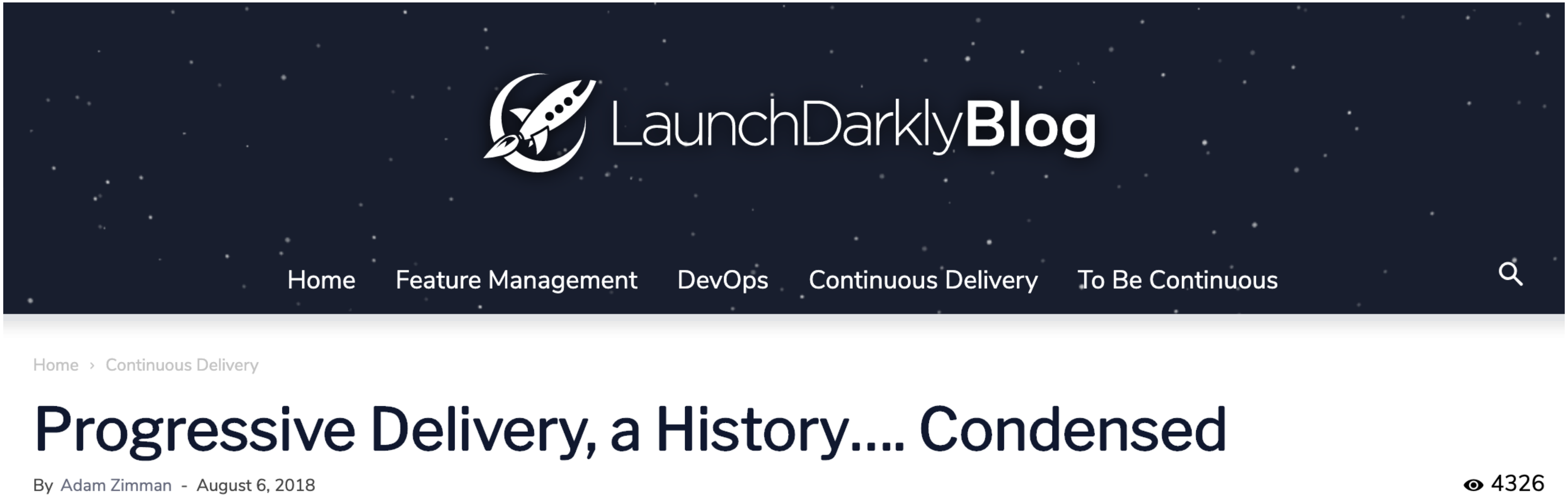
PROGRESSIVE DELIVERY

CONTINUOUS DELIVERY THE RIGHT WAY

Carlos Sanchez / csanchez.org / @csanchez

Adobe

PROGRESSIVE DELIVERY

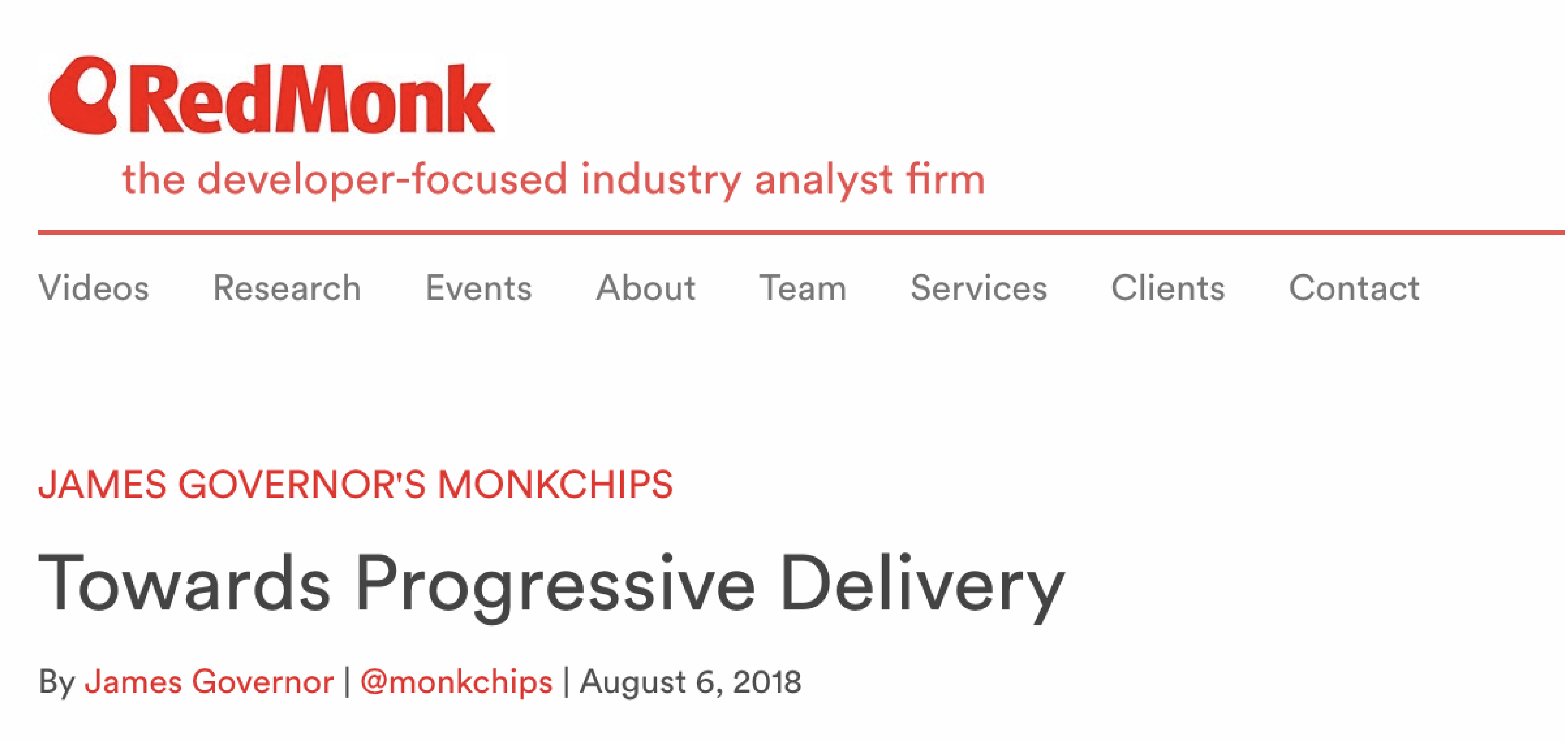


Home › Continuous Delivery

Progressive Delivery, a History…. Condensed

By Adam Zimman - August 6, 2018

4326



RedMonk

the developer-focused industry analyst firm

Videos Research Events About Team Services Clients Contact

JAMES GOVERNOR'S MONKCHIPS

Towards Progressive Delivery

By James Governor | @monkchips | August 6, 2018

Progressive Delivery is a term that includes deployment strategies that try to avoid the pitfalls of all-or-nothing deployment strategies

New versions being deployed do not replace existing versions but run in parallel for an amount of time receiving live production traffic, and are evaluated in terms of correctness and performance before the rollout is considered successful.

Continuous Delivery is hard

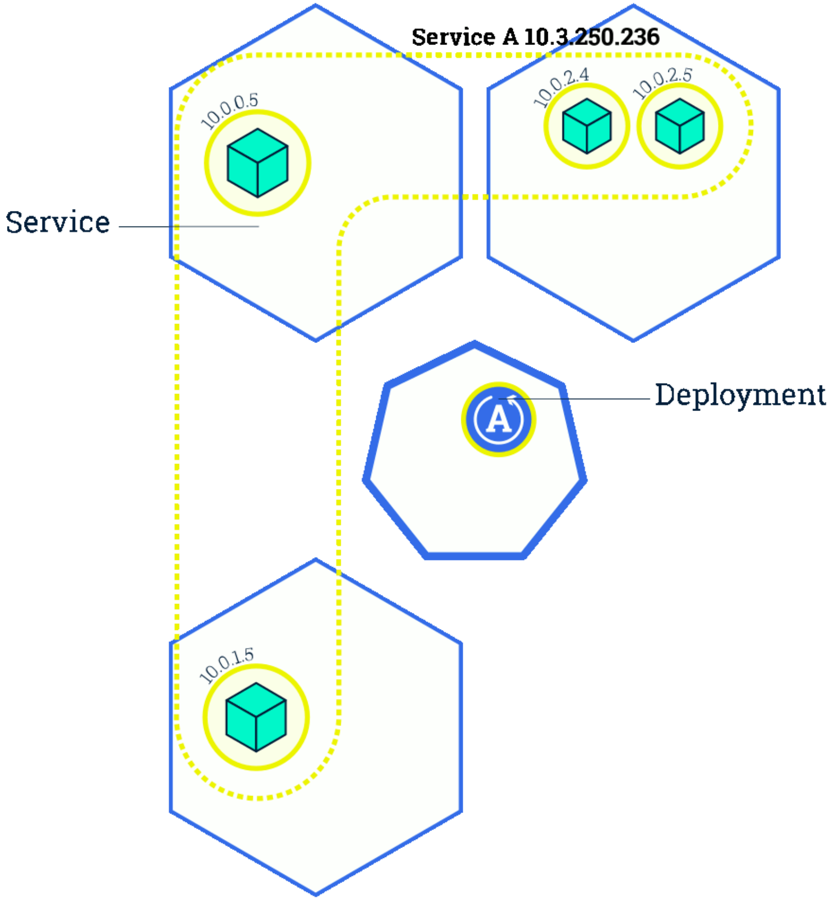
Progressive Delivery makes Continuous Delivery easier to adopt

reduces the risk associated with Continuous Delivery

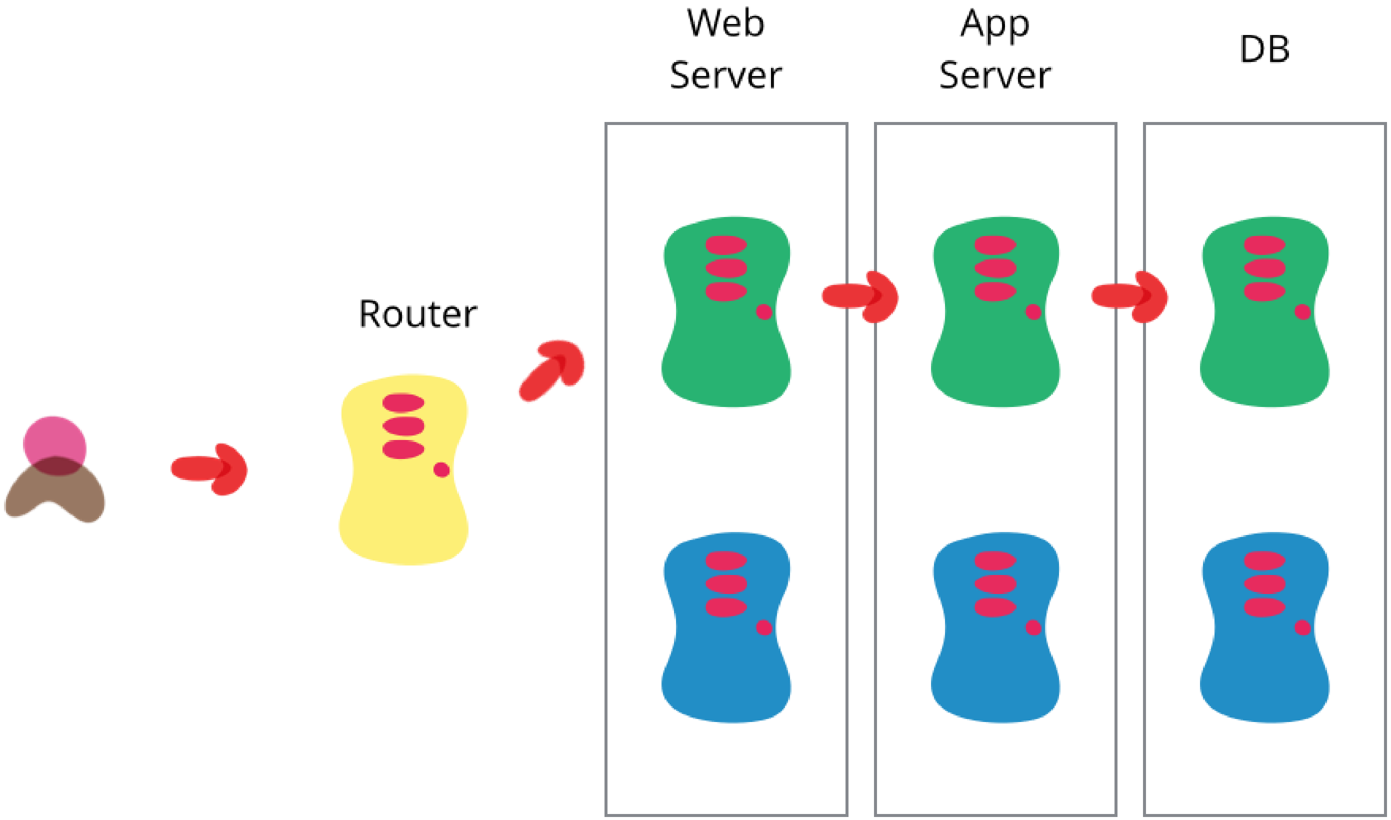
- Avoiding downtime
- Limit the blast radius
- Shorter time from idea to production

PROGRESSIVE DELIVERY TECHNIQUES

ROLLING UPDATES

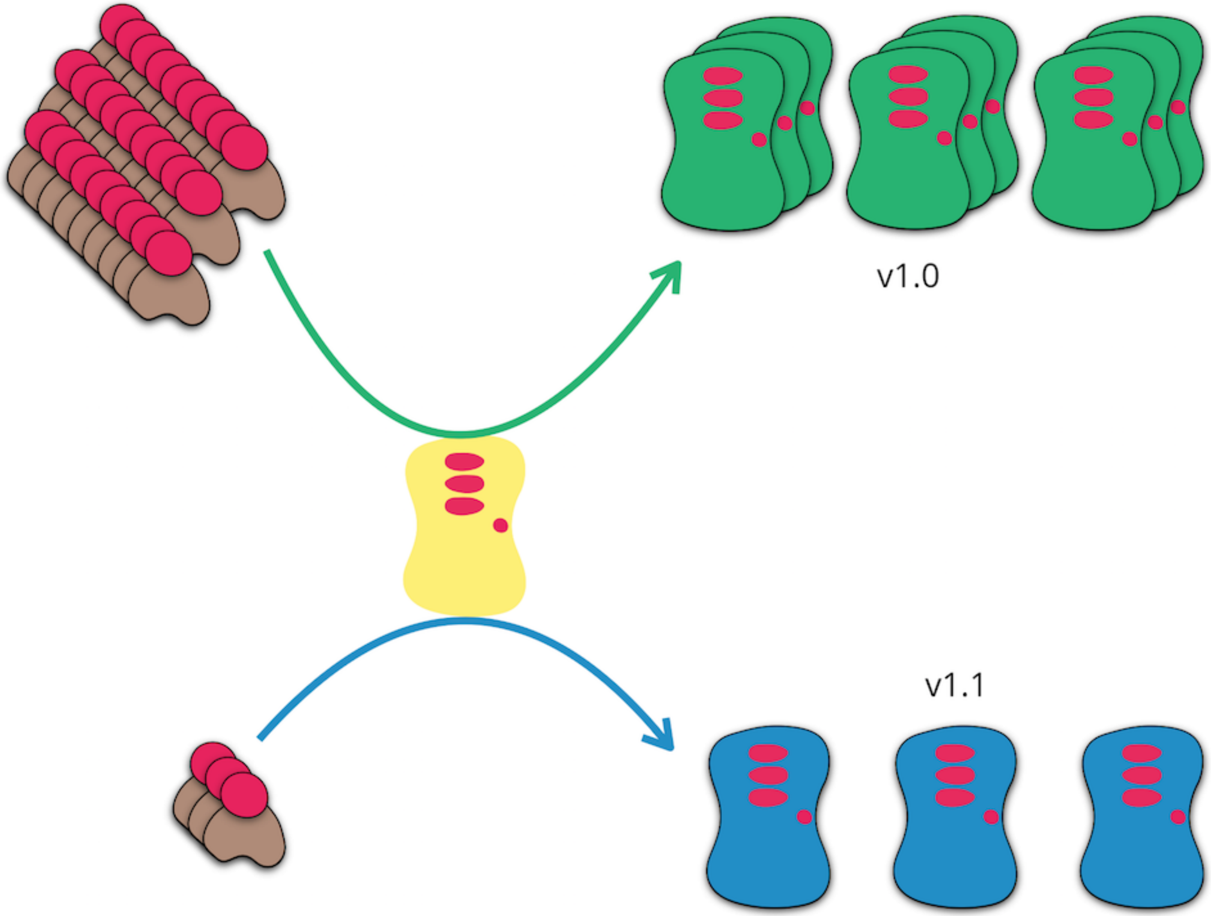


BLUE-GREEN DEPLOYMENT



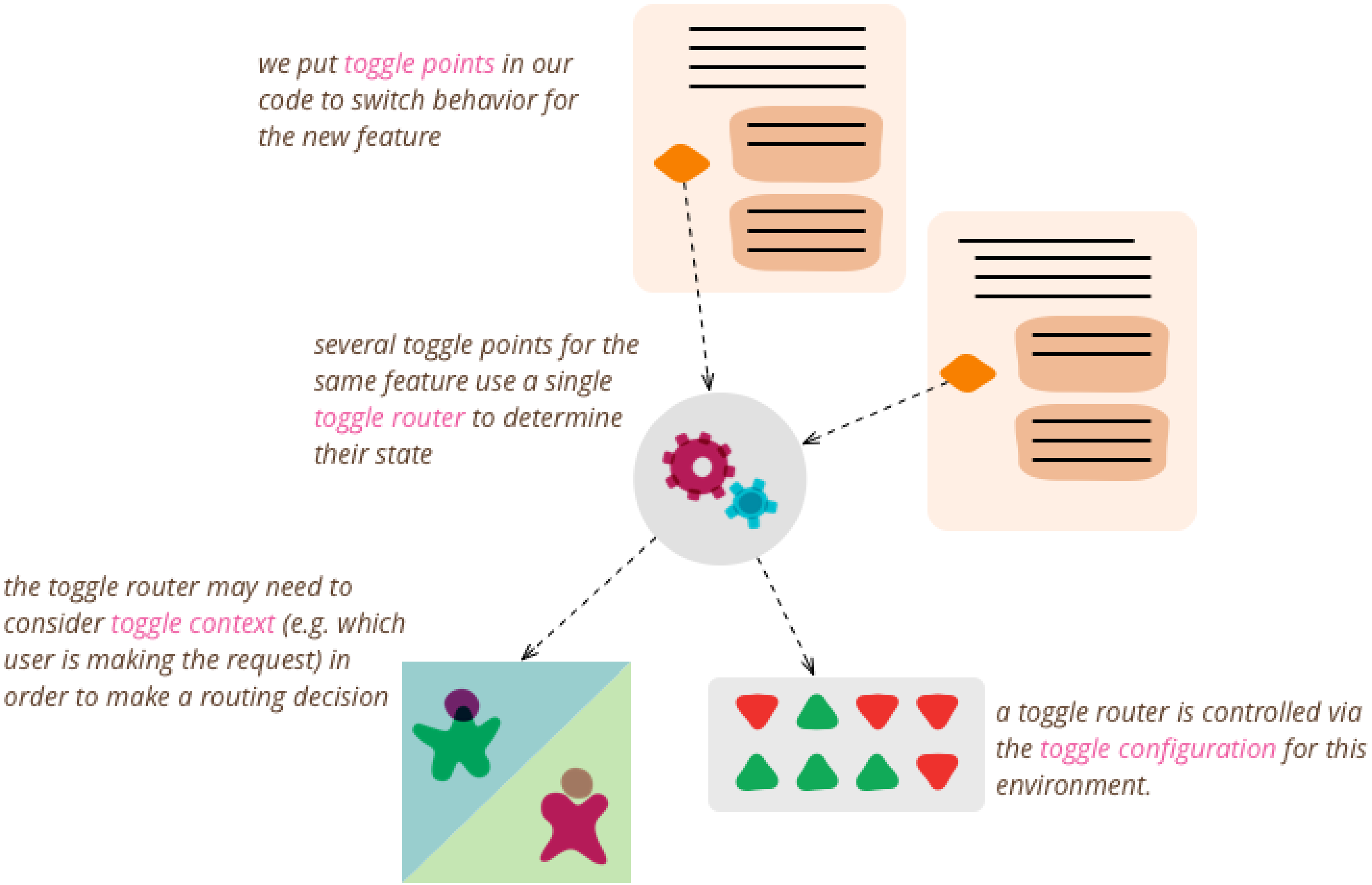
medium.com/continuous-deployment/continuous-deployment-strategies-32e2f7badd2

CANARY DEPLOYMENT



medium.com/continuous-deployment/continuous-deployment-strategies-32e2f7badd2

FEATURE FLAGS



we put toggle points in our code to switch behavior for the new feature

several toggle points for the same feature use a single toggle router to determine their state

the toggle router may need to consider toggle context (e.g. which user is making the request) in order to make a routing decision

a toggle router is controlled via the toggle configuration for this environment.

Martin Fowler martinfowler.com/articles/feature-toggles.html

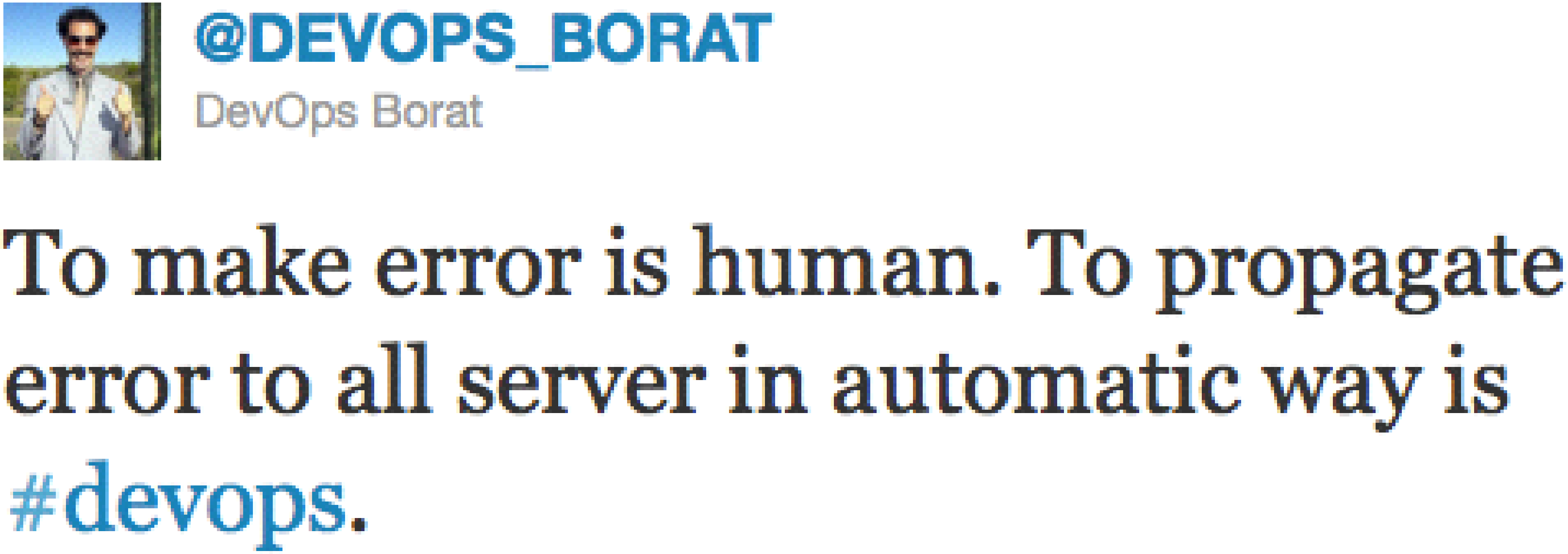
tex

MONITORING IS THE NEW TESTING

Know when users are experiencing issues **in production**

React to the issues **automatically**

Progressive Delivery requires a good amount of metrics



@DEVOPS_BORAT

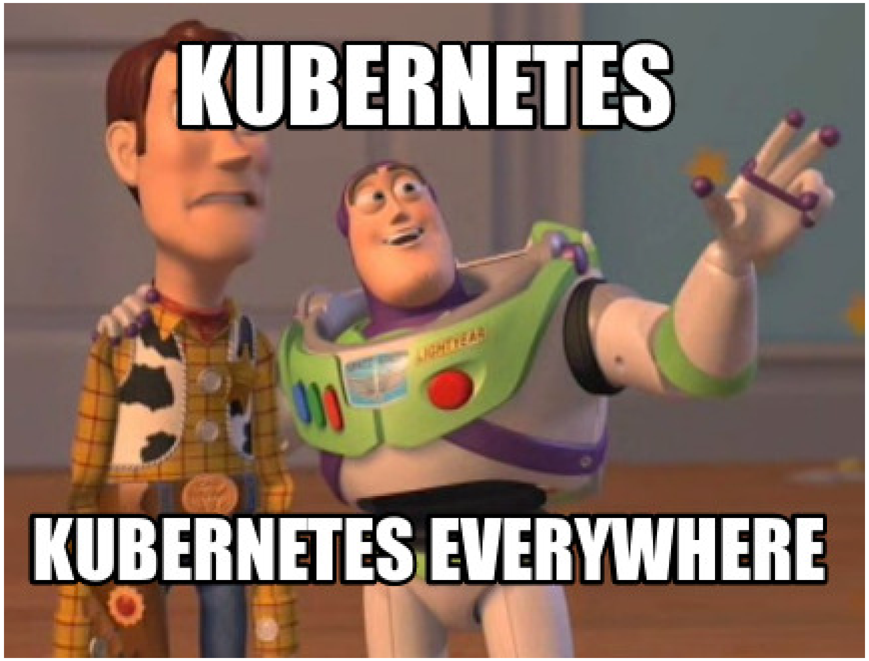
DevOps Borat

To make error is human. To propagate error to all server in automatic way is #devops.

> *If you haven't automatically destroyed something by mistake, you are not automating enough*

JENKINS X

kubernetes


KUBERNETES
KUBERNETES EVERYWHERE

docker
SKAFFOLD
DRAFT
HELM
JENKINS X
Nexus
CHARTMUSEUM
prow

TEKTON

Pipeline engine in Kubernetes

Uses Pods and containers to run the pipeline steps

Implements ChatOps

Handles GitHub webhooks

Package manager for Kubernetes

SKAFFOLD

Build Docker images with multiple backends:

- Docker build
- Kaniko
- Google Cloud Build
- Jib (Maven/Gradle)

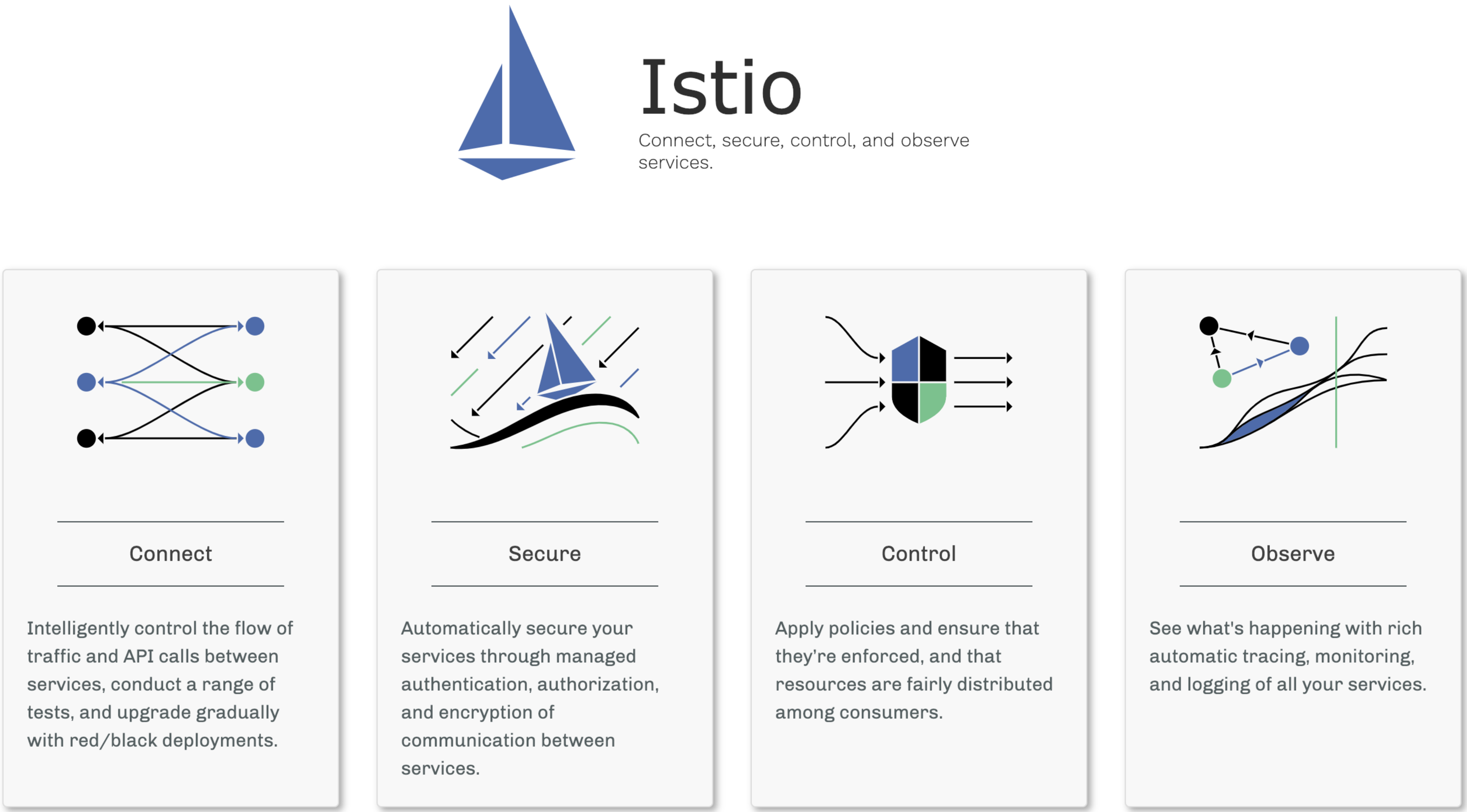
DRAFT

Generates Dockerfile and Helm charts for your project

PROGRESSIVE DELIVERY

WITH JENKINS X

jenkins-x.io/docs/managing-jx/tutorials/progressive-delivery



Istio

Connect, secure, control, and observe services.

Connect

Intelligently control the flow of traffic and API calls between services, conduct a range of tests, and upgrade gradually with red/black deployments.

Secure

Automatically secure your services through managed authentication, authorization, and encryption of communication between services.

Control

Apply policies and ensure that they're enforced, and that resources are fairly distributed among consumers.

Observe

See what's happening with rich automatic tracing, monitoring, and logging of all your services.

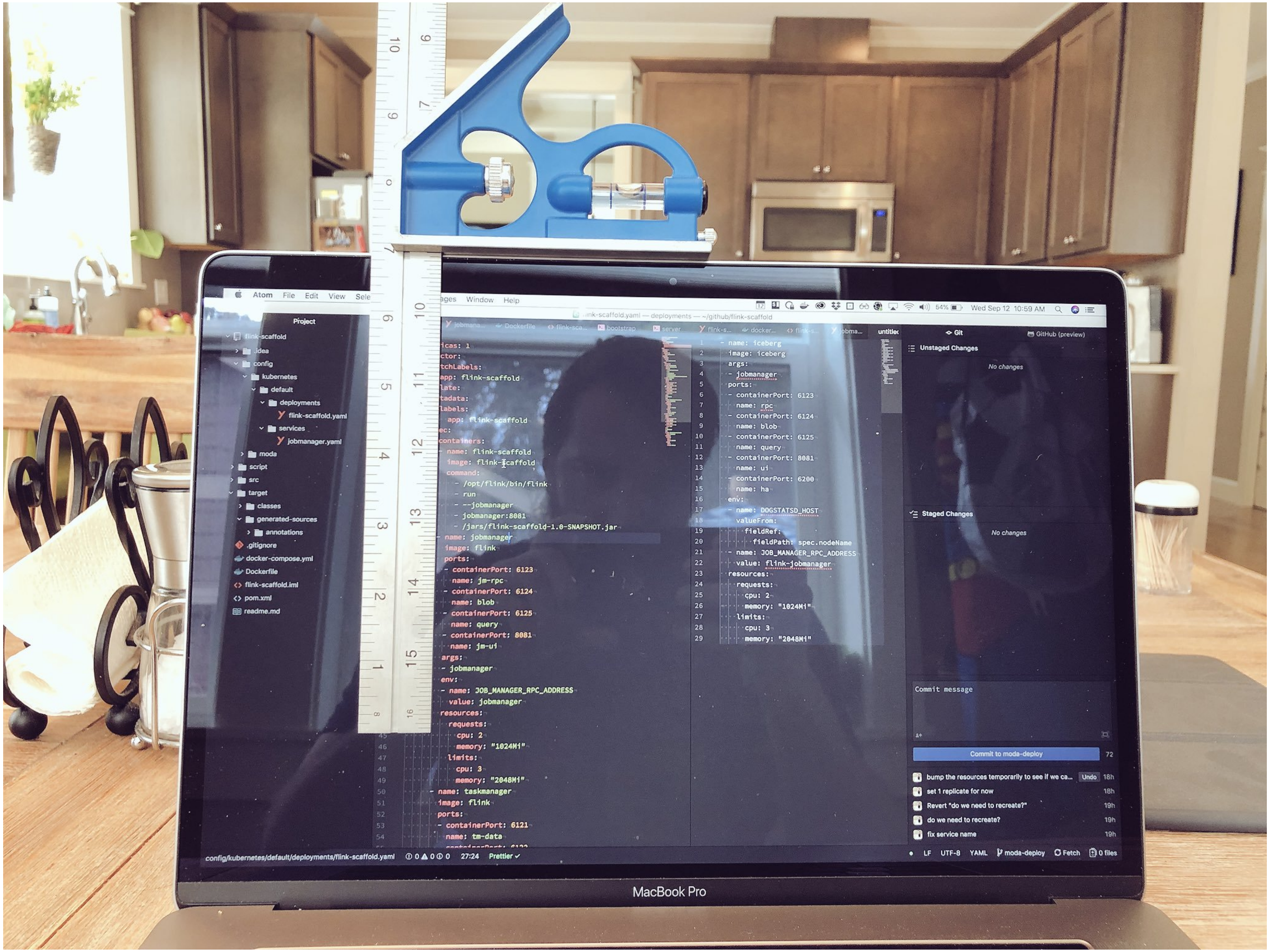
PROMETHEUS

A systems monitoring and alerting toolkit

FLAGGER

flagger.app

> *automates the promotion of canary deployments by using Istio’s traffic shifting and Prometheus metrics to analyse the application’s behaviour during a controlled rollout*



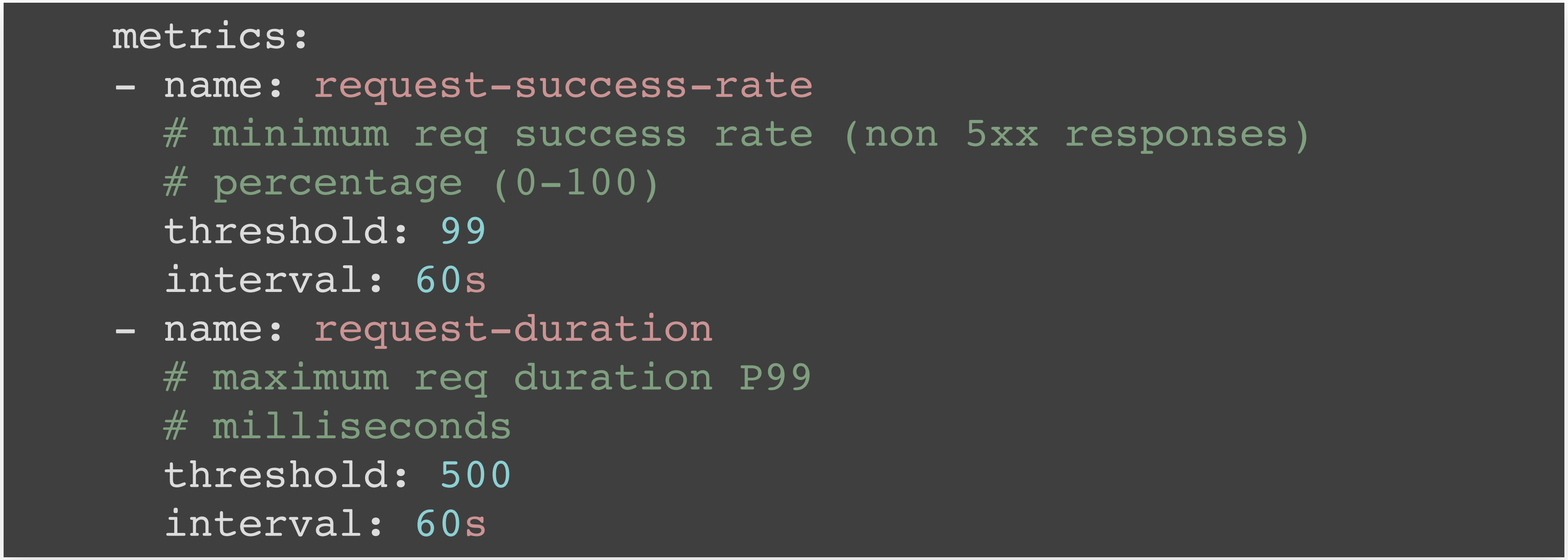
```
icas: 1
ctor:
tchLabels:
app: flink-scaffold
late:
tadata:
labels:
  app: flink-scaffold
ec:
containers:
- name: flink-scaffold
  image: flink-scaffold
  command:
    - /opt/flink/bin/flink
    - run
    - --jobmanager
    - jobmanager:8081
    - /jars/flink-scaffold-1.0-SNAPSHOT.jar
- name: jobmanager
  image: flink
  ports:
  - containerPort: 6123
    name: jm-rpc
  - containerPort: 6124
    name: blob
  - containerPort: 6125
    name: query
  - containerPort: 8081
    name: jm-ui
  args:
  - jobmanager
  env:
  - name: JOB_MANAGER_RPC_ADDRESS
    value: jobmanager
  resources:
    requests:
      cpu: 2
      memory: "1024Mi"
    limits:
      cpu: 3
      memory: "2048Mi"
- name: taskmanager
  image: flink
  ports:
  - containerPort: 6121
    name: tm-data
```

```
- name: iceberg
  image: iceberg
  args:
  - jobmanager
  ports:
  - containerPort: 6123
    name: rpc
  - containerPort: 6124
    name: blob
  - containerPort: 6125
    name: query
  - containerPort: 8081
    name: ui
  - containerPort: 6200
    name: ha
  env:
  - name: DOGSTATSD_HOST
    valueFrom:
      fieldRef:
        fieldPath: spec.nodeName
  - name: JOB_MANAGER_RPC_ADDRESS
    value: flink-jobmanager
  resources:
    requests:
      cpu: 2
      memory: "1024Mi"
    limits:
      cpu: 3
      memory: "2048Mi"
```

```
bump the resources temporarily to see if we ca...   18h
set 1 replicate for now                             18h
Revert "do we need to recreate?"                    19h
do we need to recreate?                             19h
fix service name                                    19h
```

Add the `canary` section to our application Helm chart `values.yaml`

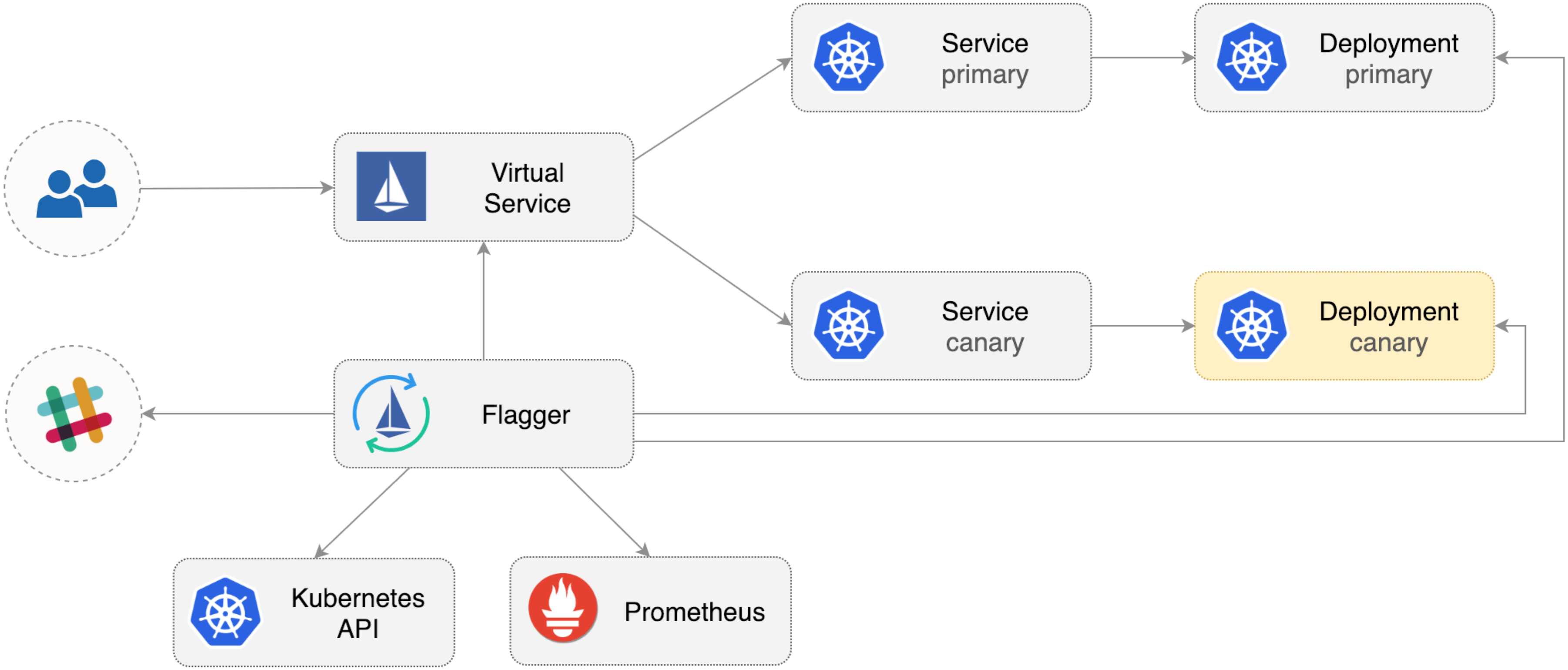
```
...
canary:
  enable: true
  service:
    hosts:
    - croc-hunter.istio.us.g.csanchez.org
    gateways:
    - jx-gateway.istio-system.svc.cluster.local
  canaryAnalysis:
    interval: 60s
    threshold: 5
    maxWeight: 50
    stepWeight: 10
```

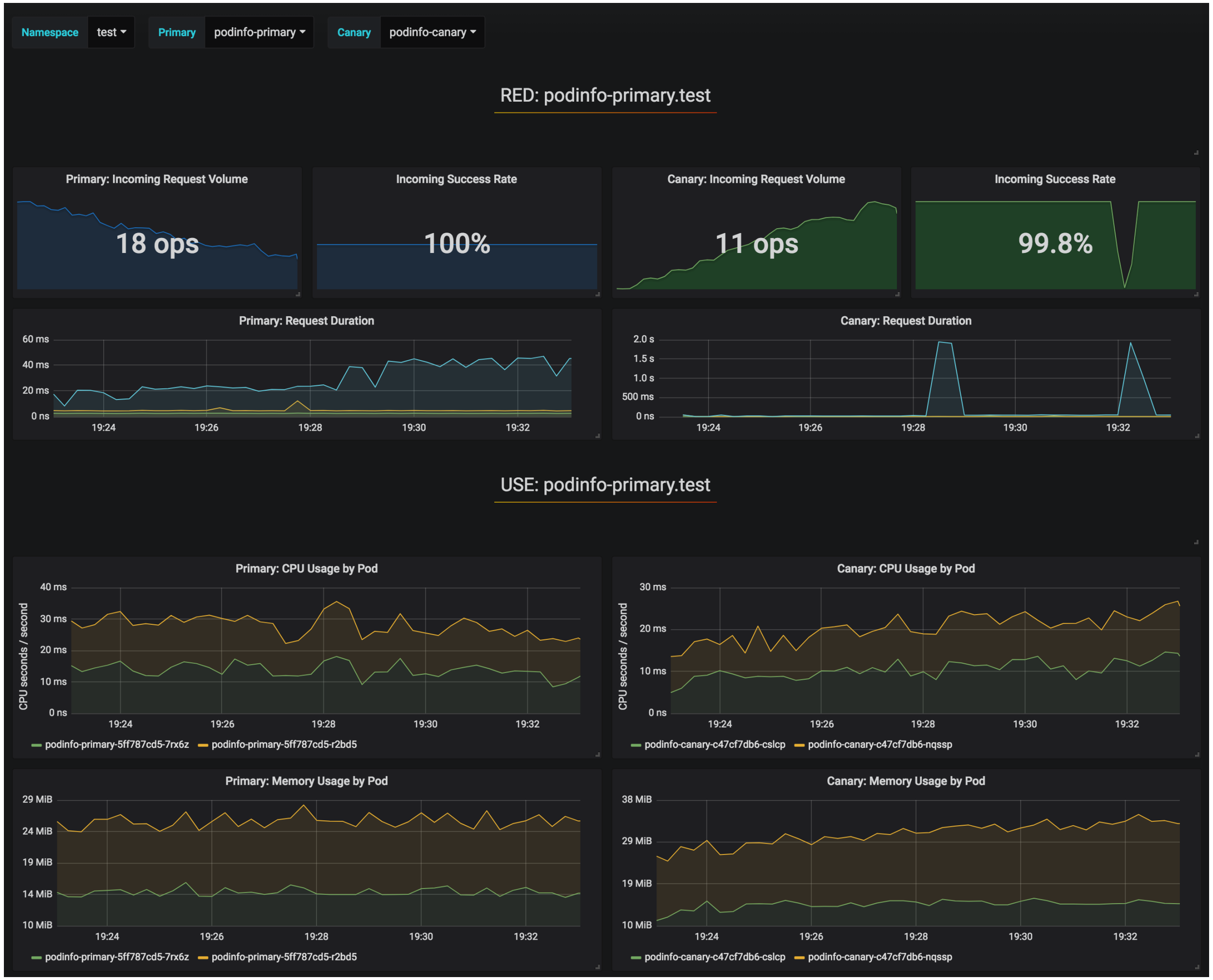


```yaml
metrics:
- name: request-success-rate
  # minimum req success rate (non 5xx responses)
  # percentage (0-100)
  threshold: 99
  interval: 60s
- name: request-duration
  # maximum req duration P99
  # milliseconds
  threshold: 500
  interval: 60s
```

PROFIT!

```
jx promote croc-hunter-java \
  --version 0.0.130 \
  --env production
```


Service
primary
Deployment
primary
Virtual
Service
Service
canary
Deployment
canary
Flagger
Kubernetes
API
Prometheus



Namespace test

Primary podinfo-primary

Canary podinfo-canary

RED: podinfo-primary.test

Primary: Incoming Request Volume
18 ops

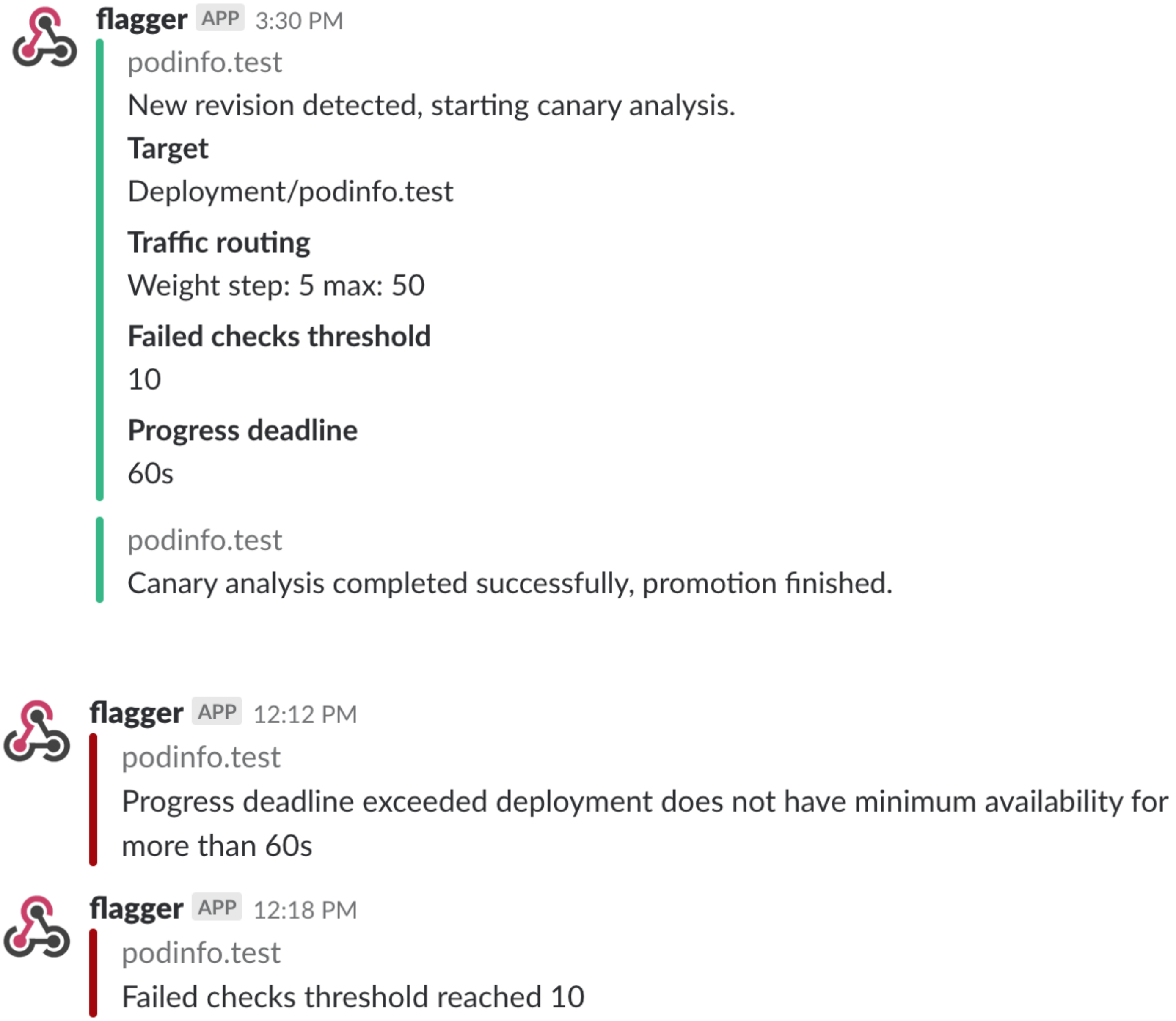
Incoming Success Rate
100%

Canary: Incoming Request Volume
11 ops

Incoming Success Rate
99.8%

USE: podinfo-primary.test

Primary: CPU Usage by Pod
CPU seconds / second
40 ms
30 ms
20 ms
10 ms
0 ns
19:24
19:26
19:28
19:30
19:32
podinfo-primary-5ff787cd5-7rx6z
podinfo-primary-5ff787cd5-r2bd5
Canary: CPU Usage by Pod
CPU seconds / second
30 ms
20 ms
10 ms
0 ns
19:24
19:26
19:28
19:30
19:32
podinfo-canary-c47cf7db6-cslcp
podinfo-canary-c47cf7db6-nqssp
Primary: Memory Usage by Pod
29 MiB
24 MiB
19 MiB
14 MiB
10 MiB
19:24
19:26
19:28
19:30
19:32
podinfo-primary-5ff787cd5-7rx6z
podinfo-primary-5ff787cd5-r2bd5
Canary: Memory Usage by Pod
38 MiB
29 MiB
19 MiB
10 MiB
19:24
19:26
19:28
19:30
19:32
podinfo-canary-c47cf7db6-cslcp
podinfo-canary-c47cf7db6-nqssp



flagger APP 3:30 PM

podinfo.test

New revision detected, starting canary analysis.

Target

Deployment/podinfo.test

Traffic routing

Weight step: 5 max: 50

Failed checks threshold

10

Progress deadline

60s

podinfo.test

Canary analysis completed successfully, promotion finished.

flagger APP 12:12 PM

podinfo.test

Progress deadline exceeded deployment does not have minimum availability for more than 60s

flagger APP 12:18 PM

podinfo.test

Failed checks threshold reached 10

QUARKUS

quarkus.io

A Kubernetes Native Java stack tailored for GraalVM & OpenJDK HotSpot, crafted from the best of breed Java libraries and standards

THE DEVOPS 2.6 TOOLKIT

Jenkins X

Viktor Farcic

CLOUD-NATIVE
KUBERNETES-FIRST
CONTINUOUS DELIVERY

csanchez.org

csanchez

carlossg

Adobe®