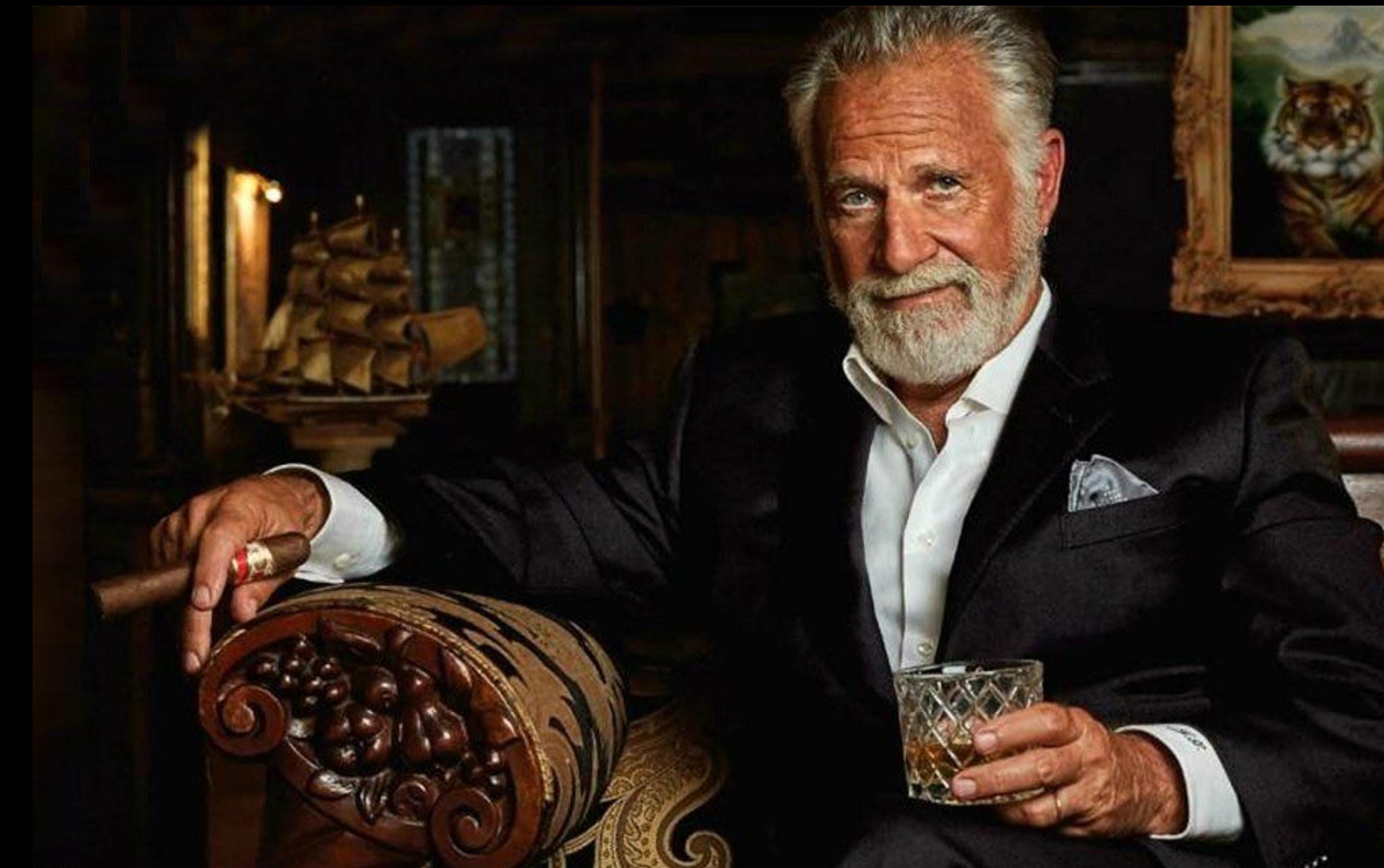


DEPLOYMENTS!

WHAT COULD POSSIBLY GO WRONG?

A LOT!

DUH.



When I _____ ... I do it in production.



When I **deploy to prod**... I do it in production.

CANARY DEPLOYS

WITH KUBERNETES AND ISTIO





JASON YEE

Technical Evangelist
Nomad & Travel Hacker
Whiskey Hunter
Pokemon Trainer

Tw: @gitbisect

Em: jyee@datadoghq.com

DATADOG

TW: @datadoghq

SaaS-based monitoring,
tracing & logging

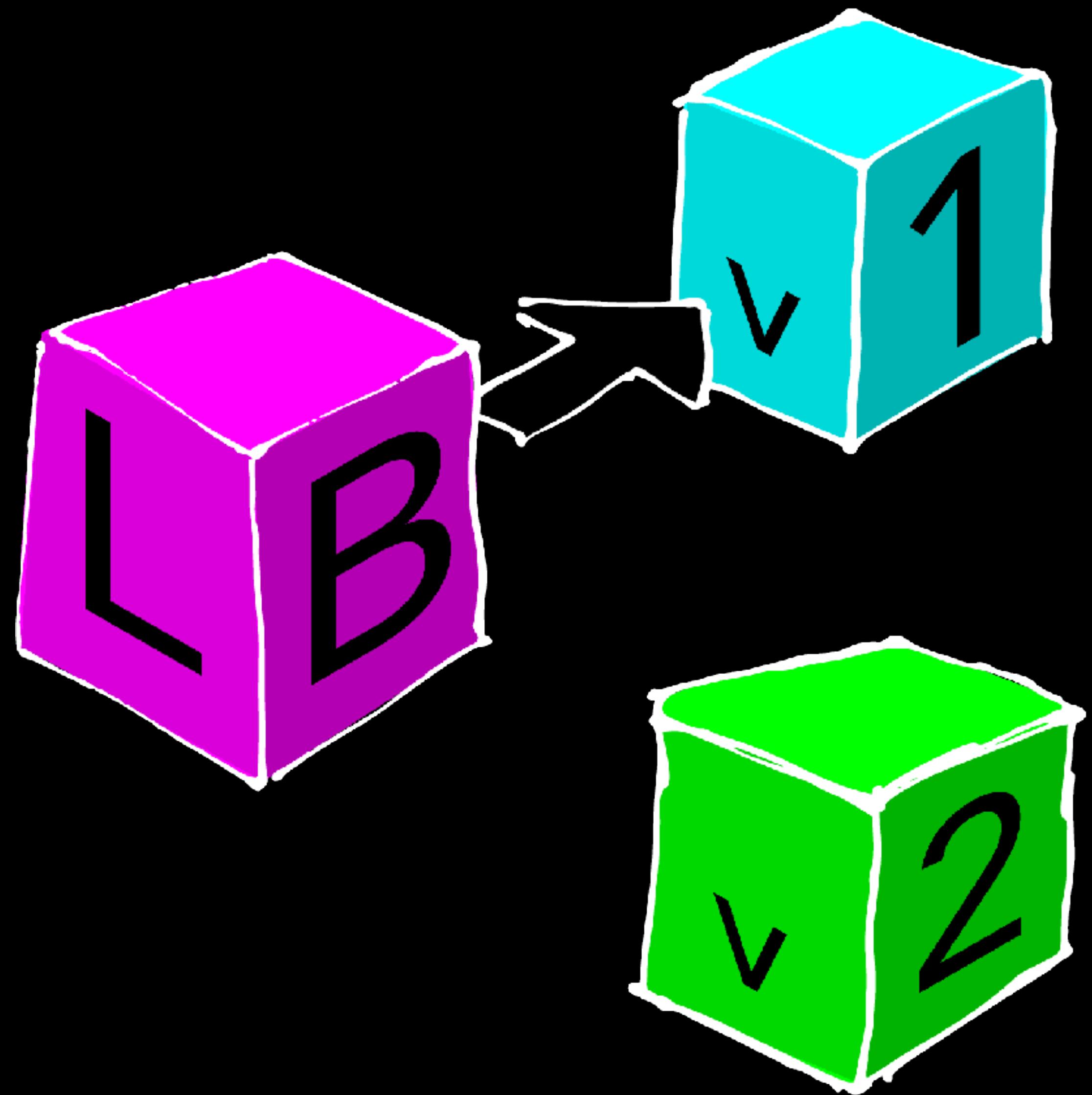
Trillions of points/day

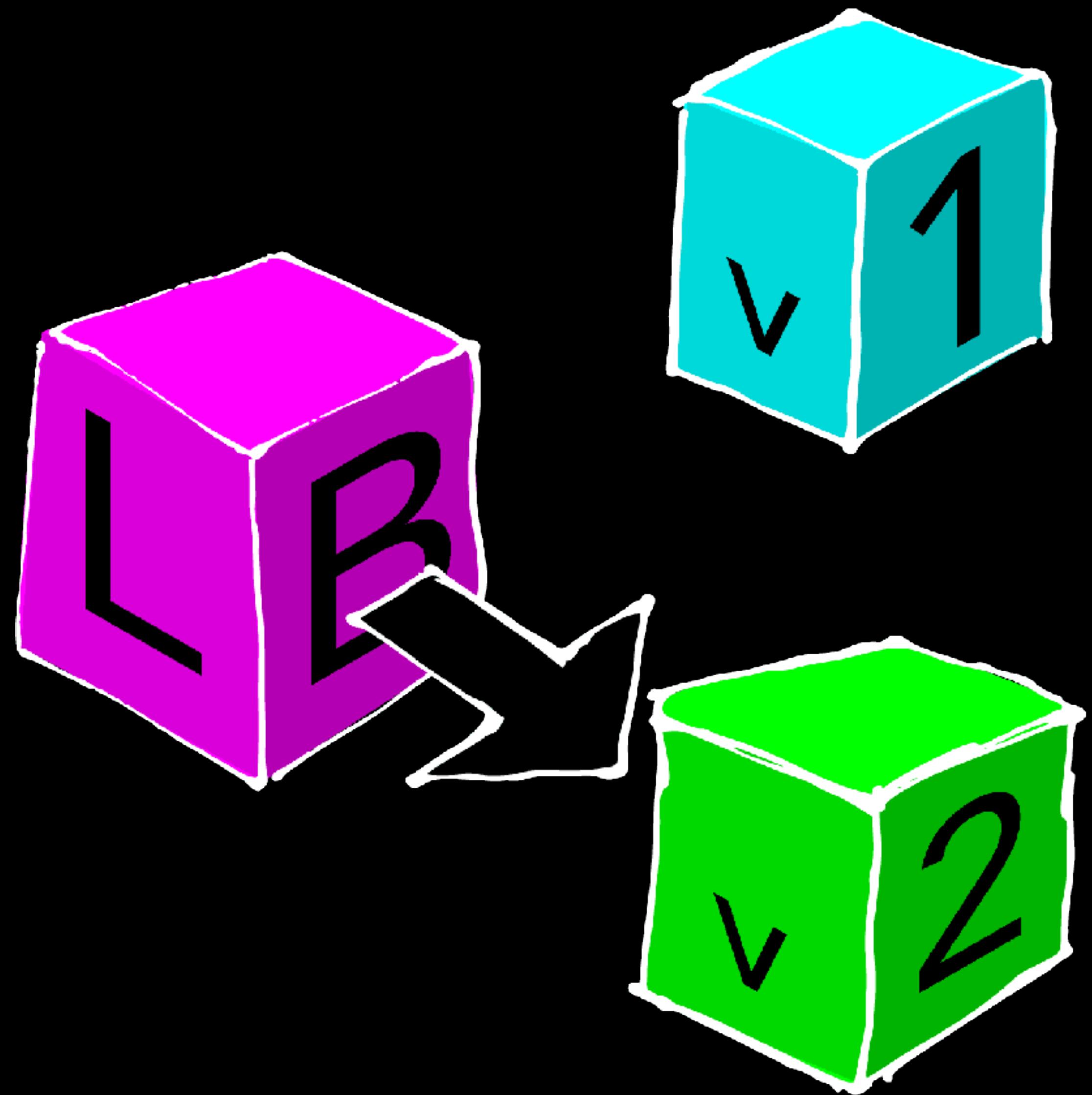
We're hiring:
jobs.datadoghq.com

Note: We're running some
production services on
Kubernetes & have been
implementing Istio.



BLUE-GREEN DEPLOYMENTS





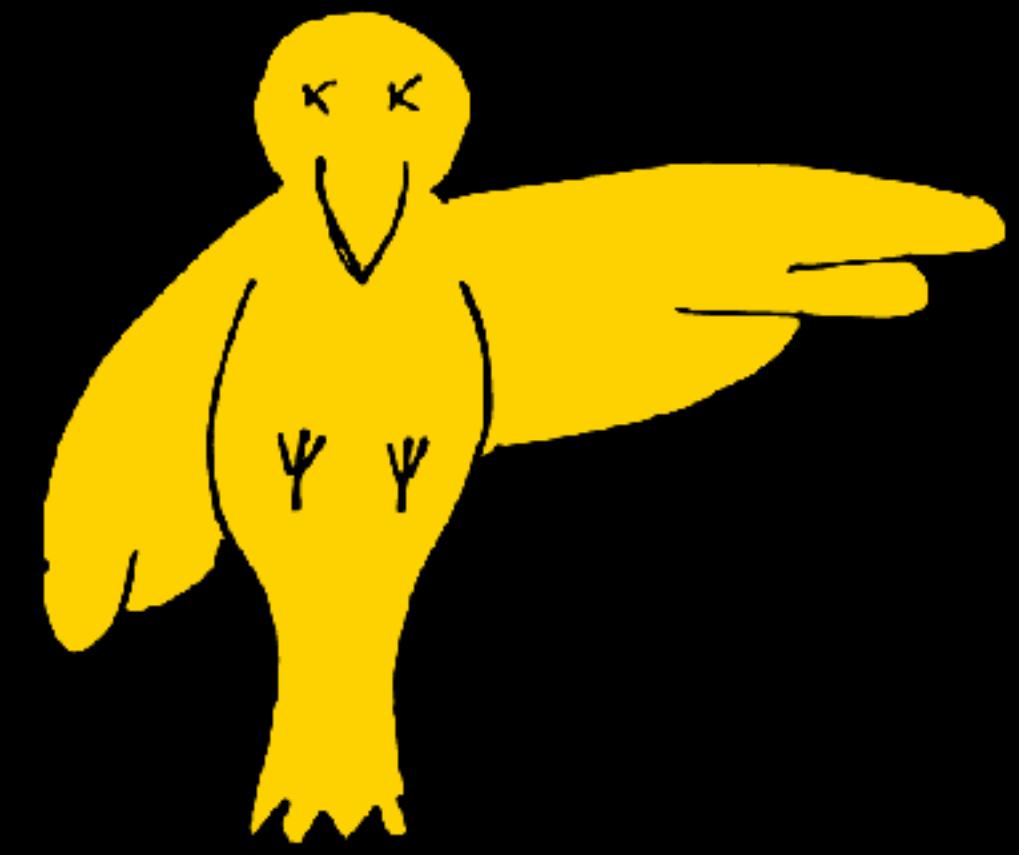
BLUE-GREEN DEPLOYMENTS

- Pros:
 - Zero-downtime deploys
 - Easy rollbacks

CTRL + Z

BLUE-GREEN DEPLOYMENTS

- Cons:
 - Easy 🤝 rollbacks... 🤝 😬

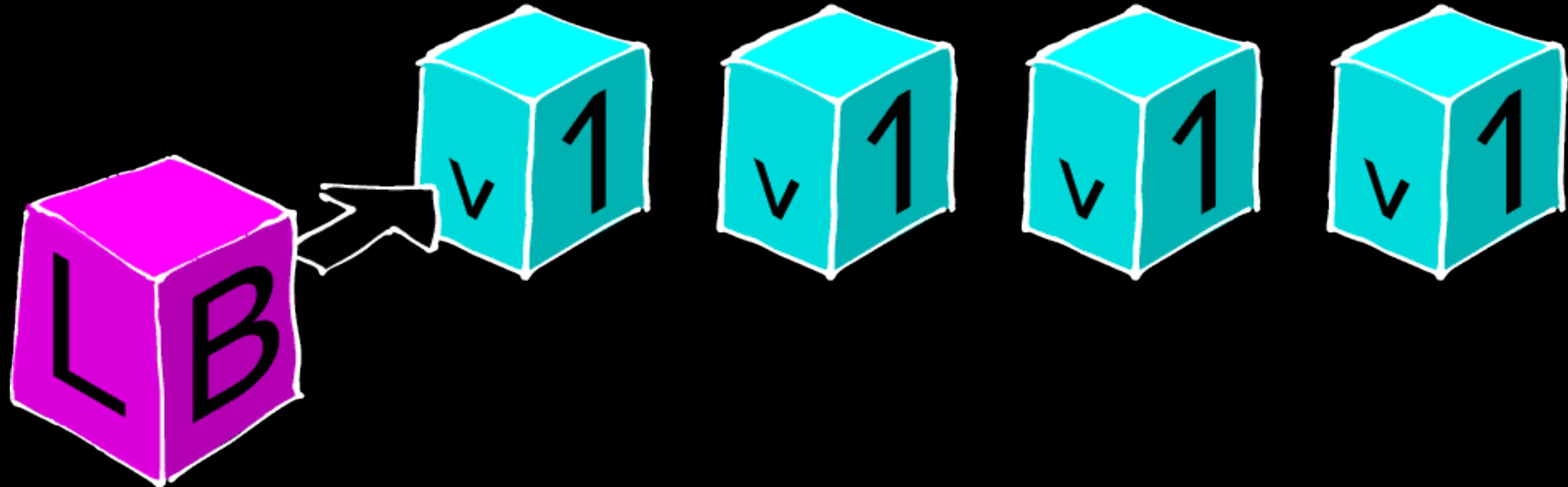


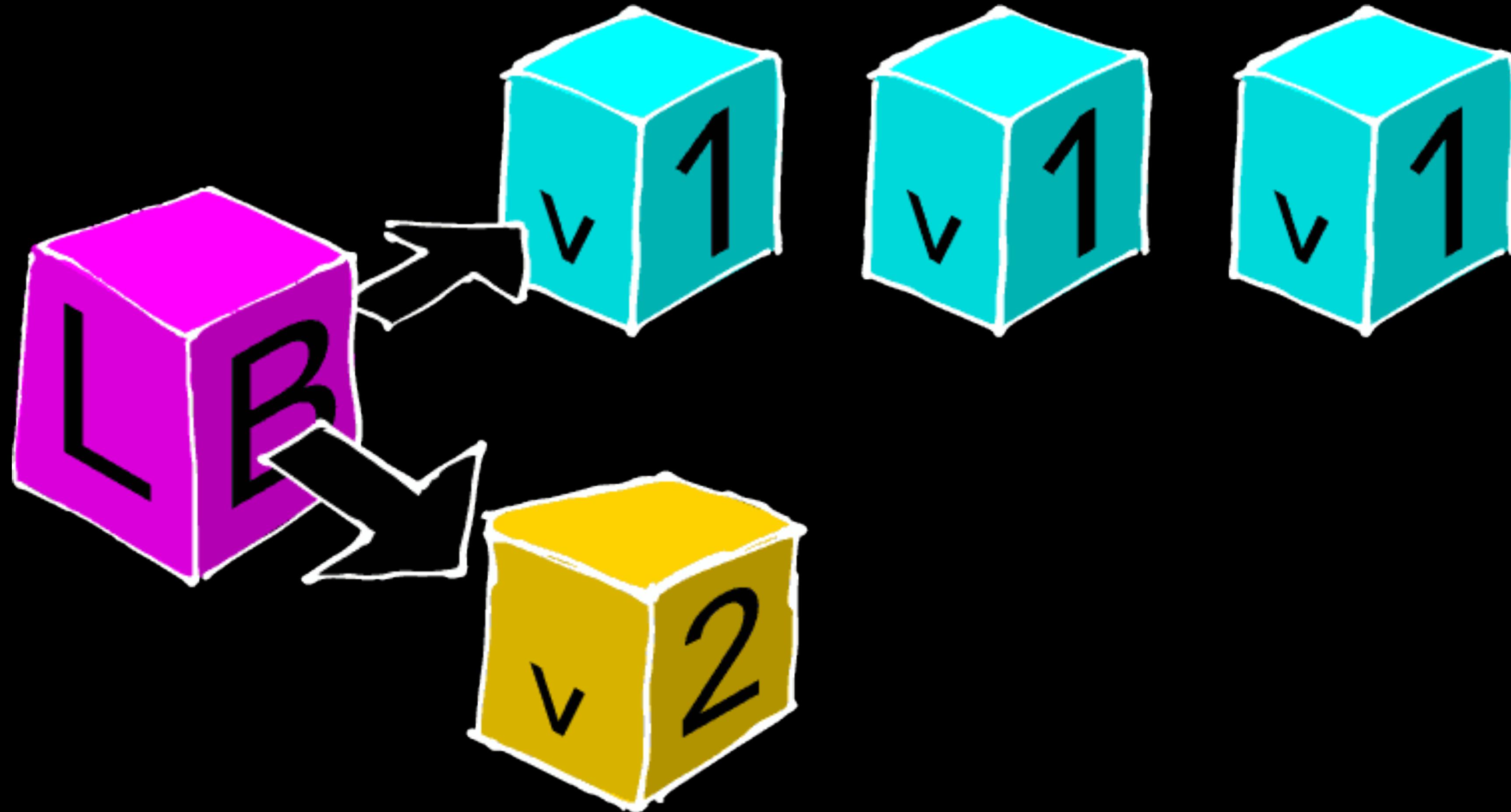
CANARY DEPLOYMENTS



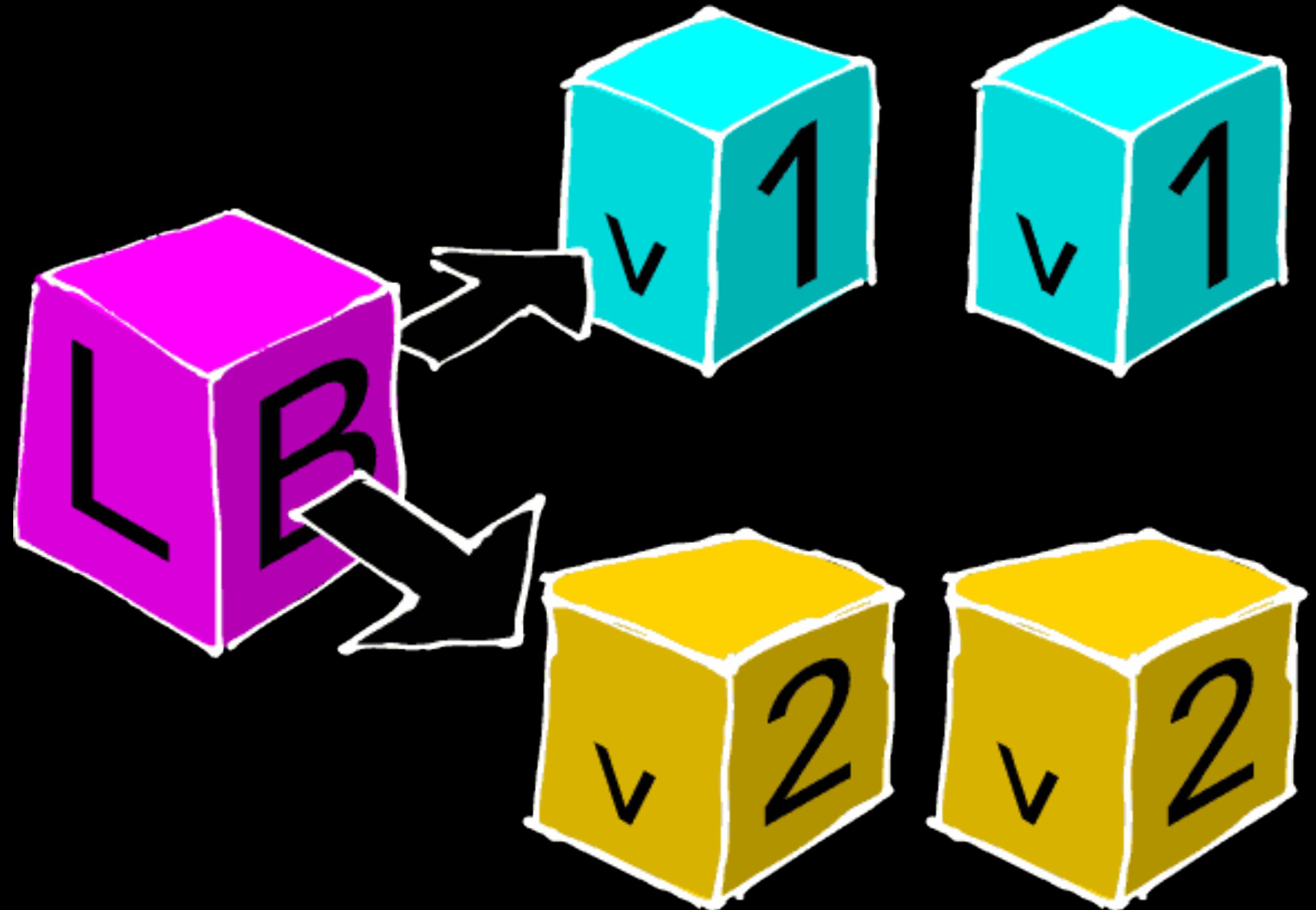
or



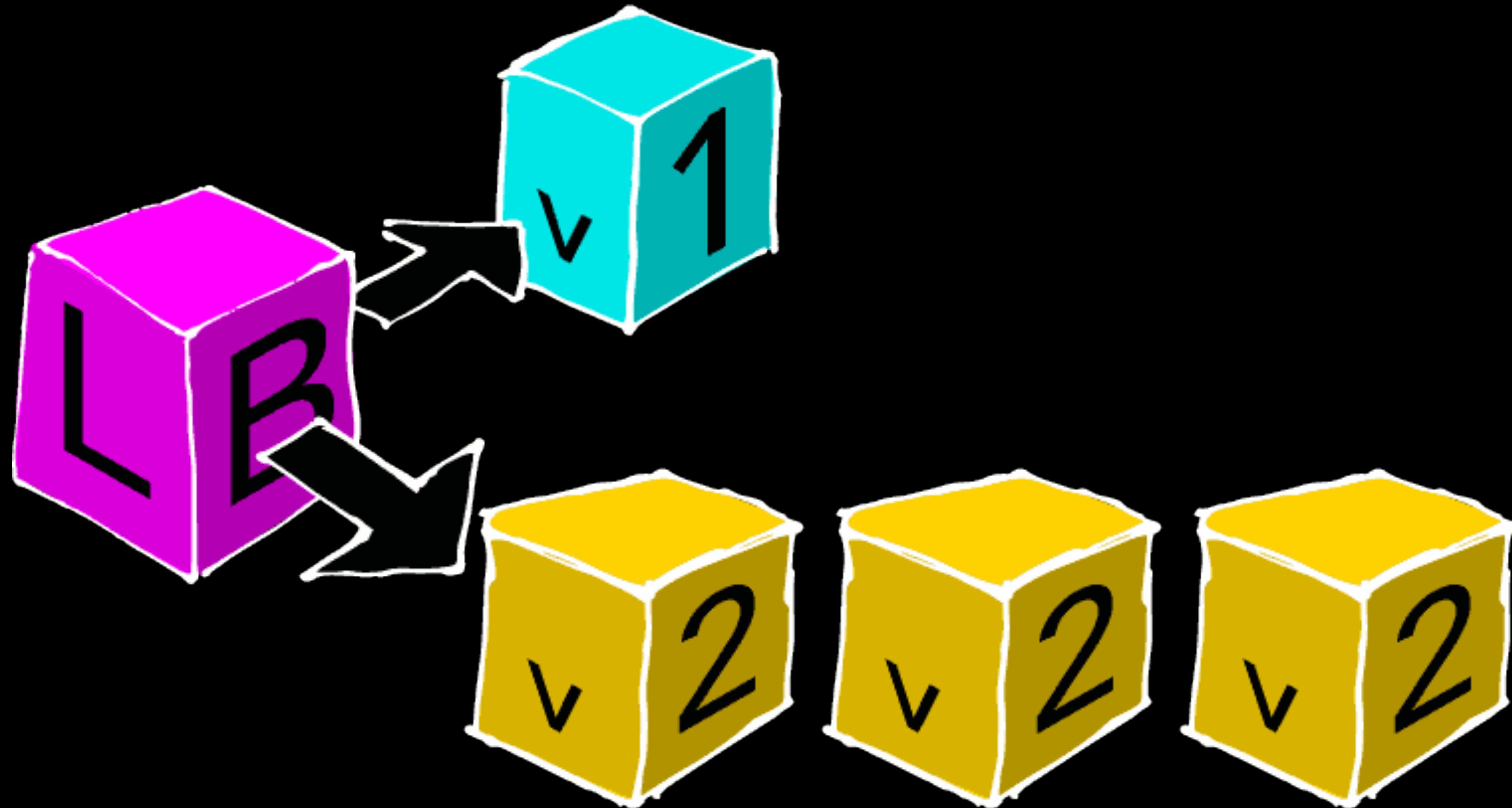




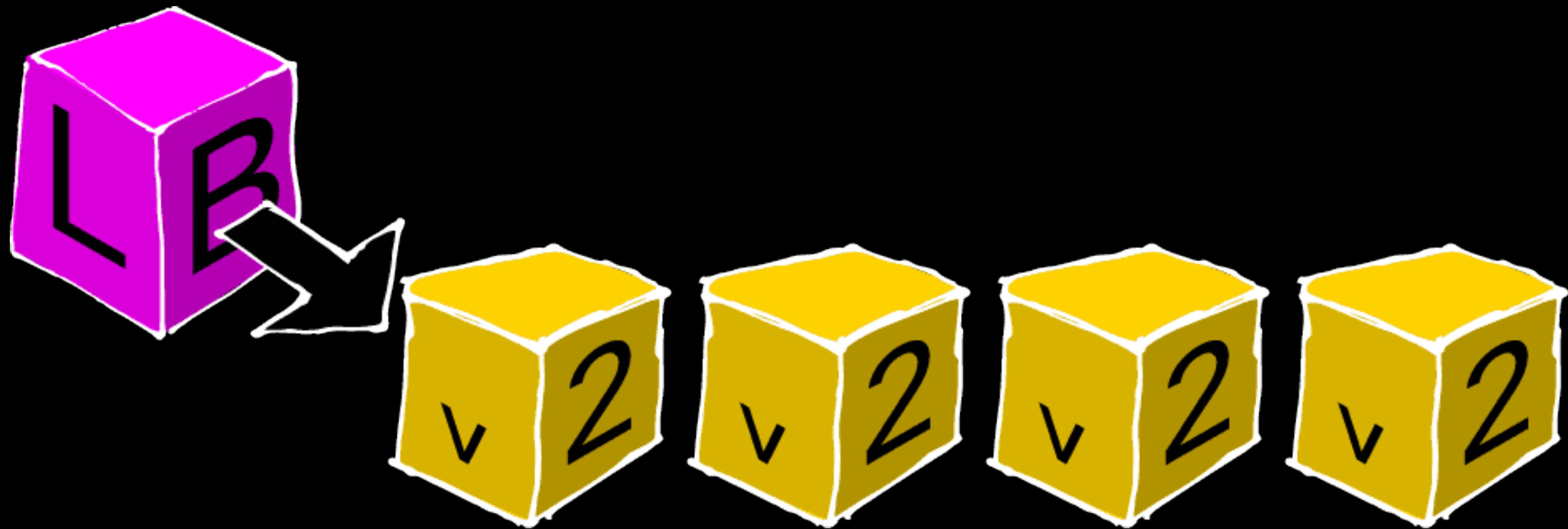
PAUSE. MONITOR.

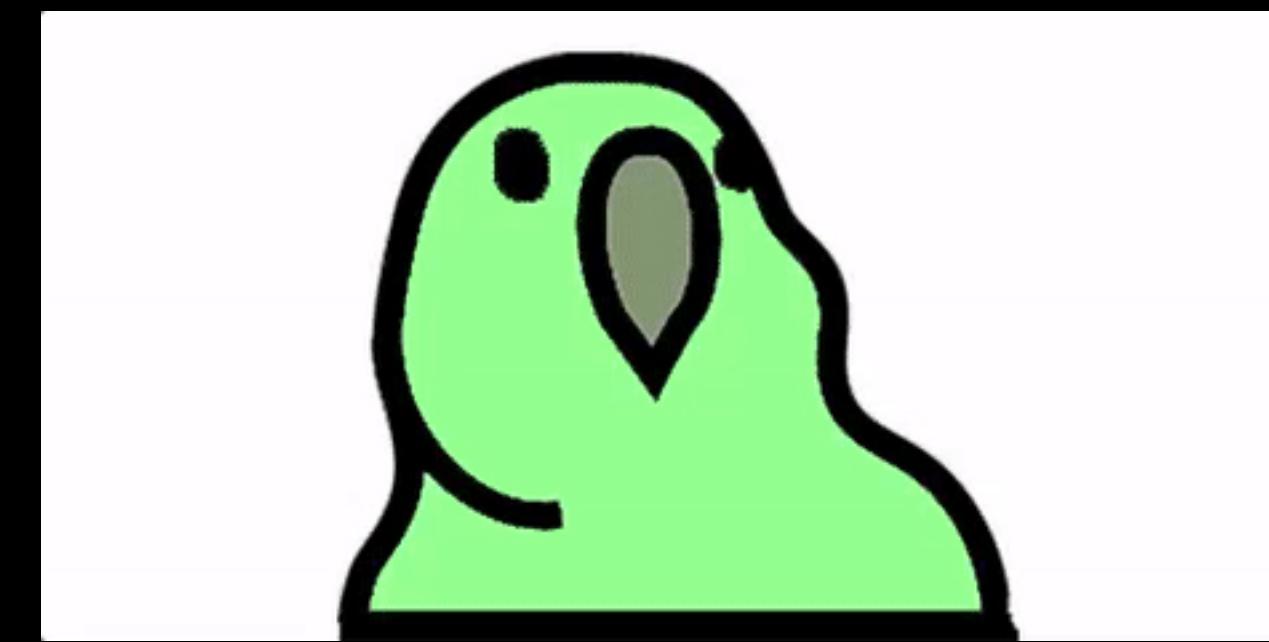


DOUBLE CHECK.



MAYBE ONE MORE TIME,
JUST TO BE CERTAIN.





PARTY!

CANARY DEPLOYMENTS

- Small scope

CANARY DEPLOYMENTS

- Small scope
- **Limited ramifications**

CANARY DEPLOYMENTS

- Small scope
- Limited ramifications
- **Easier rollbacks**

CANARY DEPLOYMENTS

- Small scope
- Limited ramifications
- Easier rollbacks
- **Load tolerant**

CANARY DEPLOYMENTS

- Small scope
- Limited ramifications
- Easier rollbacks
- Load tolerant
- **Concurrency**

CANARY STRATEGY



CANARY STRATEGY

How do you choose your sample set?

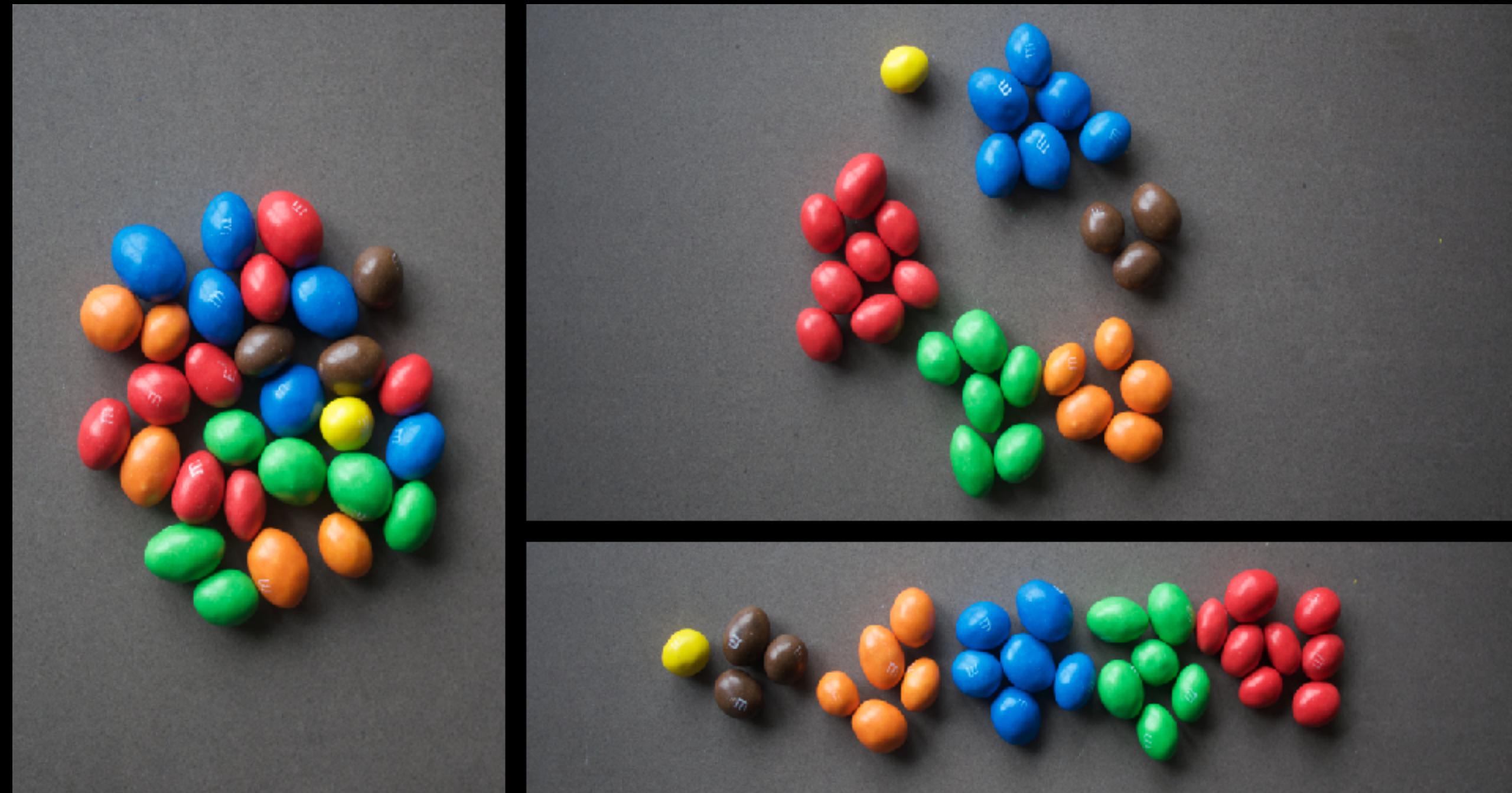
- Random



CANARY STRATEGY

How do you choose your sample set?

- Random
- **Representative**



CANARY STRATEGY

How do you choose your sample set?

- Random
- Representative
 - **Geography**
 - **Time**
 - **Use patterns**



CANARY STRATEGY

How do you choose your sample set?

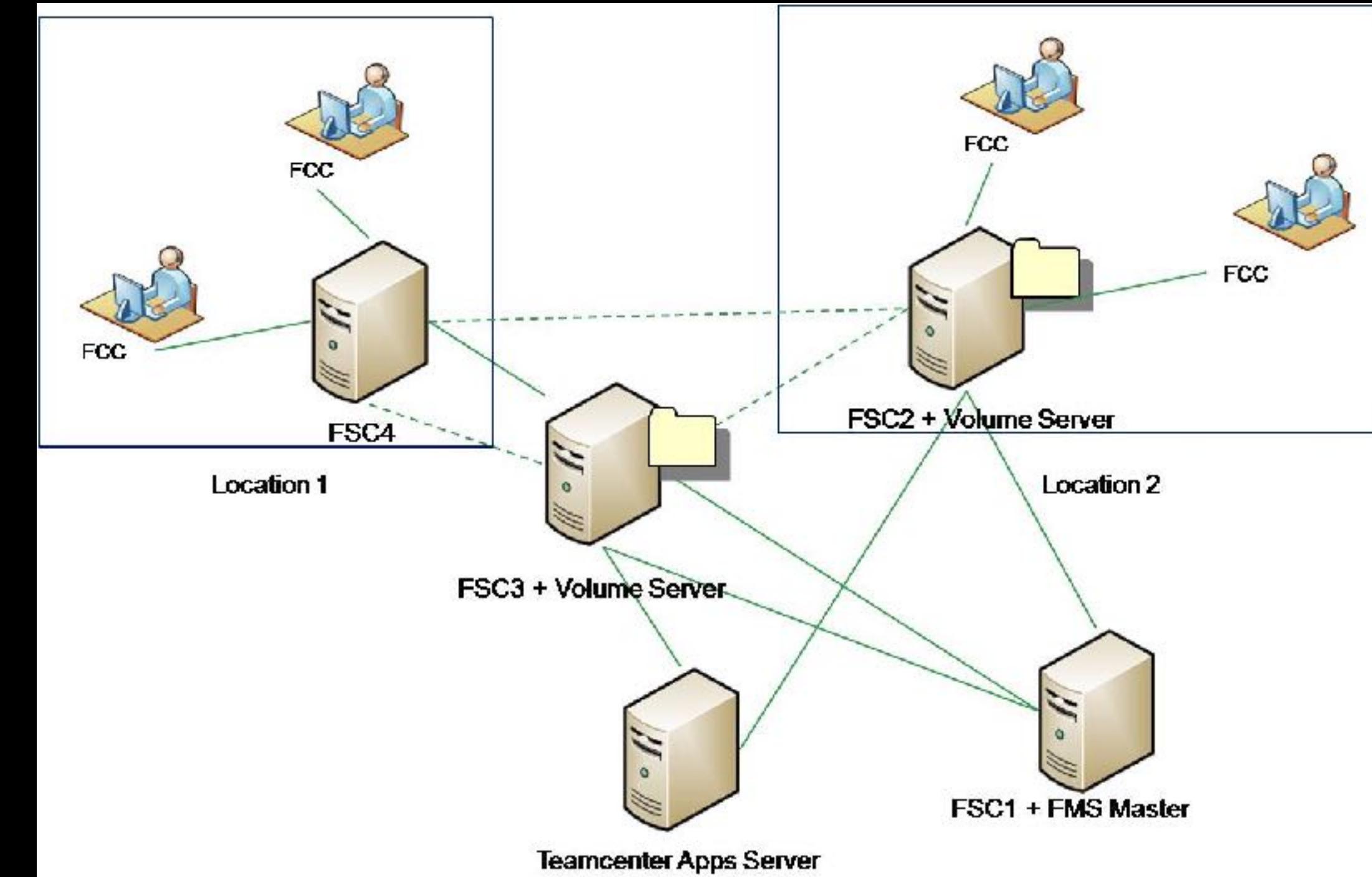
- Random
- Representative
 - Geography
 - Time
 - Use patterns
- **Granularity**



CANARY STRATEGY

How do you choose your sample set?

- Random
- Representative
 - Geography
 - Time
 - Use patterns
- Granularity
- **Resource mapping**



MONITORING STRATEGY



MONITORING STRATEGY

How do you evaluate your deployment?

- Tags! Tags! Tags! Tags! Tags!

DATAPoint

SYSTEM.NET.BYTES_RCVD 4 2016-03-02 15:00:00

METRIC NAME:
WHAT?

METRIC VALUE:
HOW MUCH?

TIMESTAMP:
WHEN?

DATAPoint

SYSTEM.NET.BYTES_RCVD 4 2016-03-02 15:00:00 [DEPLOYMENT]

METRIC NAME: WHAT? METRIC VALUE: HOW MUCH? TIMESTAMP: WHEN? TAGS: WHERE?

MONITORING STRATEGY

How do you evaluate your deployment?

- Tags!
- p90, p95, p99

MONITORING STRATEGY

How do you evaluate your deployment?

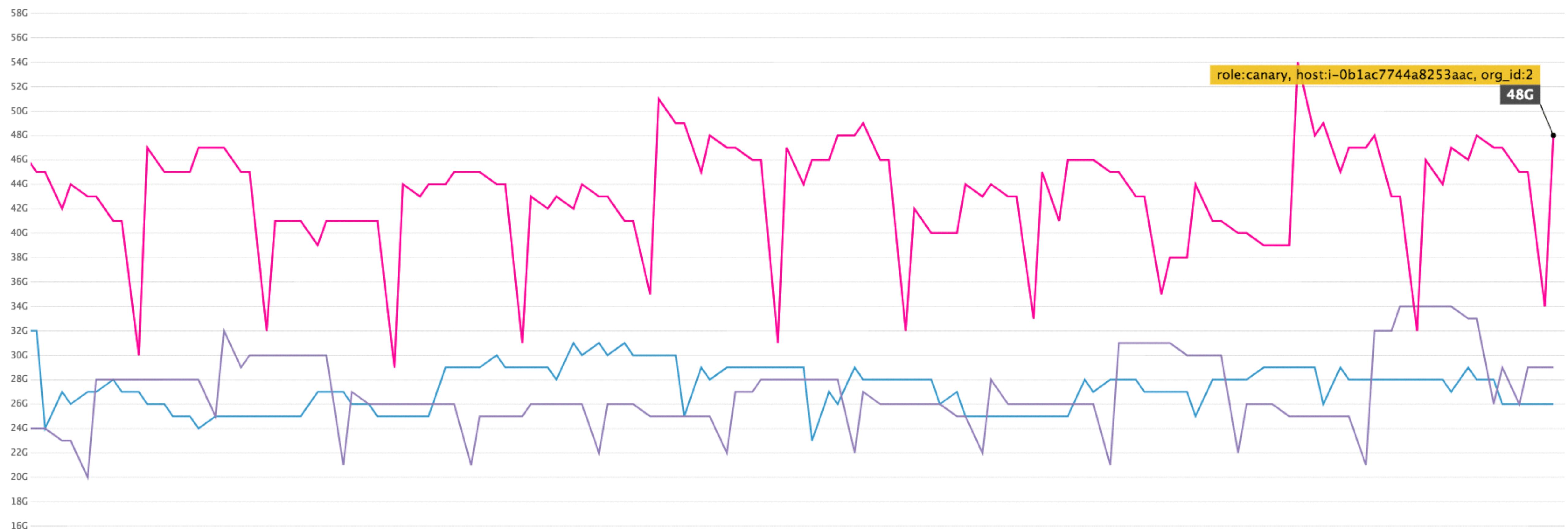
- Tags!
- p90, p95, p99
- **Outliers**

DAY 14



THEY STILL SUSPECT NOTHING

Outliers: one of these things is not like the others

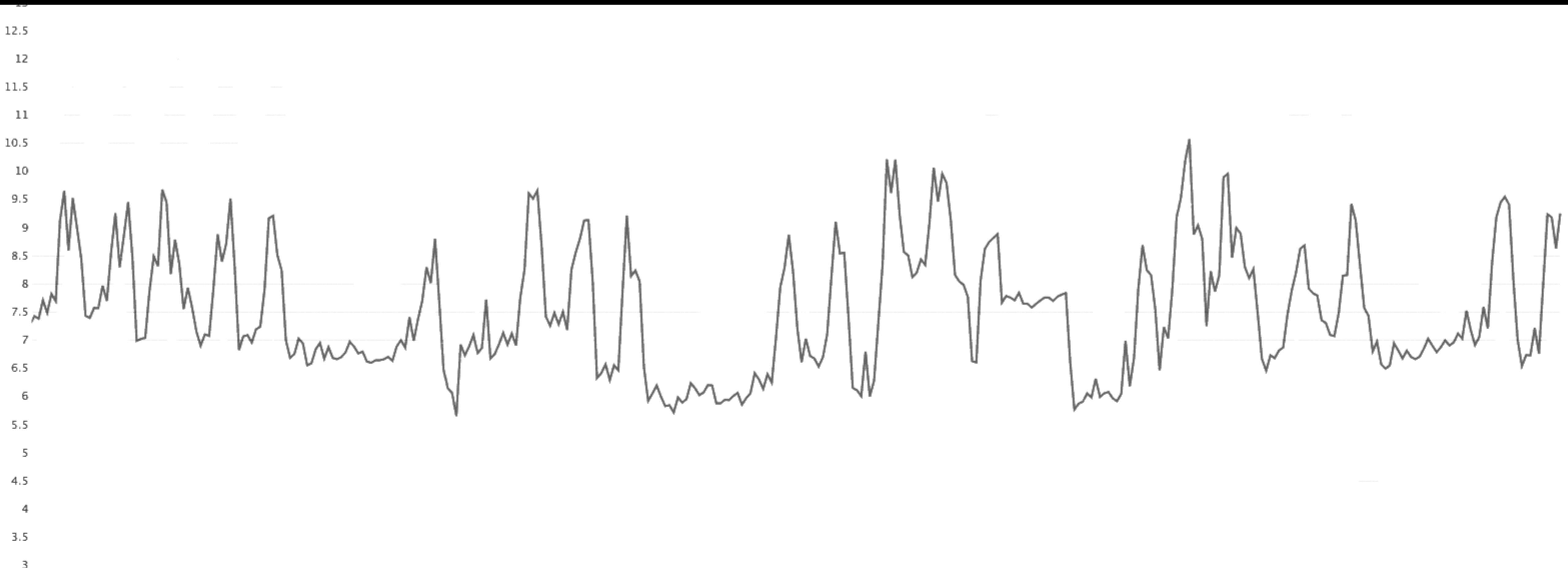


MONITORING STRATEGY

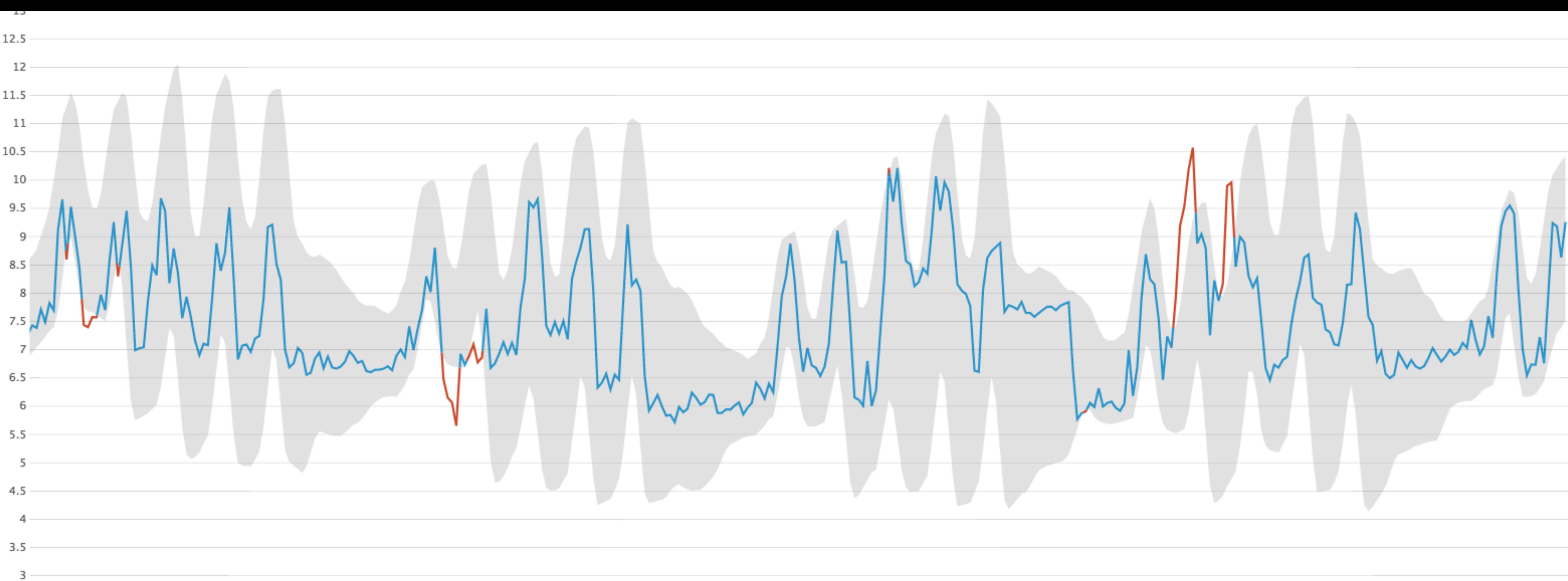
How do you evaluate your deployment?

- Tags!
- p90, p95, p99
- Outliers
- **Anomalies**

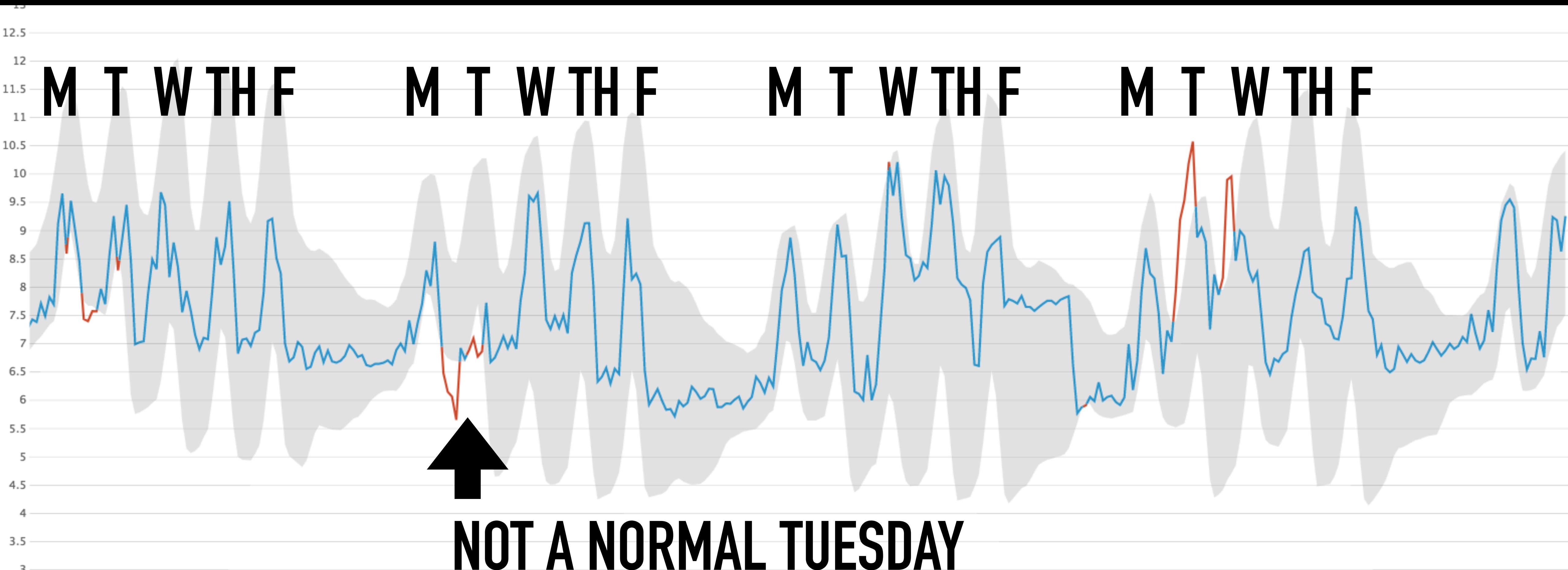
Anomalies: It wasn't like this before



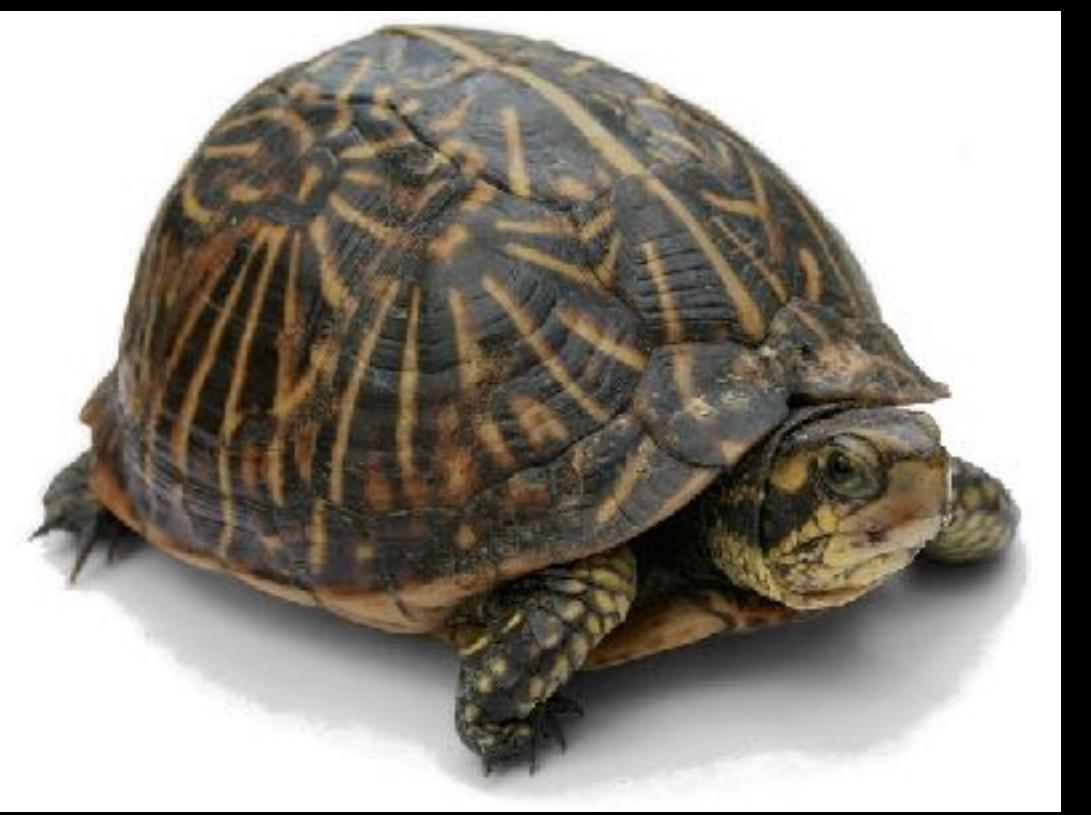
Anomalies: It wasn't like this before



Anomalies: It wasn't like this before



SIGNALS TO WATCH

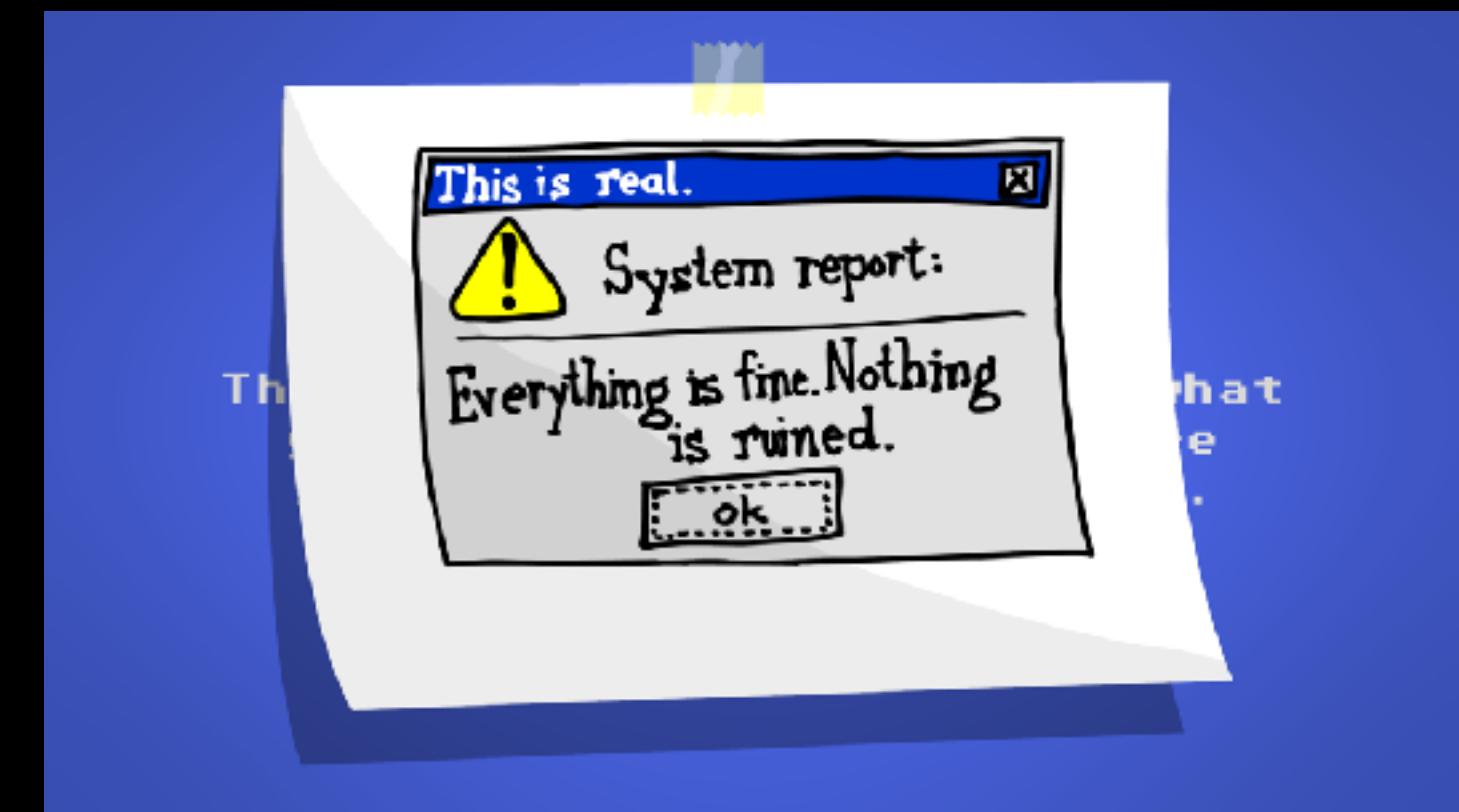


Latency

SIGNALS TO WATCH



Latency

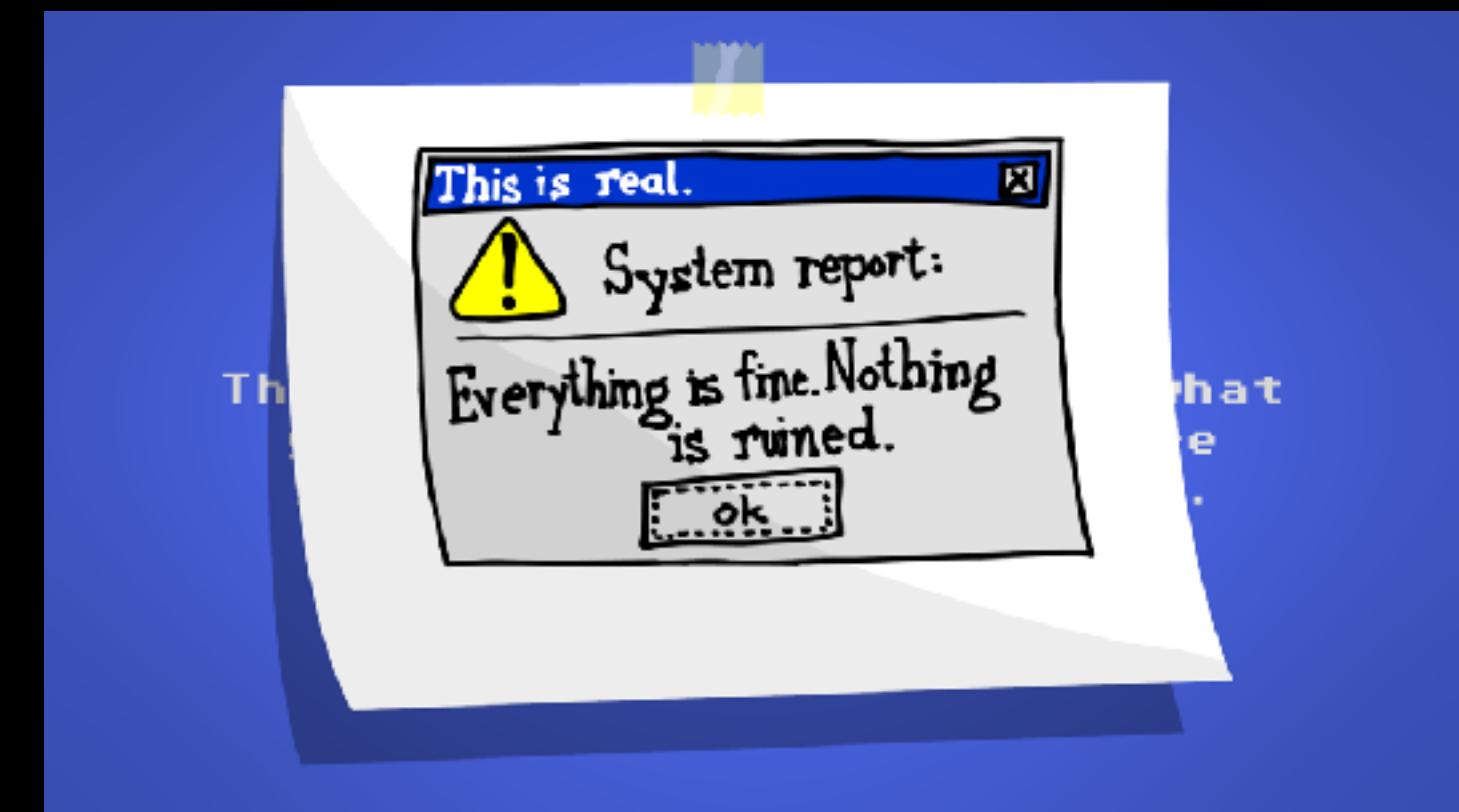


Errors

SIGNALS TO WATCH



Latency



Errors

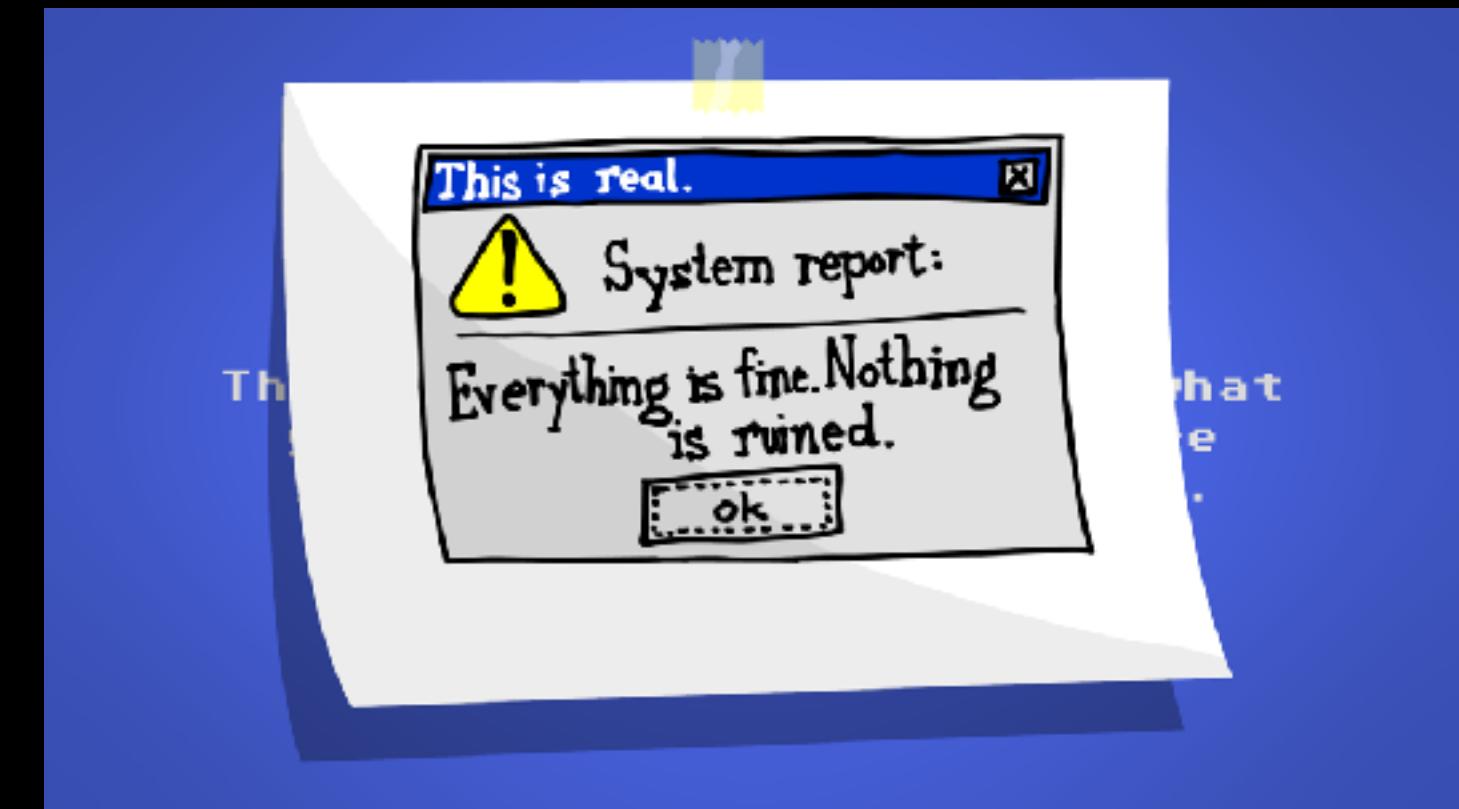


Traffic

SIGNALS TO WATCH



Latency



Errors



Traffic



Saturation



WHAT DOES KUBERNETES HAVE
TO DO WITH ANY OF THIS?



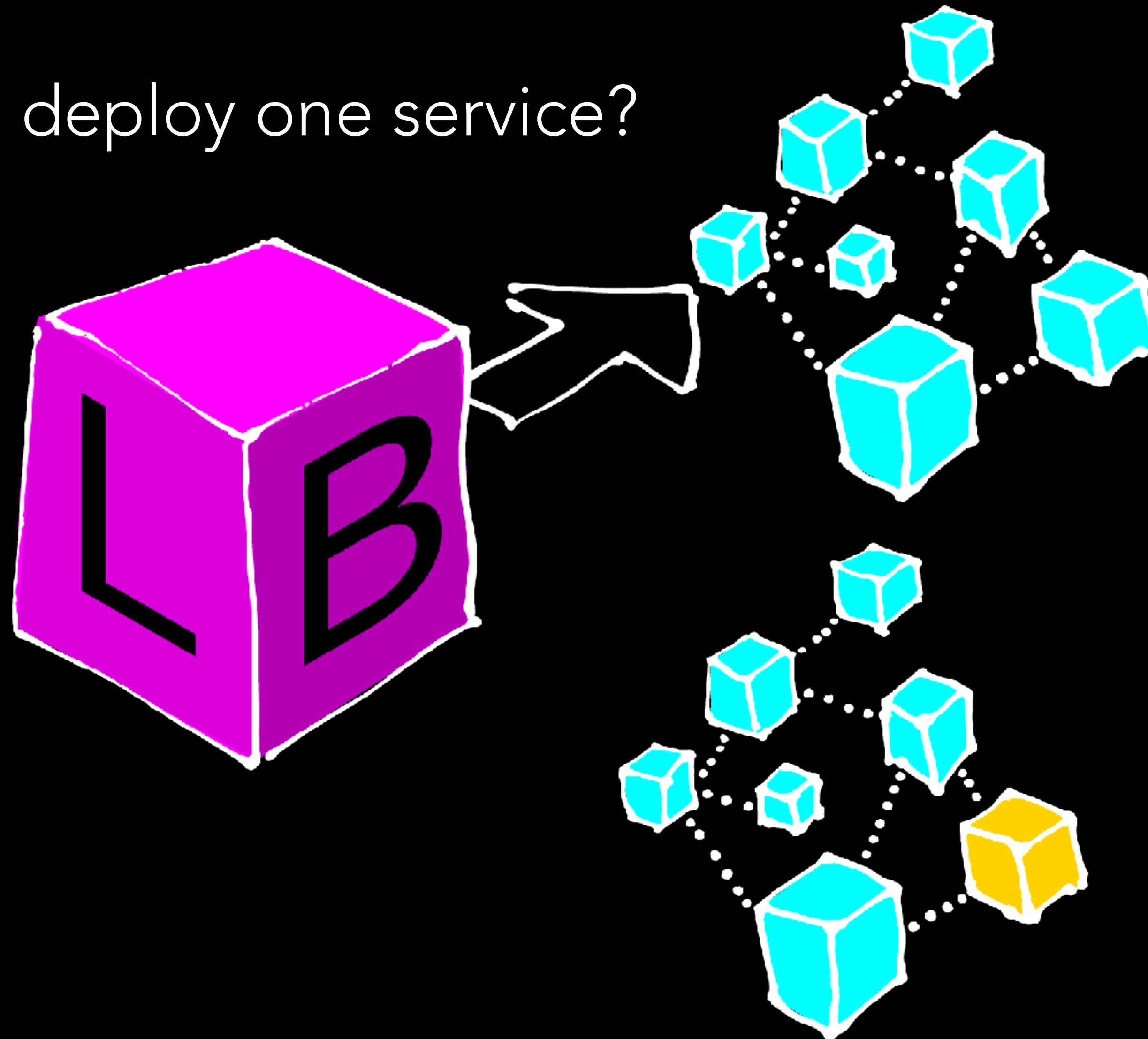
CONTAINER
ORCHESTRATOR

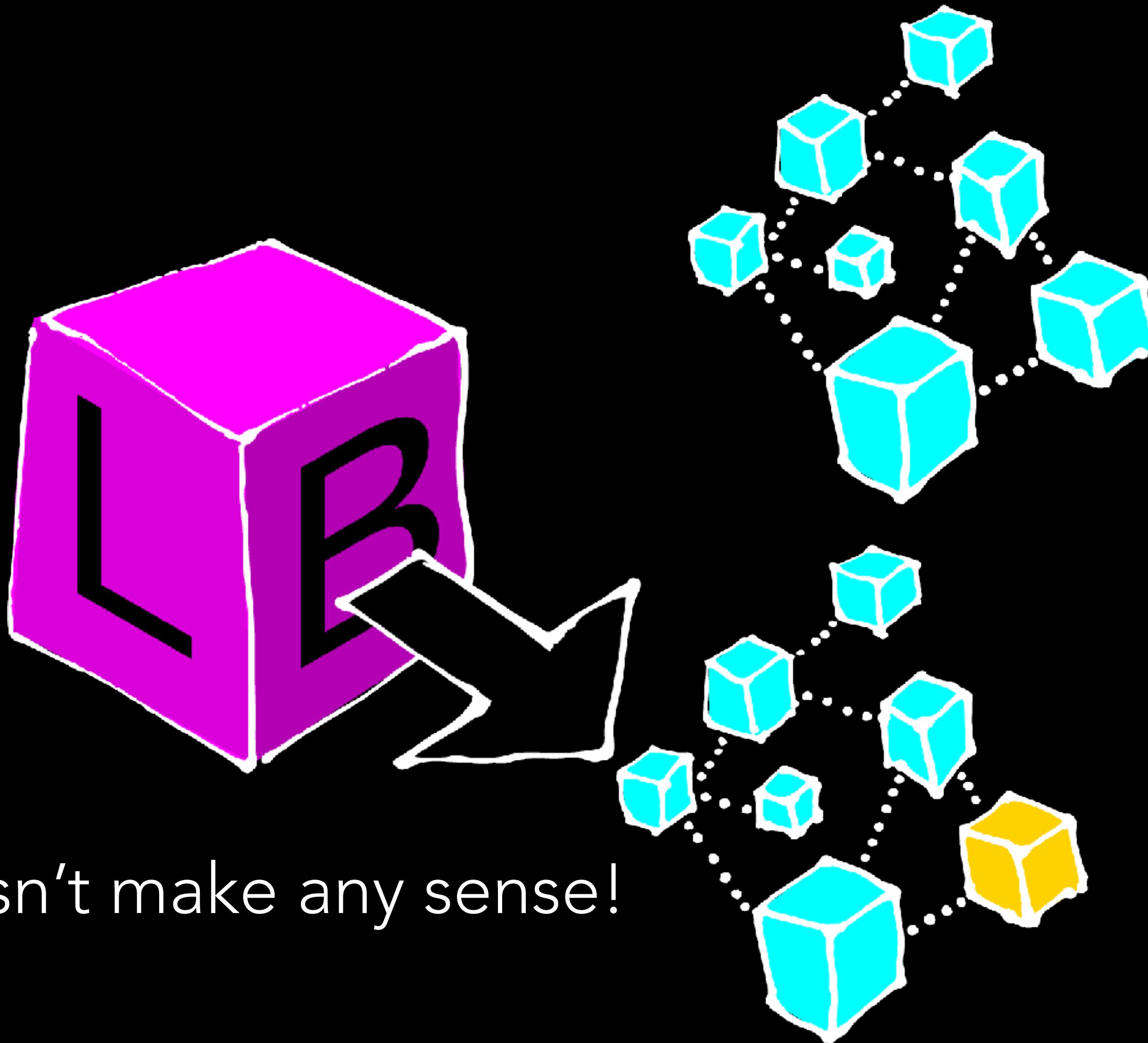


CONTAINER SERVICE ~~ORCHESTRATOR~~

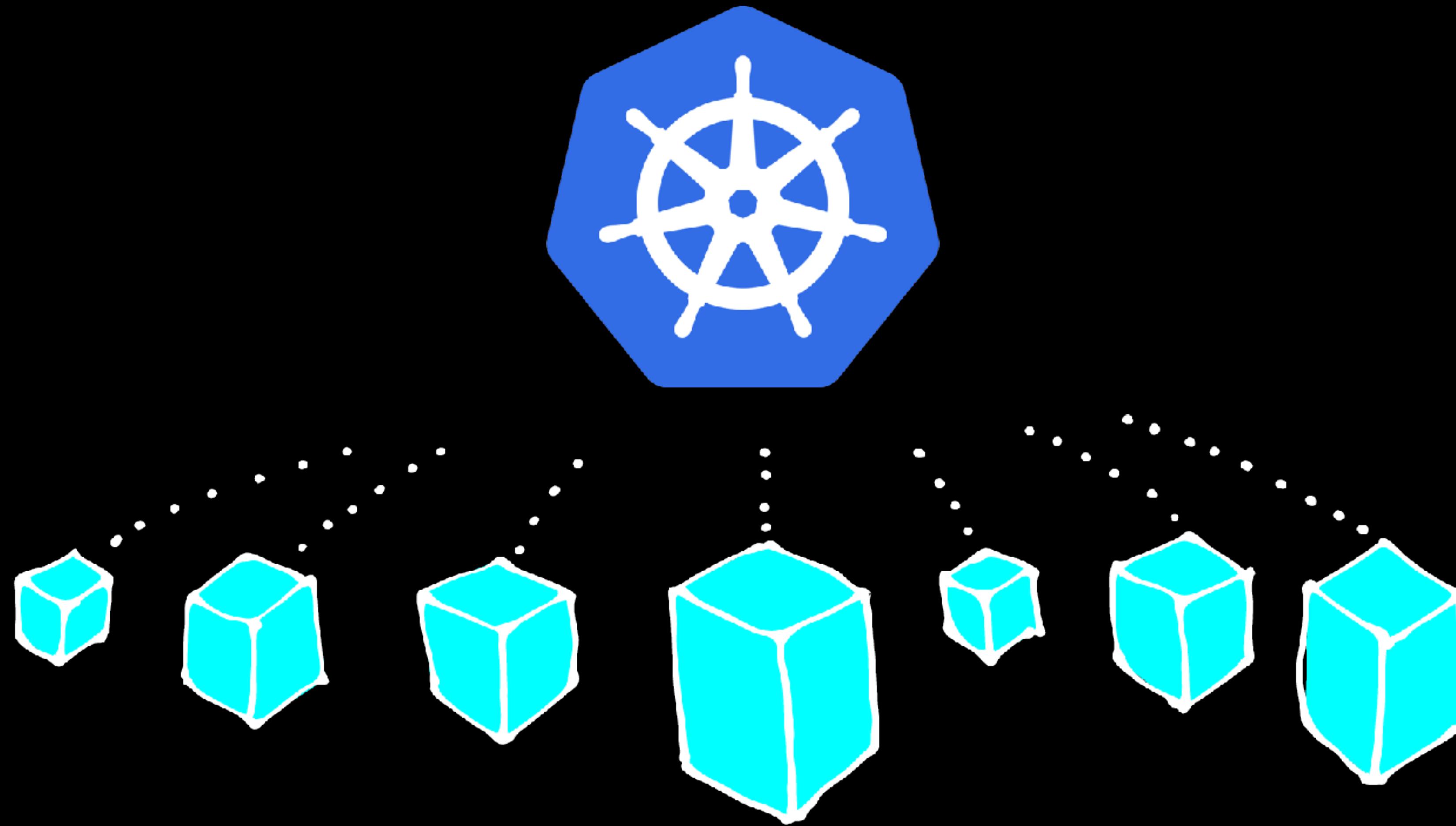
p.s. - Maybe a Squirtle orchestrator? Fun fact: Pokemon Go runs on Kubernetes!

So what if we want to deploy one service?

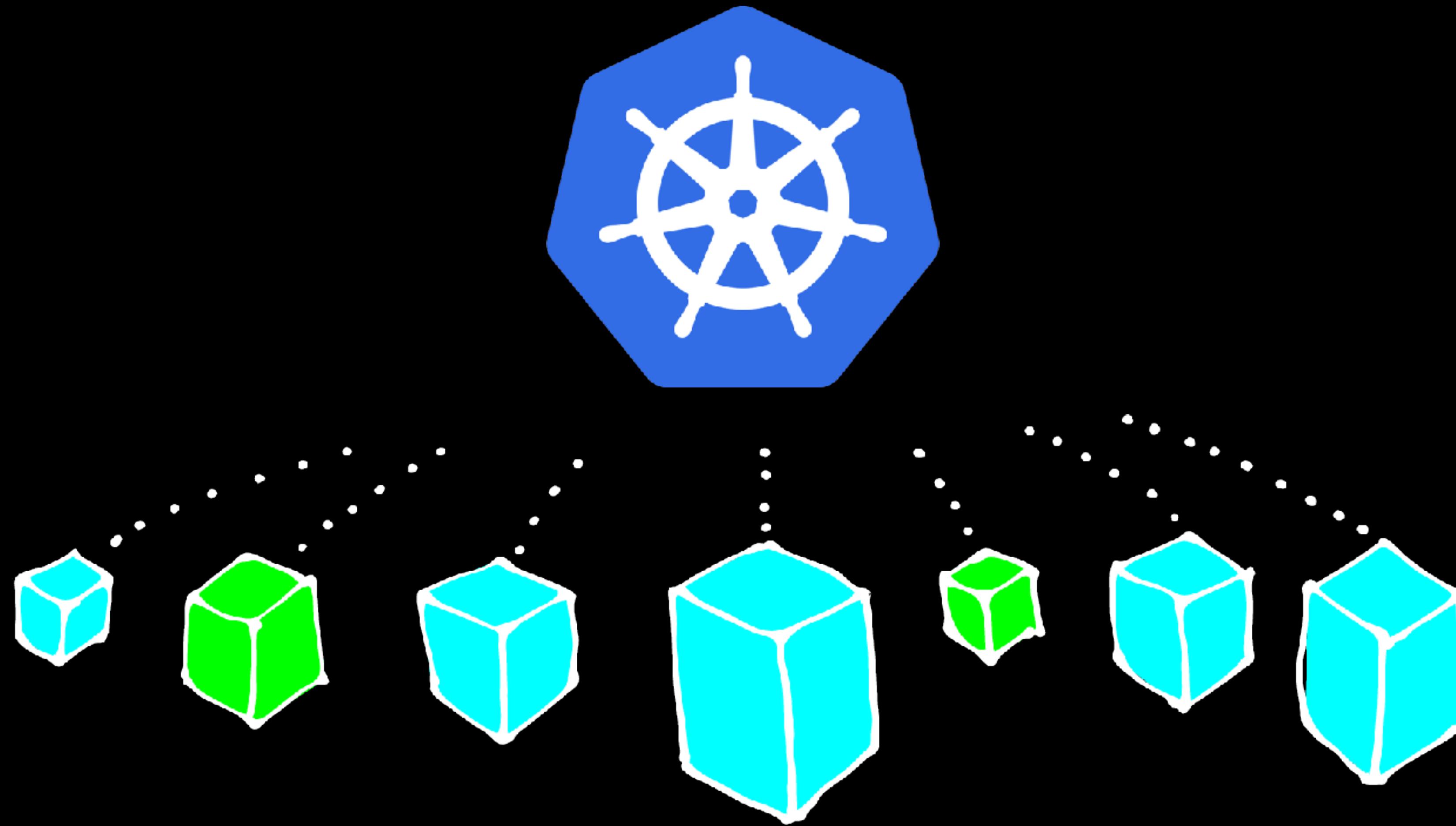




Blue-green doesn't make any sense!

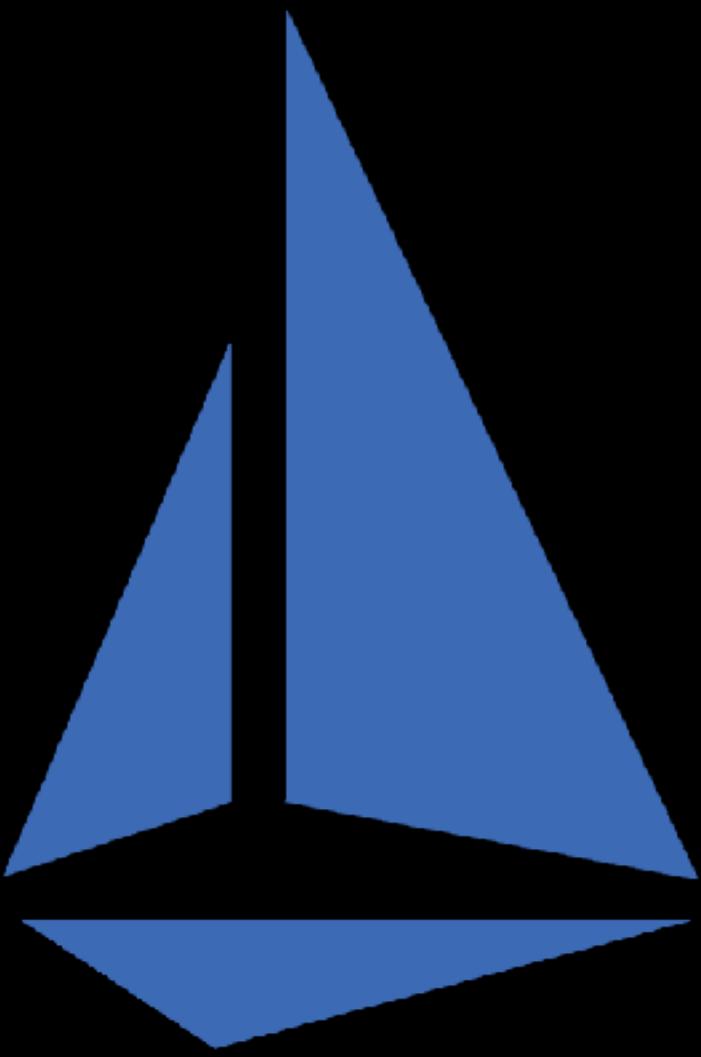


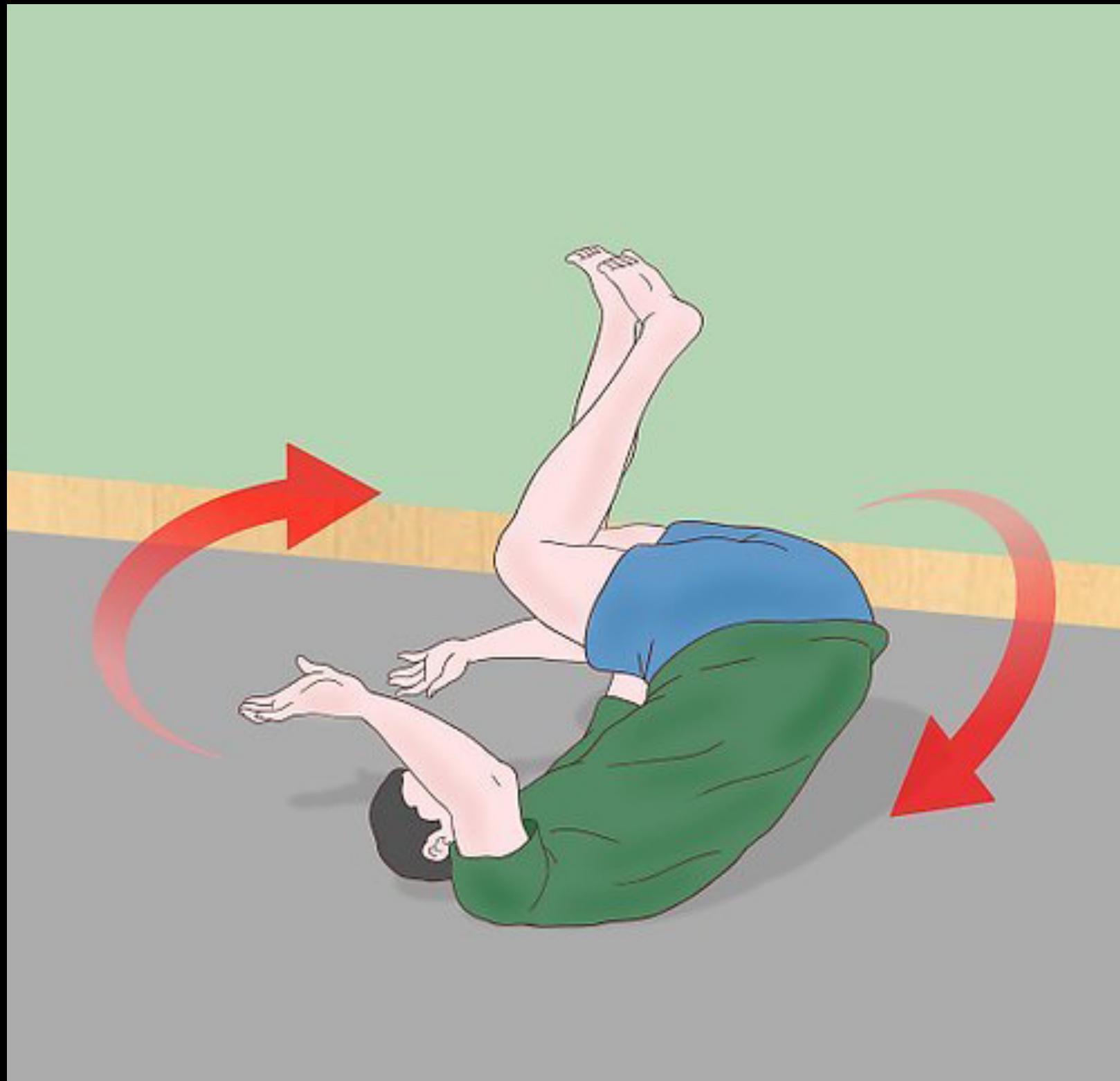
Kubernetes handles service deployments



Kubernetes handles service deployments. YAY!

WHY DO I NEED A
SERVICE MESH?





Kubernetes does
rolling deploys
really well!



Squirtledogg

Canary deploys,
not so much.

CANARY DEPLOYING WITH KUBERNETES

SERVICE

```
apiVersion: v1
kind: Service
metadata:
  name: my-app
  labels:
    app: my-app
spec:
  ports:
  - port: 80
    name: http
  selector:
    app: my-app
```

DEPLOYMENT

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: jyee/my-app:v1
          imagePullPolicy: Always
  spec:
    replicas: 3
    selector:
      matchLabels:
        app: my-app
    template:
      metadata:
        labels:
          app: my-app
    spec:
      containers:
        - name: my-app
          image: jyee/my-app:v2
          imagePullPolicy: Always
```

CANARY?

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 9
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    version: v1
  spec:
    containers:
      - name: my-app
        image: jyee/my-app:v1
        imagePullPolicy: Always
```

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    version: v2
  spec:
    containers:
      - name: my-app
        image: jyee/my-app:v2
        imagePullPolicy: Always
```

CANARY?

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 8
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    version: v1
  spec:
    containers:
      - name: my-app
        image: jyee/my-app:v1
        imagePullPolicy: Always
```

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    version: v2
  spec:
    containers:
      - name: my-app
        image: jyee/my-app:v2
        imagePullPolicy: Always
```

CANARY?

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 7
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    version: v1
  spec:
    containers:
      - name: my-app
        image: jyee/my-app:v1
        imagePullPolicy: Always
```

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    version: v2
  spec:
    containers:
      - name: my-app
        image: jyee/my-app:v2
        imagePullPolicy: Always
```

CANARY?

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 99
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
        version: v1
  spec:
    containers:
      - name: my-app
        image: jyee/my-app:v1
        imagePullPolicy: Always
```



```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
        version: v2
  spec:
    containers:
      - name: my-app
        image: jyee/my-app:v2
        imagePullPolicy: Always
```

WHAT DOES A SERVICE MESH GET YOU?

SERVICE MESHES

- Routing & load balancing

SERVICE MESHES

- Routing & load balancing
- **Service discovery**

SERVICE MESHES

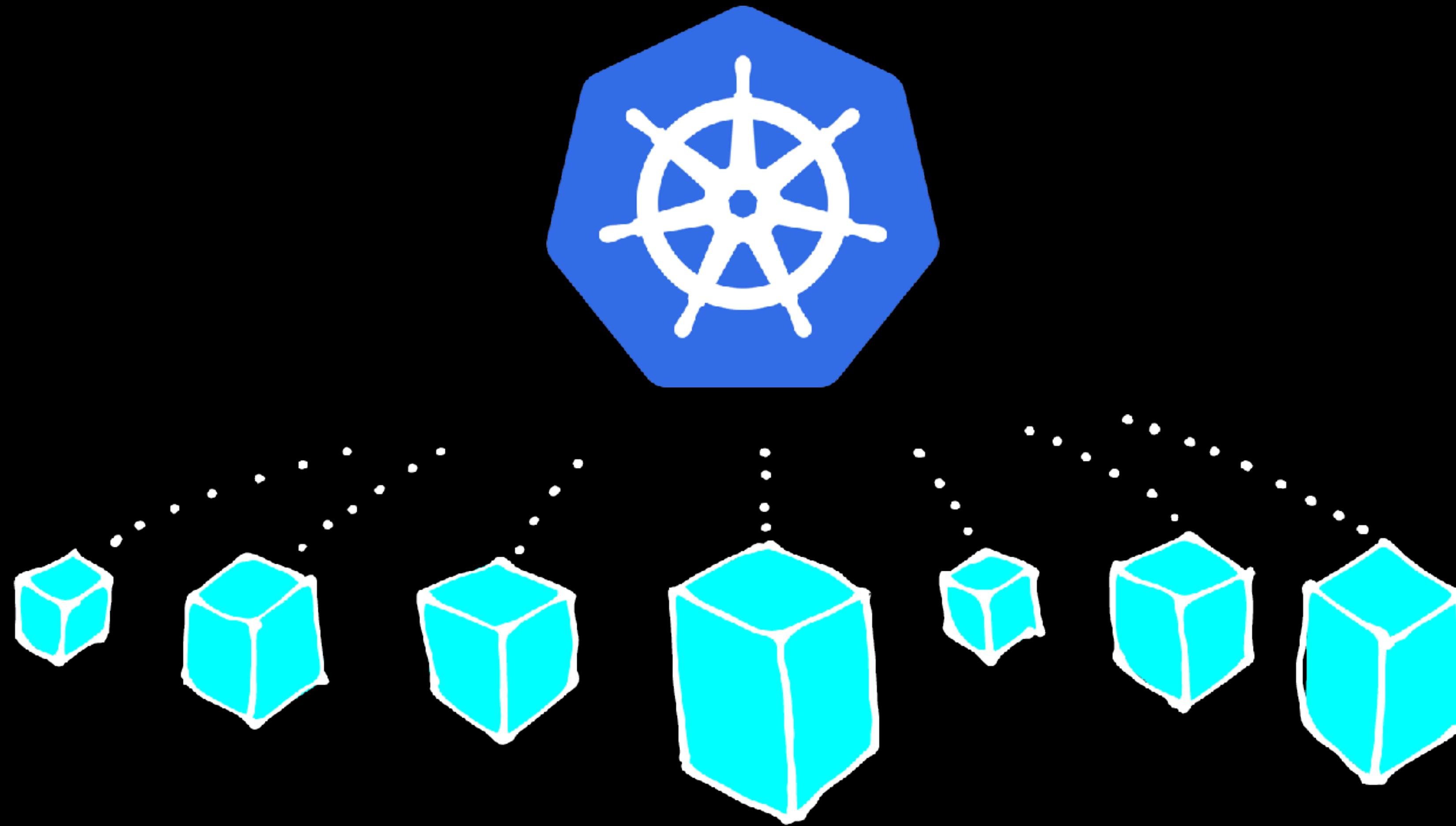
- Routing & load balancing
- Service discovery
- **Timeouts & retries**

SERVICE MESHES

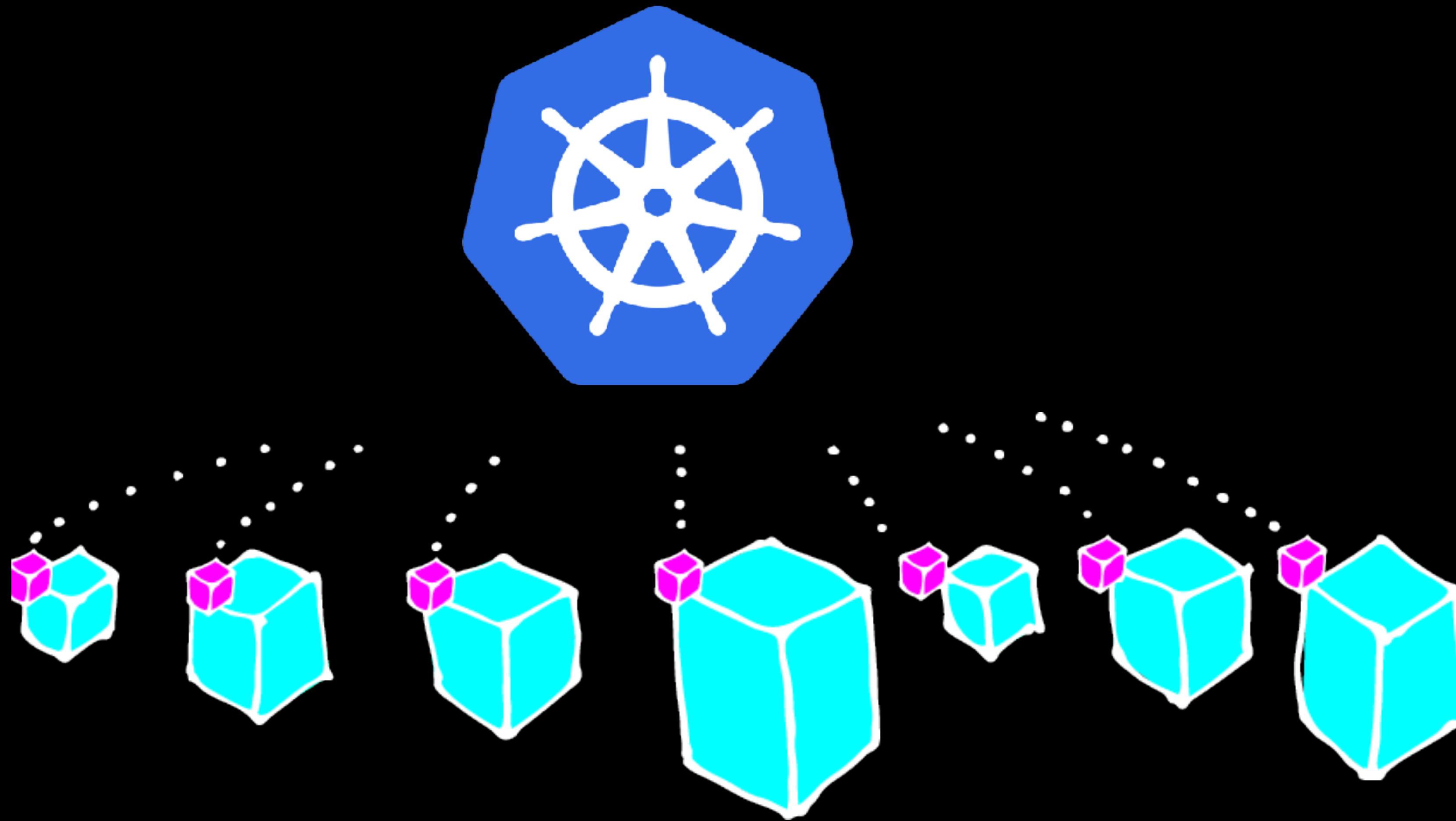
- Routing & load balancing
- Service discovery
- Timeouts & retries
- **Policy enforcement**

SERVICE MESHES

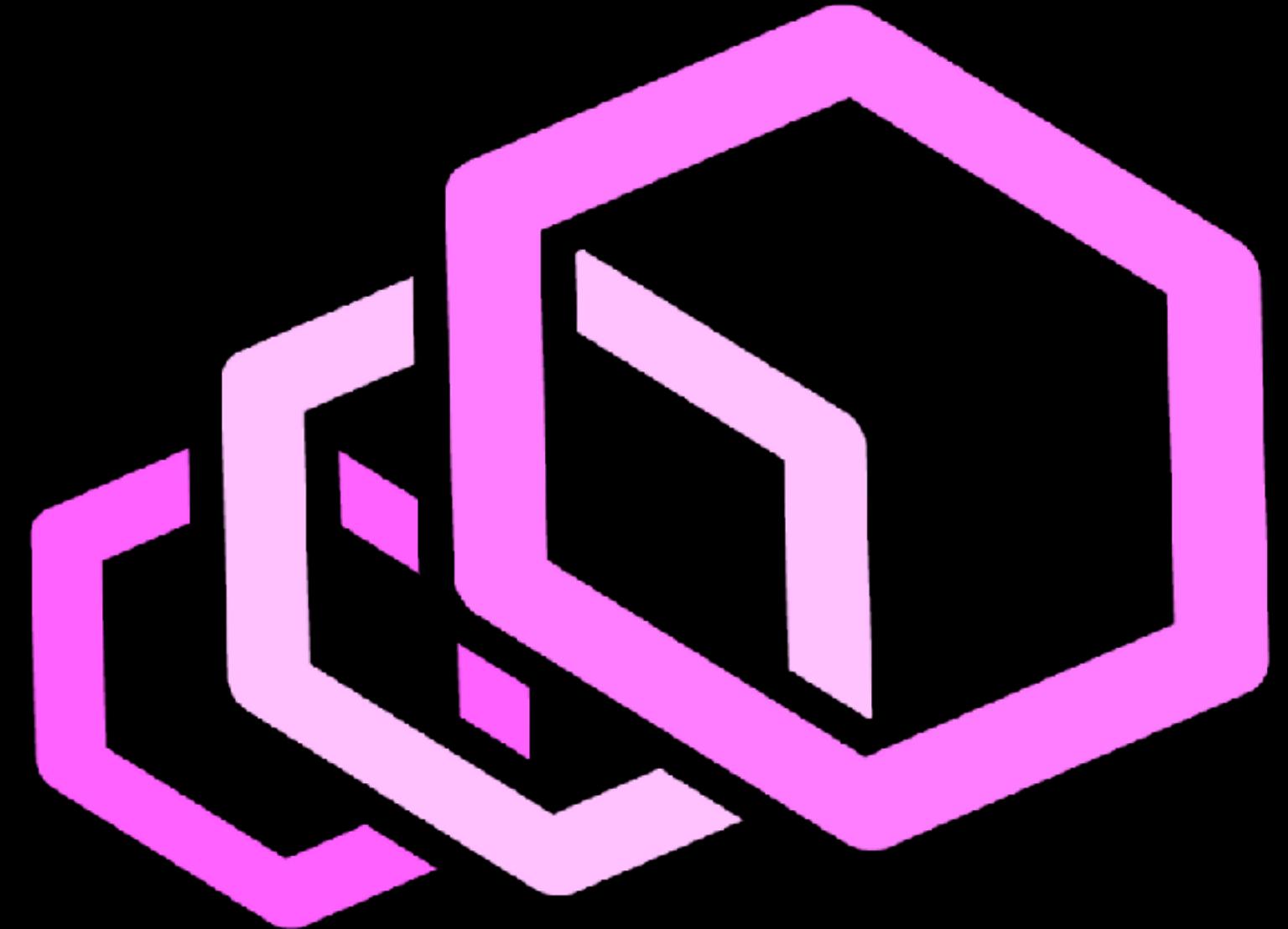
- Routing & load balancing
- Service discovery
- Timeouts & retries
- Policy enforcement
- **Monitoring & tracing**



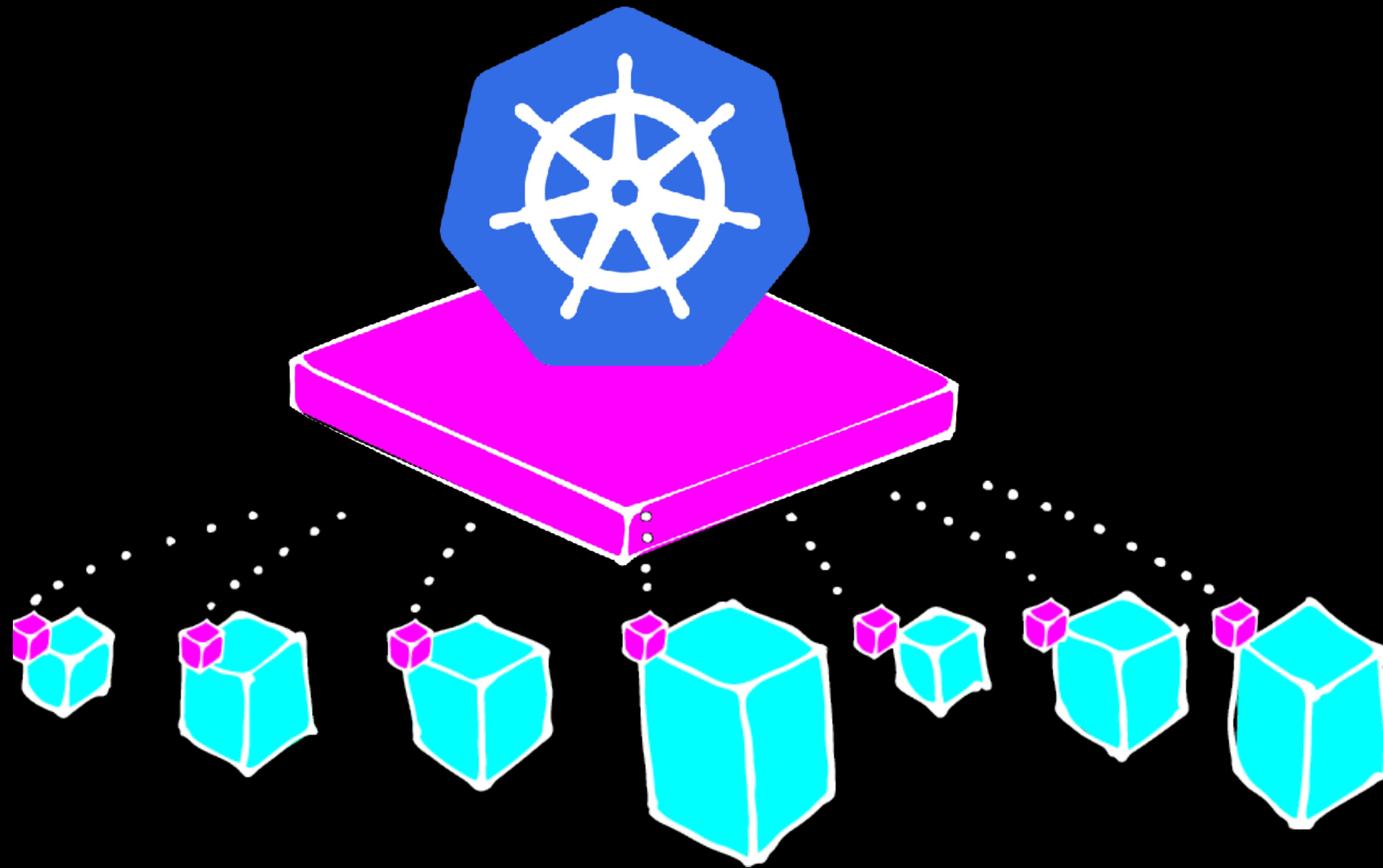
How does it work?



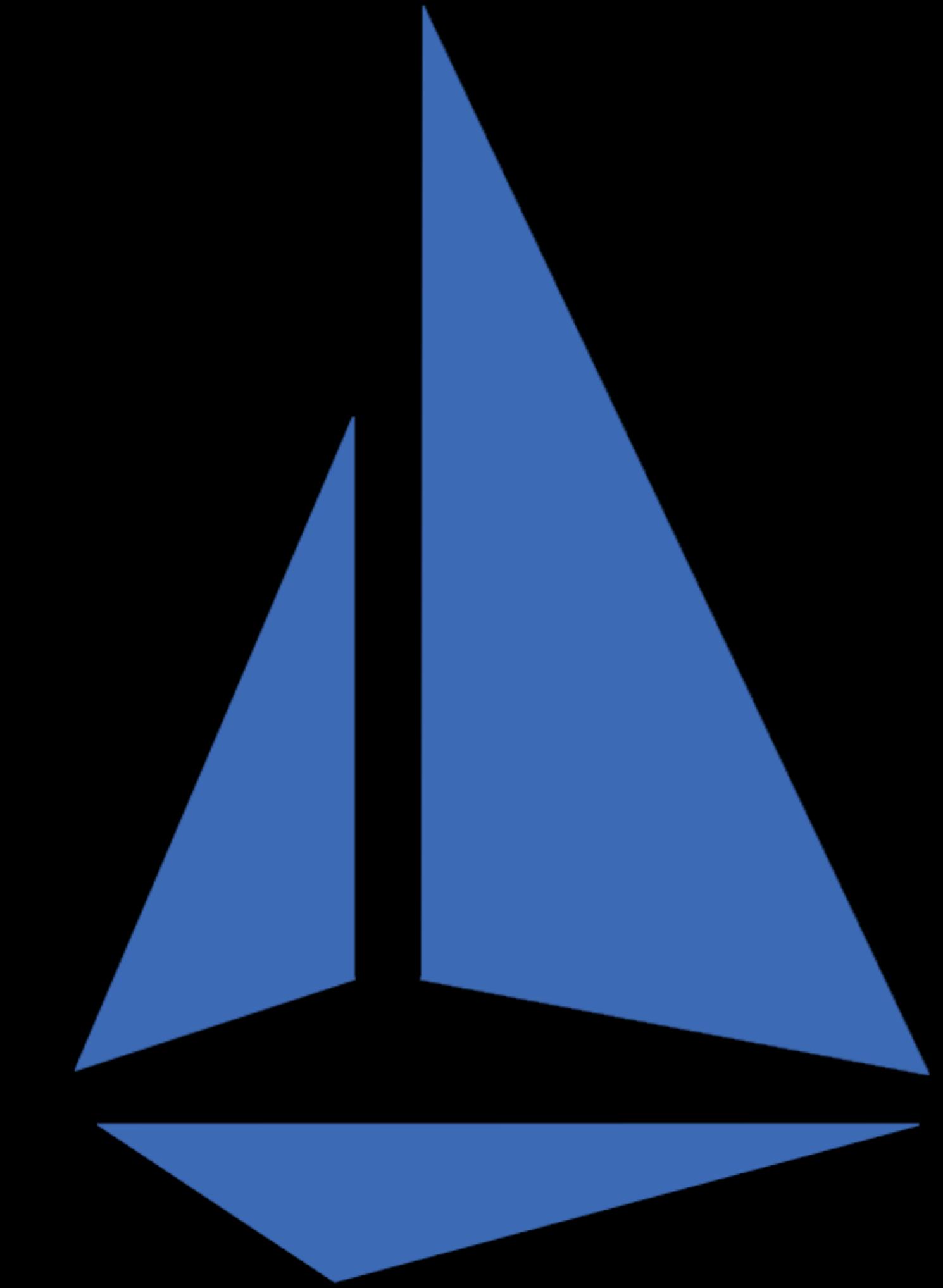
1. Add a data plane



envoy



2. Add a control plane



INSTALLING ISTIO

```
1. curl -L https://git.io/getLatestIstio | sh -
```

INSTALLING ISTIO

1. curl -L https://git.io/getLatestIstio | sh -
2. cd istio-1.0.2

INSTALLING ISTIO

1. curl -L https://git.io/getLatestIstio | sh -
2. cd istio-1.0.2
3. export PATH=\$PWD/bin:\$PATH

INSTALLING ISTIO

1. curl -L https://git.io/getLatestIstio | sh -
2. cd istio-1.0.2
3. export PATH=\$PWD/bin:\$PATH
4. kubectl apply -f install/kubernetes/istio-demo.yaml

WHAT DOES IT INSTALL?

istio-citadel

istio-egressgateway

istio-ingressgateway

istio-pilot

istio-policy

istio-sidecar-injector

istio-statsd-prom-bridge

istio-telemetry

grafana

prometheus

servicegraphng

zipkin

ISTIO SERVICES

- Istio Pilot



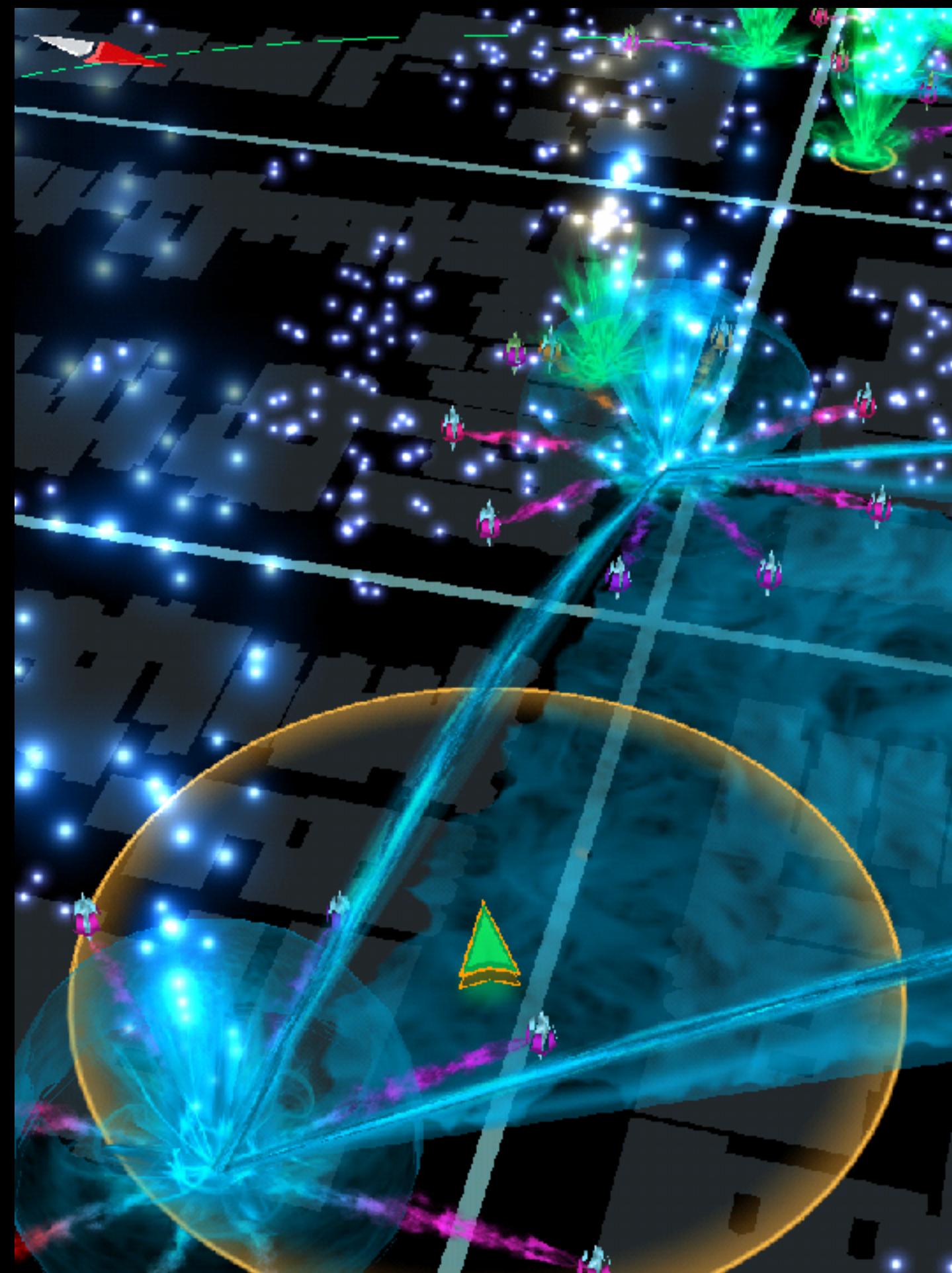
ISTIO SERVICES

- Istio Pilot
- **Istio Mixer**



ISTIO SERVICES

- Istio Pilot
- Istio Mixer
- **Istio Ingress/Egress**



Istio Pokemon Go is next!

ISTIO SERVICES

- Istio Pilot
- Istio Mixer
- Istio Ingress/Egress
- **Istio Citadel**



It has 15,782 steps!

CANARY DEPLOYING WITH ISTIO

SERVICE

```
apiVersion: v1
kind: Service
metadata:
  name: my-app
  labels:
    app: my-app
spec:
  ports:
  - port: 80
    name: http
  selector:
    app: my-app
```

SERVICE

```
apiVersion: v1
kind: Service
metadata:
  name: my-app
  labels:
    app: my-app
spec:
  ports:
  - port: 80
    name: http
  selector:
    app: my-app
```

SAME!

DEPLOYMENT

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
        version: v1
    spec:
      containers:
        - name: my-app
          image: jyee/my-app:v1
          imagePullPolicy: Always
```

DEPLOYMENT

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: jyee/my-app:v1
          imagePullPolicy: Always
```

SAME!
(sort of)

```
istioctl kube-inject -f my.yaml > mod.yaml  
kubectl apply -f mod.yaml
```

TEENAGE MUTATING WEBHOOK ADMISSION CONTROLLERS!

AKA AUTO-SIDECAR INJECTION



ISTIO VIRTUALSERVICES

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: my-app-routing
spec:
  hosts:
    - my-app
  http:
    - route:
        - destination:
            host: my-app
            subset: v1
```

ISTIO DESTINATION RULES

```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: my-app-destination
spec:
  host: my-app
  subsets:
    - name: v1
      labels:
        version: v1
```

ISTIO

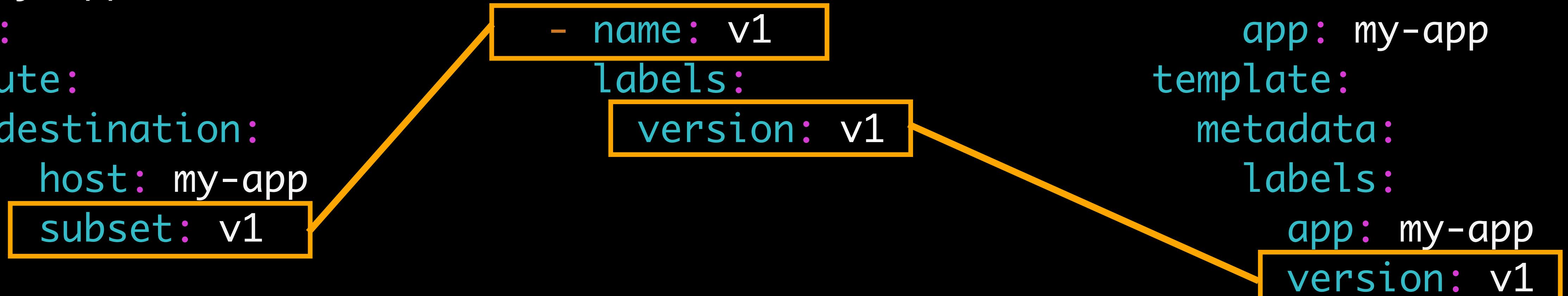
```
kind: VirtualService
...
spec:
  hosts:
    - my-app
  http:
    - route:
      - destination:
          host: my-app
          subset: v1
```

ISTIO

```
kind: DestinationRule
...
spec:
  host: my-app
  subsets:
    - name: v1
      labels:
        version: v1
```

K8S

```
kind: Deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
        version: v1
    spec:
      containers:
        ...
      
```



DEPLOYMENT

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: jyee/my-app:v1
          imagePullPolicy: Always
  spec:
    replicas: 3
    selector:
      matchLabels:
        app: my-app
    template:
      metadata:
        labels:
          app: my-app
    spec:
      containers:
        - name: my-app
          image: jyee/my-app:v2
          imagePullPolicy: Always
```

ISTIO DESTINATION RULES

```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: my-app-destination
spec:
  host: my-app
  subsets:
    - name: v1
      labels:
        version: v1
    - name: v2
      labels:
        version: v2
```

ISTIO VIRTUALSERVICES

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: my-app-routing
spec:
  hosts:
    - my-app
  http:
    - route:
        - destination:
            host: my-app
            subset: v1
            weight: 80
        - route:
            - destination:
                host: my-app
                subset: v2
                weight: 20
```

ISTIO VIRTUALSERVICES

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
...
http:
- match:
  - headers:
    cookie:
      user: my-logged-in-user
route:
- destination:
  host: my-app
  subset: v2
  weight: 20
```

WHAT ELSE CAN IT DO?
LOTS!

<https://istio.io/docs/reference/config/>

RECAP

- Service meshes give you more control

RECAP

- Service meshes give you more control
- **Canary deploys: Representative & Granular**

RECAP

- Service meshes give you more control
- Canary deploys: Representative & Granular
- **Monitoring: Tags, Outliers, Anomalies**

RECAP

- Service meshes give you more control
- Canary deploys: Representative & Granular
- Monitoring: Tags, Outliers, Anomalies
- **What to watch: Latency, Errors, Traffic, Saturation**

RECAP

- Service meshes give you more control
- Canary deploys: Representative & Granular
- Monitoring: Tags, Outliers, Anomalies
- What to watch: Latency, Errors, Traffic, Saturation
- **GO PLAY WITH ISTIO 1.0.2!!!**

QUESTIONS?

Slack: #2018addo-cloudnative

email: jyee@datadoghq.com

twitter: @gitbisect

