
Operating a Global Cloud Platform
Josh Michielsen
@jmickey_
CONDÉ NAST


GQ


W


VOGUE


Condé Nast
Traveler


THE WORLD OF
INTERIORS


TATLER


allure


AD
ARCHITECTURAL DIGEST


WIRED


HOUSE
&GARDEN


VANITY FAIR


GLAMOUR

AGENDA

01 **Landscape Overview & Introduction**
A look at the Kubernetes landscape, and what is needed to operate a cluster.

02 **Condé Nast Global Platform**
Overview of the Cloud Platform at Condé Nast built on top of Kubernetes & AWS.

03 **Logging**
Shipping logs with Fluentd makes retrieving logs in-cluster relatively simple. At Condé we pair this with ElasticSearch and Kibana.

04 **Monitoring**
Using Traefik as an ingress controller for public and private ingress for our Kubernetes clusters.

05 **Ingress**
Using Traefik as an ingress controller for public and private ingress for our Kubernetes clusters.

06 **Authentication**
How we manage identity and authentication across multiple clusters.

07 **Application Delivery**
Helm simplifies the packaging and deployment of applications running on Kubernetes.

08 **Supporting Infrastructure**
Managing out-of-cluster infrastructure with Terraform, and cultivating an inner-source community around it.

01

Landscape Overview

CNCF Cloud Native Landscape
2019-11-06T22:36:29Z 6454c43

Overwhelmed? Please see the CNCF Trail Map. That and the interactive landscape are at l.cncf.io

Greyed logos are not open source

App Definition and Development

Database

CNCF Graduated: Vitess
CNCF Incubating: TiKV

Streaming & Messaging

CNCF Incubating: cloudevents, NATS

Application Definition & Image Build

CNCF Incubating: HELM

Continuous Integration & Delivery

Orchestration & Management

Scheduling & Orchestration

CNCF Graduated: kubernetes

Coordination & Service Discovery

CNCF Graduated: CoreDNS
CNCF Incubating: etcd

Remote Procedure Call

CNCF Incubating: gRPC

Service Proxy

CNCF Graduated: envoy

API Gateway

Service Mesh

CNCF Incubating: LINKERD

Runtime

Cloud Native Storage

CNCF Incubating: ROOK

Container Runtime

CNCF Graduated: containerd
CNCF Incubating: cri-o

Cloud Native Network

CNCF Incubating: CNI

Provisioning

Automation & Configuration

Container Registry

CNCF Incubating: HARBOR

Security & Compliance

CNCF Incubating: Notary, Falco, Open Policy Agent, in-toto

Key Management

Platform

Certified Kubernetes - Distribution

Certified Kubernetes - Hosted

Certified Kubernetes - Installer

PaaS/Container Service

Observability and Analysis

Monitoring

CNCF Graduated: Prometheus

Logging

CNCF Graduated: fluentd

Tracing

CNCF Graduated: JAEGER
CNCF Incubating: OPENTRACING

Chaos Engineering

Serverless

Special

Kubernetes Certified Service Provider

Kubernetes Training Partner

Members

CLOUD NATIVE Landscape

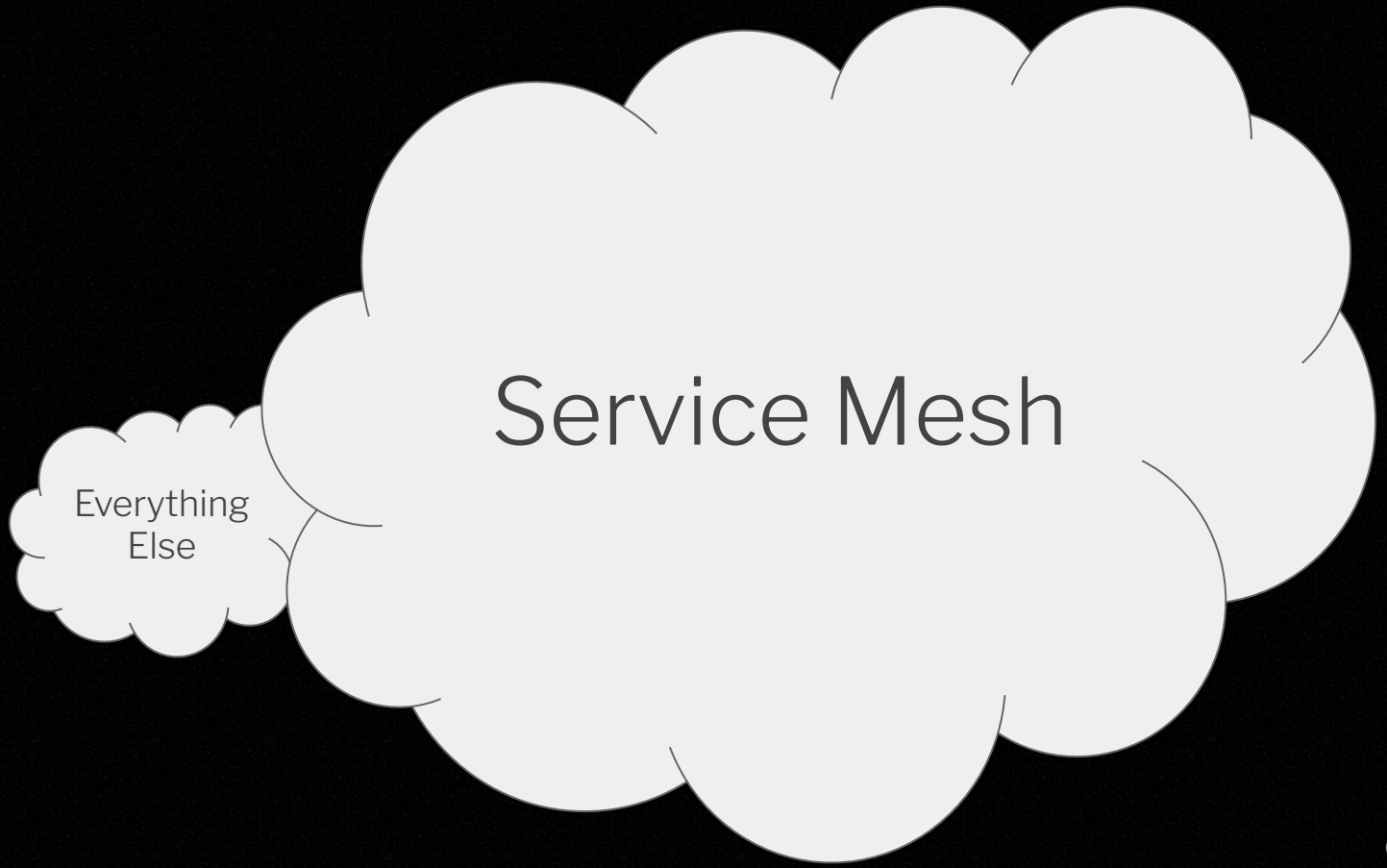
CLOUD NATIVE COMPUTING FOUNDATION

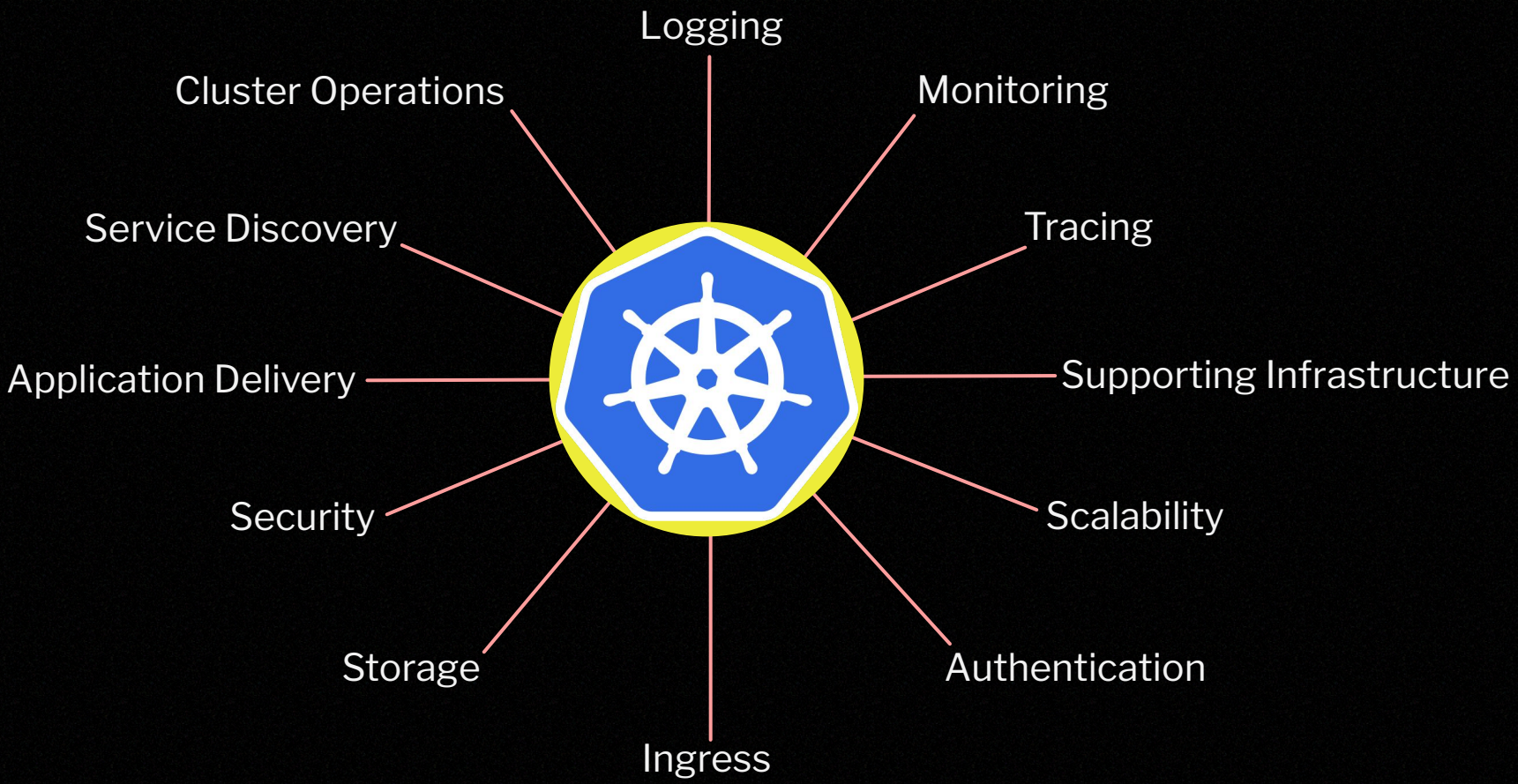
Redpoint Amplify

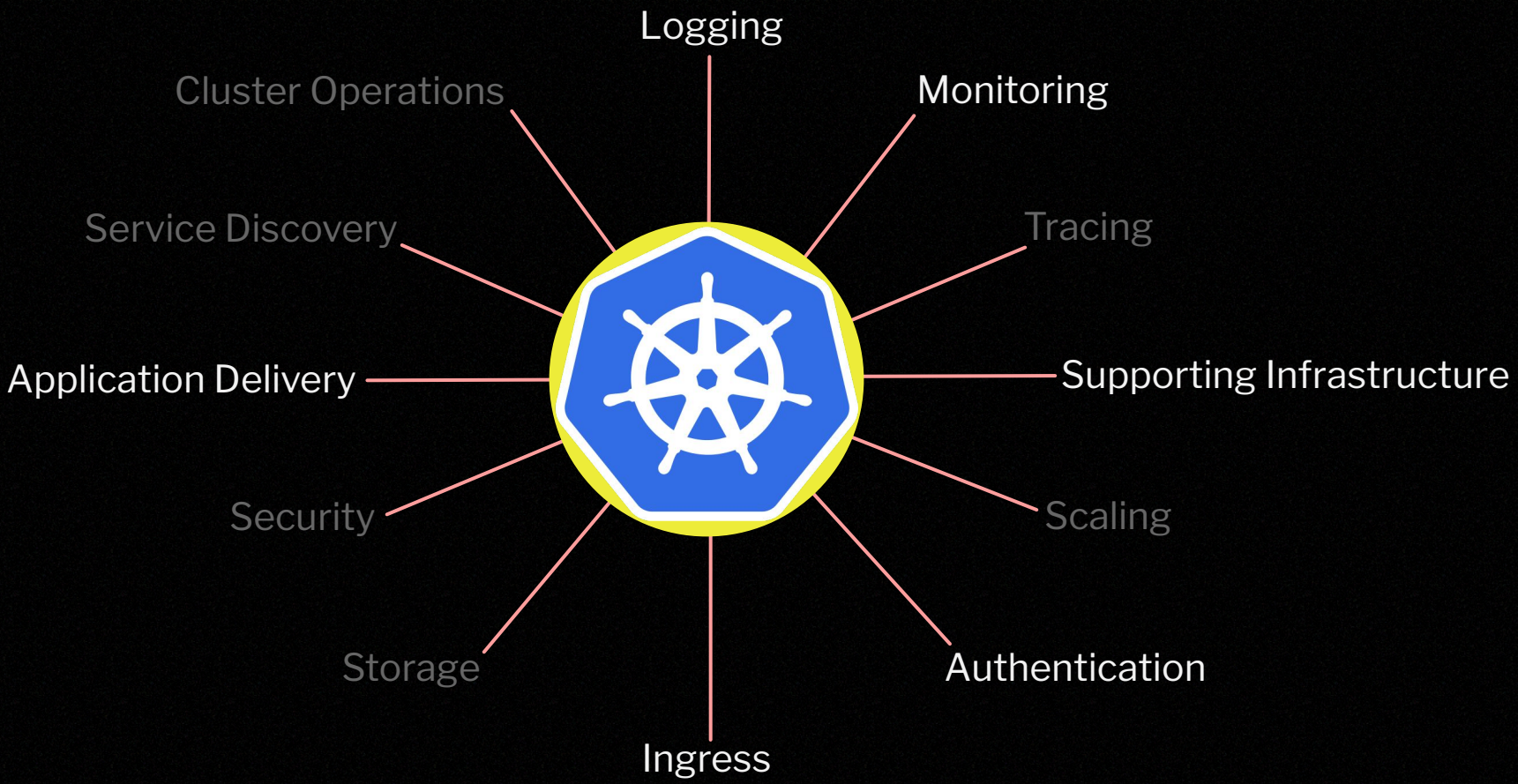
This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.

l.cncf.io

Kubernetes Landscape Word Cloud




Logging
Cluster Operations
Monitoring
Service Discovery
Tracing
Application Delivery
Supporting Infrastructure
Security
Scalability
Storage
Authentication
Ingress


Logging
Cluster Operations
Monitoring
Service Discovery
Tracing
Application Delivery
Supporting Infrastructure
Security
Scaling
Storage
Authentication
Ingress

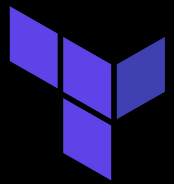
02

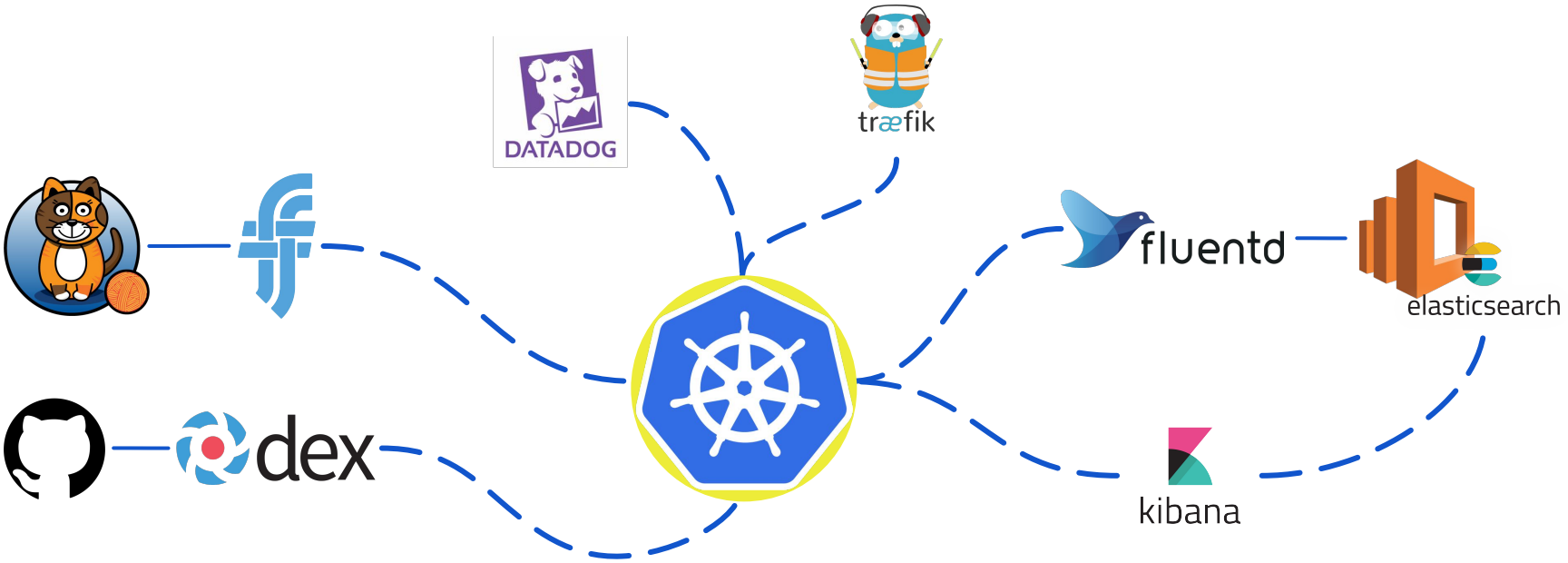
Platform Overview

Global Cloud Platform

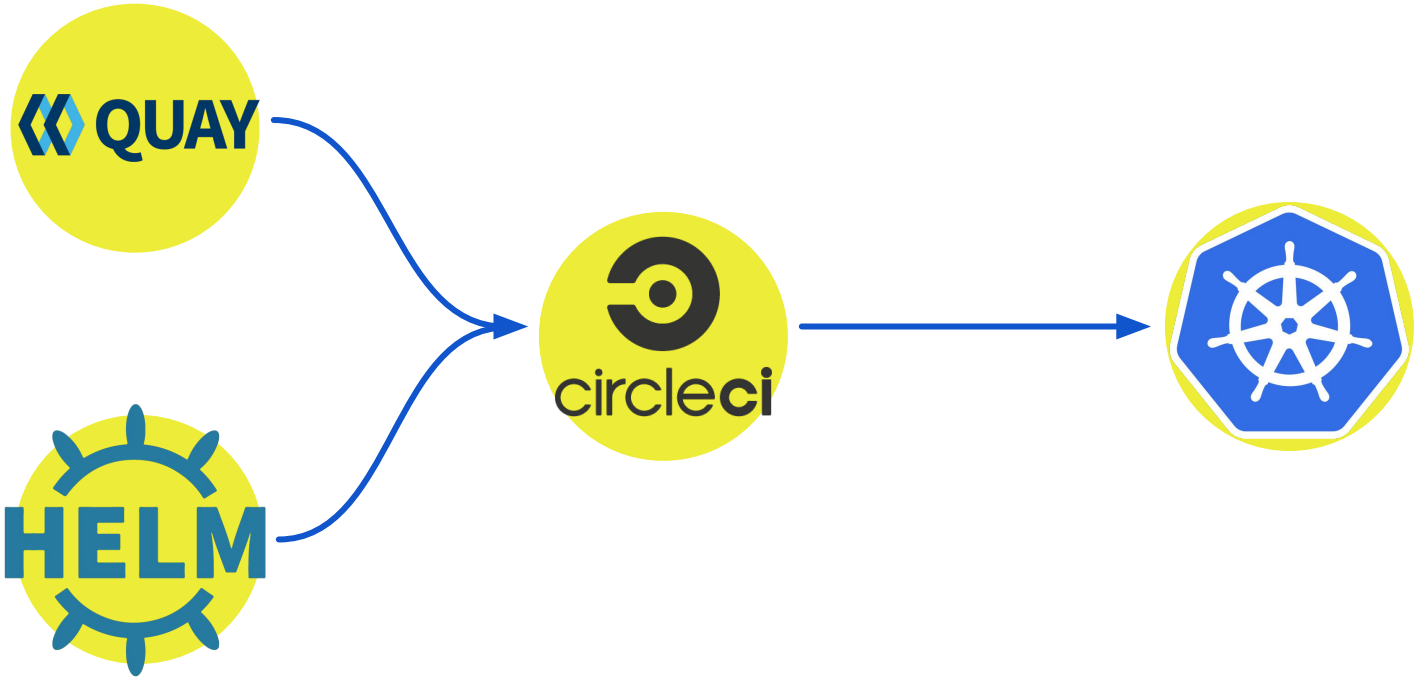
- Clusters in 4 Regions
- 11 Markets
- 180m+ Monthly Pageviews
- 23/34 Publications Migrated

CONDÉ NAST





HELM



03

Logging

fluentd

Fluentd is an open source data collector for unified logging. It provides an easy way to retrieve, process, format, and forward application logs.

Fluentd at Condé

→ Application developers configure their apps to log to stdout.
→ All development teams must adhere to our structured logging standard.
→ Fluentd is deployed as a Kubernetes DaemonSet within its own namespace.
→ Fluentd is configured with access to the local node logs, and the Kubernetes log volume.
→ Logs are processed with additional metadata (e.g. namespace, labes, env, region).
→ Logs are them forwarded to AWS ElasticSearch via a cluster local ES proxy.

CONDÉ NAST

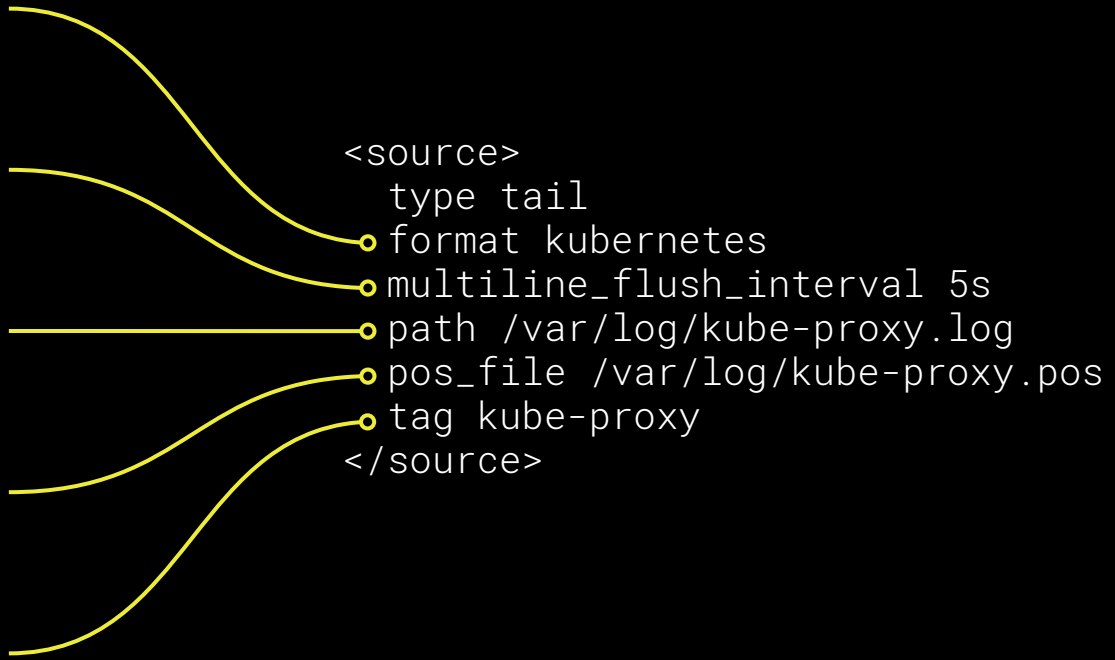
The format for the log line. In this case Kubernetes.

Interval between buffer flushing.

Location of the log file in the node file system.

Store the last position read within the log file.

Tag the log blog with the Kubernetes service.



04

Monitoring

Datadog is a cloud-based metrics and monitoring service. Commonly used for monitoring and alerting on infrastructure, as well as Application Performance Monitoring (APM).

Datadog at Condé

→ Deployed via Helm.
→ Two DaemonSets. One for master nodes, another for workers.
→ Kubernetes PriorityClass on master agents to protect from descheduling.
→ As with all monitoring and alerting, experience is heavily dependant on the implementation.
→ Very little configuration required. Great for quickly getting started.

CONDÉ NAST

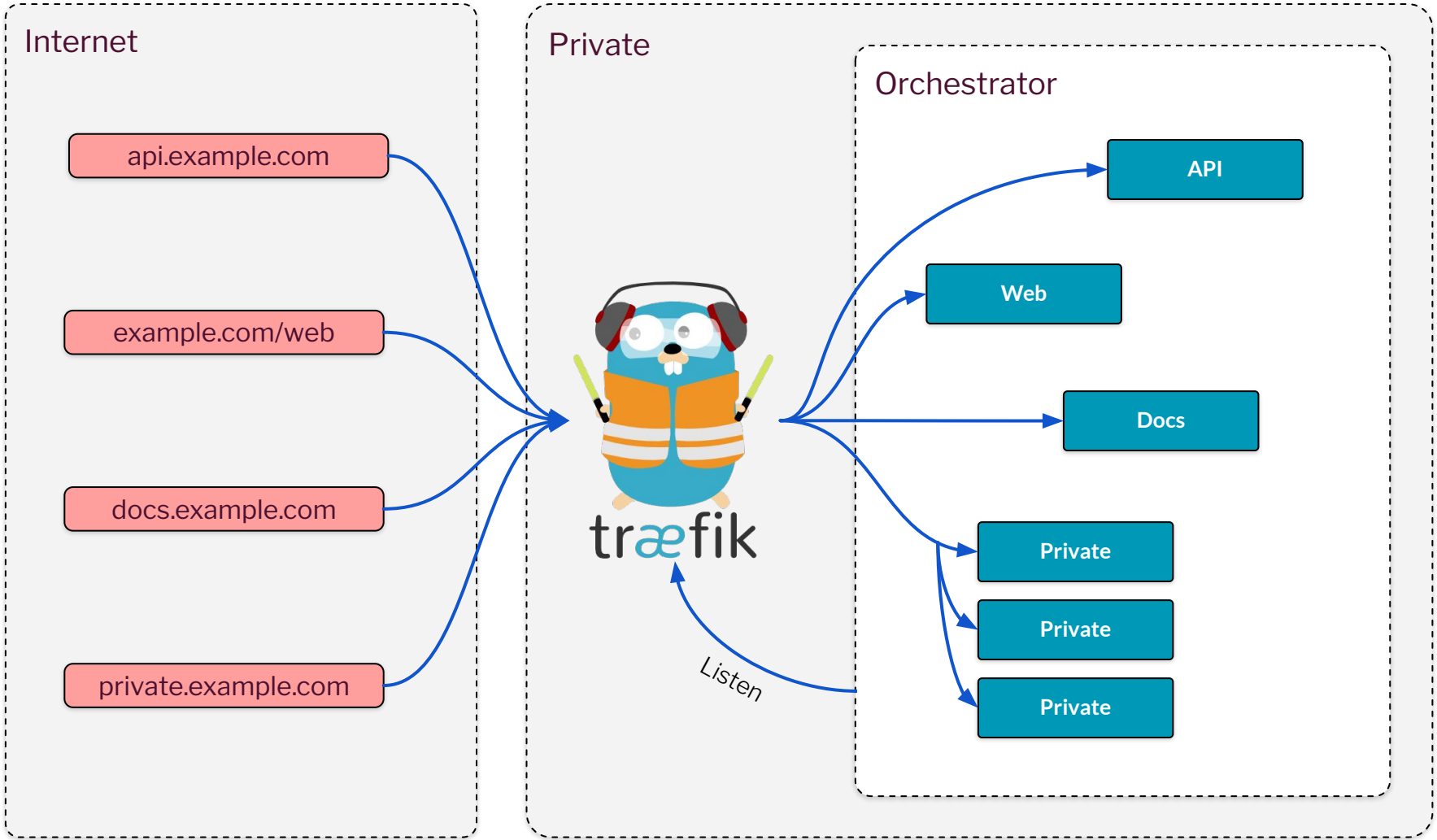
Learnings

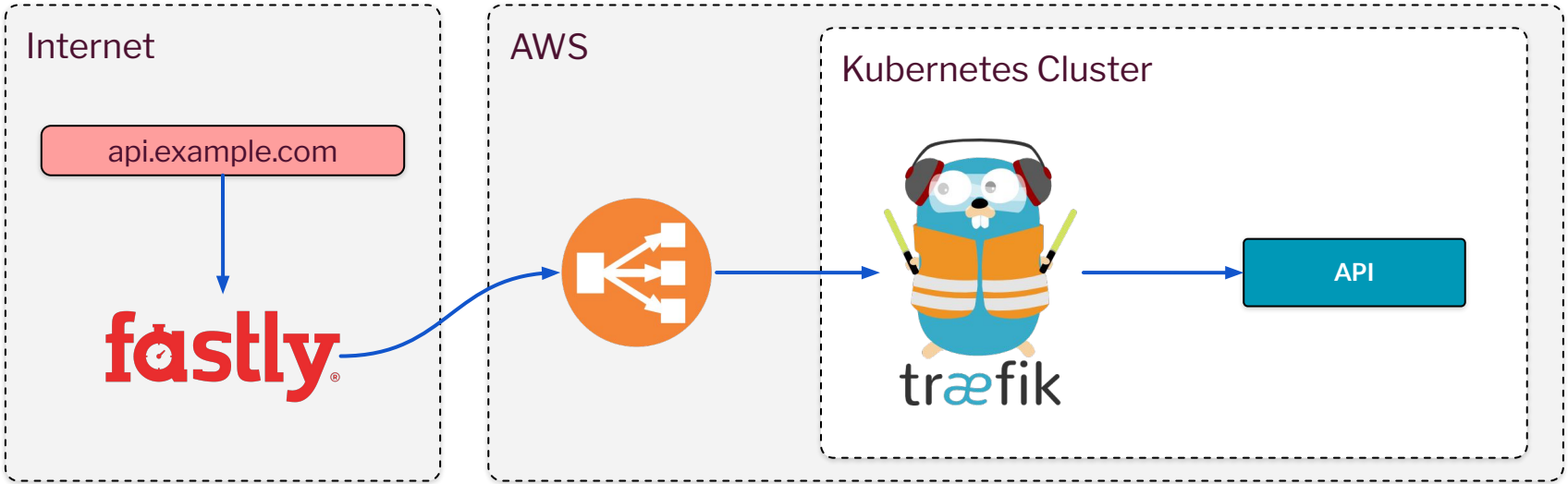
→ Can quickly become expensive as development teams increase the number of custom metrics.
→ Fairly steep learning curve for querying language and formulas.
→ Documentation could be better.
→ Investigation of Prometheus and Thanos for multi-cluster aggregation on the roadmap.

05

Ingress

A modern HTTP reverse proxy and load balancer that makes deploying microservices easy. Traefik integrates with your existing infrastructure components and configures itself “automatically and dynamically”.


Internet
api.example.com
example.com/web
docs.example.com
private.example.com
Private
Orchestrator
API
Web
Docs
Private
Private
Private
træfik
Listen


Internet
api.example.com
fastly
AWS
Kubernetes Cluster
træfik
API

Traefik at Condé

- → Each development team has a namespace.
- → Each namespace has a public ingress, and a private ingress.
- → Certificates are configured on AWS ELBs via AWS ACM.
- → Ingress rules are managed via an `ingress` configuration block within the Helm chart.
- → Enables developers to manage their own application ingress rules. Including allow and block lists.

CONDÉ NAST

```yaml
ingress:
  enabled: true
  traefik: traefik-namespace.private.r53.dns
  type: external
  annotations:
    traefik.ingress.kubernetes.io/frontend-entry-points: http,https
    traefik.ingress.kubernetes.io/redirect-entry-point: https
  rules:
    - host: api.myapp.private.r53.dns
      path: /
```

06

Authentication

Federated OpenID Connector (OIDC) by CoreOS. It acts as a portal that defers authentication to third-party identity providers (IDP) such as Active Directory, SAML, or cloud providers like GitHub and Google.

Auth at Condé

→ GitHub is our IDP, and permissions are managed via GitHub “teams” and Kubernetes RBAC.
→ Okta adopted since the launch of the platform. Migration from GitHub to Okta planned.
→ Custom developer authentication portal that provides a simplified workflow for authenticating with clusters.
→ Service account token are provided within CI/CD pipelines - not visible to developers and rotated periodically.

Kubernetes Auth

Welcome to our Kubernetes cluster 👋

Have a nice day!

SIGN IN

Dependencies

You will need to install `kubectl` before continuing, and to ensure that you keep your own version upto date. Failure to do so could result in interrupted Kubernetes operational usage.

macOS

```
brew install kubernetes-cli
```

linux

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://sto
chmod +x ./kubectl
sudo mv ./kubectl /usr/local/bin/kubectl
```

windows

```
choco install kubernetes-cli
```

Congratulations 🎉

You have successfully authenticated with your authentication provider to enable access to our Kubernetes cluster.

Run the following command locally, to ensure `kubectl` has the appropriate configuration for this environment.

COPY COMMAND

```
kubectl config set-cluster prod_eu-central-1 --server=
kubectl config set-context prod_eu-central-1 --cluster=prod_eu-central-1 --user=github@m
kubectl config set-credentials github@mickey.dev-prod_eu-central-1 --auth-provider=oidc
kubectl config use-context prod_eu-central-1
```

If this is your **first time** connecting to this environment, use the following to setup your default namespace.

```
kubectl config set-context $(kubectl config current-context) --namespace=<a namespace>
```

To confirm everything is working as expected, and that you can access this cluster, please test by running the command `kubectl get pods` .

▶ I'm feeling 🙂

https://github.com/conde-nast-international/kubernetes-auth

@jmickey_

Learnings

→ Inconsistent permissions management between GitHub and Okta. Not a massive issue, but does have a small overhead.
→ Authentication is not federated across clusters. Devs need to authenticate to each cluster they want to query.

07

Application Delivery

A Kubernetes package manager that simplifies the packaging, configuration, and deployment of applications and services onto Kubernetes clusters

Helm Basics

Provides a templating language that can be used to generate standard resource configurations. Charts can be provided a set of override values.

Helm charts can have dependencies, allowing you to modularise your Helm configurations.

When executed, Helm:

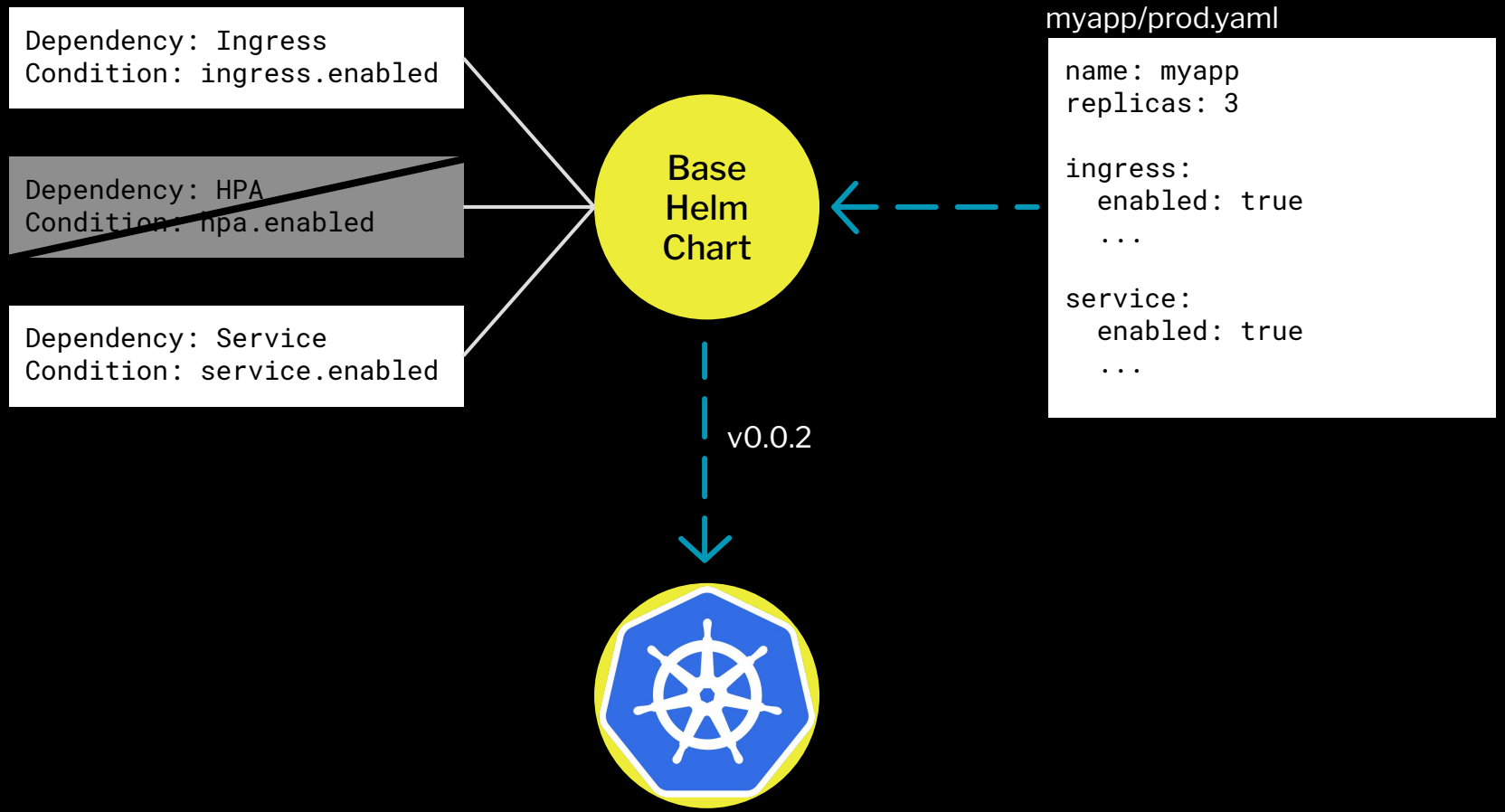
→ Replaces the values in the configuration
→ Builds the resource definitions
→ Deploys them to Kubernetes, and keeps track of all those associated resources
→ All while versioning them as a set (A.K.A a “release”)

```
$ helm create myapp
$ cat myapp/templates/deployment.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ include "myapp.fullname" . }}
  labels:
{{ include "myapp.labels" . | indent 4 }}
spec:
  replicas: {{ .Values.replicaCount }}
  selector:
    matchLabels:
      app.kubernetes.io/name:
        {{ include "myapp.name" . }}
      app.kubernetes.io/instance:
        {{ .Release.Name }}
...
```

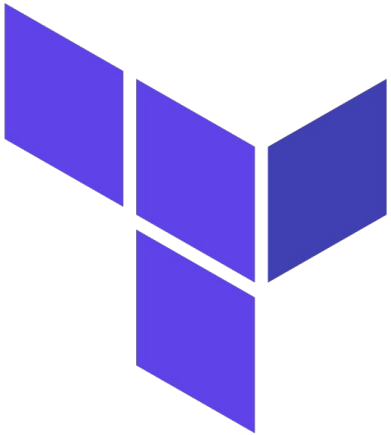
Helm at Condé

- → Single base helm chart used across all development teams.
- → YAML file to provide values for each environment, stored in the application repo.
- → Conditionals on dependencies means developers can choose the features they want to use by simply specifying the config for that feature.
- → We set non-negotiable Helm configuration items that must be included (e.g. Limits).
- → Deployed to Kubernetes from CircleCI.


Dependency: Ingress
Condition: ingress.enabled
Dependency: HPA
Condition: hpa.enabled
Dependency: Service
Condition: service.enabled
Base Helm Chart
myapp/prod.yaml
name: myapp
replicas: 3
ingress:
enabled: true
...
service:
enabled: true
...
v0.0.2

08

Supporting Infrastructure



Terraform provides a declarative language for provisioning, changing, and managing infrastructure for a wide range of tools and services.

Terraform at Condé

- → Terraform code is declared once and reused across environments and regions through variable injection.
- → Continuous delivery pipelines are configured so that devs can update infrastructure without waiting for platform teams to apply changes.
- → Central modules repo that anyone can contribute to.
- → Devs are encouraged to write their own infrastructure code, with PRs being approved by platform.

Terraform at Condé

```
terraform/
├── route53/
│   ├── main.tf
│   ├── variables.tf
│   ├── backend.tf
├── rds/
│   ├── main.tf
│   ├── variables.tf
│   ├── backend.tf
prod/
│   ├── eu-central-1/
│   │   ├── route53/
│   │        ├── terraform.tfvars
│   │        ├── backend.tfvars
staging/
...
```

```
$ cd prod/eu-central-1/route53

$ terraform plan -var-file=terraform.tfvars
  -out=prod-eu-central-1-route53.plan
  ../../../terraform/route53

$ terraform apply prod-eu-central-1-route53.plan
```

CONDÉ NAST

Learnings

- → We were overzealous with modules.
- → The automation of planning and applying terraform is mostly held together by bash scripts. These can be difficult to maintain.
- → IAM permissions for automation CI/CD keys took a little while to get right.
- → Plans are reviewed manually and manual approval is required in CD before apply can happen. Investigating ways to run checks against plans so that this can be automated a bit more.

09

The Future

Prometheus → The introduction of tools like Thanos and Cortex have made managing Prometheus across multiple clusters, envs, and even namespaces much easier.

Weaveworks Flux → GitOps for Kubernetes. Git becomes the single source of truth, and Flux executes automatic remediation when drift occurs.

Service Mesh → mTLS throughout the cluster, retries, service discovery, load balancing, auth(n/z).

Thanks for Listening
Please Rate this Session
We're Hiring! Come Chat

CONDÉ NAST

mickey.dev
@jmickey_
j@mickey.dev
jmichielsen
jmickey

CONDÉ NAST