ORY // summit-21

Architecting Ory Cloud

Andreas Bucksteeg, Lead Engineer at Ory

2021-10-29 | Munich

Goals
Beginning with the end in mind

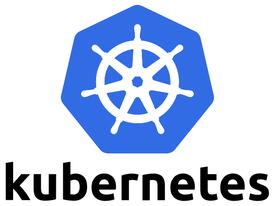
Designed with Security in Mind

Globally available

Highly Automated

Synergies with Open Source

How to deploy our workloads



GitOps
A short intro

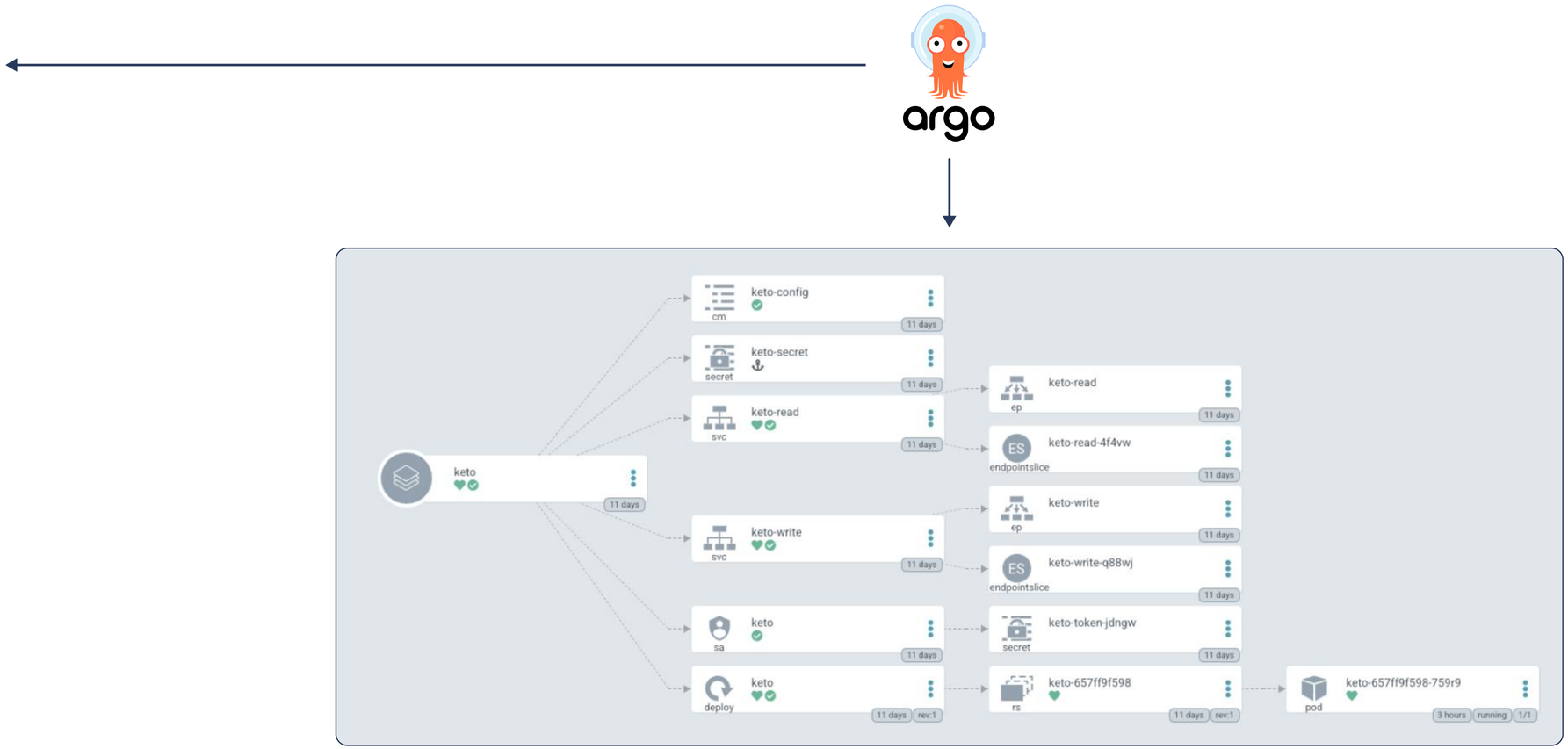
Declarative System Description

Stored in Git

Changes are automatically applied to System

Ory / Keto
Simple example

```yaml
---
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
 finalizers:
   - resources-finalizer.argocd.argoproj.io
 name: keto
 namespace: gitops
spec:
 destination:
   namespace: oasis
   server: https://kubernetes.default.svc
 project: default
 source:
   chart: keto
   helm:
     values: |-
       "deployment":
         "resources":
           "limits":
             "cpu": "250m"
             "memory": "256Mi"
           "requests":
             "cpu": "10m"
             "memory": "32Mi"
       ....
   repoURL: https://k8s.ory.sh/helm/charts
   targetRevision: 0.19.5
 syncPolicy:
   ...
```



Deploying all workloads

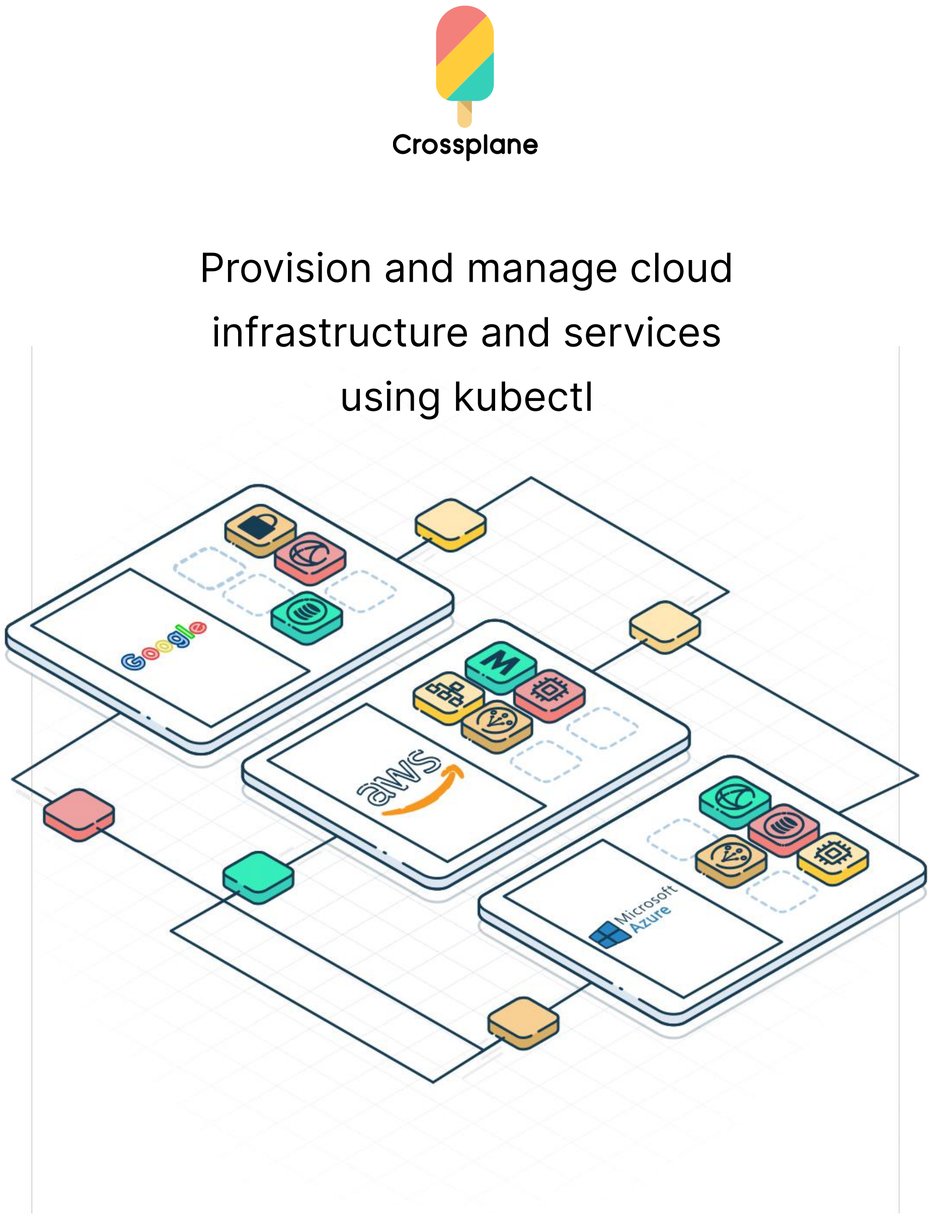
GitHub

Access to private repository

kubernetes

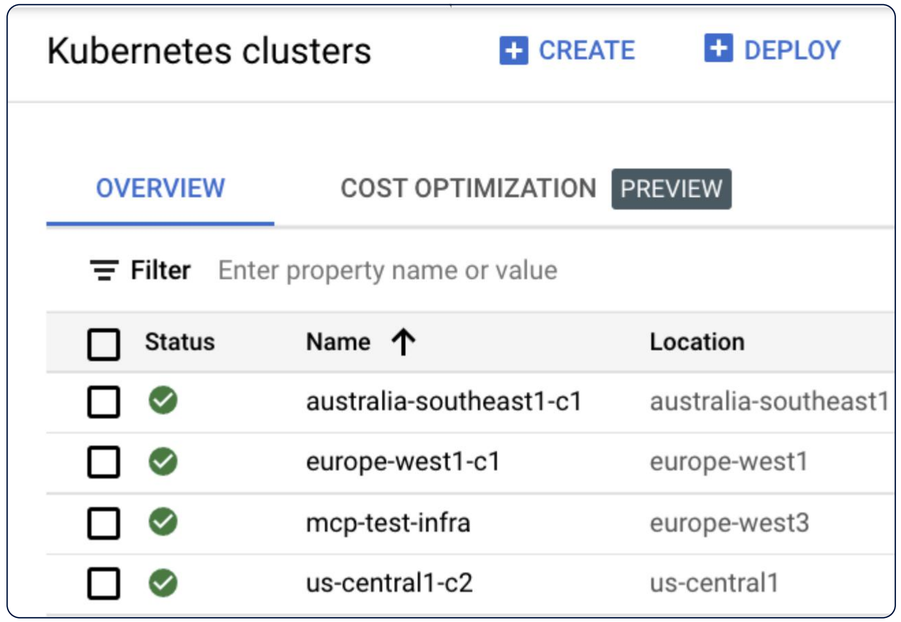
GitOps	Ingress	Observability	Ory
ArgoCD	NGINX Ingress	Prometheus	Kratos
Sealed Secrets	External-DNS	Loki	Keto
ArgoCD Notifications	Cert-Manager	Jaeger	Oathkeeper
		Grafana	Back-Office
			Console

But...
how to bootstrap the infrastructure

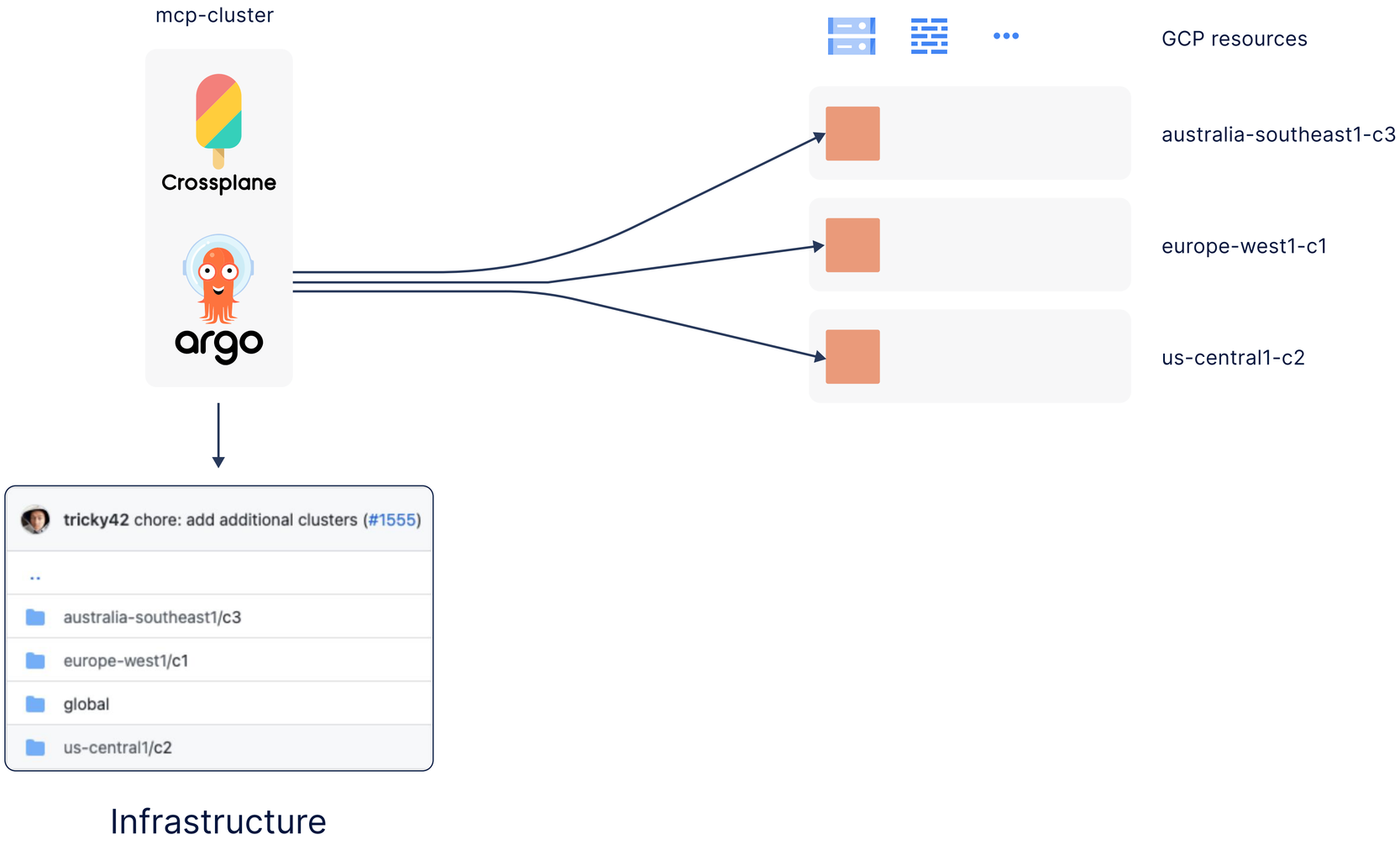


Simple infrastructure example

```yaml
apiVersion: container.gcp.crossplane.io/v1beta1
kind: GKECluster
metadata:
 name: gke-crossplane-cluster
spec:
 forProvider:
   initialClusterVersion: "1.21"
   network: "projects/labs/opsnet"
   subnetwork: "projects/labs/regions/..."
   ipAllocationPolicy:
     useIpAliases: true
   defaultMaxPodsConstraint:
     maxPodsPerNode: 110
   addonsConfig:
     ...
   location: us-central1-a
   ...
   monitoringService: "none"
---
apiVersion: container.gcp.crossplane.io/v1alpha1
kind: NodePool
metadata:
 name: gke-crossplane-np
 ...
```

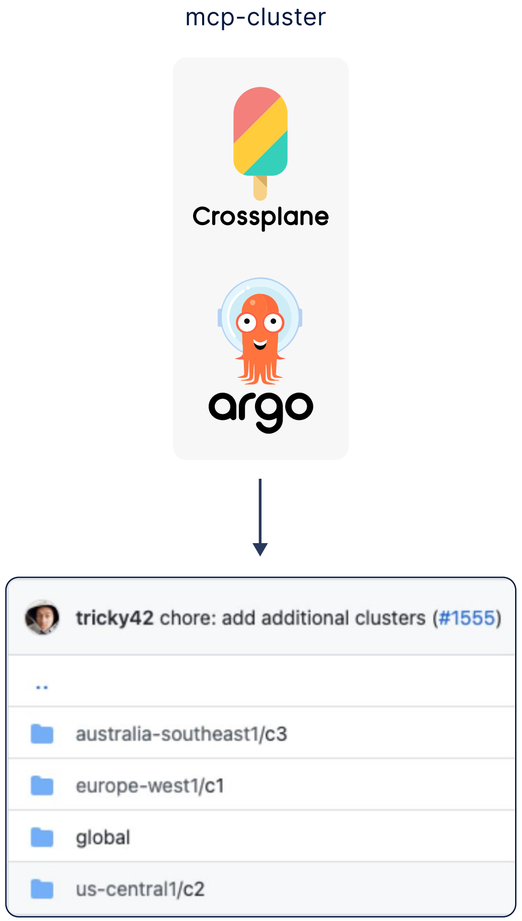


Provisioning a complete environment



Infrastructure

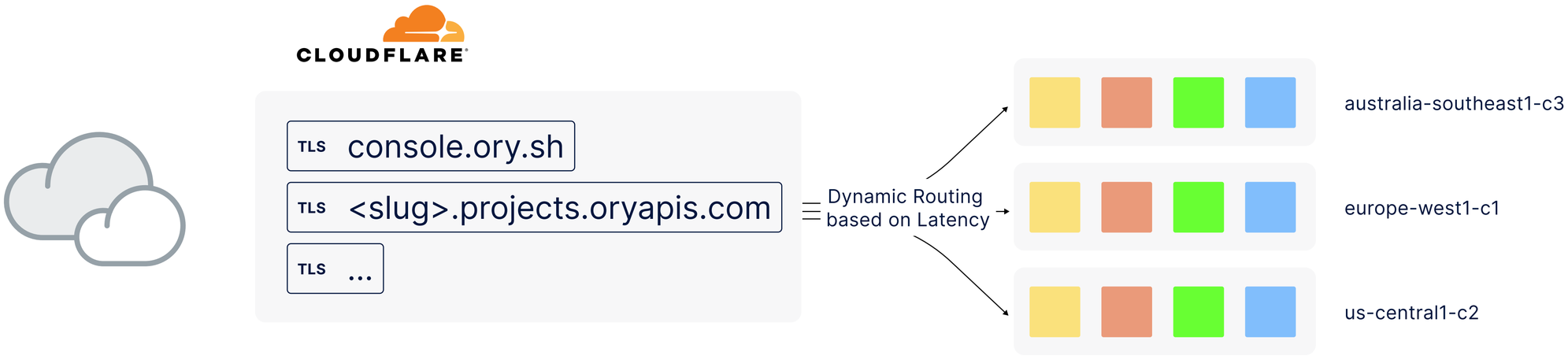
Provisioning a complete environment



Infrastructure

Workload

Accessing the environment



DNS
Load-Balancing
TLS
Custom Hostnames
DDos Protection
Rate-Limiting
CDN

Automated using Pulimi + Metadata from Crossplane

Future: Contribute to Crossplane Cloudflare Provider and switch

Wrapping up

Learnings

Automation can be a double edged sword -> invest heavily in testing and chaos engineering

Validating new versions of third party solutions is hard -> e.g. new versions of kube-prometheus-stack

Next Steps

Fully switch to Crossplane (contribute to Cloudflare Provider)

Integrate Chaos Mesh into our test regime

Globalize Environments

Provide SBOM for Opensource and Ory Cloud

Prepare for Certifications (SOC2, ...)