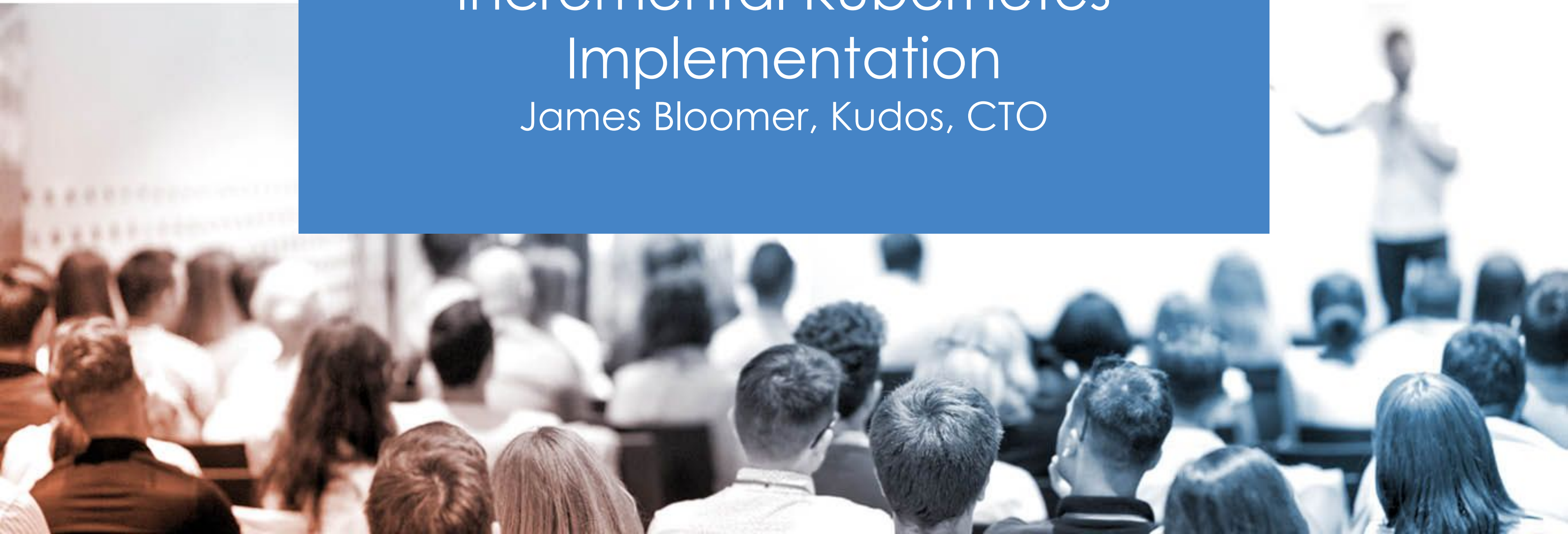
Incremental Kubernetes Implementation

James Bloomer, Kudos, CTO



What we do

For research to have impact, the people who can apply it need to find it and understand it.

This requires effective communication and reach beyond traditional academic activities, such as publishing in journals and giving talks at highly specialist conferences. But researchers have limited time and expertise to reach these broader audiences or to provide evidence of their reach and engagement. **Kudos Pro** fills that gap. We enable researchers to professionalise their communications –without needing a degree in marketing! www.growkudos.com

Who we are

- ✓ Established 2013
- ✓ Kudos is based in Oxford with 29 staff
- ✓ Kudos ‘basic edition’ launched in 2014
- ✓ Kudos Pro is launching Q3 2019
- ✓ 300,000 researcher-users (12% of global researcher market*)
- ✓ Users linked to 7 million publications
- ✓ Users based in 10,000 research institutions in over 190 countries

*OECD estimate 2.4m researchers based within Higher Education Institutions

KUDOS

Before Kubernetes

- ✓ AWS EC2 servers
- ✓ Auto Scaling Group for web servers
- ✓ Other servers not in ASGs
- ✓ Ruby app with capistrano deployment
- ✓ Sidekiq for background jobs
- ✓ MySQL database

KUDOS

Reasons for using Kubernetes

- ✓ Reduce risk of “pet” servers
- ✓ Decrease deployment risk
- ✓ Decrease deployment time
- ✓ Containers over configuration management tools
- ✓ Kubernetes ubiquity
- ✓ Portability between vendors eased the risk

KUDOS

Reasons for using GKE

- ✓ Small team of 5
- ✓ No specialised ops engineers
- ✓ Self hosted Kubernetes too hard for us
- ✓ Low entry cost
- ✓ Spike proved that getting started was trivial

KUDOS

How we started

- ✓ A single, independent, simple service
- ✓ Written in Go, not much code
- ✓ No datastore
- ✓ Simple local Docker build
- ✓ JIT GKE learning
- ✓ Smallest cluster possible

KUDOS

Creating the cluster

- ✓ Tried Google Deployment Manager
- ✓ Tried Terraform
- ✓ Both deleted cluster on change
- ✓ Finally used `gcloud`

KUDOS

Creating the cluster

```
# https://cloud.google.com/sdk/gcloud/reference/container/clusters/create

# Create a GKE cluster with a default node pool
gcloud config set project kudos-prod-1234
gcloud beta container clusters create kudos-production \
  --num-nodes=3 \
  --machine-type=n1-standard-1 \
  --zone=europe-west2-a \
  --disk-size=10 \
  --enable-autoscaling \
  --min-nodes=3 \
  --max-nodes=5 \
  --enable-autorepair \
  --enable-autoupgrade
```

Delivery pipeline in Cloud Build

- ✓ Elastic and cheap (free!) pricing
- ✓ Simple concept yet very flexible
- ✓ YAML config, simple to get started
- ✓ Visualisations of pipelines lacking
- ✓ Good support from team on Slack
- ✓ Deploy with `kubectl apply`

KUDOS

Monitoring

- ✓ Stackdriver for logging and monitoring
- ✓ One tool so one less decision
- ✓ Low cost for our usage
- ✓ GCP metrics in there by default
- ✓ Included our AWS metrics
- ✓ Log metrics created from log statements

Gradual increase in usage

- ✓ Larger number of services
- ✓ Templates for new service creation
- ✓ Utility packages for common functionality
- ✓ Google Datastore for service storage
- ✓ Google Pubsub for events
- ✓ Elasticsearch for time-series database

KUDOS

Multiple cloud vendors

- ✓ Existing system on AWS
- ✓ Interaction between both clouds
- ✓ Calls into AWS via Lamdas
- ✓ Calls into GCP via Google Endpoints
- ✓ Metrics for whole system in Stackdriver

Scaling up the cluster

- ✓ More services, not many pods per node
- ✓ Decided to increase machine size
- ✓ Enables more chance for better utilisation
- ✓ Created a new node pool and switched

Scaling up the cluster

```
# Migrate between node pools
# https://cloud.google.com/kubernetes-engine/docs/tutorials/migrating-node-pool

## List existing nodes
kubectl get pods -o=wide

## List nodes in a specific node pool
kubectl get nodes -l cloud.google.com/gke-nodepool=default-pool

## Cordon nodes
for node in $(kubectl get nodes -l cloud.google.com/gke-nodepool=default-pool
-o=name); do kubectl cordon "$node"; done

## Check nodes are SchedulingDisable
kubectl get nodes

## Drain nodes
for node in $(kubectl get nodes -l cloud.google.com/gke-nodepool=default-pool
-o=name); do kubectl drain --force --ignore-daemonsets "$node"; done

## Check pods have switched node pool
kubectl get pods -o=wide

## Delete old node pool
gcloud container node-pools delete default-pool --zone=europe-west2-a --cluster
kudos-production
```

Extracting services from the Monolith

- ✓ Refactoring complicated functionality
- ✓ New services small and understandable
- ✓ Critical to product functioning
- ✓ Real-time user interaction
- ✓ Zero downtime release

KUDOS

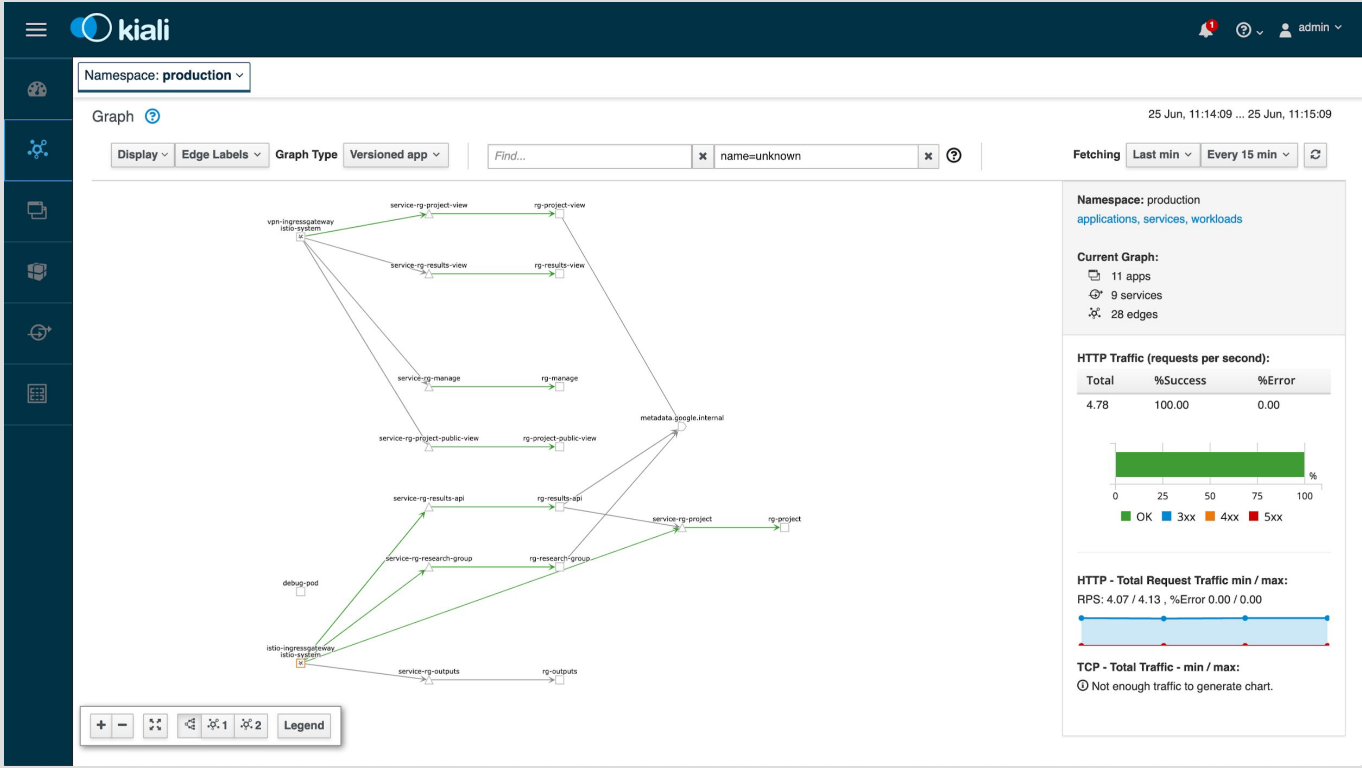
All in on new product

- ✓ User facing via website
- ✓ AWS Application Load Balancer to redirect to new product in GKE
- ✓ Cross cloud VPN to enable redirect
- ✓ Let the cluster auto-scale up nodes
- ✓ Istio for observability and ingress gateway

Istio

- ✓ Beta GKE add-on
- ✓ One click to enable
- ✓ New k8s namespace for services
- ✓ Sidecar auto injection
- ✓ Istio ingress gateway over Google Endpoints
- ✓ Not much best practice advice
- ✓ Kiali for visual graph

Kiali



25 Jun, 11:14:09 ... 25 Jun, 11:15:09

name=unknown

Namespace: production

applications, services, workloads

Current Graph:

11 apps

9 services

28 edges

HTTP Traffic (requests per second):

Total	%Success	%Error
4.78	100.00	0.00

HTTP - Total Request Traffic min / max:

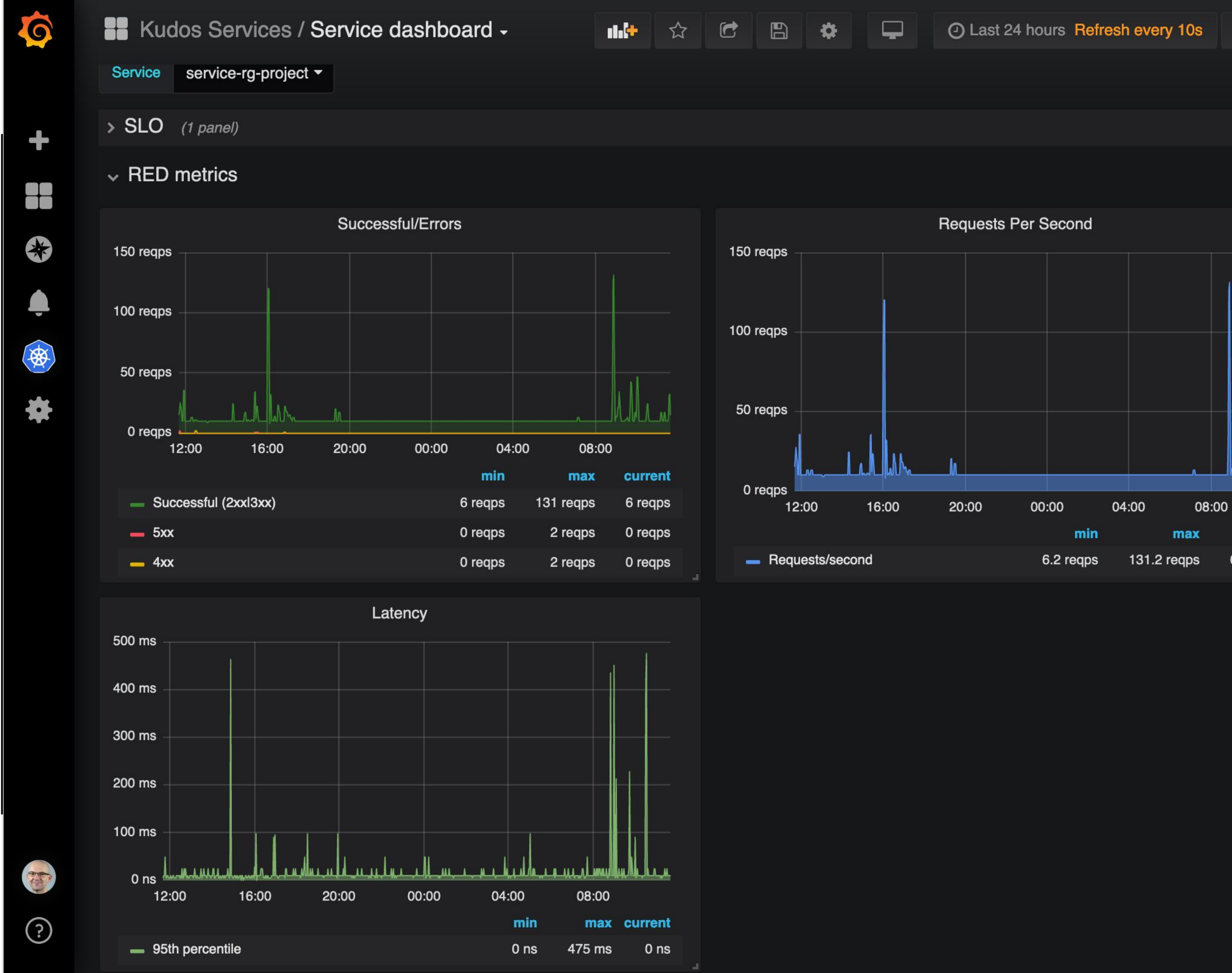
RPS: 4.07 / 4.13 , %Error 0.00 / 0.00

TCP - Total Traffic - min / max:

ⓘ Not enough traffic to generate chart.

KUDOS

Grafana and Prometheus



Kudos Services / Service dashboard

Last 24 hours Refresh every 10s

Service: service-rg-project

SLO (1 panel)

RED metrics

Successful/Errors

	min	max	current
Successful (2xxl3xx)	6 reqps	131 reqps	6 reqps
5xx	0 reqps	2 reqps	0 reqps
4xx	0 reqps	2 reqps	0 reqps

Requests Per Second

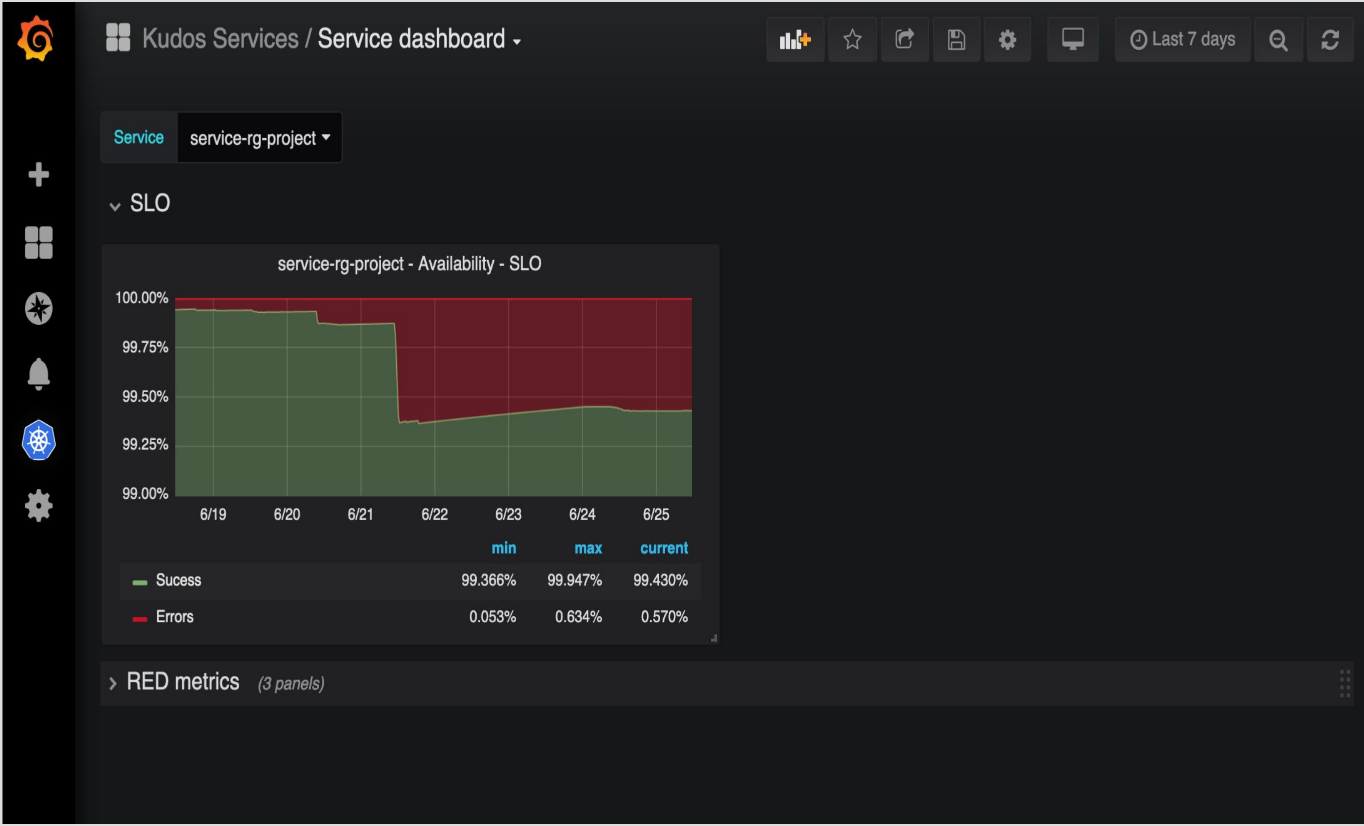
	min	max
Requests/second	6.2 reqps	131.2 reqps

Latency

	min	max	current
95th percentile	0 ns	475 ms	0 ns

KUDOS

SLOs and Error Budgets



	min	max	current
🟩 Sucess	99.366%	99.947%	99.430%
🟥 Errors	0.053%	0.634%	0.570%

Coping with more load

- ✓ Load test a service
- ✓ Observe latency and errors in Grafana
- ✓ Add horizontal pod auto-scaler
- ✓ Increase maximum number of nodes in node auto-scaler

Adding resiliency

- ✓ Rogue service using PIDs and crashing nodes
- ✓ Google support identified the problem
- ✓ Set pod anti-affinity
- ✓ Change number of pods for Istio add-on
- ✓ Added readiness probes and pre-stop script

KUDOS

Now

- ✓ Engineering team of 11
- ✓ 19 supporting services
- ✓ 23 business services
- ✓ ~14 node n1-standard-2 cluster
- ✓ 4 persistent volumes
- ✓ On average around 20 deploys a day

KUDOS

Future

- ✓ Istio for authorisation
- ✓ Istio for security
- ✓ Canary releases
- ✓ Spinnaker?

KUDOS

What we have learned

- ✓ You don’t have to know everything about Kubernetes
- ✓ You don’t need a big team
- ✓ Start small, be incremental
- ✓ An Ops engineer helps with growth
- ✓ Automating was essential

KUDOS

Questions?

James Bloomer

CTO, Kudos

james.bloomer@growkudos.com

Find out how Kudos can help your group achieve greater visibility and impact for your research, through more **effective**, **targeted** and **collaborative** communications.