 17, 2018 3:49:12 PM	2.3 KB	Standard

Viewing 1 to 15

s3 plugin updates
the index.yaml on
each version
upload

DEPLOY YOUR UMBRELLA ...

```
kubectl create namespace msa-umbrella
helm dep build ./helm/msa-umbrella
helm install --name v1 --namespace msa-umbrella ./helm/msa-umbrella
```



DEPLOY YOUR UMBRELLA ...

```
kubectl create namespace msa-umbrella
helm dep build ./helm/msa-umbrella
helm install --name v1 --namespace msa-umbrella ./helm/msa-umbrella
```



CHECK YOUR UMBRELLA

...

helm list

kubectl -n msa-umbrella get po,svc,deploy

view helm releases which are installed

view msa-umbrella pods, services & deployments



RECAP

- ▶ Helm is awesome -> (completes Kubernetes)
 - ▶ Application centric solution
 - ▶ Flexible templating language
 - ▶ Well integrated into workflows such as GitKube, Scafold and others

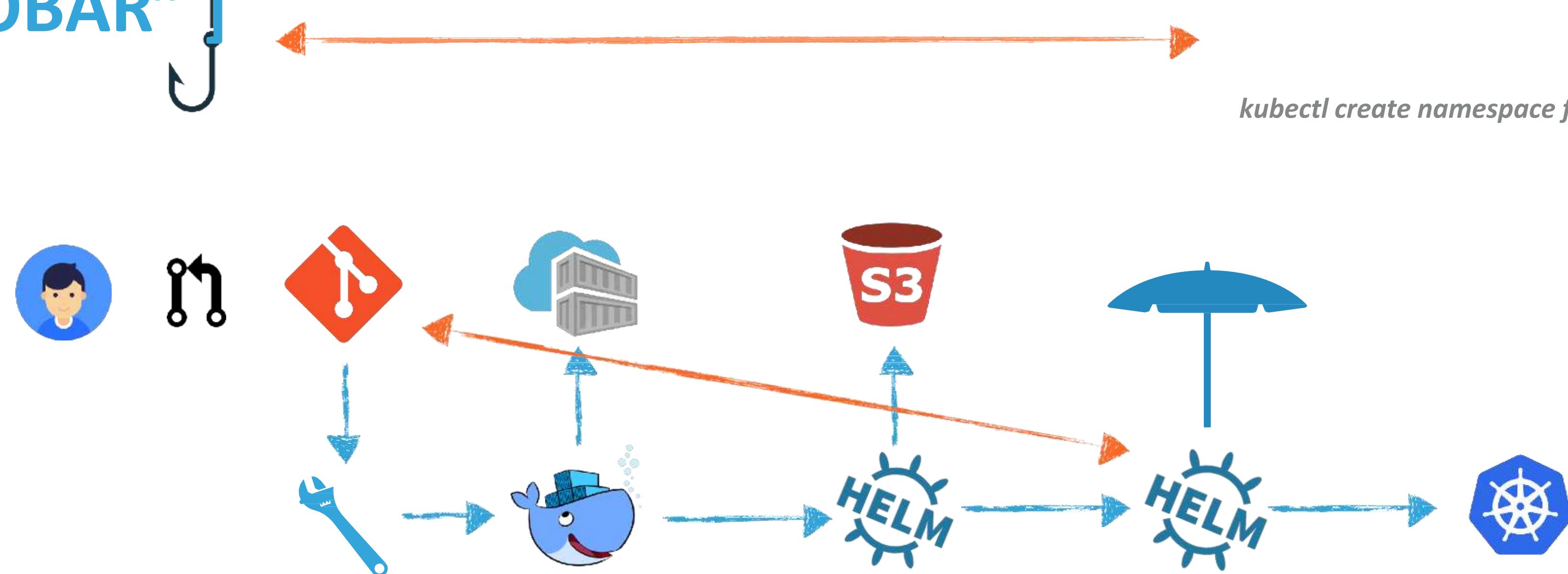
USE [DON'T OVER USE] YOUR IMAGINATION

- ▶ Custom Resource Definitions { e.g istio }  | {{ if .Values.istio.enabled }}
- ▶ Common monitoring RD's { e.g prometheus }  | {{ if .Values.monitoring.enabled }}
- ▶ Common logging RD's { e.g fluentd }  | {{ if .Values.logging.enabled }}
- ▶ Ingresses (yes / no)  | {{ if .Values.ingress.enabled }}
- ▶ Services (type: NodePort, ClusterIp, LoadBalancer)  | {{ if .Values.service.type }}
- ▶ Provider specific configs { on-prem vs. cloud }



So Whats Next ? ...

CONTINUES INTEGRATION PROCESS [JENKINS || GITLAB || “FOOBAR”]



INTRODUCTION TO HELM

JENKINS

Pipeline

Jenkins → msa-api → Pipeline msa-api

ENABLE AUTO REFRESH

[Status](#) [Changes](#) [Build with Parameters](#) [Delete Pipeline](#) [Configure](#) [Full Stage View](#) [Open Blue Ocean](#) [Rename](#) [Embeddable Build Status](#) [Build Review](#) [Pipeline Syntax](#)

[Recent Changes](#)

[add description](#) [DISABLE PROJECT](#)

Stage View

Setup	Checkout	Build	Prepare Test Env	System Tests	Deploy	Deployment env. setup	HELM package	HELM dependency update	HELM deploy
22ms	8s	12s	16ms	22ms	67ms	49s	8s	5s	2min 55s
Average stage times: (Average full run time: ~1min 12s)									
#15 Oct 18 10:59 2 commits	21ms	7s	26s	17ms	79ms	97ms	18s	10s	5s
#14 Oct 17 15:48 2 commits	17ms	7s	4s	14ms	16ms	57ms	13s	8s	911ms

W TECH

PIPELINE LIBRARY - NOW SUPPORTS HELM LIFECYCLE



Skip to... Pull requests Issues Marketplace Explore

tikalk / tikal-pipelib Unwatch ▾ 15 Star 1 Fork 0

Code Issues 0 Pull requests 0 Projects 1 Wiki Insights Settings

Tikal Jenkins Pipeline Library Edit

Manage topics

303 commits 3 branches 0 releases 4 contributors

Branch: master ▾ New pull request Create new file Upload files Find file Clone or download ▾

Tom Gur fixed helm upgrade command Latest commit 712997e 6 days ago

src fixed helm upgrade command 6 days ago

vars AWS configure 16 days ago

.gitignore Initiate tikal-pipelib a year ago

README.md rename and replace tpl_ with tpl prefix a year ago

CREDITS

- ▶ Mark Kirshner & Anatoly Rabkin
- ▶ Helm documentation -> <https://docs.helm.sh/>
- ▶ ChaosKube helm chart -> <https://github.com/helm/charts/tree/master/stable/chaoskube>
- ▶ Icons -> <https://github.com/octo-technology/kubernetes-icons>
- ▶ Kubernetes docs -> <https://kubernetes.io/docs/home/?path=users>



We Love
Thank you for tuning in
Tech



FullStack Developers Israel

