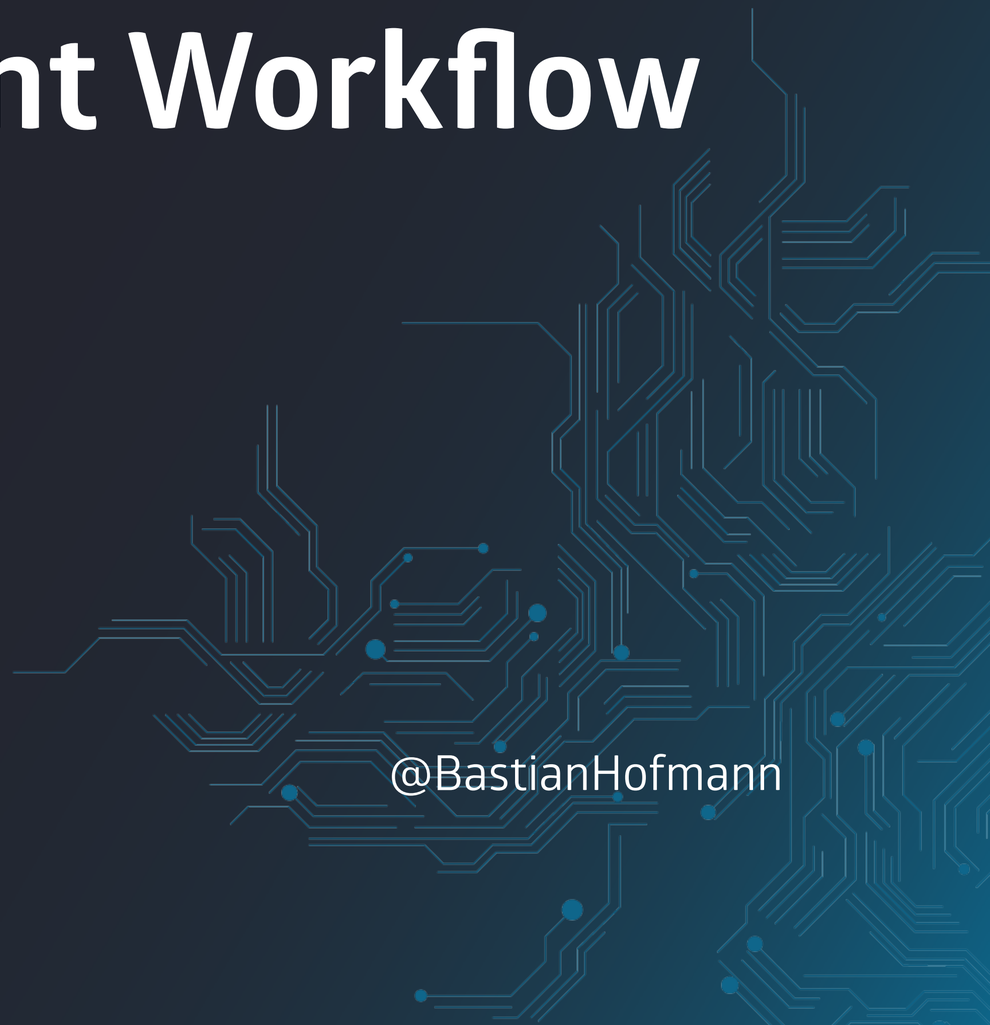
SysEleven
engage · build · run

Creating a fast Kubernetes Development Workflow

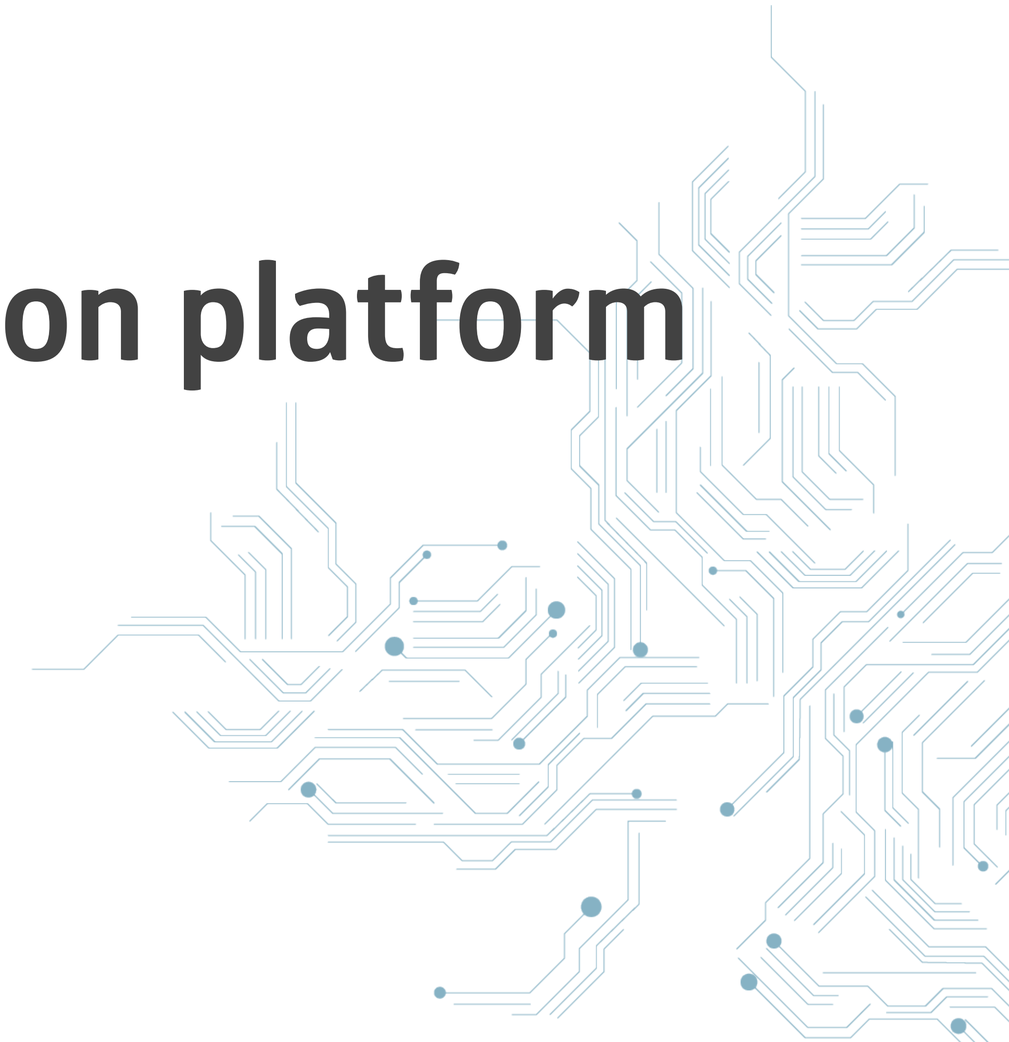


Bastian Hofmann

@BastianHofmann

kubernetes

Container orchestration platform



Deploy, run and scale your services in isolated containers

No vendor lock in

Runs on

Your laptop

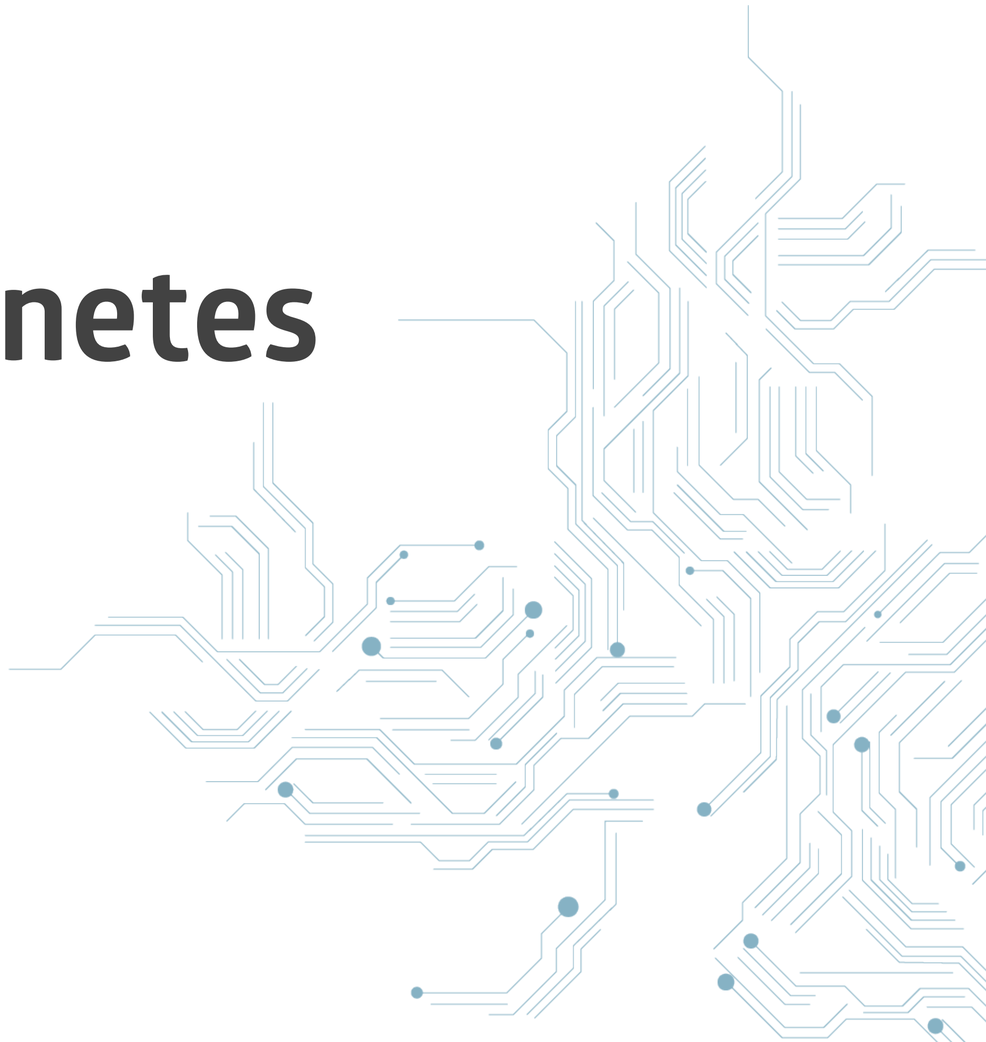

Kubernetes
General
File Sharing
Disk
Advanced
Proxies
Daemon
Kubernetes
Reset
Enable Kubernetes
Start a Kubernetes single-node cluster when starting Docker Desktop.
Deploy Docker Stacks to Kubernetes by default
Make Kubernetes the default orchestrator for "docker stack" commands
(changes "~/.docker/config.json")
Show system containers (advanced)
Show Kubernetes internal containers when using Docker commands.
Apply
Docker Desktop is running

Bare metal

Cloud Providers

And if you don't want to install and maintain Kubernetes yourself

Managed Kubernetes



CNCF Cloud Native Interactive Landscape

CLOUD NATIVE COMPUTING FOUNDATION

The Cloud Native Trail Map (png, pdf) is CNCF's recommended path through the cloud native landscape. The cloud native landscape (png, pdf) and serverless landscape (png, pdf) are dynamically generated below. Please open a pull request to correct any issues. Greyed logos are not open source. Last Updated: 2019-02-12 06:44:45Z

You are viewing 33 cards with a total of 180 stars, market cap of $4.01T and funding of $1.19B.

Landscape | Card Mode | Serverless

Share 462

Platform - Certified Kubernetes - Hosted (33)

Alibaba Cloud Container Service for Kubernetes MCap: $434B
Alibaba Cloud

Amazon Elastic Container Service for Kubernetes (EKS) MCap: $782B
Amazon Web Services

Azure (ACS) Engine ★174
Microsoft MCap: $808B

Azure Kubernetes Service (AKS) MCap: $808B
Microsoft

Azure Stack (ACS) Engine ★4
Microsoft MCap: $808B

Baidu Cloud Container Engine MCap: $58.6B
Baidu

BoCloud BeyondcentContainer Funding: $15.3M
Bocloud

Catalyst Kubernetes Service ★2
Catalyst Cloud

Cisco Container Platform MCap: $214B
Cisco

DigitalOcean Kubernetes Funding: $305M
DigitalOcean

EasyStack Kubernetes Service (EKS) Funding: $110M
EasyStack

eBaoCloud
eBaoTech Corporation

eKing Cloud Container Platform
Hainan eKing Technology

ELASTX Private Kubernetes
ELASTX

Google Kubernetes Engine (GKE) MCap: $763B
Google

HarmonyCloud Container Platform
Hangzhou Harmony Technology

Hasura Funding: $1.6M
Hasura

Huawei Cloud Container Engine (CCE)
Huawei Technologies

IBM Cloud Kubernetes Service MCap: $122B
IBM

Intellect FABRIC MCap: $22.4B
Intellect Design Arena

Nirmata Managed Kubernetes
Nirmata

Nutanix Karbon MCap: $9.48B
Nutanix

Oracle Container Engine MCap: $184B
Oracle

OVH Managed Kubernetes Service Funding: $737M
OVH.com

Rackspace Kubernetes-as-a-Service Funding: $17.8M
Rackspace

Samsung SDS Kubernetes Service
Samsung SDS

SAP Certified Gardener MCap: $126B
SAP

SysEleven MetaKube
SysEleven

Tencent Kubernetes Engine (TKE) MCap: $421B
Tencent Holdings

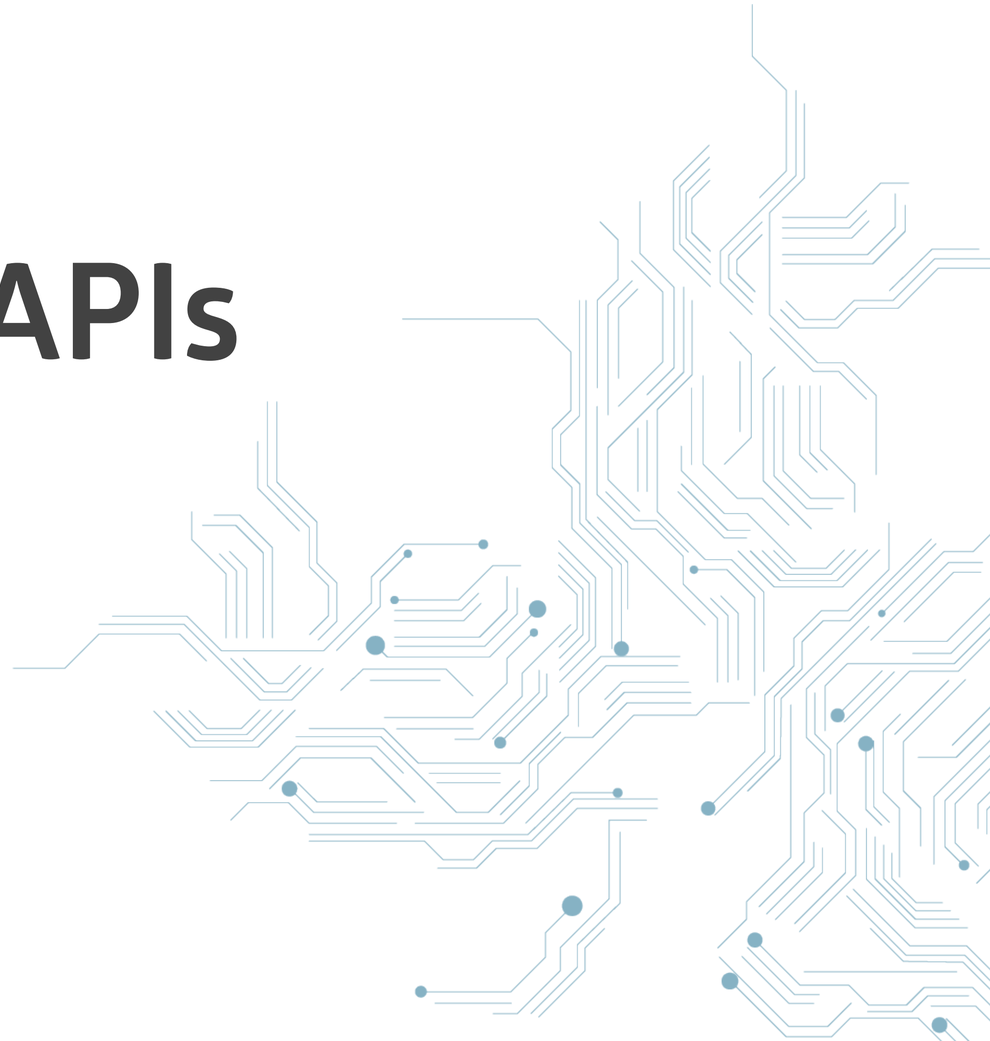
TenxCloud Container Engine (TCE)
TenxCloud

VEXXHOST Kubernetes Container Service 0
VEXXHOST

VMware Cloud PKS MCap: $65.2B
VMware

ZTE TECS
ZTE

Standardized APIs

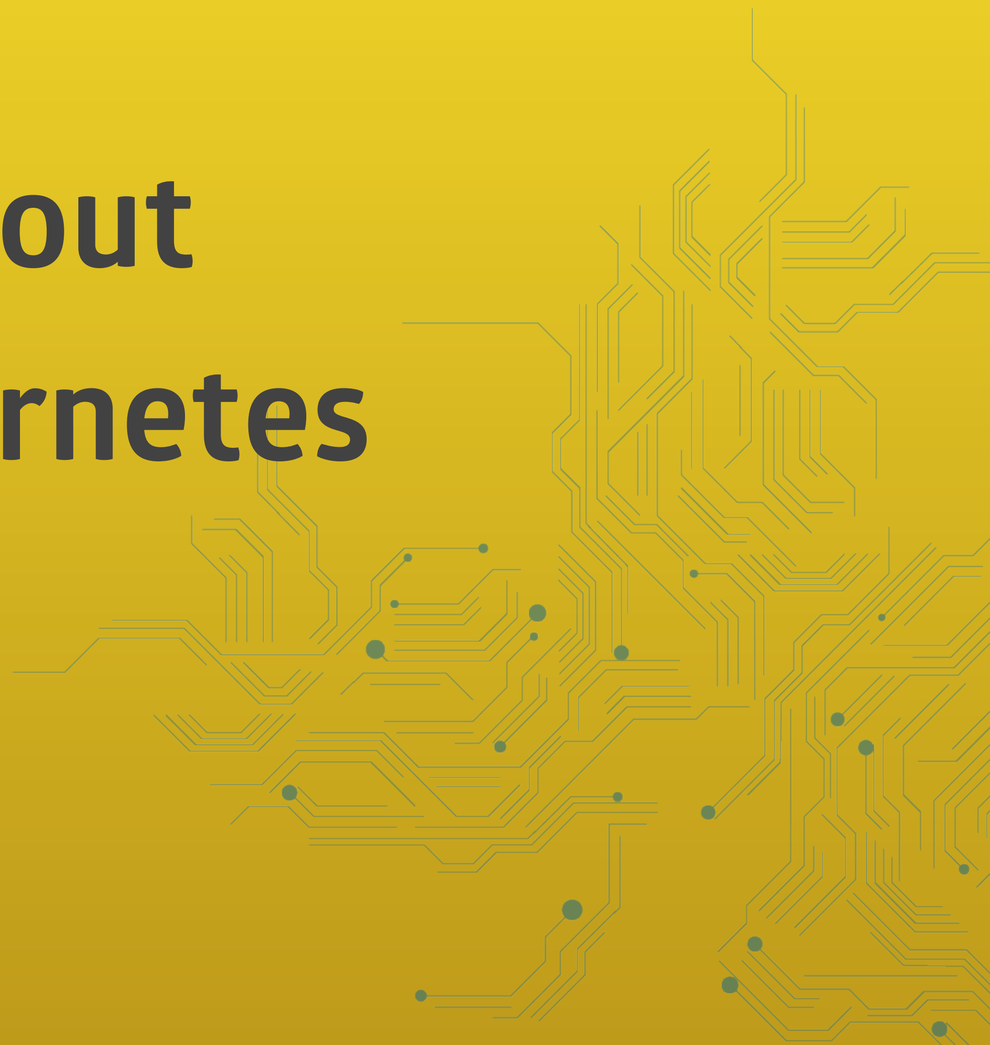


It works the same everywhere*

It works the same everywhere*

*mostly

This talk is about how to use Kubernetes



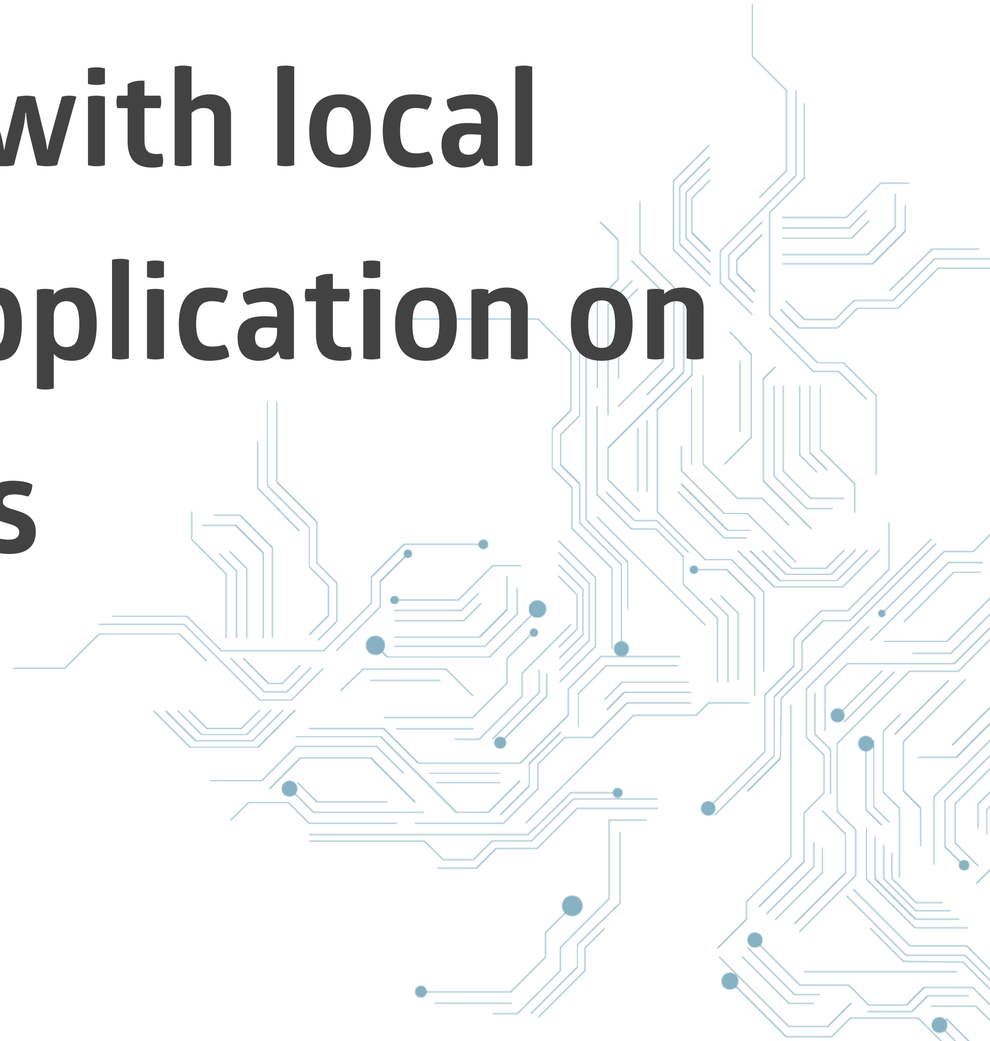
Not only for production workloads

But in your development workflows

Agenda

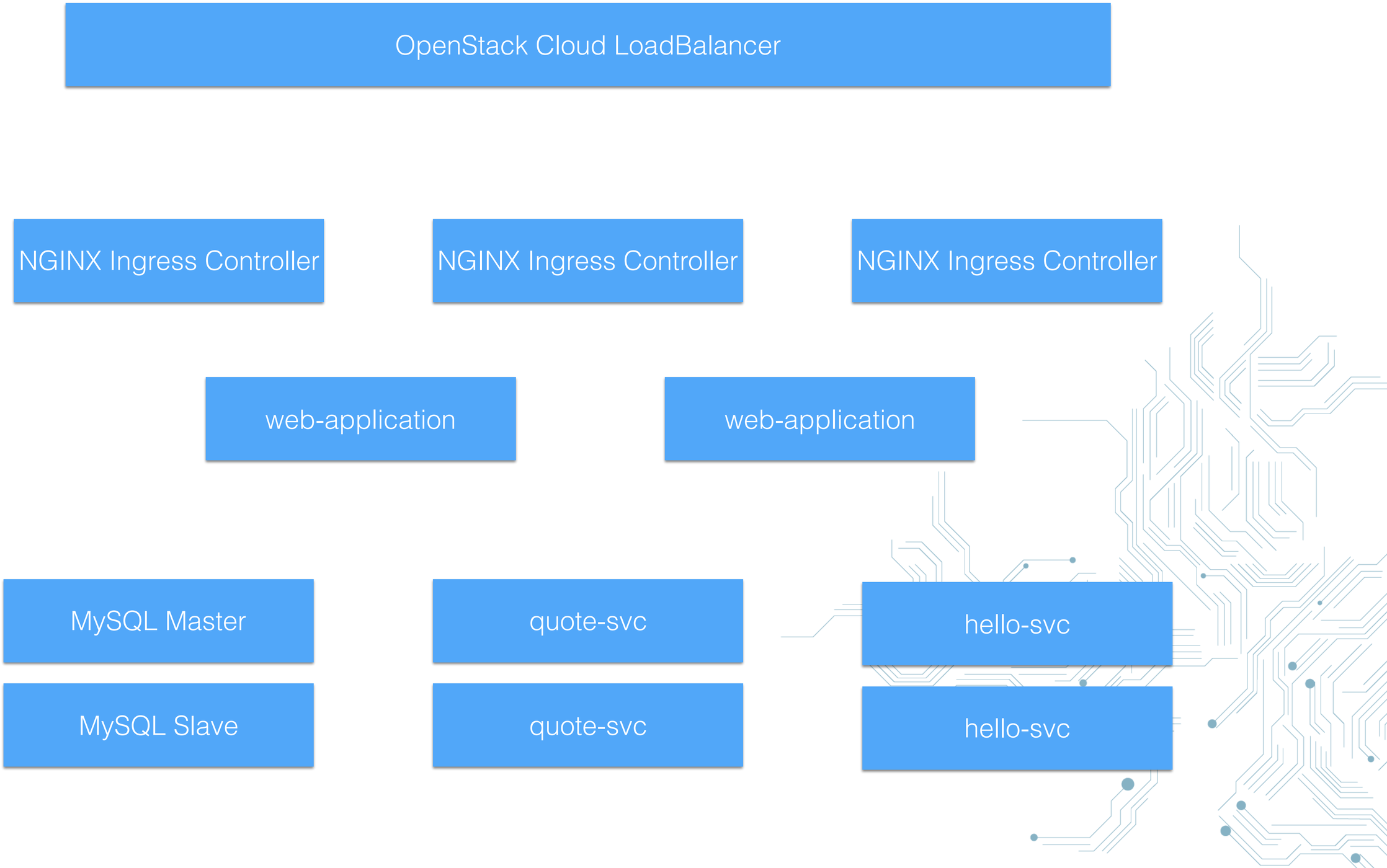
Deployment of a micro-service application

Some tools to help with local development of this application on Kubernetes

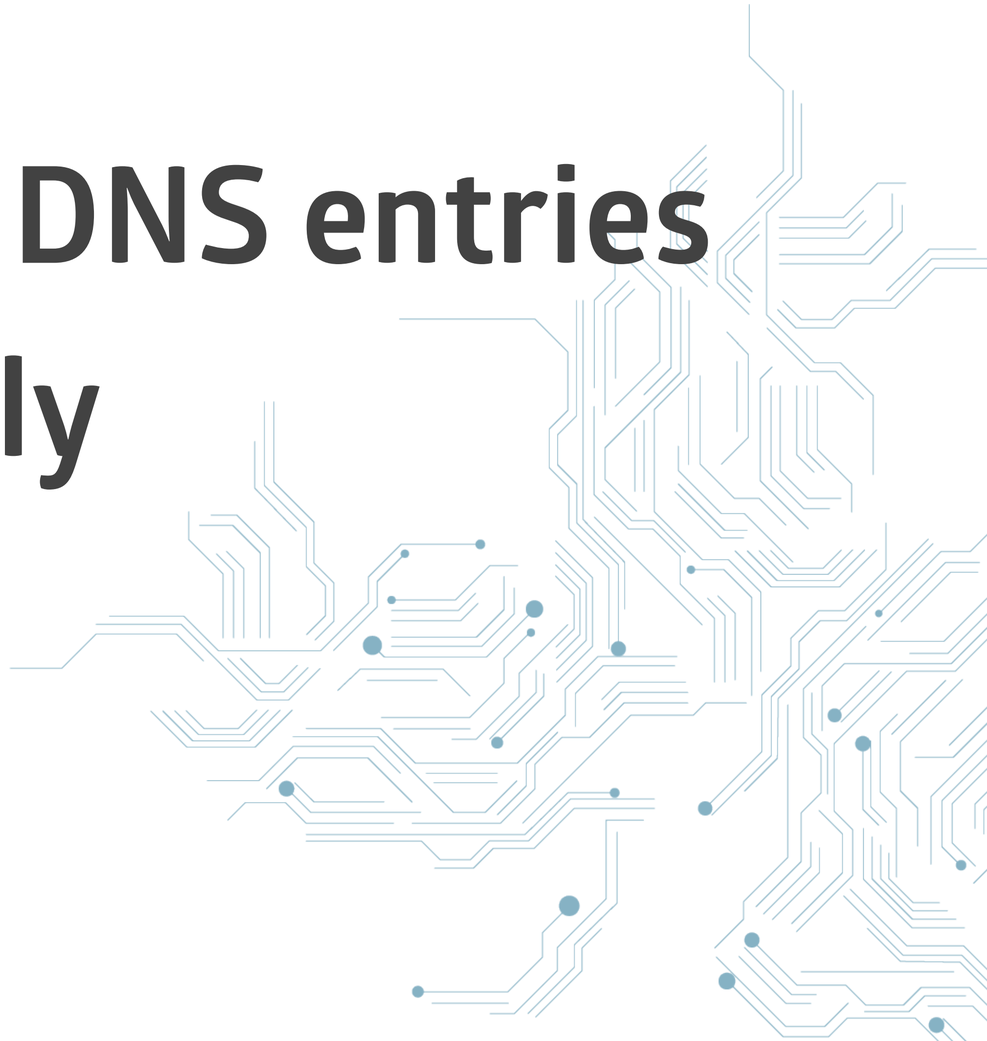


Let's have a look at the sample application


METAKUBE

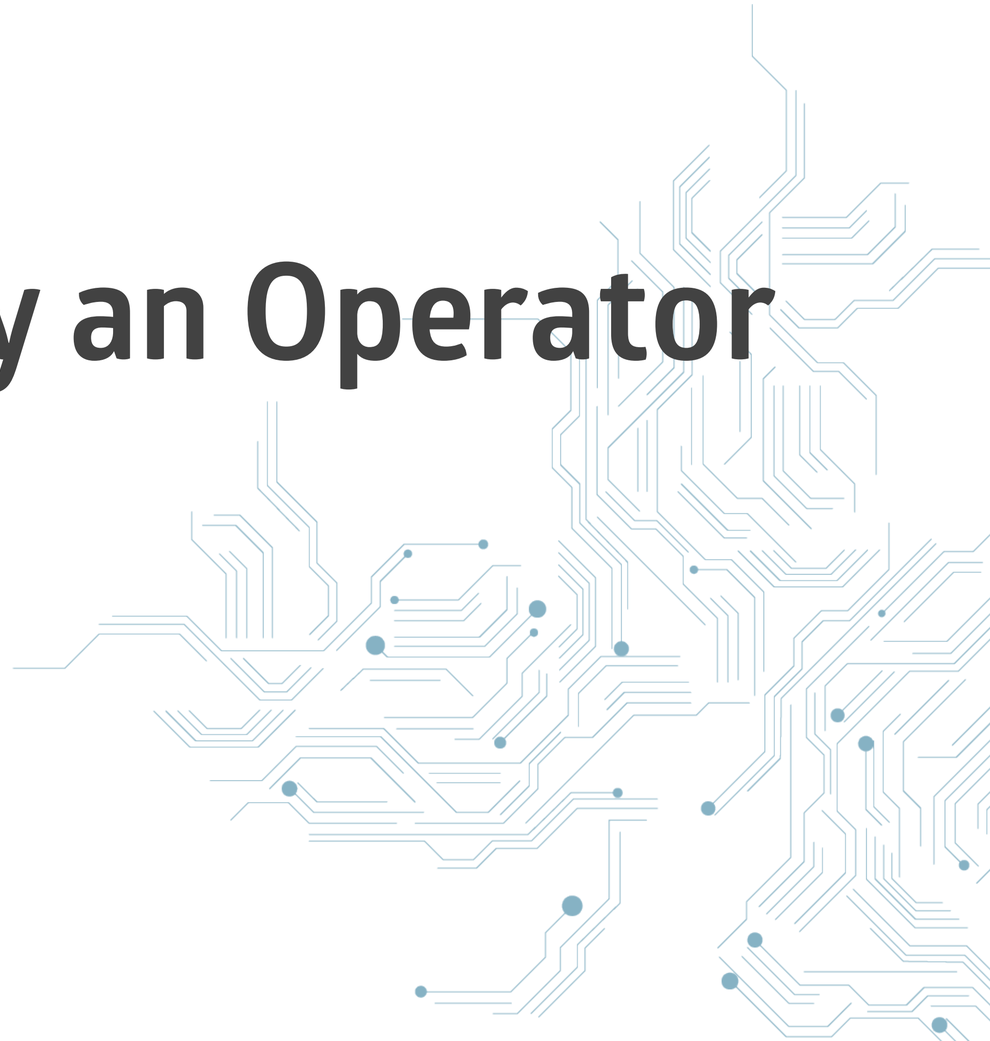

OpenStack Cloud LoadBalancer
NGINX Ingress Controller
NGINX Ingress Controller
NGINX Ingress Controller
web-application
web-application
MySQL Master
MySQL Slave
quote-svc
quote-svc
hello-svc
hello-svc

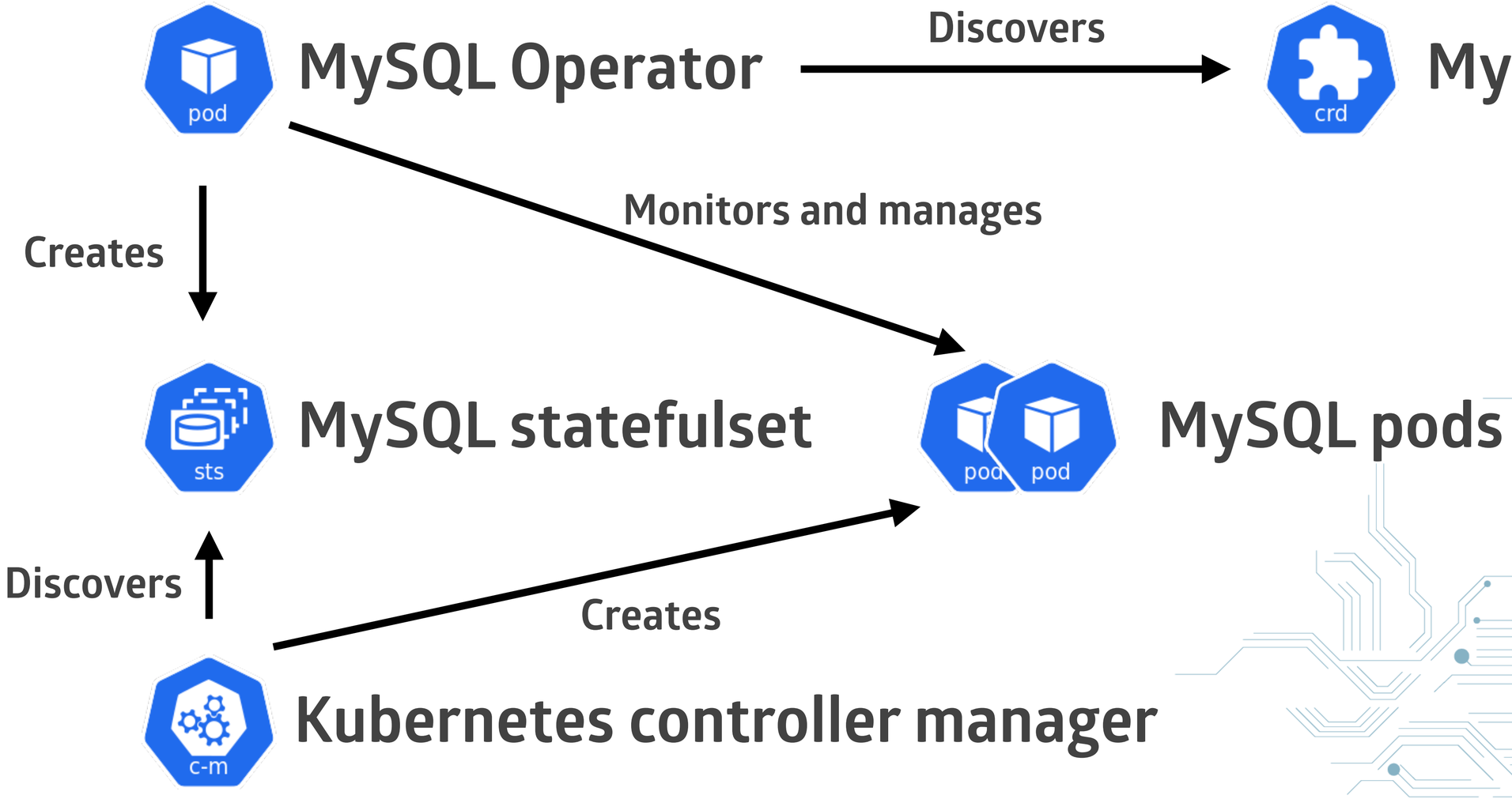
external-dns to create DNS entries automatically



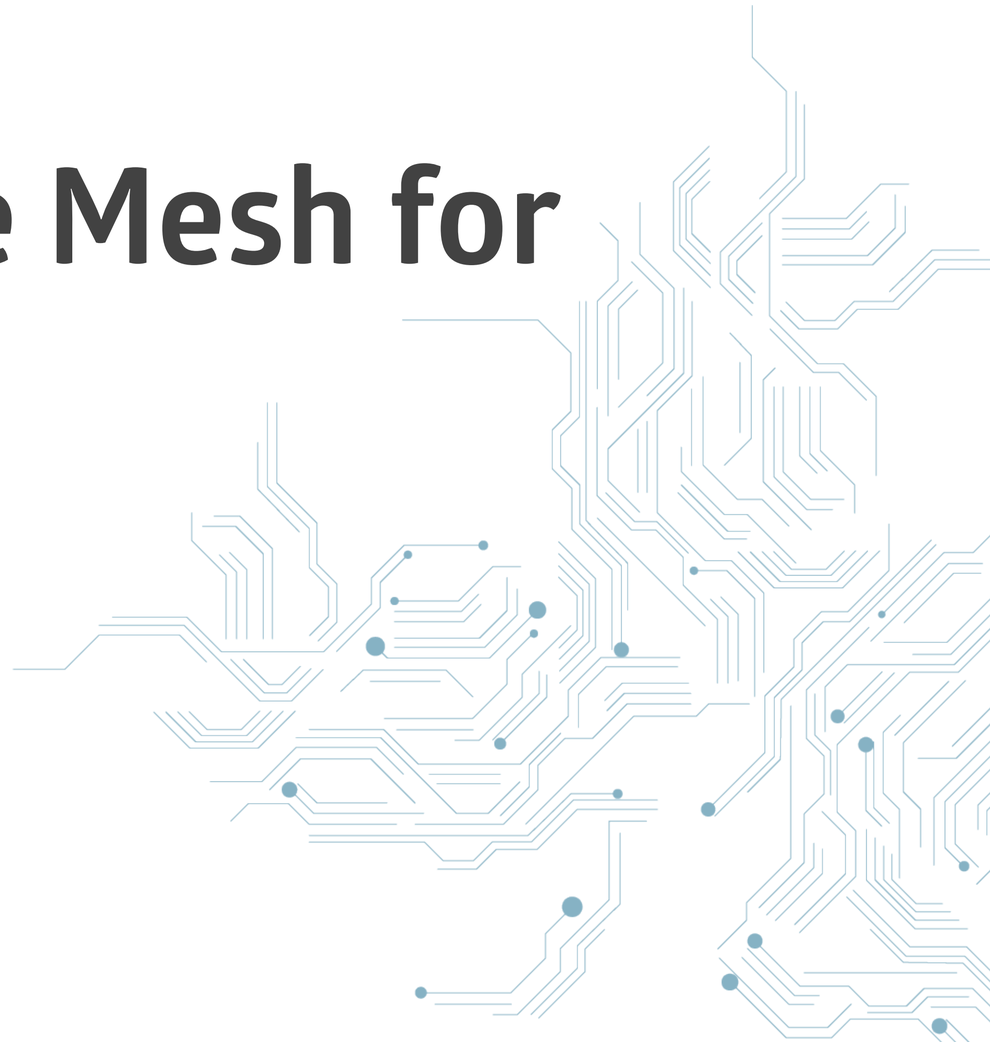
cert-manager to retrieve Let's Encrypt certificates automatically

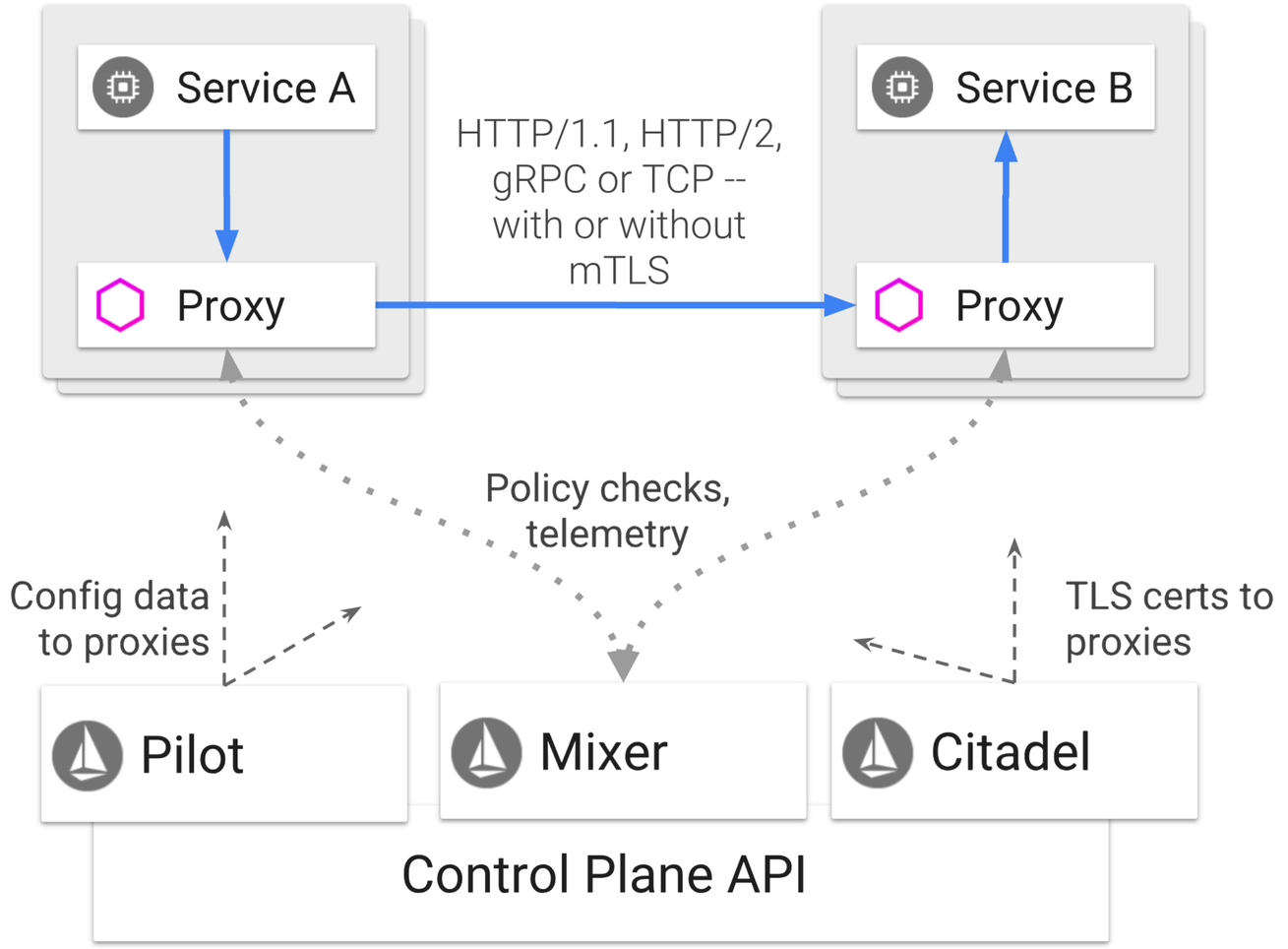
Database is managed by an Operator




pod
MySQL Operator
Discovers
crd
MySQLCluster
Creates
Monitors and manages
sts
MySQL statefulset
pod
pod
MySQL pods
Discovers
Creates
c-m
Kubernetes controller manager

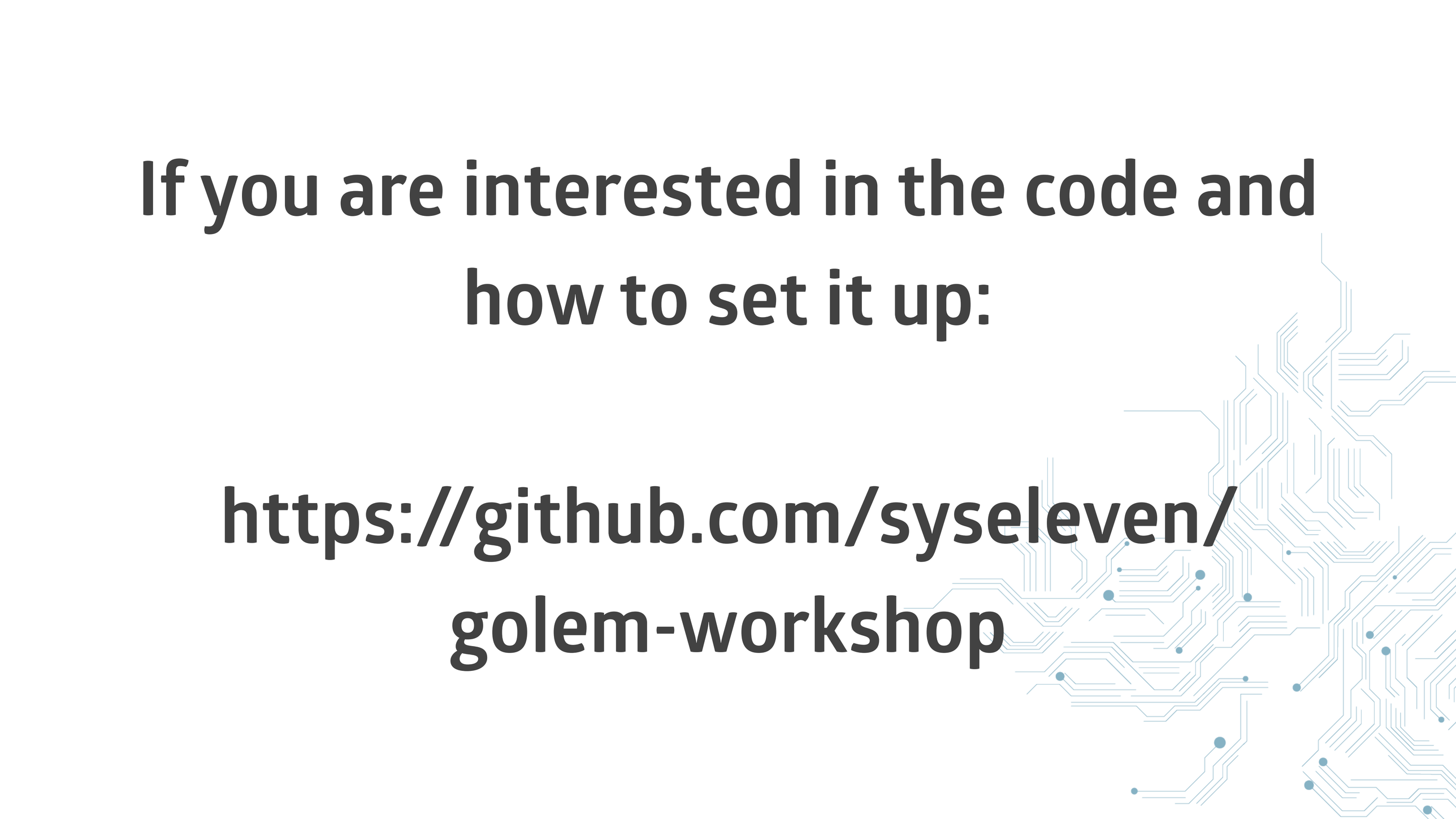
LinkerD as a Service Mesh for Telemetry




Service A
Service B
HTTP/1.1, HTTP/2,
gRPC or TCP --
with or without
mTLS
Proxy
Proxy
Policy checks,
telemetry
Config data
to proxies
TLS certs to
proxies
Pilot
Mixer
Citadel
Control Plane API

If you are interested in the code and how to set it up:

https://github.com/syseleven/golem-workshop



Demo

Writing this YAML files is tedious

YAML files are tied to a specific version and a specific environment

Production

Staging

Development

Per Development team

Per branch

Per developer

We need to maintain multiple, very similar YAML files with slightly different versions and configuration

"Templating"

Great tools because of standardized Kubernetes API

Helm

HELM

Get Helm Blog Docs

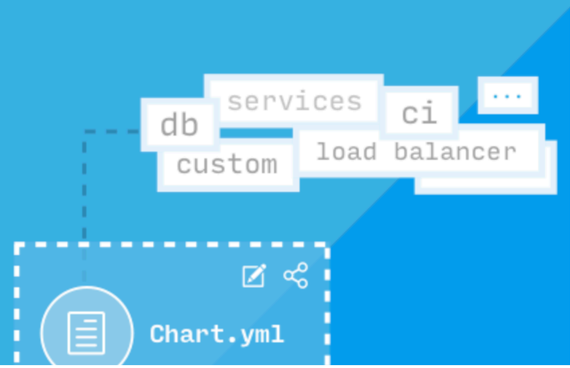
The package manager for Kubernetes

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

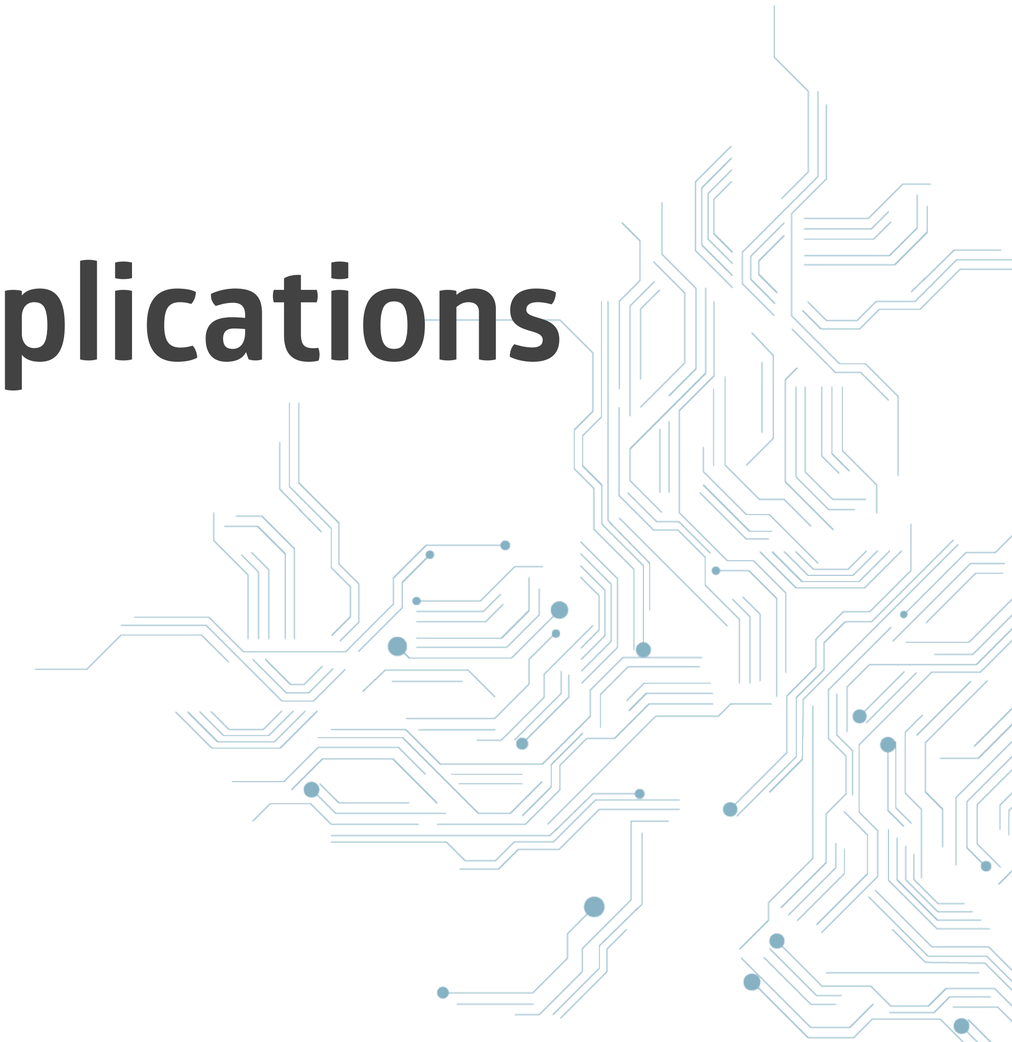
What is Helm?

Helm helps you manage Kubernetes applications — Helm Charts helps you define, install, and upgrade even the most complex Kubernetes application.

Charts are easy to create, version, share, and publish — so start using Helm and stop the copy-and-paste



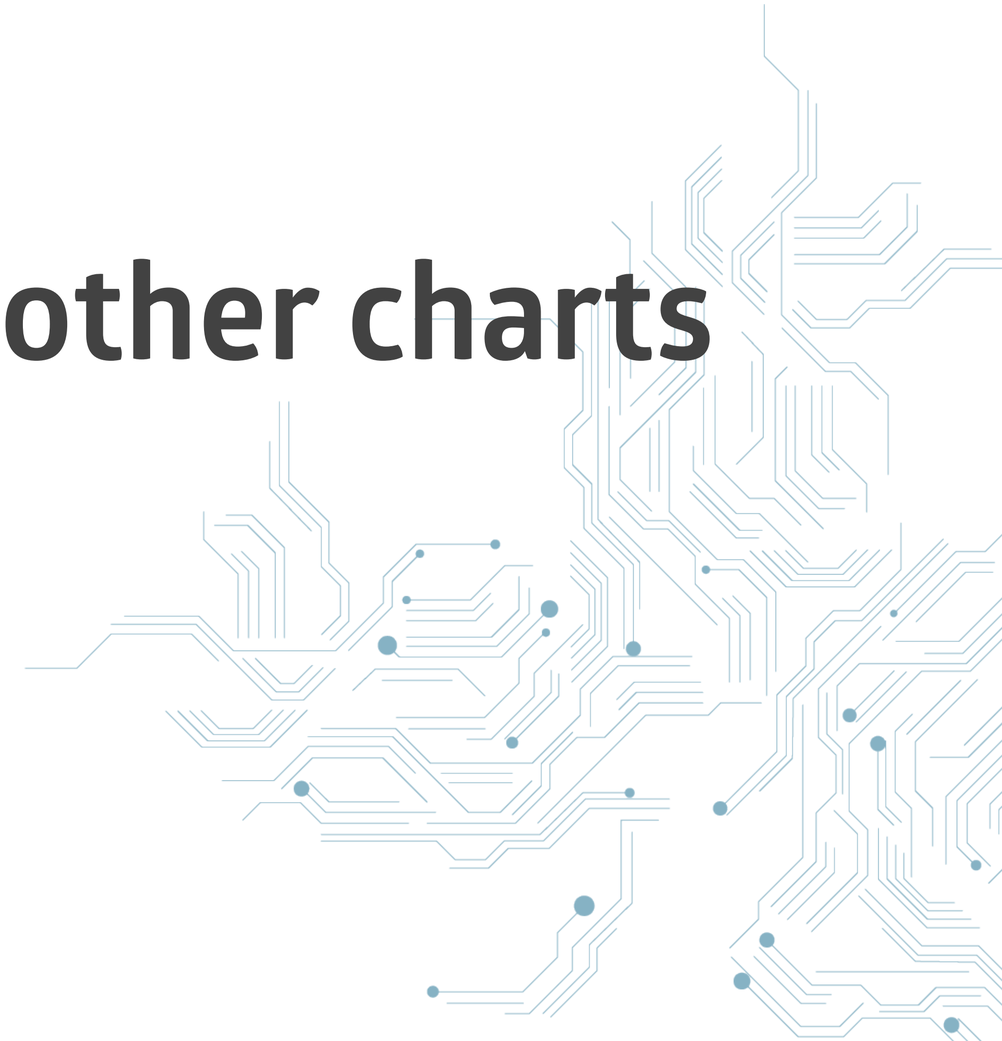
Allows to install applications



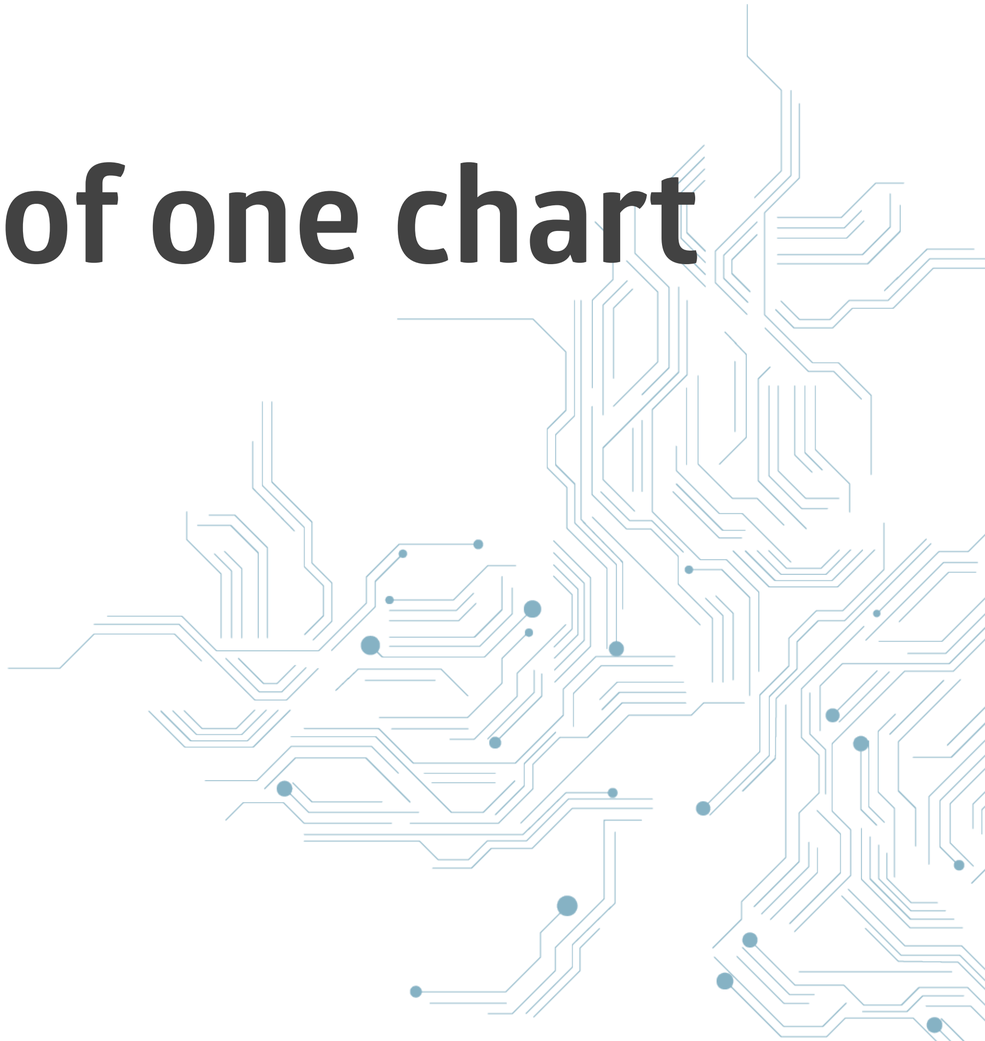
So called "charts"

```
$ helm install stable/wordpress \
  --name my-blog \
  --namespace blog
```

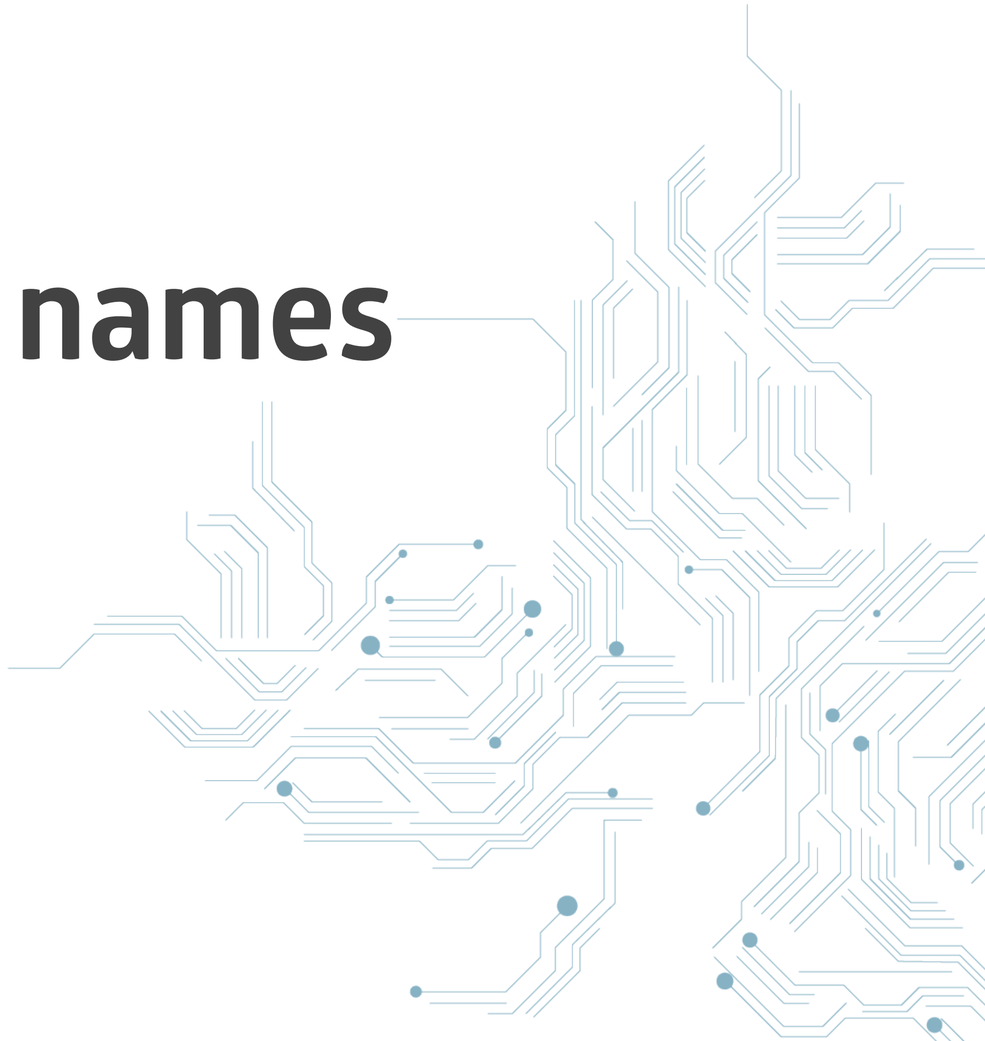
Charts can depend on other charts



Multiple deployments of one chart possible

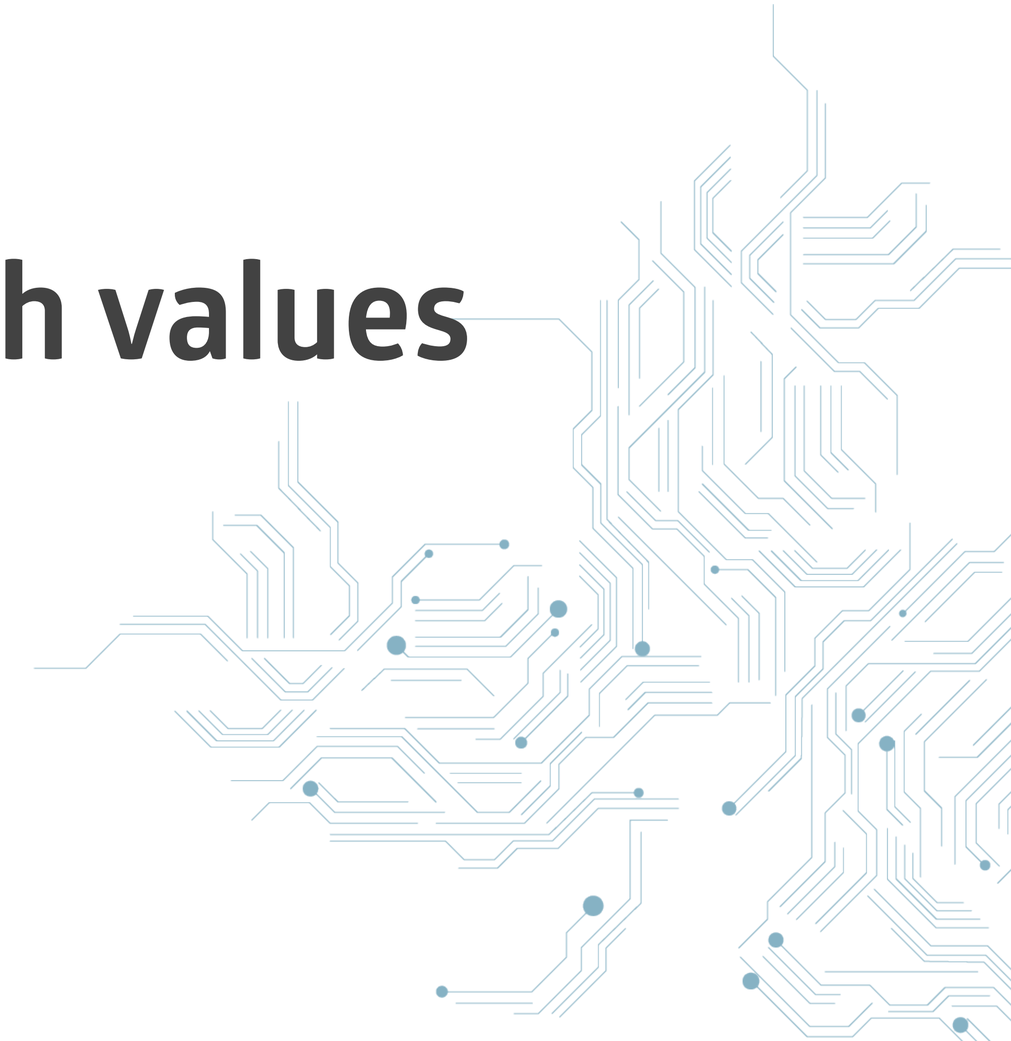


Different release names



Different namespaces

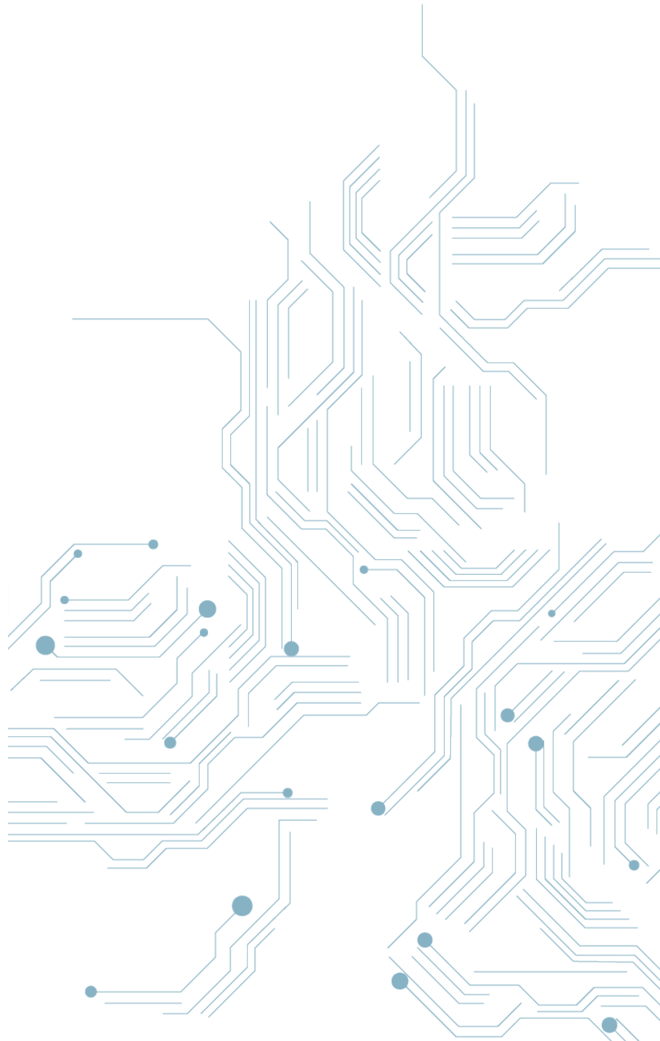
Configuration with values



Configuration

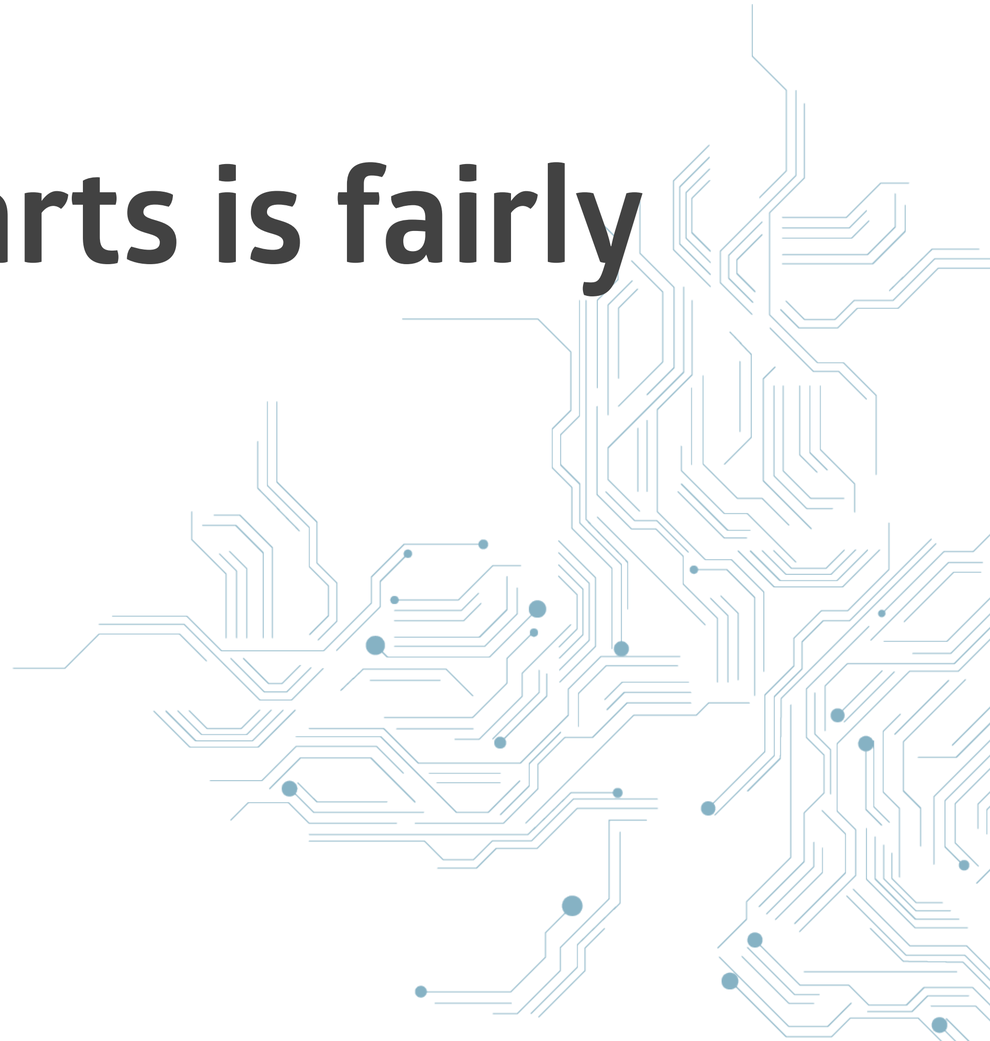
The following table lists the configurable parameters of the WordPress chart and their default values.

Parameter	Description	Default
`image.registry`	WordPress image registry	`docker.io`
`image.repository`	WordPress image name	`bitnami/wordpress`
`image.tag`	WordPress image tag	`{VERSION}`
`image.pullPolicy`	Image pull policy	`Always` if `imageTag` is `latest`, else `IfNotPresent`
`image.pullSecrets`	Specify image pull secrets	`nil`
`wordpressUsername`	User of the application	`user`
`wordpressPassword`	Application password	*random 10 character long alphanumeric string*
`wordpressEmail`	Admin email	`user@example.com`
`wordpressFirstName`	First name	`FirstName`
`wordpressLastName`	Last name	`LastName`
`wordpressBlogName`	Blog name	`User's Blog!`
`wordpressTablePrefix`	Table prefix	`wp_`
`allowEmptyPassword`	Allow DB blank passwords	`true`
`smtpHost`	SMTP host	`nil`
`smtpPort`	SMTP port	`nil`
`smtpUser`	SMTP user	`nil`
`smtpPassword`	SMTP password	`nil`
`smtpUsername`	User name for SMTP emails	`nil`
`smtpProtocol`	SMTP protocol [`tls`, `ssl`]	`nil`



```
$ helm install stable/wordpress \
  --name my-blog \
  --namespace blog \
  -f my-config-values.yaml
```

Writing your own charts is fairly easy



Scaffolding to get started

```
$ helm create quote-svc
```

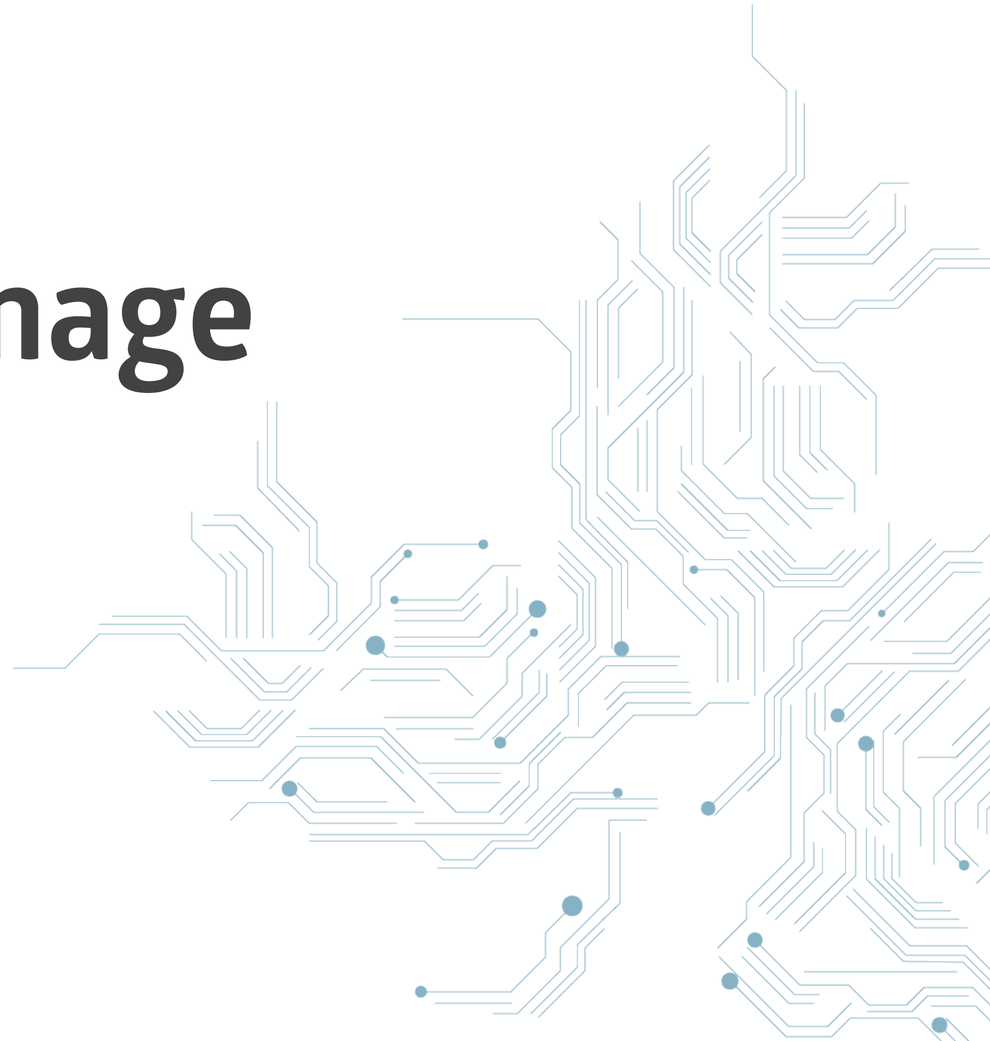
```
$ helm install ./quote-svc \
  --namespace dev-bastian \
  --name dev-bastian-quote-svc \
  --values dev.yaml --values bastian.yaml
```

Demo

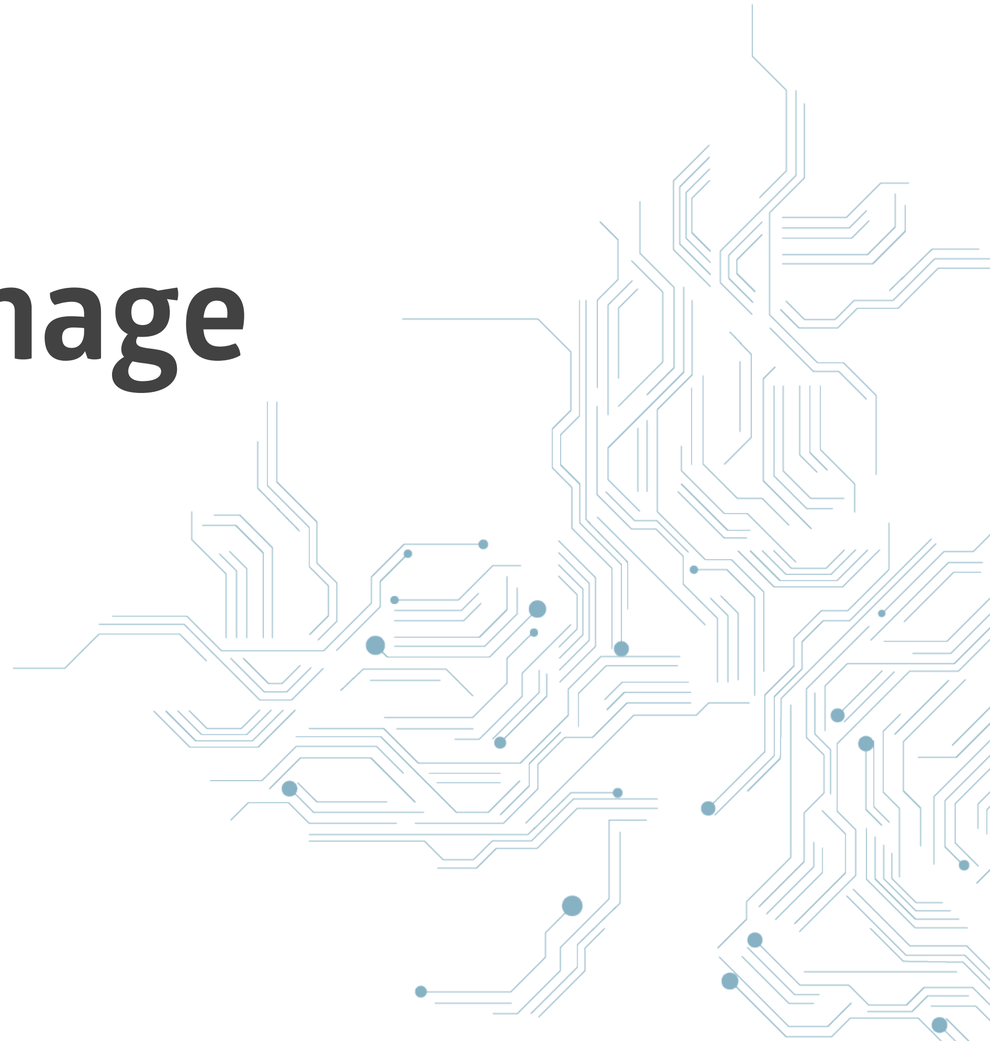
Still, for development:

Make a code change

Build docker image



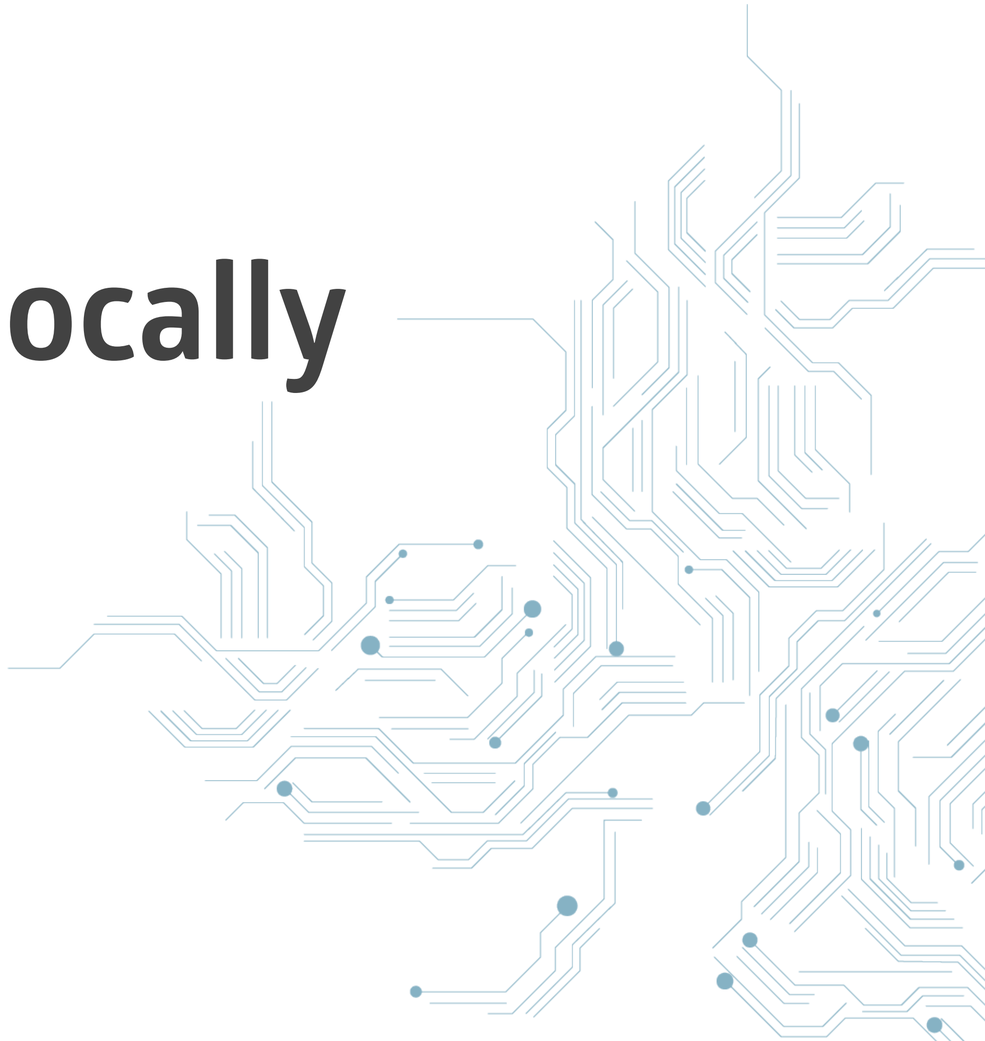
Push docker image



Run helm install/upgrade with new image version

Can this be quicker?

Run everything locally



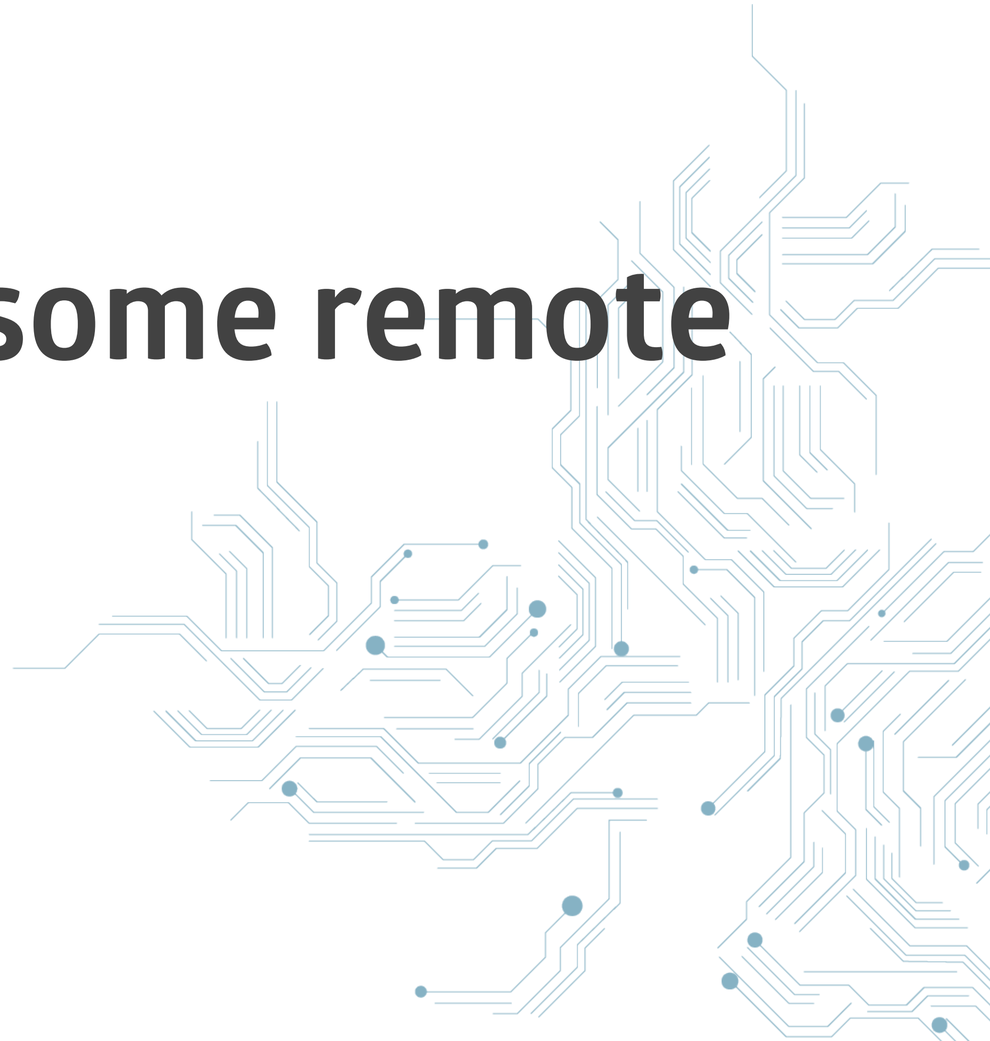
docker-compose

Duplication of the definition of how to run a container

Inconsistencies

If you have a lot of services, you have to run a lot locally

Some services locally, some remote



Service Discovery

Not every service is exposed to the Internet

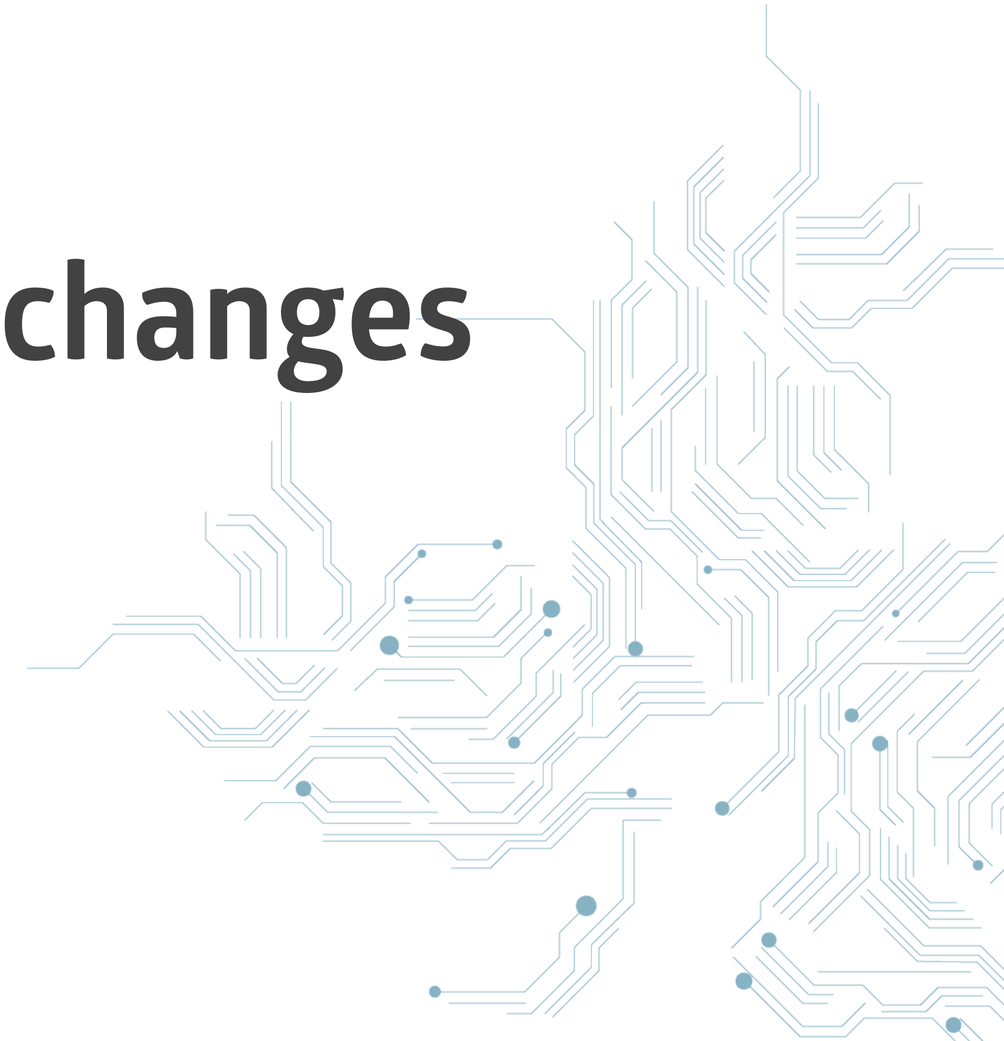
Shared resources with other developers?

Other options?

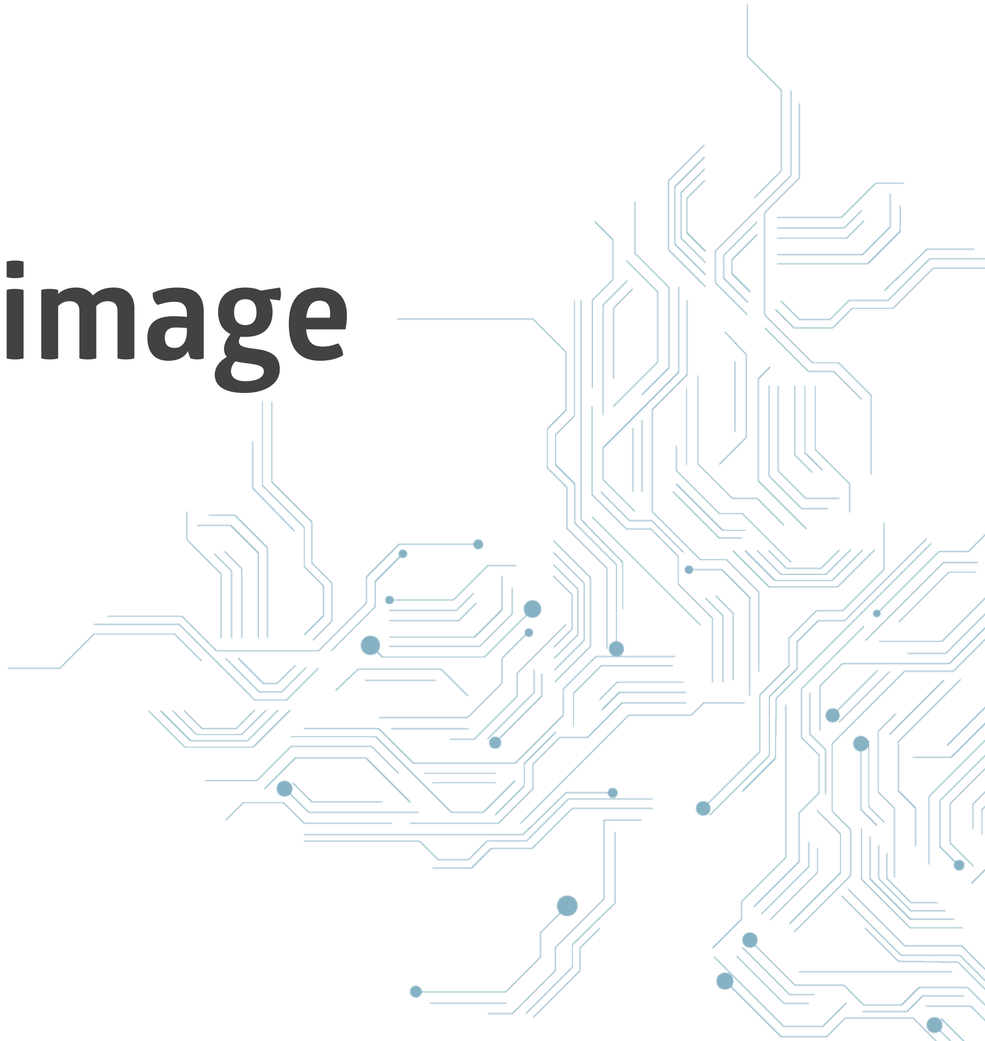
Tilt

```
$ tilt up
```

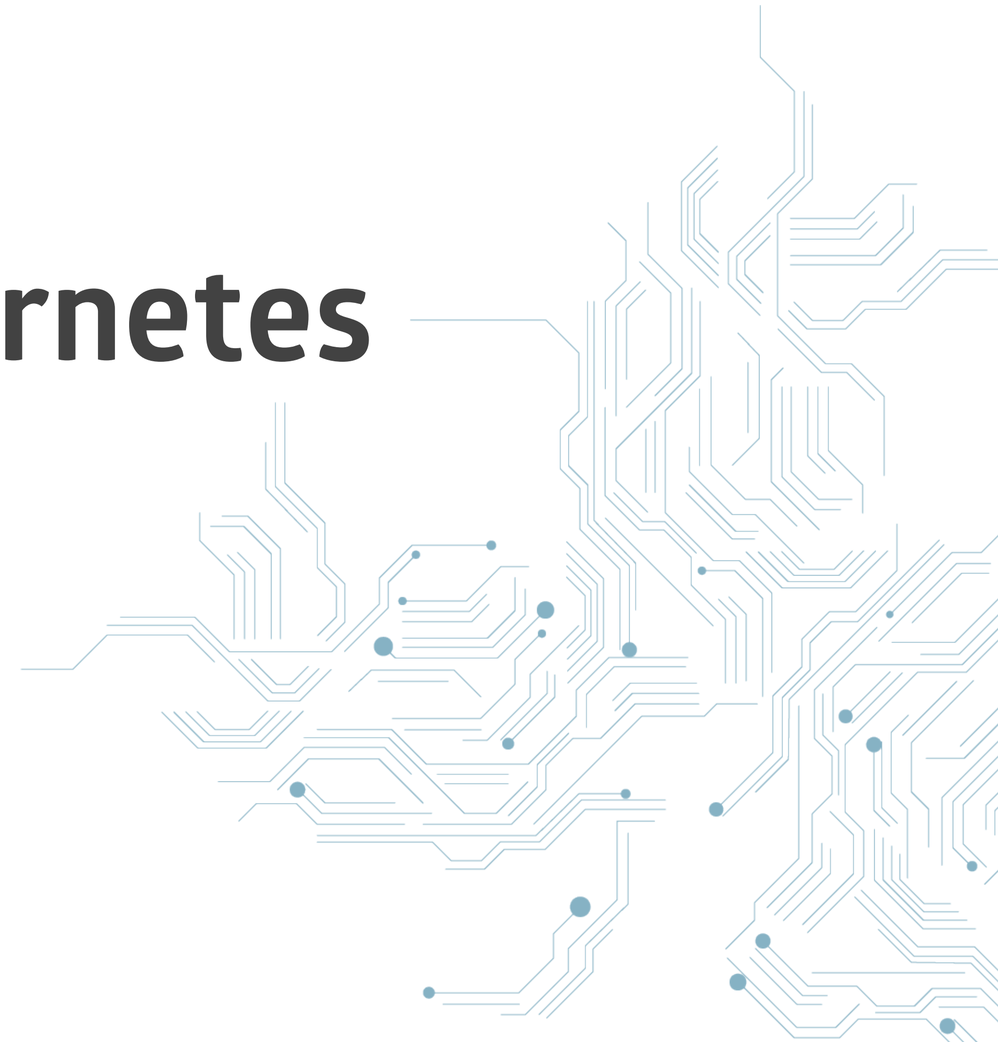
Watches for code changes



Rebuilds docker image

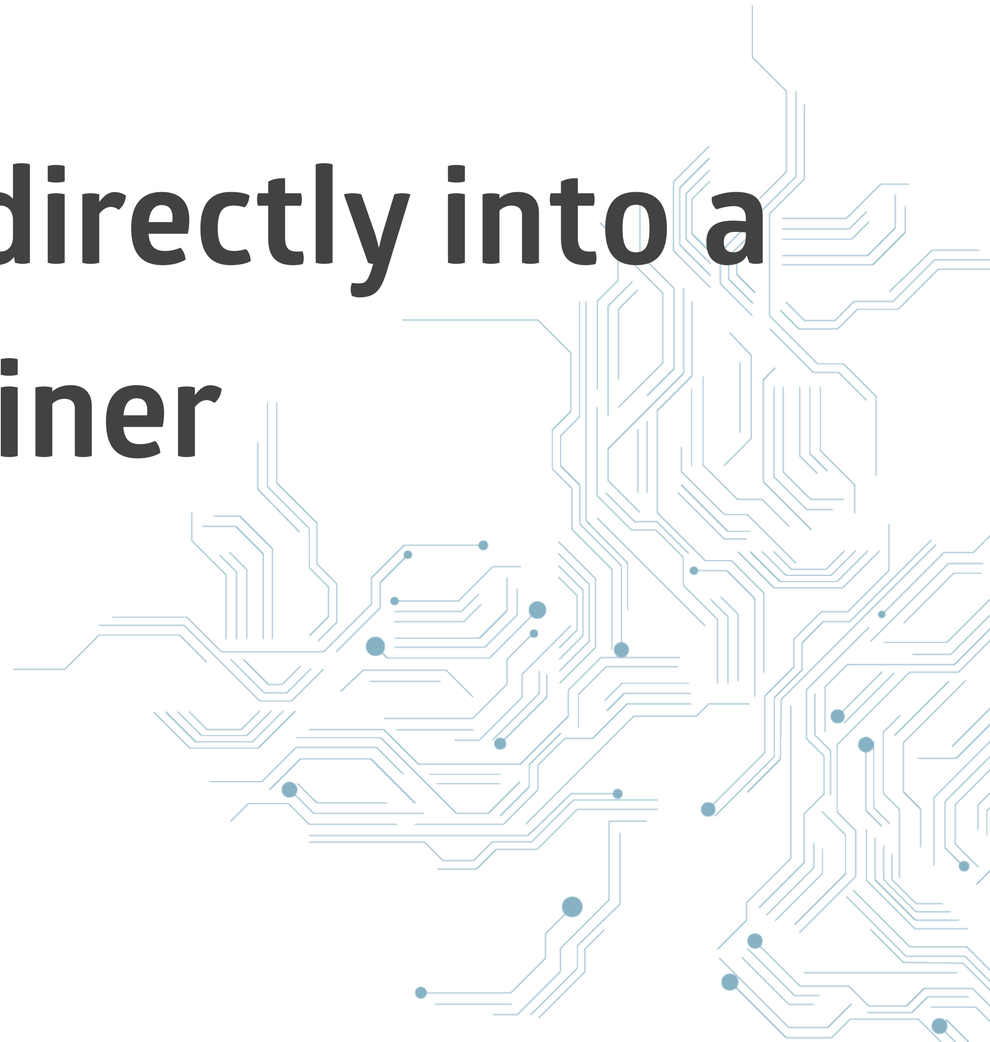


Deploys to Kubernetes



Sets up port-forwarding

Can sync changed files directly into a running container



Demo

Another approach

TELEPRESENCE

FAST, LOCAL DEVELOPMENT FOR KUBERNETES AND OPENSHIFT MICROSERVICES

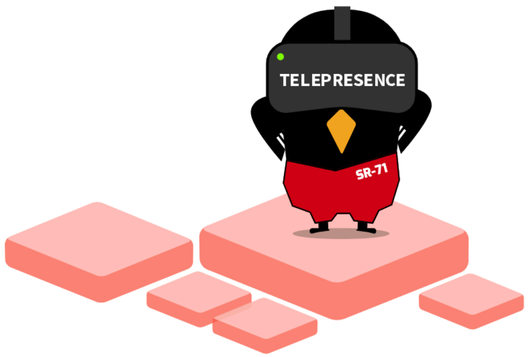
GET STARTED

Star 1,244 Fork 73

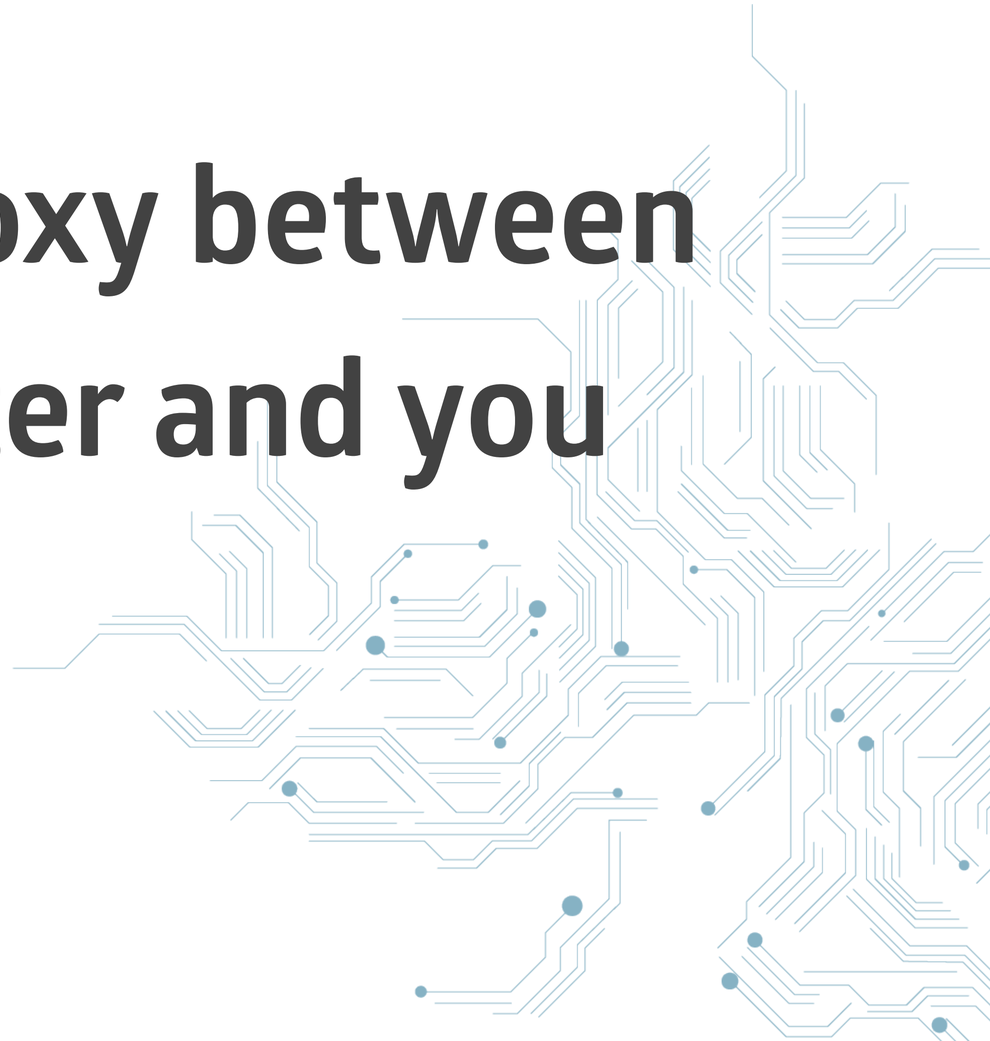
0.92

Telepresence 0.92 is now available

Read the CHANGELOG



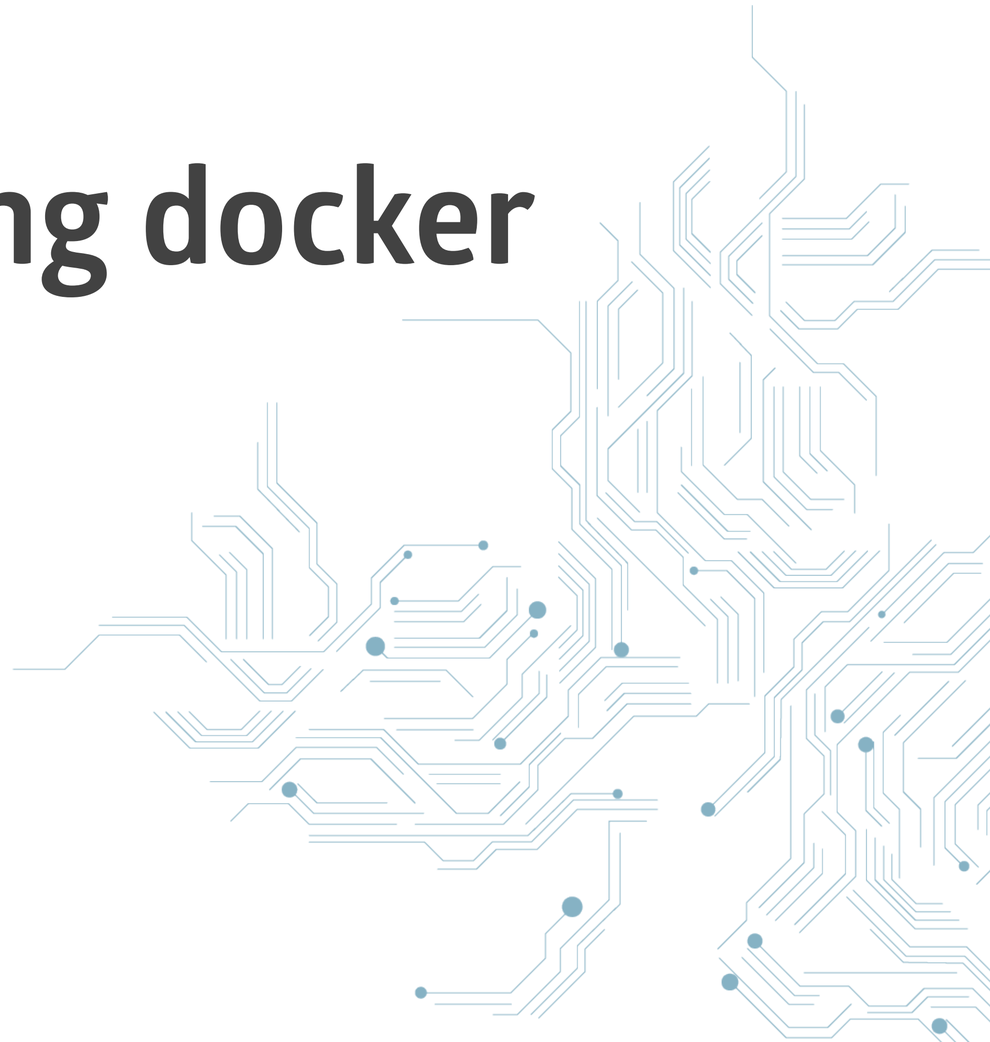
Creates a two-way proxy between the Kubernetes cluster and you



```
$ telepresence
T: Starting proxy with method 'vpn-tcp'...

@fhgbvx65xg|bash-3.2$ curl http://quote-svc/quote | jq '.'
[
  {
    "ID": 503,
    "title": "stefan sagmeister",
    "content": "<p>...</p>\n",
    "link": "https://quotesondesign.com/stefan-
sagmeister-2/"
  }
]
```

Swap a running deployment in the cluster with a local process

… or a locally running docker container



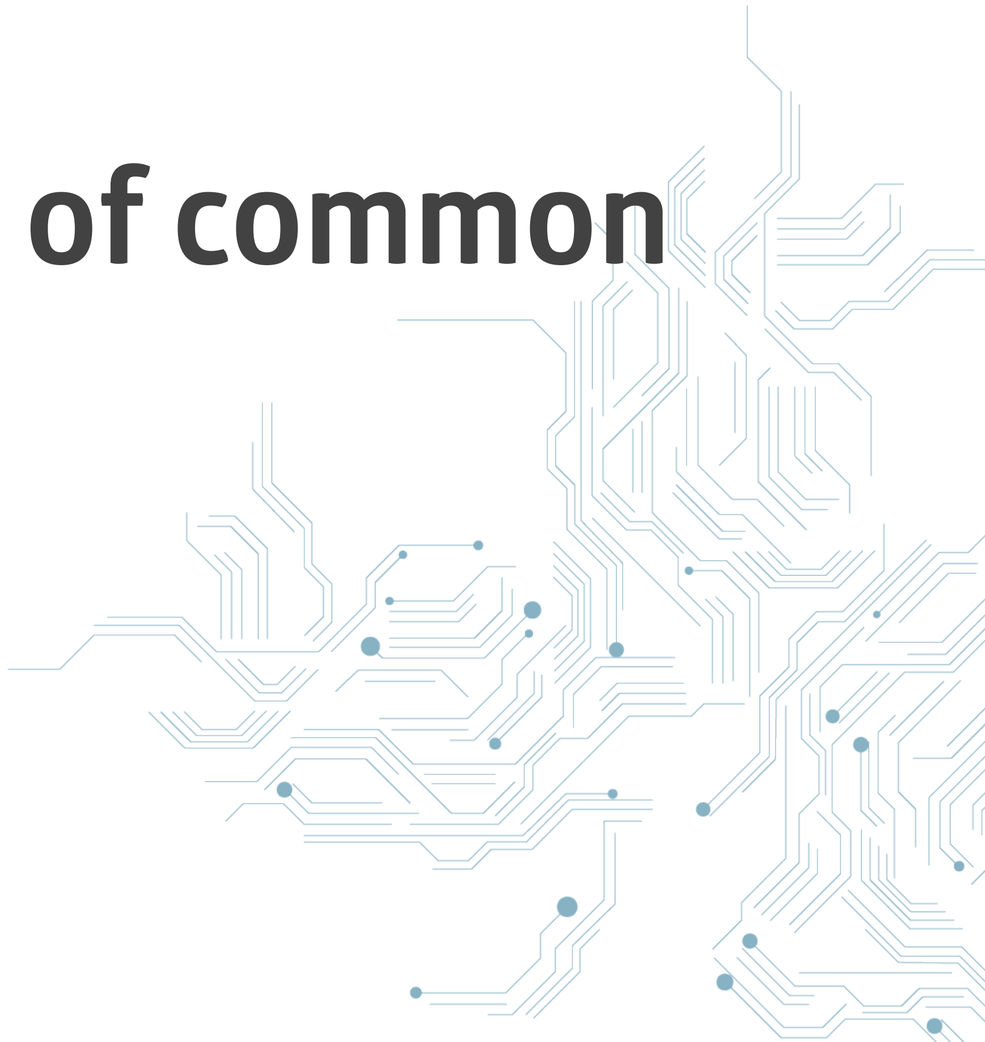
```
$ telepresence --swap-deployment quote-svc --namespace
dev-flow-demo --expose 3000 --run npm run debug

T: Starting proxy with method 'vpn-tcp',...
T: Forwarding remote port 3000 to local port 3000....

> quote-svc@1.0.0 debug /Users/bhofmann/forge_test/quote-
svc
> nodemon --inspect quote-svc.js
[nodemon] watching: *.*
[nodemon] starting `node --inspect quote-svc.js`
Debugger listening on ws://127.0.0.1:9229/83aa27ac-
d879-4b50-a228-440354cca791
quote svc listening on port 3000!
```

Demo

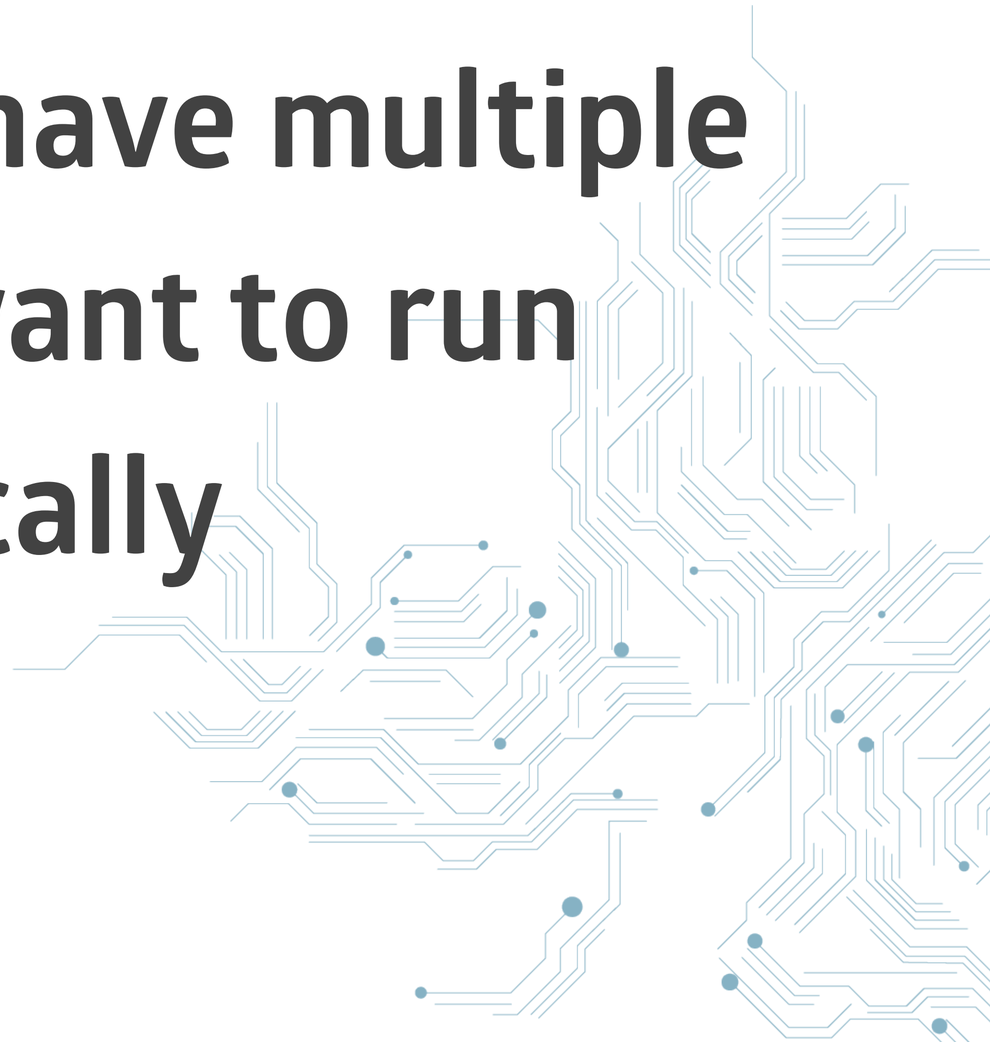
Summary

Powerful

Great tooling because of common APIs



Especially great if you have multiple services and don't want to run everything locally



I just picked helm, tilt and telepresence. There is more for different use-cases.

mail@bastianhofmann.de

https://twitter.com/BastianHofmann
http://speakerdeck.com/u/bastianhofmann
https://github.com/syseleven/golem-workshop