



KubeCon



CloudNativeCon

North America 2017



# Moving from Mesos to Kubernetes *without anyone noticing\**

Anubhav Mishra



**Anubhav Mishra**

@anubhavm



**Anubhav Mishra**

@anubhavm



# Hootsuite®

Chrome File Edit View History Bookmarks People Window Help

https://hootsuite.com/dashboard

Send to... Compose message...

Instagram Twitter Facebook Bread & Coffee + Add Stream + Add Social Network

Home hootsuite 10 mins God's artwork: the #Eldoret #sunset. The best art there is... 3 likes Add a comment...

My Posts hootsuite 20 hours hootsuite 142 likes 1 comment Add a comment...

#hootsuitelife hootsuite 2 hours adrietsa 15 likes 3 comments @enrifzaparick legit come visit upstairs! WHO IS THIS? Add a comment...

Hootsuite hootsuite 9 Apr 21, 7:22pm jaimestein 38 likes 2 comments Latergram from yesterday's Game of Thrones drop from STELLUS. Soneho House Lannister! #OptikDelivers #HootsuiteLife @mccigure\_kat Add a comment...

thejanuaryfall 20 hours hand placement!! fun times! 1 like Add a comment...

hootsuite 2 days graceless\_kelly 4 hours allie\_rus 9 Apr 21, 6:19pm telus 1 like The Lannisters send their regards. thepantheonians 1 like That's some wicked box art! Add a comment...

socialatte 10 mins thepantalonians 1 like Apr 21, 7:26pm



**Anubhav Mishra**

@anubhavm



**Atlantis**



**Anubhav Mishra**

@anubhavm



**Atlantis**



HashiCorp



**Anubhav Mishra**

@anubhavm



Atlantis



HashiCorp

**Terraform**



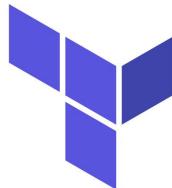
Anubhav Mishra

@anubhavm





Atlantis



HashiCorp

**Terraform**



Anubhav Mishra

@anubhavm





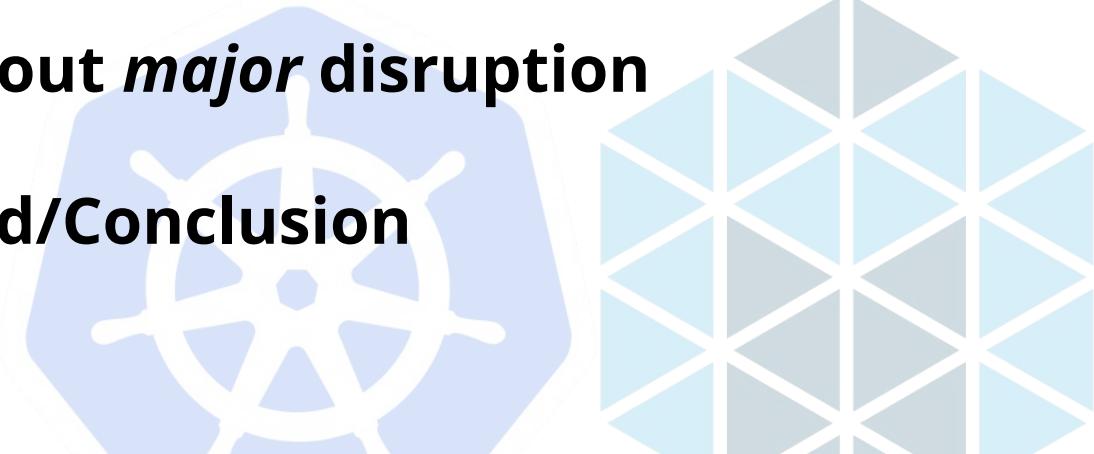
**VS**





# Agenda

- **Hootsuite's Journey from Mesos to Kubernetes**
- **Microservices pipeline**
  - Mesos and Marathon
  - Kubernetes
- **Migration without *major* disruption**
- **Live demo!** 🙏
- **Lessons learned/Conclusion**



# Hootsuite Now

# Numbers

- **120+** developers
- **50+** microservices
- **2** cluster schedulers
- **1500+** servers on **AWS**

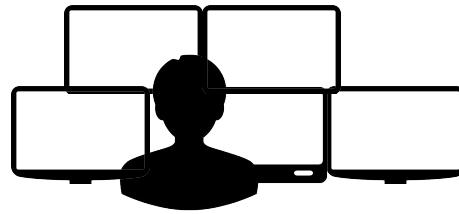
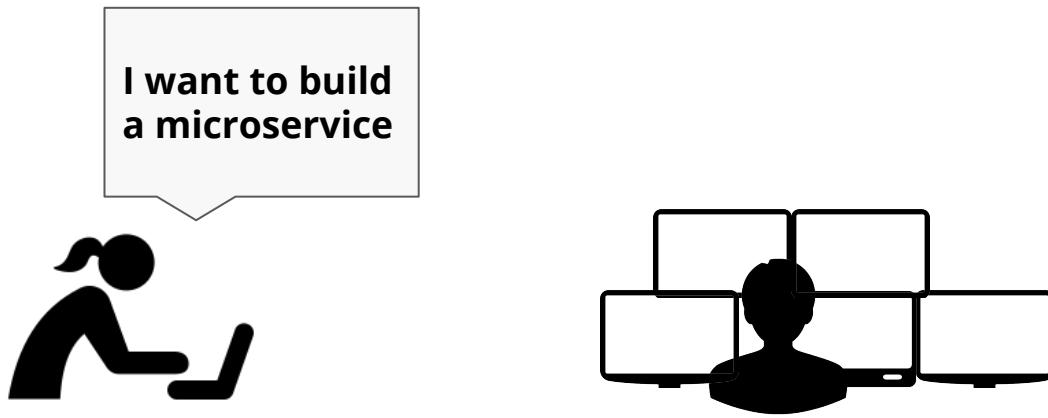


2014



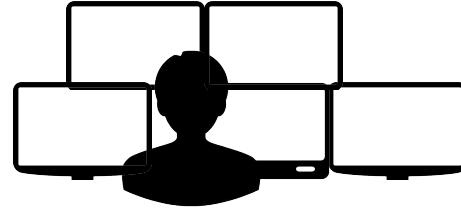
I want to build  
a microservice





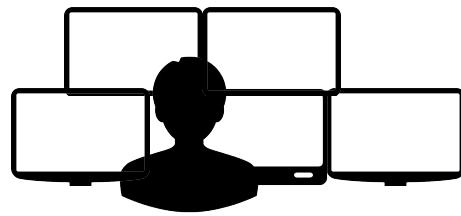


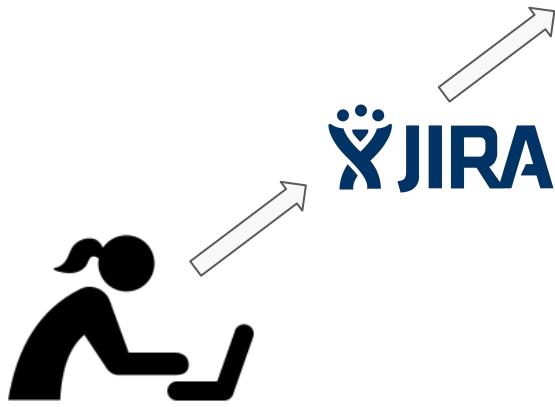
I want to build  
a microservice



Oh! A “microservice”?  
Hmm.. seems to be the  
new thing huh. Yep, just  
create a JIRA ticket.

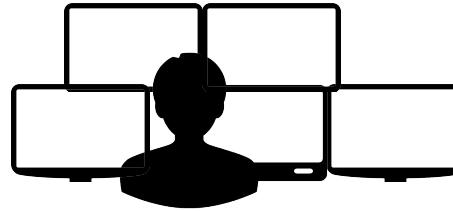
**Minutes later....**

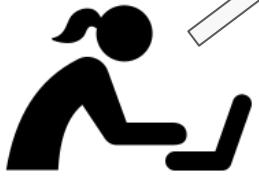




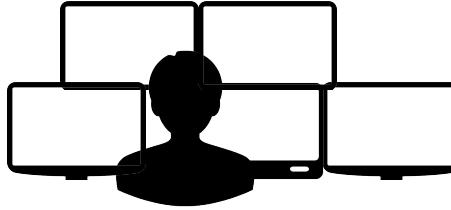
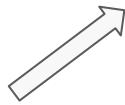
JIRA

A screenshot of a JIRA issue detail page. At the top, there are buttons for Edit, Comment, Assign, More, Start Progress, Resolve Issue, and Workflow. On the right, there's an Export button. The main section shows issue details: Type: Bug, Priority: Major, Labels: None; Status: Open (View Workflow), Resolution: Unresolved. To the right, under People, it lists Assignee: Admin, Reporter: Admin, Votes: Vote for this issue, and Watchers: Stop watching this issue. Below that, Dates show Created: 25 minutes ago and Updated: 8 minutes ago. The Activity tab is selected, showing a feed of comments from users Fred, christina, and Admin. A Drag and Drop area below the feed says "Drop files here to attach them".





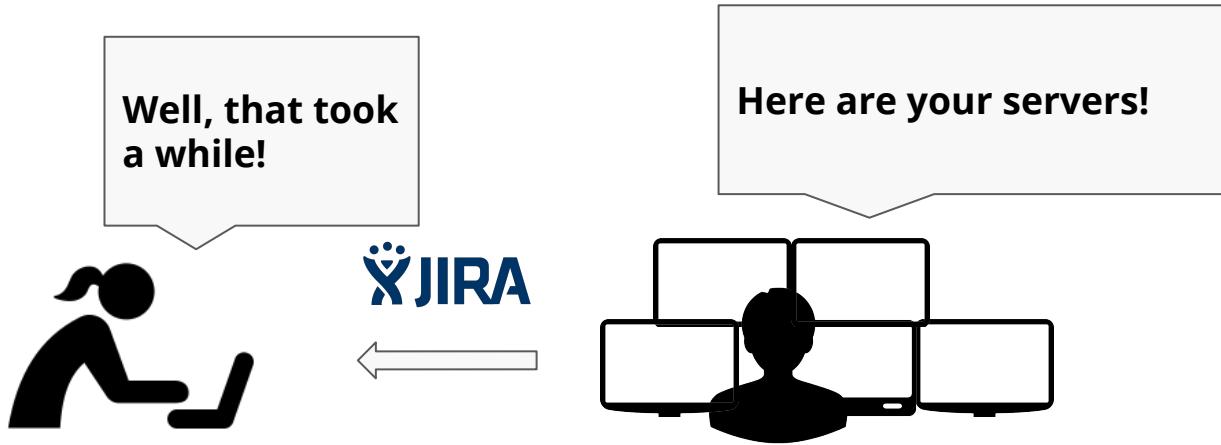
# JIRA



The image shows three screenshots of the Atlassian Confluence software interface:

- Screenshot 1 (Left):** A dashboard view showing a "Details" section with fields for "Type," "Priority," and "Labels." Below it is a "Description" section and a "Space Directory" sidebar listing various spaces like "Development," "Documentation," and "Marketing."
- Screenshot 2 (Middle):** A "Space Directory" view titled "All Spaces" showing a list of spaces with their descriptions and categories. Examples include "We build stuff. Good stuff." (Category: Development), "We like to write. A lot." (Category: Documentation), and "We care about you. Really." (Category: Marketing).
- Screenshot 3 (Right):** A "People Directory" view showing a list of users such as "All People," "People with Personal Spaces," and "People with Recent Activity." It includes a search bar and a "Recent Activity" section.

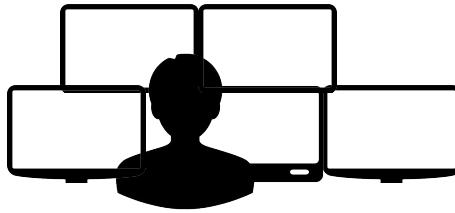
**Weeks later....**



Ok! Now I only need  
Java, Sensu checks  
and a Jenkins  
pipeline top deploy  
to the servers



JIRA





**2016-2017**





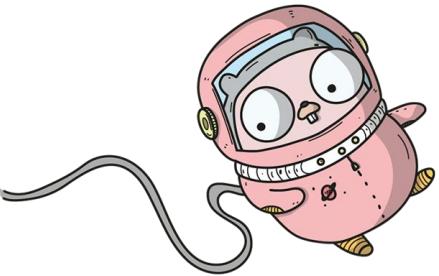
I want to build  
a microservice

**5 minutes later....**

I just deployed  
a microservice  
to production!



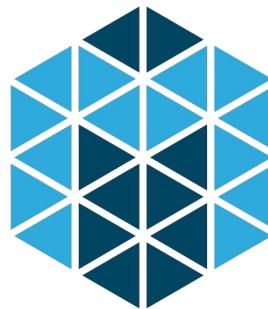
# Microservice Pipeline



`./project-generator`



**Pipeline as Code**



**Mesos**



**Marathon**

# Project Skeleton

```
#####
# Modify these vars

# Name of your service, lowercase, dash separated. ie. [a-z\-]
# !!! SHOULD NOT have "service" as prefix or suffix. We're trying to get rid of that.
# Will be used for Skyline and Kubernetes names as well as statsd prefixes, etc.
#name: my-awesome-service
name: new-service

# Human readable title. Used for docs and such
#nameNice: My Awesome Service
nameNice: My New Service

# Long form description
#description: Skyline sample service generated from the skeleton based off of Play! framework, dockerized, and
description: Hello World

#maintainers:
# - name: Firstname Lastname
#   email: firstname.lastname@hootsuite.com
# - name: Joe Blough
#   email: joe.blough@hootsuite.com
maintainers:
- name: Anubhav Mishra
  email: anubhav.mishra@hootsuite.com

# Pick a project type. Can either be: scala, go, go-grpc, idl etc.
# This can be set with the command line arg `--project-type` which will allow you to
# create multiple project types with the same config. This is useful for projects like `go-grpc` + `idl`
# projectType: go
projectType: go
```

# Project Skeleton

```
#####
# Modify these vars

# Name of your service, lowercase, dash separated. ie. [a-z\-]
# !!! SHOULD NOT have "service" as prefix or suffix. We're trying to get rid of that.
# Will be used for Skyline and Kubernetes names as well as statsd prefixes, etc.
#name: my-awesome-service
name: new-service

# Human readable title. Used for docs and such
#nameNice: My Awesome Service
nameNice: My New Service

# Long form mode (optional)
#description: Skyline sample service generated from the skeleton based off of Play! framework, dockerized, and
description: Hello World

#maintainers:
# - name: Firstname Lastname
#   email: firstname.lastname@hootsuite.com
# - name: Joe Blough
#   email: joe.blough@hootsuite.com
maintainers:
- name: Anubhav Mishra
  email: anubhav.mishra@hootsuite.com

# Pick a project type. Can either be: scala, go, go-grpc, idl etc.
# This can be set with the command line arg `--project-type` which will allow you to
# create multiple project types with the same config. This is useful for projects like `go-grpc` + `idl`
# projectType: go
projectType: go
```



```
Dockerfile
Dockerfile-dev
Jenkinsfile
Makefile
Makefile.base.mk
README.md
bin
  coverage.sh
  envtpl
  tdd.sh
conf
  about.json
  default.yml
  dev.yml
  production.yml
  staging.yml
  vagrant.yml
deploy
  application.yml
  dev.yml
  production.yml
  staging.yml
  vagrant.yml
  deploy-test.py
docs
  todo
index.go
main.go
main_test.go
middleware.go
server.go
skeleton-vars.yml
status.go
status_test.go
```

# Pipeline as Code

```
pipeline {  
    agent any  
  
    stages {  
        stage('Build') {  
            steps {  
                sh 'make'  
            }  
        }  
        stage('Test'){  
            steps {  
                sh 'make test'  
                junit 'reports/**/*.xml'  
            }  
        }  
        stage('Deploy') {  
            steps {  
                sh 'make deploy'  
            }  
        }  
    }  
}
```



- Dockerfile
- Dockerfile-dev
- Jenkinsfile
- Makefile
- Makefile.base.mk
- README.md
- bin
  - coverage.sh
  - envtpl
  - tdd.sh
- conf
  - about.json
  - default.yml
  - dev.yml
  - production.yml
  - staging.yml
  - vagrant.yml
- deploy
  - application.yml
  - dev.yml
  - production.yml
  - staging.yml
  - vagrant.yml
- deploy-test.py
- docs
  - todo
- index.go
- main.go
- main\_test.go
- middleware.go
- server.go
- skeleton-vars.yml
- status.go
- status\_test.go

# Pipeline as Code

Jenkins Jenkins Open Blue Ocean search

Production Delivery (github organization) mishra-go-skeleton master

Up Status Changes Build Now View Configuration GitHub Full Stage View Job Config History GitHub Pipeline Syntax

Recent Changes

## Branch master

Full project name: production-delivery (github organization)/mishra-go-skeleton/master

### Stage View

Average stage times:

setup	unit test	build service	push	deploy dev	deploy staging	deploy production
2s	8s	2s	3s	1s	3s	3s
2s	1s	1s	2s	1s	3s	12s (paused for 10s)
4s	19s	3s	6s	1s	3s	1s (paused for 10s) aborted
4s	23s	3s	6s	1s	3s	1s (paused for 6s) aborted
4s	1s	1s	2s	1s	3s	1s (paused for 4s) aborted

Build History trend →

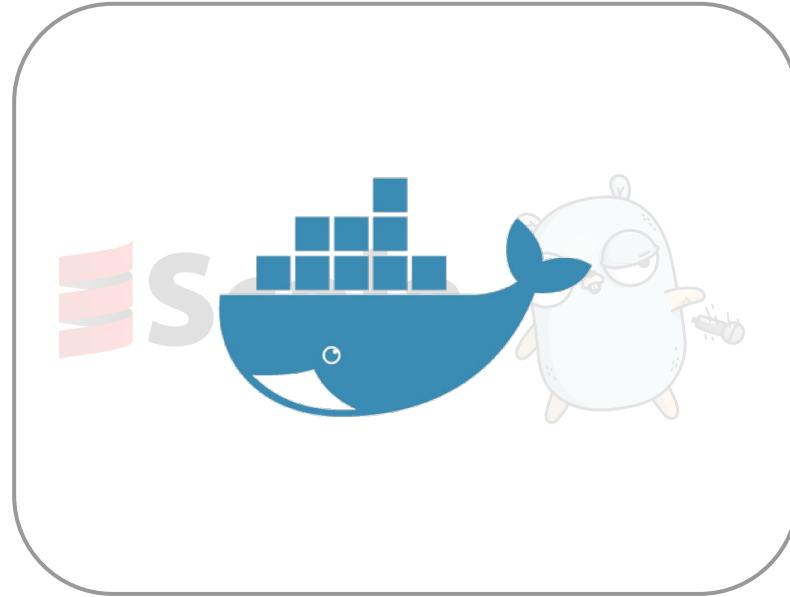
- 8-161010b Sep 16 18:30 No Changes
- 7-161010b Sep 16 18:30 No Changes
- 6-161010b Sep 16 18:28 No Changes
- 5-161010b Sep 16 15:34 1 commits
- 4-56c3eb7 Sep 16 2:54 PM
- 3-5008c81

The Scala logo, featuring the word "Scala" in a bold, black, sans-serif font. To the left of the "S" is a red graphic element consisting of three horizontal bars of decreasing height from left to right.

Scala



# Packaging



# Deployment Files

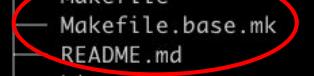
```
replicas: 1
resources:
  cpu: 2
  memory: 200M
healthChecks:
...
...
```



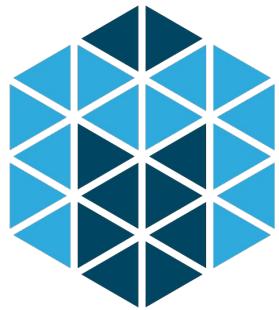
```
Dockerfile
Dockerfile-dev
Jenkinsfile
Makefile
Makefile.base.mk
README.md
bin
  coverage.sh
  envtpl
  tdd.sh
conf
  about.json
  default.yml
  dev.yml
  production.yml
  staging.yml
  vagrant.yml
deploy
  application.yml
  dev.yml
  production.yml
  staging.yml
  vagrant.yml
  deploy-test.py
docs
  todo
index.go
main.go
main_test.go
middleware.go
server.go
skeleton-vars.yml
status.go
status_test.go
```

# Makefile

- make deploy-dev
- make deploy-staging
- make deploy-production



```
. └── Dockerfile
    └── Dockerfile-dev
    └── Jenkinsfile
    └── Makefile
    └── Makefile.base.mk (highlighted)
    └── README.md
    └── bin
        └── coverage.sh
        └── envtpl
        └── tdd.sh
    └── conf
        └── about.json
        └── default.yml
        └── dev.yml
        └── production.yml
        └── staging.yml
        └── vagrant.yml
    └── deploy
        └── application.yml
        └── dev.yml
        └── production.yml
        └── staging.yml
        └── vagrant.yml
    └── deploy-test.py
    └── docs
        └── todo
    └── index.go
    └── main.go
    └── main_test.go
    └── middleware.go
    └── server.go
    └── skeleton-vars.yml
    └── status.go
    └── status_test.go
```



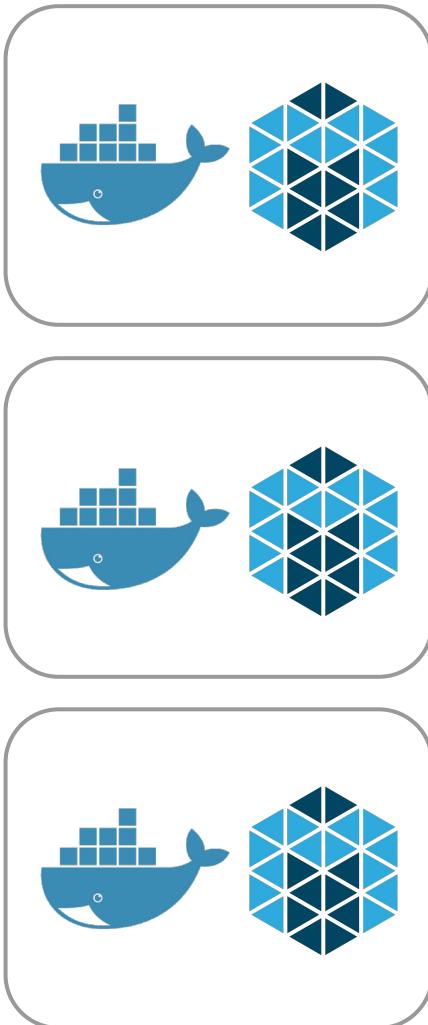
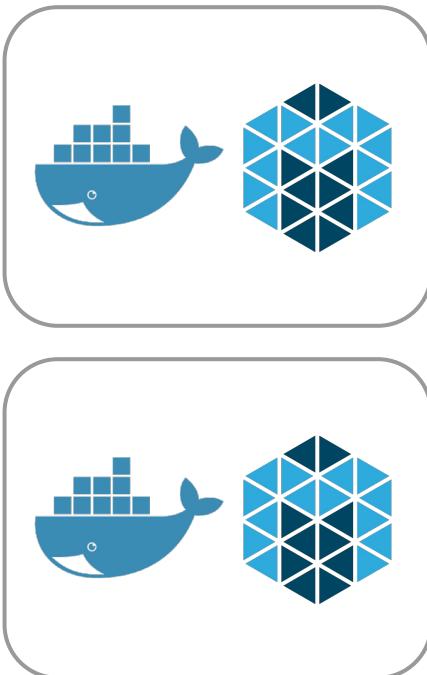
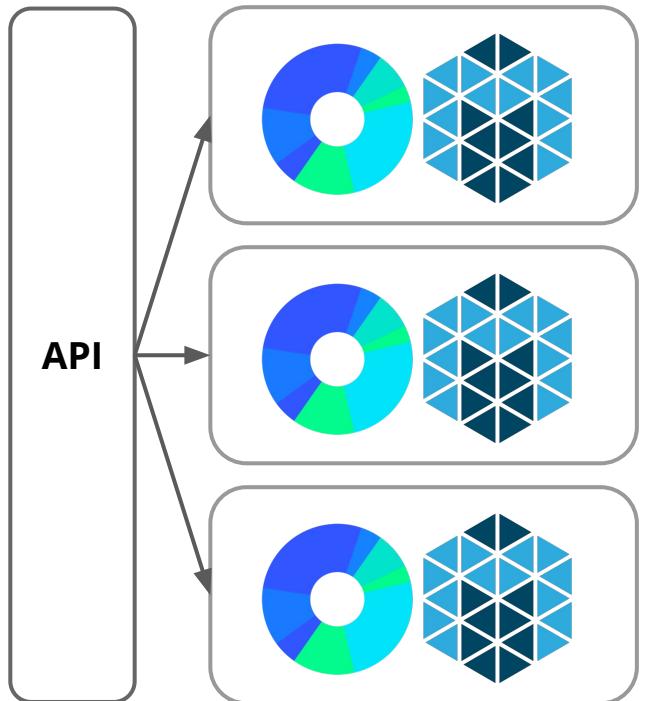
**Mesos**



**Marathon**



make deploy



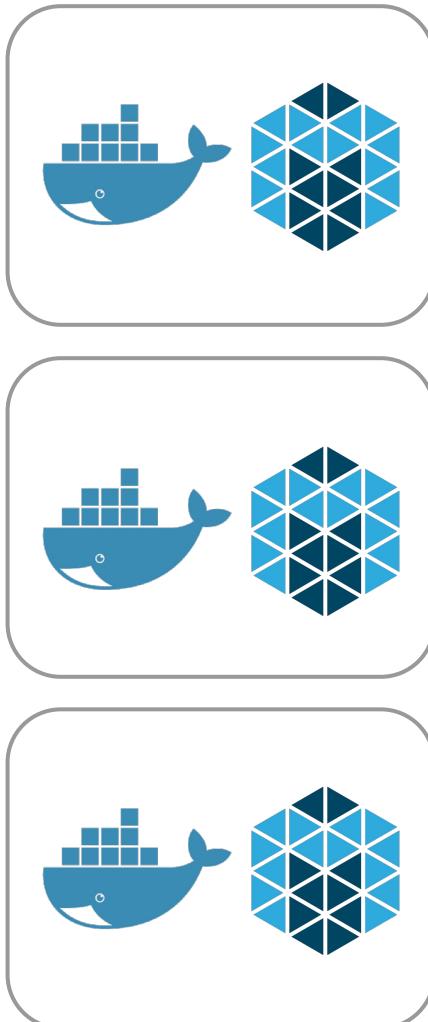
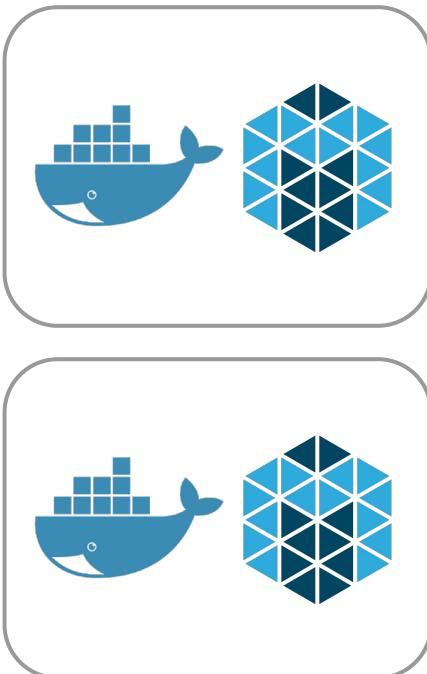
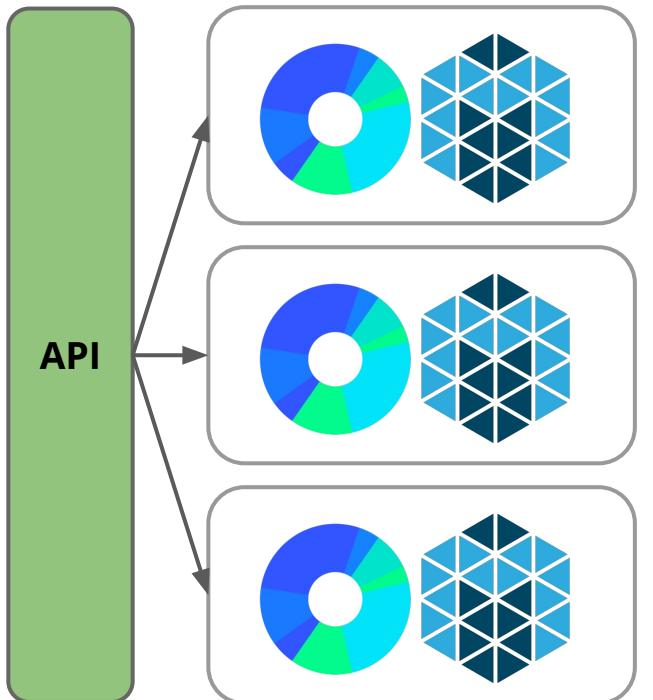


make deploy

POST  
{

"id": "service-1",  
"cpus": 0.1,  
"mem": 10.0,  
"instances": 1

}



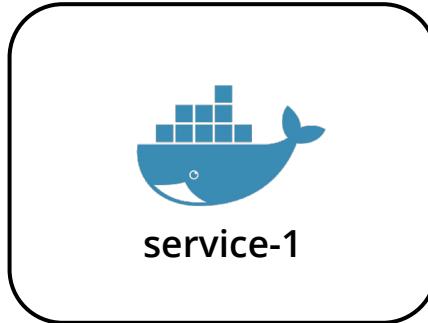
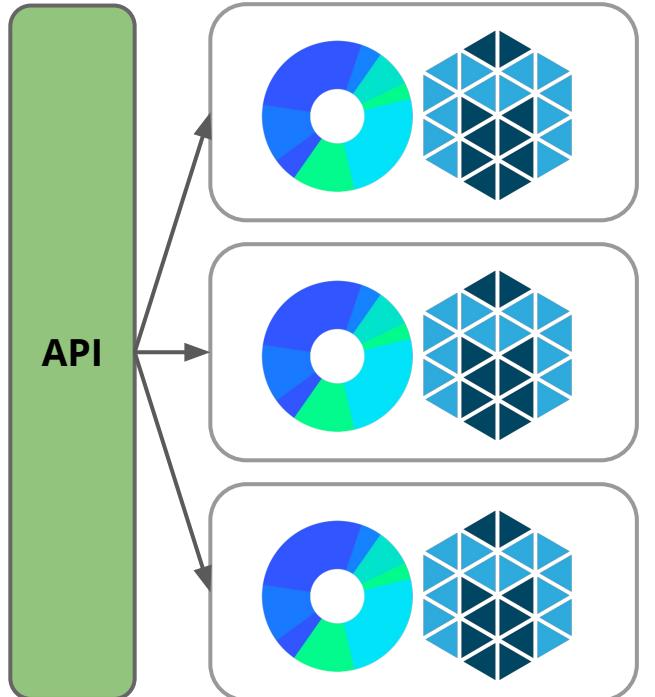


make deploy

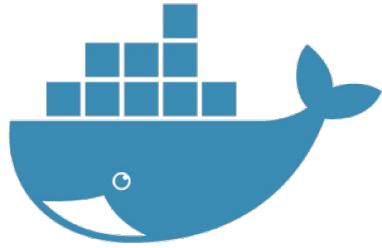
POST  
{

"id": "service-1",  
"cpus": 0.1,  
"mem": 10.0,  
"instances": 1

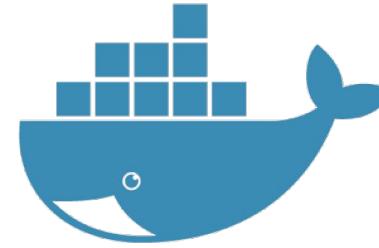
}



# Routing

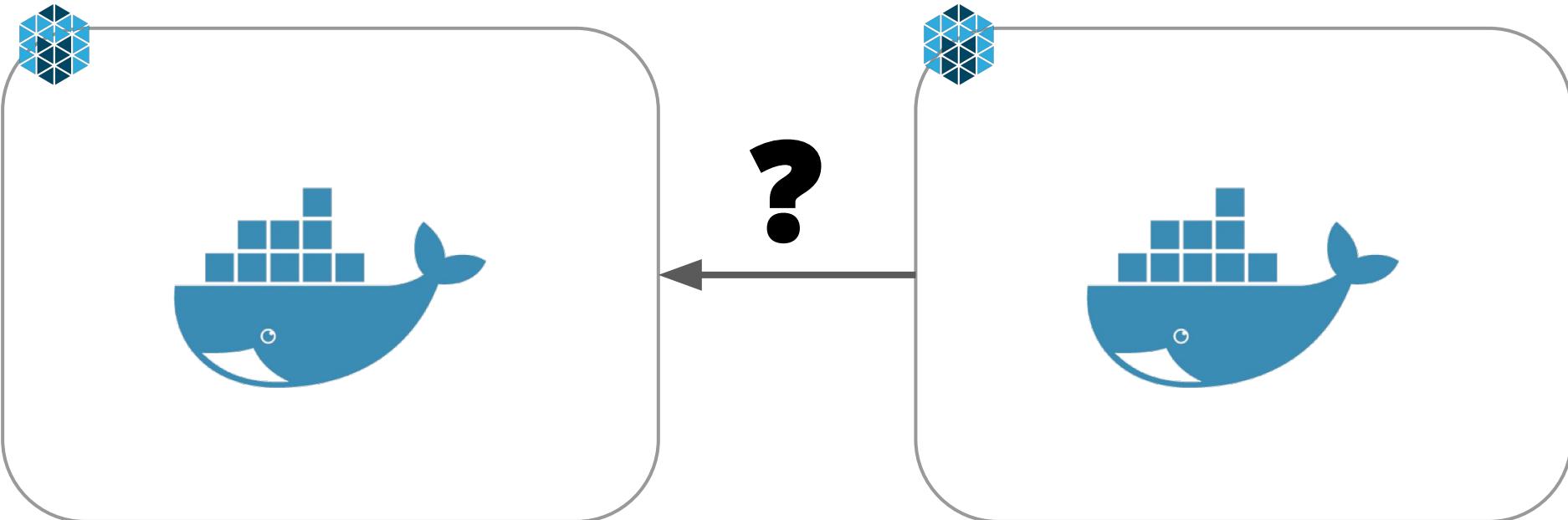


**service-1**  
**10.0.10.1**



**service-2**  
**10.0.10.2**

# Routing



**service-1**  
10.0.10.1

**service-2**  
10.0.10.2

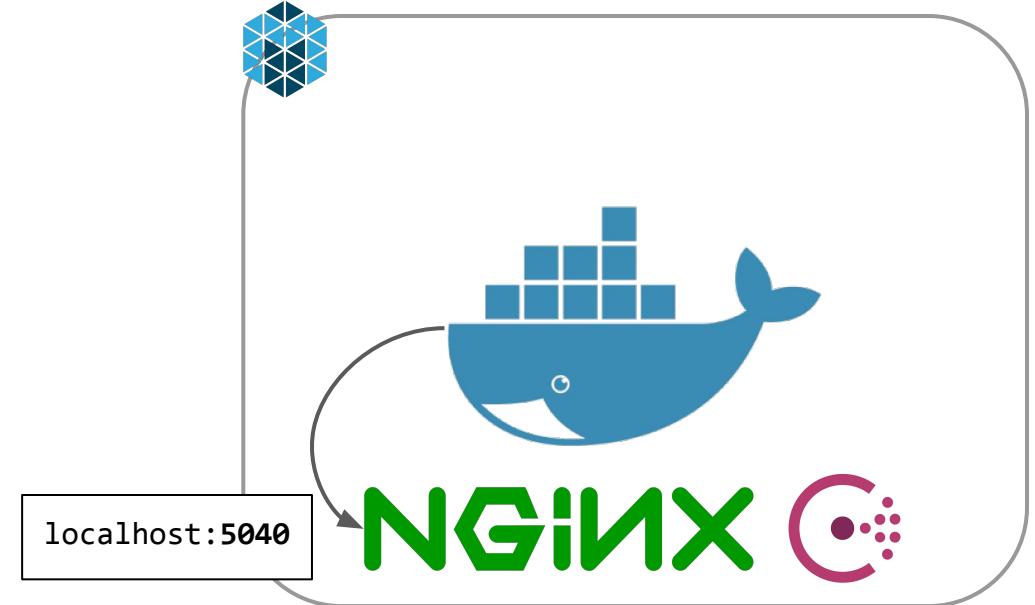
# Routing - Fat Middleware



**service-1**  
10.0.10.1

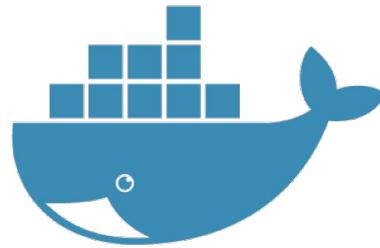


**service-2**  
10.0.10.2



```
curl http://localhost:5040/service/service-1/endpoint
```

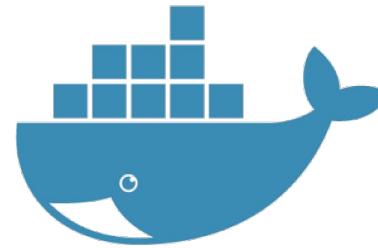
```
{
  upstream service-1 {
    server 10.0.10.1:5041;
    ....
  }
}
```



**NGINX** 

**service-1**  
**10.0.10.1**

```
curl https://10.0.10.1:5041/service/service-1/endpoint
```



**NGINX** 

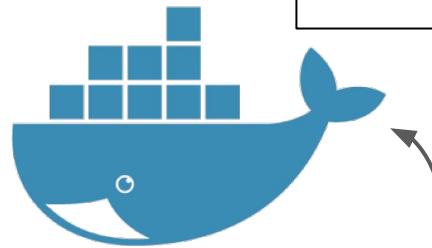
**service-2**  
**10.0.10.2**

```
{
    upstream service-1 {
        server 10.0.10.1:5041;
        ....
    }
}
```



**NGINX** 

**service-1**  
**10.0.10.1**



localhost:8080

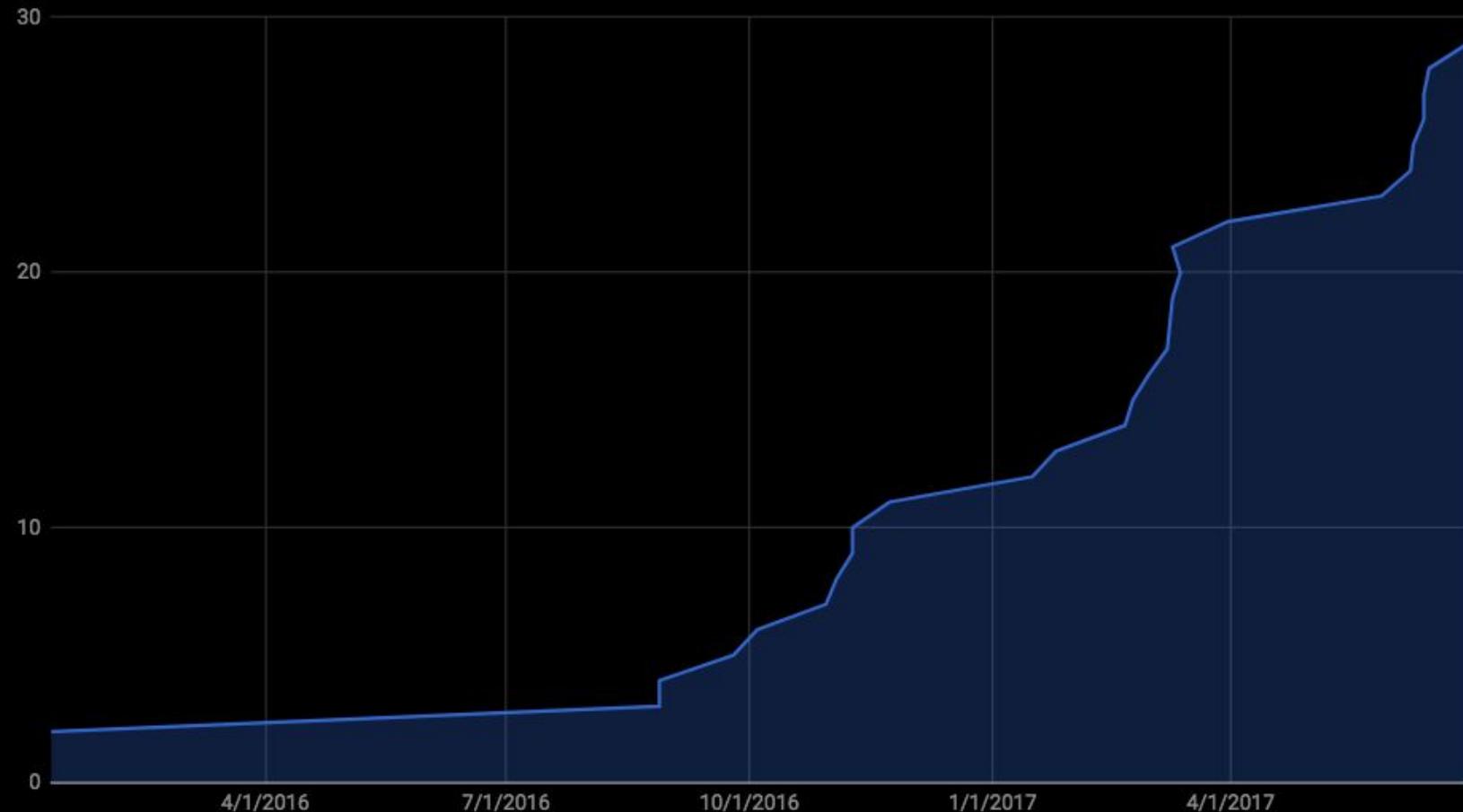


**NGINX** 

**service-2**  
**10.0.10.2**

```
{  
  upstream service-1 {  
    server 10.0.10.1:5041;  
    ....  
  }  
}
```









# Why Kubernetes?



Co-ops

High School Students

Development

[Go to Hootsuite.com](#)

Careers

## Migrating Container Orchestrators – Mesos, Kubernetes, or Nomad?

Hootsuite's recent transition from [Monolith to Microservices](#), like any large scale change, has met many challenging issues. As we move towards a SOA (Service Oriented Architecture), we need to build new infrastructure, tools, and pipelines. This new architecture requires our web applications to be made up of many small components, which can be done via [containerization](#). Containers then need to be orchestrated in order to truly run as a distributed system. An issue arose when our initial container orchestrator choice slowly became outclassed by other alternatives. Looking at where we wanted to be in the near future, we decided to migrate to a new platform. This blog will describe our technical decisions that drove this change.

### What is container orchestration/scheduling? Why should you care?

Consider a web developer who just finished building a simple PHP application. He has made sure every component functions properly on his localhost server. He uploads his code and assets to a host on the internet for the public, but is he guaranteed that everything will function the same as it did locally? The answer is 'no'. The system environment could be very different between his local server and the external internet server. There might be missing dependencies, or the operating system could be entirely different. This is where a containerization tool called [Docker](#) comes in. Docker solves the inconsistency issue by stuffing the *entire* environment, along with the web app, into a "Docker image". Any server with Docker installed can then run this image, regardless of the environment.

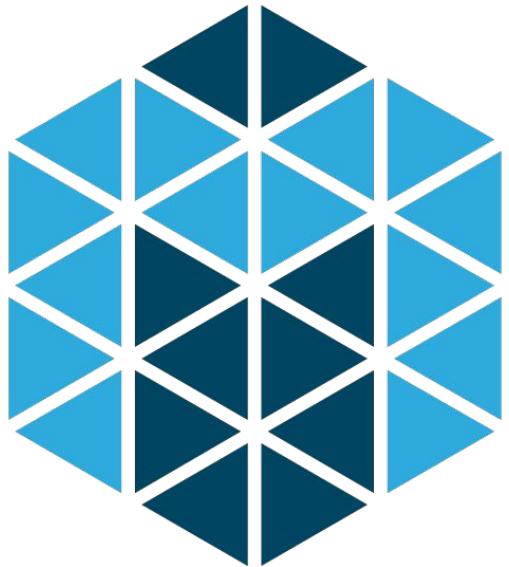
# 4 months

x





K8S DANCE!



# **Microservices on Mesos and Marathon**

- **Project Skeleton**
  - Golang or Scala
- **Pipeline as Code**
  - Jenkinsfile
  - Makefile
- **Docker images for packaging**
- **API on top of Marathon**
- **Dynamic service discovery**
- **Fat middleware using Consul and NGINX**

# Microservices on Kubernetes

- **Project Skeleton**
  - Golang or Scala
- **Pipeline as Code**
  - Jenkinsfile
  - Makefile
- **Docker images for packaging**
- ***API on top of Marathon***
- **Dynamic service discovery**
- **Fat middleware using Consul and NGINX**
- **Documentation for getting started**
- **./mesos2k8s**

# ./mesos2k8s

```
-> ./mesos2k8s
做人 🚶
mesos2k8s 🚶
▶ Skytrain Deployment Directory: deploy
▶ Kubernetes Deployment Directory: kubernetes
```



# Project Skeleton

```
#####
# Modify these vars

# Name of your service, lowercase, dash separated. ie. [a-z\-]
# !!! SHOULD NOT have "service" as prefix or suffix. We're trying to get rid of that.
# Will be used for Skyline and Kubernetes names as well as statsd prefixes, etc.
#name: my-awesome-service
name: new-service

# Human readable title. Used for docs and such
#nameNice: My Awesome Service
nameNice: My New Service

# Long form description
#description: Skyline sample service generated from the skeleton based off of Play! framework, dockerized, and
description: Hello World

#maintainers:
# - name: Firstname Lastname
#   email: firstname.lastname@hootsuite.com
# - name: Joe Blough
#   email: joe.blough@hootsuite.com
maintainers:
- name: Anubhav Mishra
  email: anubhav.mishra@hootsuite.com

# Pick a project type. Can either be: scala, go, go-grpc, idl etc.
# This can be set with the command line arg `--project-type` which will allow you to
# create multiple project types with the same config. This is useful for projects like `go-grpc` + `idl`
# projectType: go
projectType: go
```

```
. Dockerfile
  Dockerfile-dev
  Jenkinsfile
  Makefile
  Makefile.base.mk
  README.md
  bin
    coverage.sh
    tdd.sh
  conf
    about.json
    default.yml
    dev.yml
    local.yml
    production.yml
    staging.yml
  docs
    CONTRIBUTING.md
    todo
  index.go
  kubernetes
    application.yml
    dev.yml
    minikube.yml
    production.yml
    staging.yml
  main.go
  main_test.go
  middleware.go
  policy.hcl
  server.go
  skeleton-vars.yml
  status.go
  status_test.go
```

# Deployment Files

- `make deploy-k8s-dev`
- `make deploy-k8s-staging`
- `make deploy-k8s-production`



```
.           Dockerfile
           Dockerfile-dev
           Jenkinsfile
           Makefile
           Makefile.base.mk
           README.md
           bin
             coverage.sh
             tdd.sh
           conf
             about.json
             default.yml
             dev.yml
             local.yml
             production.yml
             staging.yml
           docs
             CONTRIBUTING.md
             todo
             index.go
           kubernetes
             application.yml
             dev.yml
             minikube.yml
             production.yml
             staging.yml
             main.go
             main_test.go
             middleware.go
             policy.hcl
             server.go
             skeleton-vars.yml
             status.go
             status_test.go
```

# Pipeline as Code

Jenkins Jenkins Open Blue Ocean search

Production Delivery (github organization) mishra-go-skeleton master

Up Status Changes Build Now View Configuration GitHub Full Stage View Job Config History GitHub Pipeline Syntax

Recent Changes

## Branch master

Full project name: production-delivery (github organization)/mishra-go-skeleton/master

### Stage View

Average stage times:

setup	unit test	build service	push	deploy dev	deploy staging	deploy production
2s	8s	2s	3s	1s	3s	3s
2s	1s	1s	2s	1s	3s	12s (paused for 10s)
4s	19s	3s	6s	1s	3s	1s (paused for 10s) aborted
4s	23s	3s	6s	1s	3s	1s (paused for 6s) aborted
4s	1s	1s	2s	1s	3s	1s (paused for 4s) aborted

Build History trend →

- 8-161010b Sep 16 18:30 No Changes
- 7-161010b Sep 16 18:30 No Changes
- 6-161010b Sep 16 18:28 No Changes
- 5-161010b Sep 16 15:34 1 commits
- 4-56c3eb7 Sep 16 2:54 PM
- 3-5008c81

# Pipeline as Code

Jenkins    Open Blue Ocean    search    ?

Jenkins    Production Delivery (github organization)    mishra-go-skeleton    master

Up    Status    Changes    Build Now    View Configuration    GitHub    Full Stage View    Job Config History    GitHub    Pipeline Syntax

Recent Changes

Branch master

Full project name: production-delivery /mishra-go-skeleton

Stage View

push    dev    deploy staging    deploy production

3s    3s

3s    12s (paused for 10s)

3s    1s (paused for 10s) aborted

1s    3s    1s (paused for 10s) aborted

6s    1s    2s    1s    3s    1s (paused for 10s) aborted

4s    1s    1s    2s    1s    3s    1s (paused for 10s) aborted

8-161010b    Sep 16 18:30

7-161010b    Sep 16 18:30 No Changes

6-161010b    Sep 16 18:28 No Changes

5-161010b    Sep 16 18:28

4-56c3eb7    Sep 16 15:34 1 commits

3-5008c81

Build History

trend

find

8-161010b Sep 16, 2016 6:30 PM

7-161010b Sep 16, 2016 6:30 PM

6-161010b Sep 16, 2016 6:28 PM

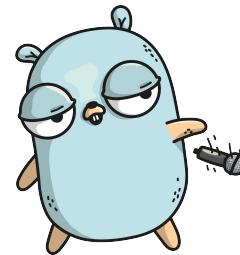
5-161010b Sep 16, 2016 3:34 PM

4-56c3eb7 Sep 16, 2016 2:54 PM 1 commits

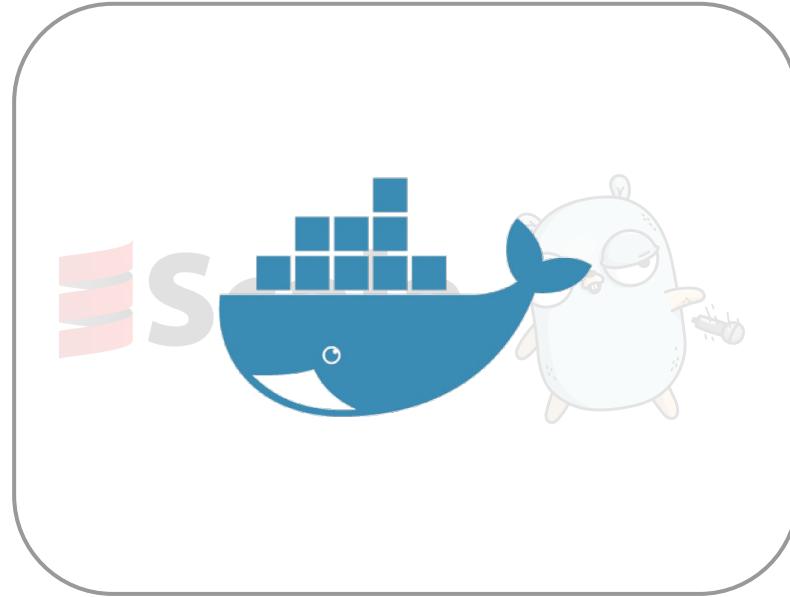
3-5008c81

# Packaging

 Scala



# Packaging



# Routing in Mesos

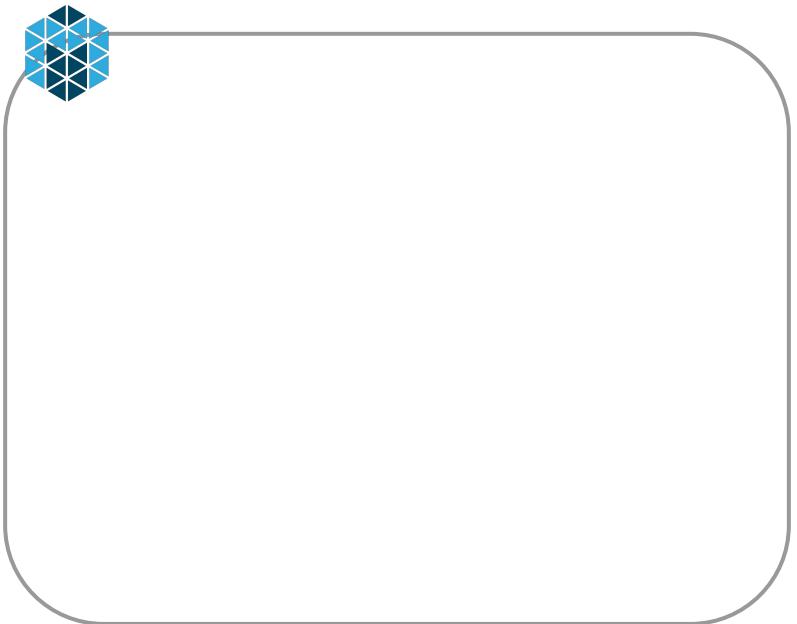


**service-1**  
10.0.10.1



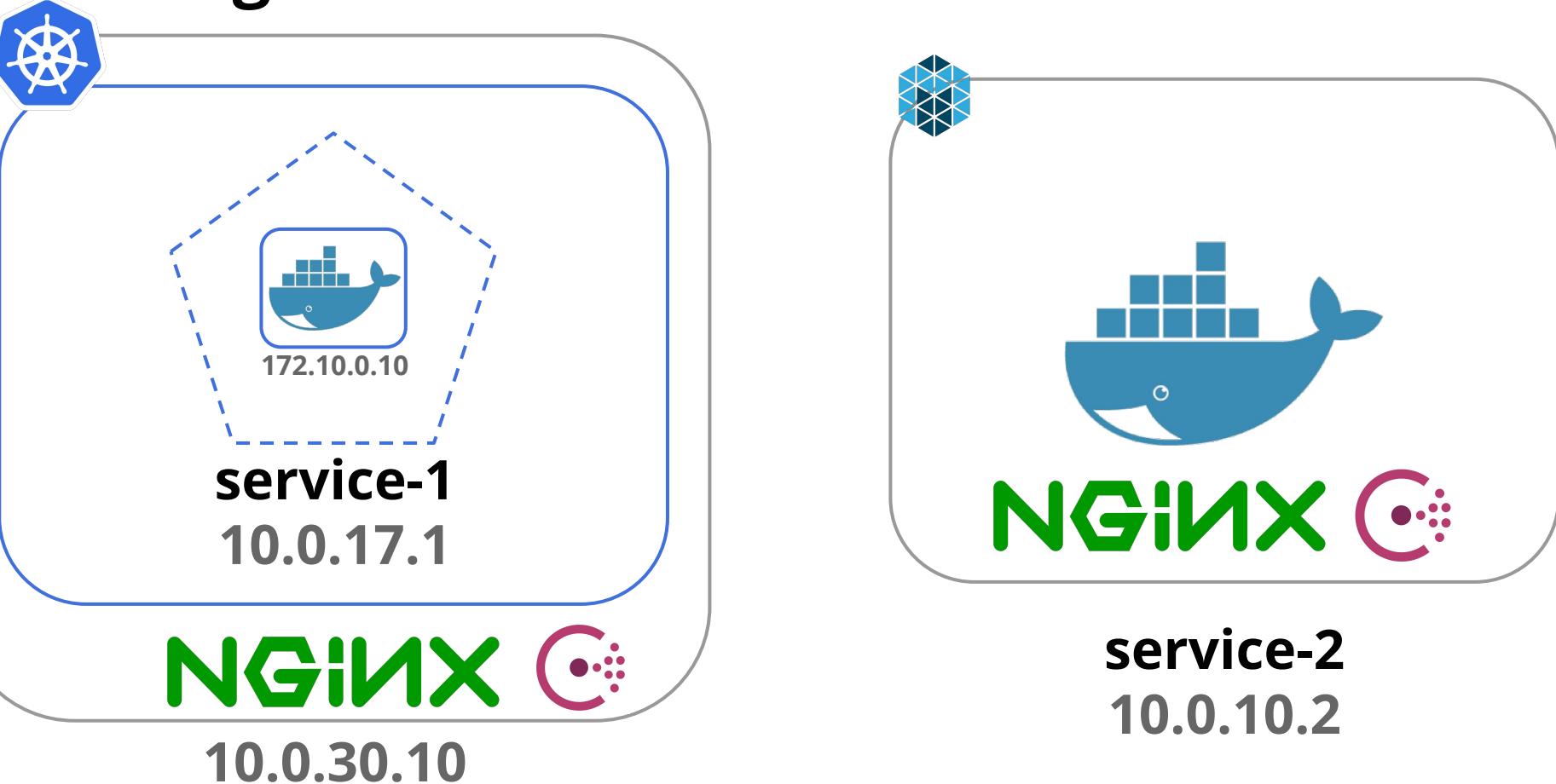
**service-2**  
10.0.10.2

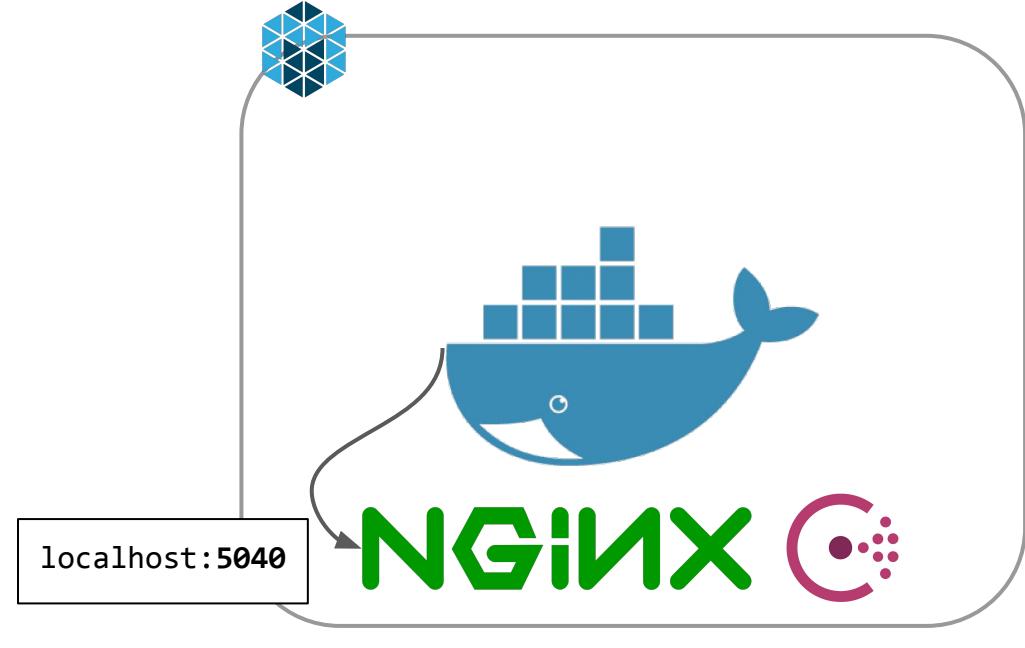
# Routing in Mesos



**service-2**  
**10.0.10.2**

# Routing to K8s





```
curl http://localhost:5040/service/service-1/endpoint
```

```
{  
    upstream service-1 {  
    }  
    upstream bridge-1 {  
        server 10.0.20.1:5041;  
        ....  
    }  
}
```



**bridge-1**  
(multi-dc aware)





## bridge-1 (multi-dc aware)

```
curl https://10.0.20.1:5041/service/service-1/endpoint
```



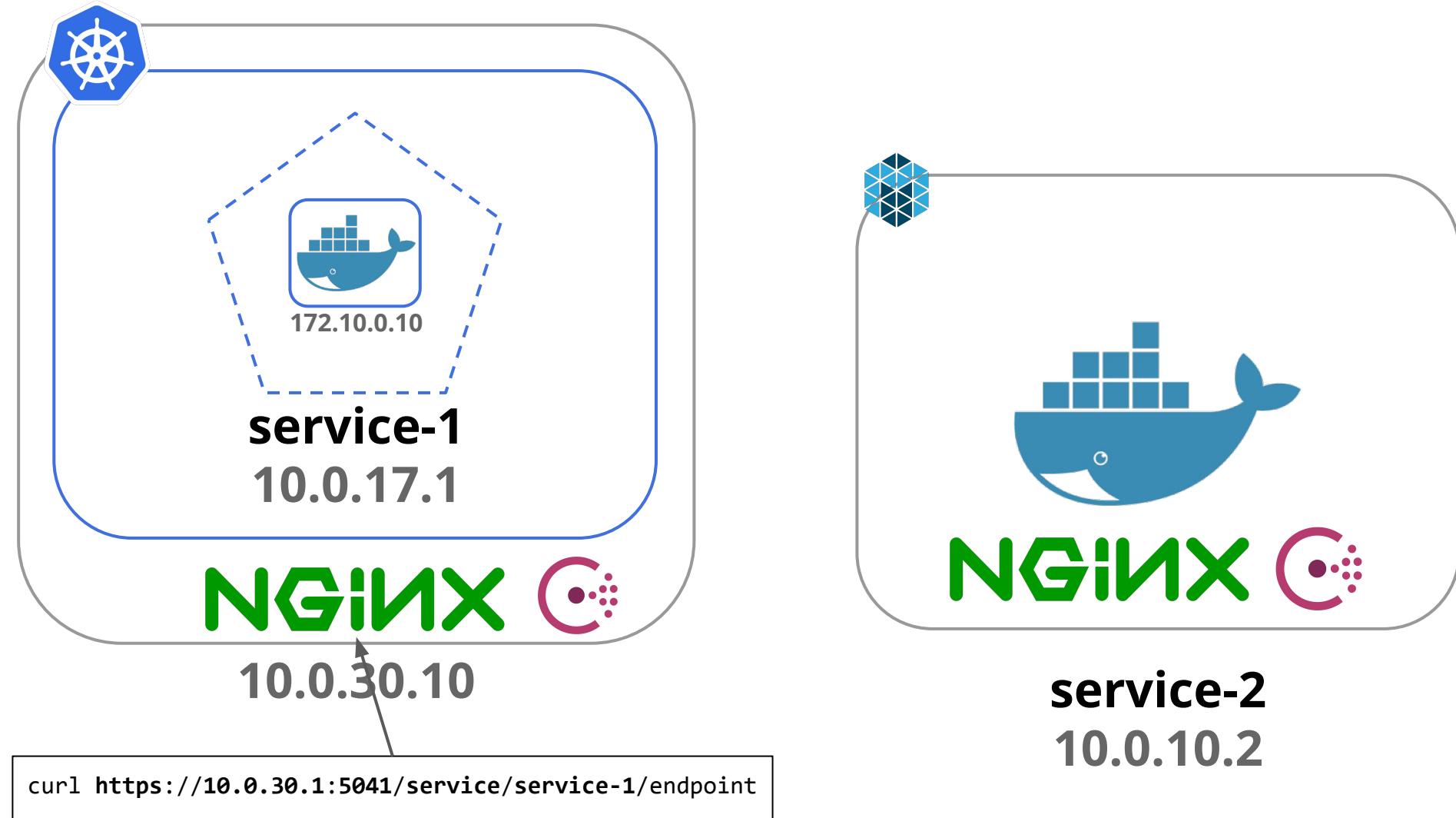


**NGINX** 

## **bridge-1** **(multi-dc aware)**

```
curl https://10.0.30.1:5041/service/service-1/endpoint
```







`http://service-1.default.svc.cluster.local:8080`



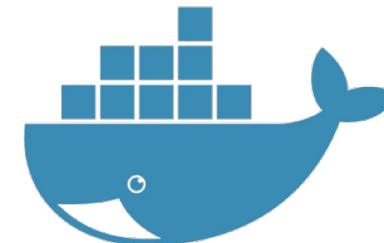
`172.10.0.10`

**service-1**  
`10.0.17.1`

**NGINX**

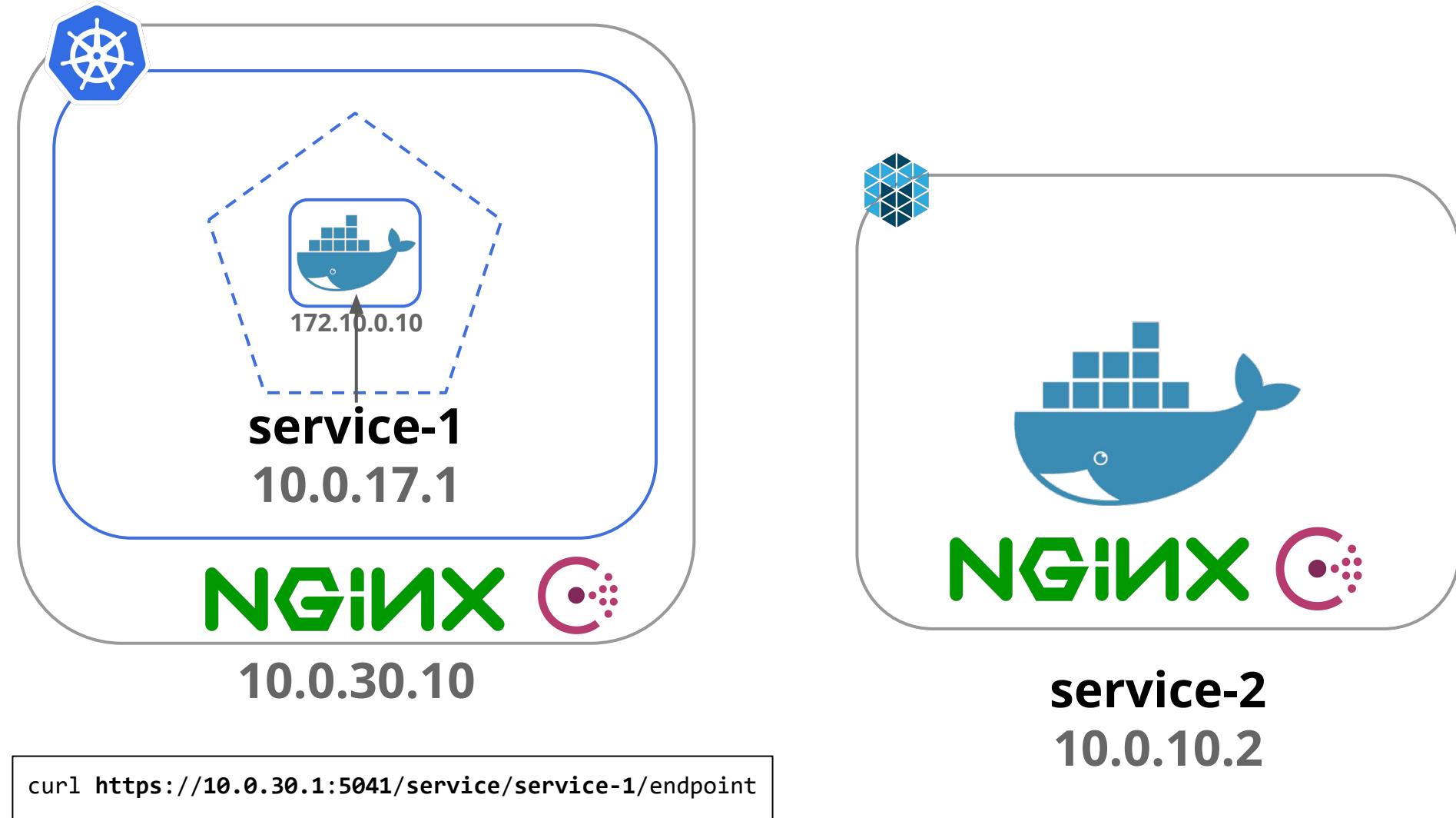
`10.0.30.10`

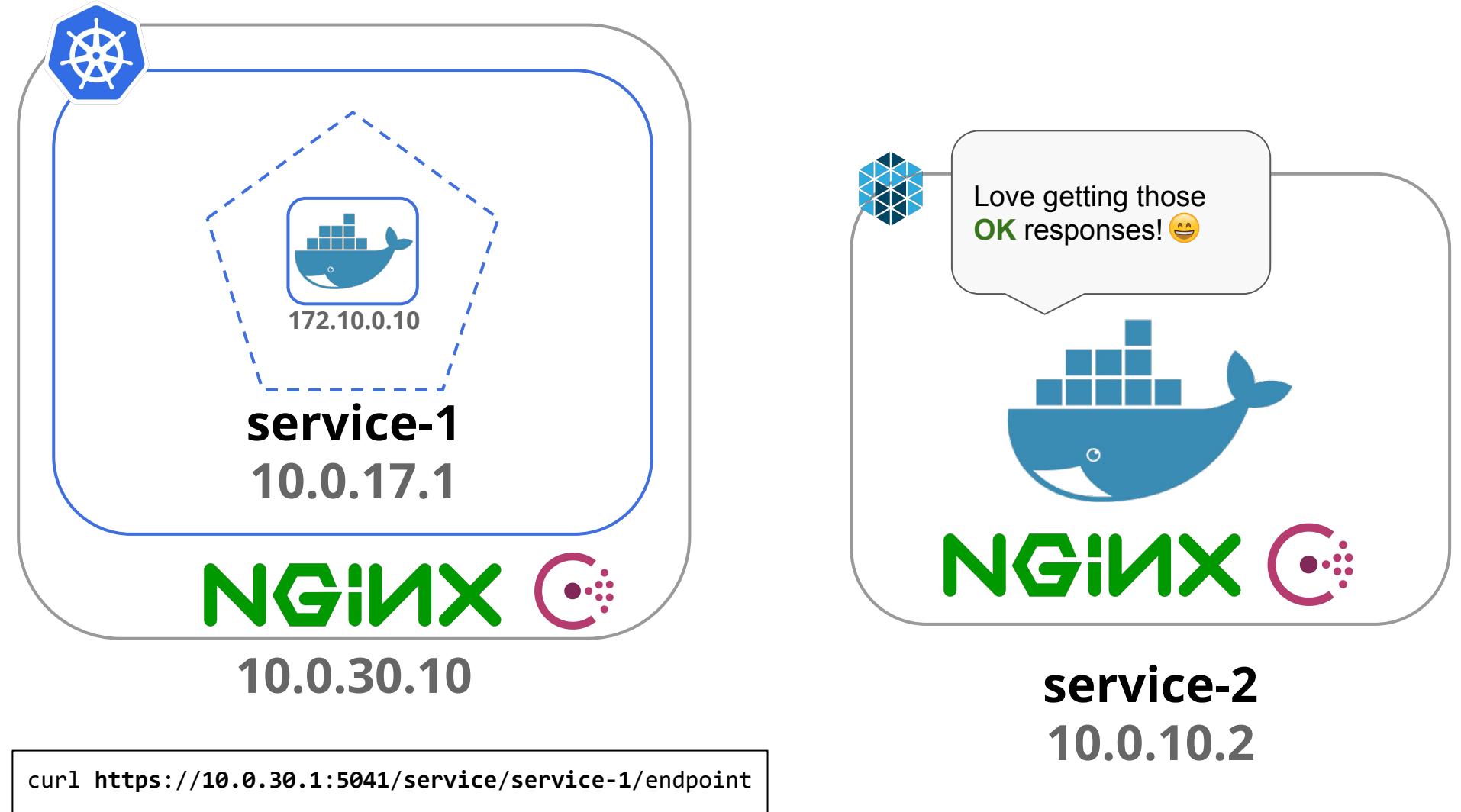
`curl https://10.0.30.1:5041/service/service-1/endpoint`



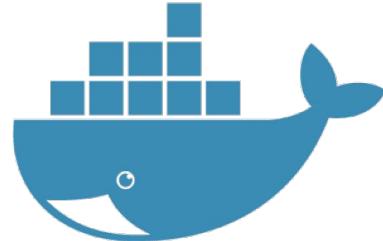
**NGINX**

**service-2**  
`10.0.10.2`



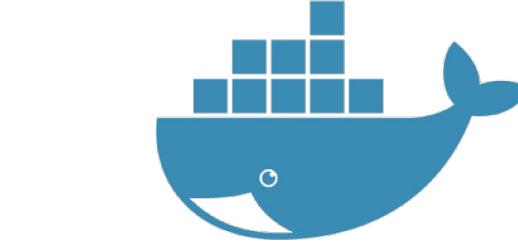


# Rollback



**NGINX**

**service-1**  
**10.0.10.1**



**NGINX**

**service-2**

```
{  
    upstream service-1 {  
        server 10.0.10.1:5041;  
    }  
    upstream bridge-1 {  
        server 10.0.20.1:5041;  
        ....  
    }  
}
```



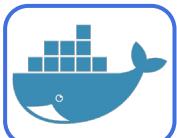
**service-1**  
172.10.0.10

**NGINX**

**10.0.30.10**



**foo**  
**10.0.10.4**



**service-1**  
172.10.0.10

**foo**  
10.0.17.100

**NGINX**

**10.0.30.10**



**foo**  
**10.0.10.4**

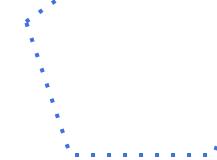
```
apiVersion: v1
kind: Service
metadata:
  name: foo
  labels:
    app: foo
spec:
  ports:
    - port: 5040
      protocol: TCP
      name: http
  selector:
    app: nginx-skyline-router
```



```
curl http://foo:5040/endpoint
```



**service-1**  
172.10.0.10



**foo**  
10.0.17.100

**NGINX**

**10.0.30.10**



**foo**  
**10.0.10.4**

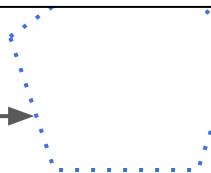


```
curl http://foo:5040/endpoint
```



**service-1**  
172.10.0.10

**foo**  
10.0.17.100



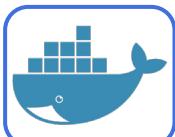
**foo**  
**10.0.10.4**



**10.0.30.10**



```
curl http://foo:5040/endpoint
```



**service-1**  
172.10.0.10

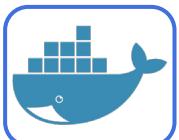
**foo**  
10.0.17.100

**NGINX** A green "NGINX" logo with a magenta circular icon containing dots.

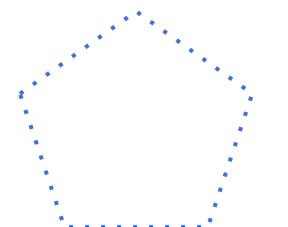
**10.0.30.10**



**foo**  
**10.0.10.4**



**service-1**  
172.10.0.10



**foo**

10.0.17.100

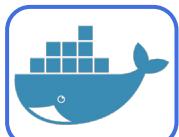


10.0.30.10

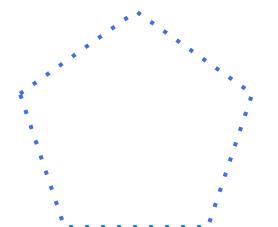


**foo**  
10.0.10.4

```
# match on:  
# service.namespace.svc.cluster.local  
# service.namespace  
# service  
server_name REGEX  
....  
location / {  
    rewrite ^/(.*)$ /service/$service/$1 break;  
    proxy_pass https://egress_bridge;  
}
```



**service-1**  
172.10.0.10



**foo**  
10.0.17.100

**NGINX**

**10.0.30.10**

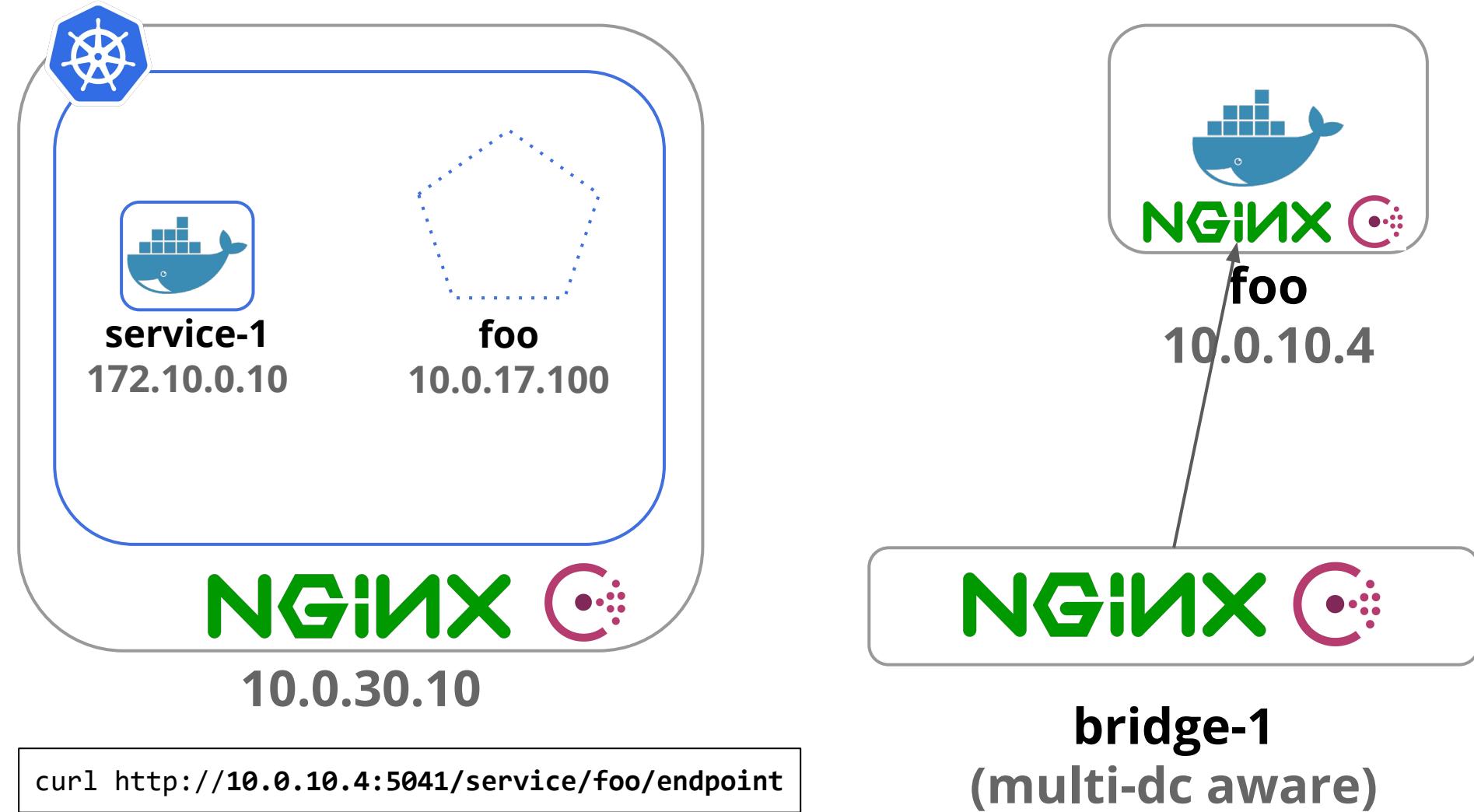
```
curl http://bridge1:5041/service/foo/endpoint
```



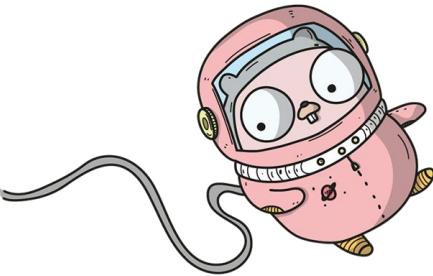
**foo**  
**10.0.10.4**

**NGINX**

**bridge-1**  
**(multi-dc aware)**



# Ship it!



**This branch has no conflicts with the base branch**

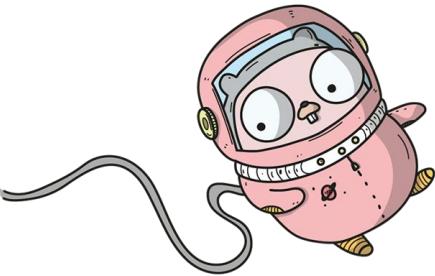
Merging can be performed automatically.

**Merge pull request**

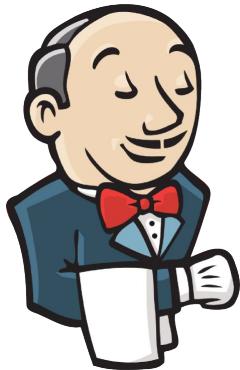
You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

`./project-generator`

# Microservice Pipeline



`./project-generator`



**Pipeline as Code**



**Kubernetes**

# Documentation

## Migrating from Mesos



Kubernetes

Getting Started

Minikube

Creating a New Service

Calling Other Services

Calling Your Service

Darklaunching

Encrypting Config

Health Checks

Jenkins

Logging

Sensu and PagerDuty

Managing an Existing Service

Migrating from Mesos

## Migrating from Mesos

#

### ⚠ Talk To Us First

TALK TO US IN #KUBERNETES AND WE'LL BOOK A MEETING TO RUN THROUGH THIS WITH YOU. DON'T DO IT YOURSELF

Let's see how we can migrate from a project or service running in Mesos to Kubernetes

This guide will go through the following steps to move your service running in Mesos to Kubernetes:

- › Change your config files: [Makefile](#), [Jenkinsfile](#)
- › Generate Kubernetes deployment config files
- › **Code Change:** Modify the urls you're using to call other services
- › Allow access to non-Mesos services calling your service

### Makefile

#

In the [Makefile.base.mk](#) we will add the following things:

- › Names of the Kubernetes clusters

# Live Demo





# Things fail 😞, Let's talk about it....

- “The bad config outage”
- “The classic security group fail”

**502 Bad Gateway**

---

hootsuite/1.6.0

# Lessons Learned/Conclusion

Choose the least important service

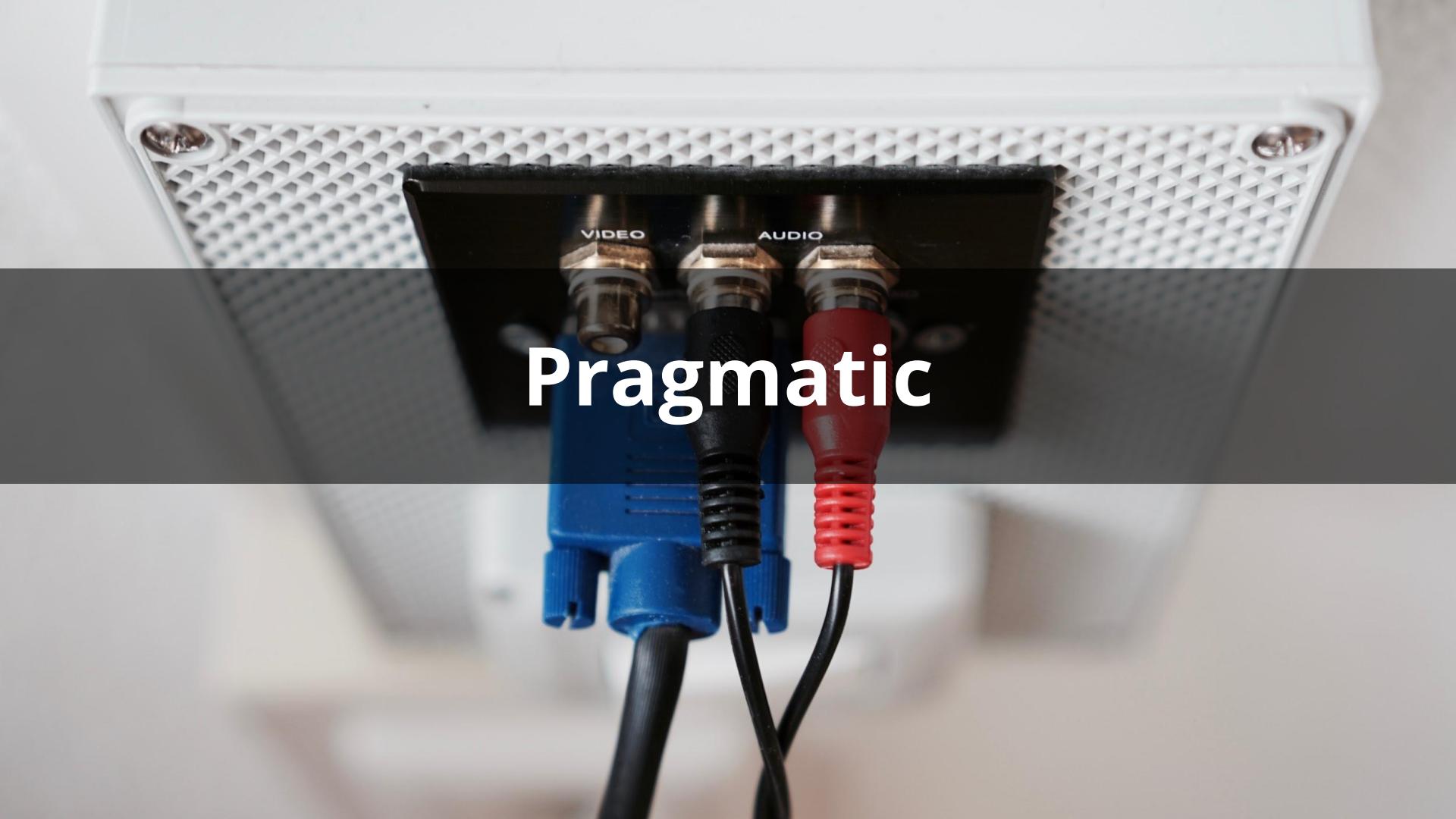


Have a rollback plan

# Write down what your deployment pipeline looks like



# Documentation should be written for humans to read



Pragmatic

A photograph of four people in an office environment. On the left, a woman with dark hair and a black t-shirt is smiling broadly. In the center, a man with dark hair and a beard, wearing a white and grey striped t-shirt, is looking towards the camera. To his right, another woman with long dark hair and a grey textured top is laughing heartily. On the far right, a man with glasses and a dark t-shirt is looking directly at the camera with a neutral expression. The background shows office equipment and fluorescent lighting on a concrete ceiling.

Minimizing disruption = Great  
Adoption

# Links

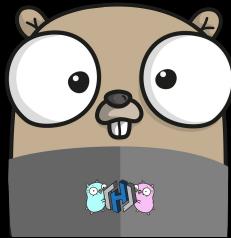
- Migrating Container Schedulers:

<http://code.hootsuite.com/migrating-container-orchestrators-mesos-kubernetes-nomad/>

- Abstracting Marathon Deployment Details from Microservices:

<http://code.hootsuite.com/abstracting-marathon-deployment-details-from-microservices/>

# Thank you!



Anubhav Mishra

@anubhavm