Scaling containers w/ Multicluster GKE and Istio

Christopher Grant
About Me

- Customer Engineer w/ Google Cloud
- Focus on Containers Microservices DevOps
- Previously AppDev & Architecture - Fortune 50

Find me @
  Web: cgrant.io
  Twitter: cgrant
  Github: cgrant
  AOL: user2735
  MySpace: webdude7913

* no i don’t really use myspace & aol
This session

Single cluster to Enterprise Scale

Progress from simple multi-tenant concepts to a global multicloud GKE mesh with Istio

Highlight tools and practices enroute

Create a cluster

Deploy Globally

How to draw an owl

1.

https://www.flickr.com/photos/centralasian/5229725173
Agenda

- Multi-Tenant
  - Multi-Cluster
    - Resilience
    - Independence
    - Integration
Congratulations !!!!

Your hard work on that Kubernetes POC was a huge success!
Now what?
1 Multi Tenant
Software Multi-tenancy

single instance of software runs on a server and serves multiple tenants.
Kubernetes Multi-tenancy

Providing isolation and fair resource sharing between multiple users and their workloads within a cluster.
Multi-tenancy use cases in Kubernetes

Enterprise

SaaS (Software as a Service)

KaaS (Kubernetes as a Service)
How do I share Kubernetes Engine with other teams?
multi-tenancy modes

multi-tenancy features

trust

access control

isolation

policy management

resource usage

billing

scheduling

preventing contention
Do you trust...

- Your compiler
- Operating system
- Dependencies
- Deployment pipeline
- Container runtime

...
Levels of trust software multi-tenancy

- **Trusted**: The code comes from an audited source, built and run by trusted components (a.k.a. “the dream”)
- **Semi-trusted**: Trusted code, but has 3rd party dependencies or software not fully audited (a.k.a. most people)
- **Non-trusted**: The code comes from potentially hostile users, cannot assume good intent (a.k.a. hosting providers)
Multi-tenancy use cases in Kubernetes

Enterprise ★

SaaS (Software as a Service)

KaaS (Kubernetes as a Service)
Infrastructure & tenants

Give your tenants a safe place to live.

Don’t tell them how to decorate.
Kubernetes Cluster vs Namespace
Kubernetes Engine primitives

Access Control
- Admission Control
- IAM
- RBAC

Resource Sharing
- Quotas
- Pod Priority
- Pod Affinity/Anti-Affinity
- Limit Range

Pod Security Policy
- Sandbox Pods
- Network Policy

Runtime Isolation
Kubernetes Engine multi-tenancy primitives

Auth related:
- IAM
- RBAC
- Admission Control
- Network Policy
- Pod Security Policy

Scheduling related:
- Quotas
- Pod Priority
- Pod Affinity
- Pod Security Context
- Limit Range
- Pod Affinity
- Sandbox Pods

Context:
- Pod Affinity
- Limit Range
- Sandbox Pods
Want to know more?

Watch:
Multi-Tenancy Best Practices for Google Kubernetes Engine (Cloud Next '18)

https://www.youtube.com/watch?v=RkY8u1_f5yY
You wrote all these **policies**, but how do you deploy and manage them in practice?

Keeping Kubernetes/IAM policies up to date across **namespaces / clusters / projects** is difficult!
Policy workflow

Automation and cooperation win the day.
Policy as code

Branch → Validate → Review → Merge → Deploy

Google Cloud
GKE Policy Management **ALPHA**

- Centrally define your entire Cloud Services Platform configuration in code across Istio and Kubernetes
- Integrates with on-prem code repositories for auditability and easy workflow
- Flexible configurations for:
  - Separating different cluster groups (e.g. “dev” or “prod”)
  - Cross-cluster namespaces for development
  - Hierarchical, inheritable access control rules for complex organizations

Participate in alpha: goog.page.link/kpm-alpha
K8s API Server

**Policy Importer**

- **Source of truth**
  - Hierarchical
  - Policies are inherited
  - Namespaces at the leaf nodes

- **Multi-cluster install**
  - Controller containers
  - Sync with Source of Truth and write locally
  - Flattened hierarchy with compiled namespaces and policies
  - Admission controller manages quota across namespaces

**Syncer**

**Quota Controller**

Google Cloud
Git as a source of truth

Directory is the hierarchy

YAML is the YAML

Includes pre-commit validation
Demo: GKE Policy Management
Kubernetes Multi-tenancy Limitations

Kubernetes API:
- Currently API calls are not rate limited, open to DoS from tenants, impacting others.

Networking:
- Networking is not a scheduled resource in Kubernetes, yet (cannot use with limits/requests)
- Tenants can still discover each other via Kubernetes DNS

Many more...
Cluster per Tenant

Tenant (Blue Team) → K8s API → Default Namespace → Cluster

Google Cloud
One Tenant, Many Clusters

Blue Team

Google Cloud
Multi-cluster, Multi-tenant

Blue Team

Green Team

Google Cloud
Early efforts: Federation

Kubernetes Federated Services

Deploy once, deployed everywhere

Good start but left some challenges
Multi Cluster Use Cases

HA Multi Region Deployment

Jurisdictional / Data Sovereignty
- Germany
- US
- Secure records

IT policy

Development Lifecycle
- Prod
- Test
- Dev

Organizational Structures
- Team 1
- Team 2
- Team 3

Hybrid
- Cloud
- On-prem
- Hosted CoLo

Multi-Cloud

Google Cloud
Multiple Clusters
- Resilience
- Independence
- Integration
Increasing Resiliency
The easy way
**Zonal Clusters**

**Standard Cluster:** Utilize a Master with 99.5% SLA, multiple node pools in a single zone

Google Kubernetes Engine, Zonal Cluster

- **Kubernetes Master**
  - us-central1-a

- **Node, us-central1-a**
  - default-pool

- **Node, us-central1-a**
  - node-pool-n

- **Node, us-central1-a**
  - node-pool-n
Multi-zone and **Regional Clusters**

**Regional Clusters:** Enables zero-downtime upgrades and 99.95% uptime by deploying multiple masters across a region.

![Google Kubernetes Engine, Regional Cluster diagram](image-url)
Regional Clusters

As usage continues to grow teams tend to look next toward availability concerns. How to extend beyond one zone to include additional zones and regions.

Masters and nodes spread across 3 zones in a region giving your cluster 99.95%

Single Zone Cluster
```
gcloud container clusters create my-regional-cluster --zone=us-east1-c
```

Regional Cluster
```
gcloud container clusters create my-regional-cluster --region=us-east1
```
Demo: Regional Cluster
Multiple Clusters
- Resilience
- Independence
- Integration
Multi Cluster Use Cases

HA Multi Region Deployment

Jurisdictional / Data Sovereignty

IT policy

Development Lifecycle

Organizational Structures

Team 1  Team 2  Team 3

Hybrid

Cloud  On-prem  Hosted CoLo
Don’t forget about “Microservices?!”
Systems, Services, Microservices
Helpful Tools
**kubectx & kubens**

Manage Context & Namespace

Visit [this GitHub repository](https://github.com/ahmetb/kubectx) to learn more about `kubectx` and `kubens` tools.

This repository provides both `kubectx` and `kubens` tools.

`kubectx` helps you switch between clusters back and forth:

```
$ kubectx coffee
$ minikube
$ test
$ kubectx coffee
Switched to context "coffee".
```

---

[Google Cloud logo]
kube-ps1

Context & Namespace Prompt

https://github.com/onmosco/kube-ps1

kube-ps1: Kubernetes prompt for bash and zsh

A script that lets you add the current Kubernetes context and namespace configured on `kubectl` strings (i.e. the `$PS1`).

Inspired by several tools used to simplify usage of `kubectl`.

→ ~ ( Idol|minikube:default)

→ ~ ( Idol|minikube:default)

→ ~ ( Idol|minikube:default)
Aliases for common commands

This repository contains a script to generate hundreds of convenient kubectl aliases programatically.

Examples
Some of the 800 generated aliases are:

```bash
alias k='kubectl'
alias kg='kubectl get'
alias kgp='kubectl get pod'
```
Cluster Registry

Cluster listing and associated metadata

User Guide

What is the cluster registry?

The cluster registry is a Kubernetes-style API that provides an endpoint for interacting with a list of clusters and associated metadata. If it helps, you can think of the cluster registry as a hosted kubeconfig file. However, since it's a Kubernetes-style API, the cluster registry allows custom annotations and filtering across labels, can be used with `kubectl` and Kubernetes-style generated...

https://github.com/kubernetes/cluster-registry
Cluster Registry

Foundational Piece
- Discovery mechanism
- Common names, labels, annotations

What does it solve:
- Eliminates hard coded lists of clusters or roll your own DBs
- Can operate on consistent lists of clusters with defined meanings (labels/tags)

What problems does it NOT solve:
- Making changes to clusters, or anything else. It is just an index.

Cluster Registry
- Policy-Based Placement
- Shell scripts
- Cluster Lifecycle Service
- Istio Registrar
- Replication Service

Cluster #1
- zone = us-central1-a
- compliance = pci-dss
- type = prod

Cluster #2
- zone = europe-west3-a
- gpus = k80
- type = dev

Cluster #3
- zone = on-prem-europe
- compliance = corp
- type = test
Welcome Jon Friedman, Freight Farms, Co-Founder

Demo: Cluster Registry
Multiple Clusters
- Resilience
- Independence
- Integration
Integration

Traffic can be routed to clusters with DNS or GCLB

Cross Cluster Service communication can be enabled with Istio
Istio Value Proposition

- Service Level Security
- Uniform Observability
- Routing Traffic Shaping
## Istio Value Proposition

<table>
<thead>
<tr>
<th>Service Level</th>
<th>Uniform</th>
<th>Routing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Security</strong></td>
<td><strong>Observability</strong></td>
<td><strong>Traffic Shaping</strong></td>
</tr>
<tr>
<td>mTLS Auth</td>
<td>Distributed Tracing</td>
<td>Traffic Routing</td>
</tr>
<tr>
<td>RBAC</td>
<td>Service Graphs</td>
<td>Request Routing</td>
</tr>
<tr>
<td>- namespace-level</td>
<td>Log with Fluentd</td>
<td>Discovery &amp; Load Balancing</td>
</tr>
<tr>
<td>- service-level</td>
<td>Collect TCP Metrics</td>
<td>Rules Configuration</td>
</tr>
<tr>
<td>- method-level</td>
<td>Query Metrics</td>
<td>Fault Injection</td>
</tr>
<tr>
<td></td>
<td>Visualize Metrics</td>
<td>Failure Handling</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Timeouts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Bounded retries</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Concurrent connection limits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Active health checks</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Fine-grained circuit breakers</td>
</tr>
</tbody>
</table>

---

### Security
- namespace-level
- service-level
- method-level

### Observability
- Distributed Tracing
- Service Graphs
- Log with Fluentd
- Collect TCP Metrics
- Query Metrics
- Visualize Metrics

### Routing
- Traffic Routing
- Request Routing
- Discovery & Load Balancing
- Rules Configuration
- Fault Injection
- Failure Handling
  - Timeouts
  - Bounded retries | timeout budgets
  - Concurrent connection limits
  - Active health checks
  - Fine-grained circuit breakers
Istio on GKE

Automatic install and upgrades of istio Control Plane and Nodes

Quickly create a cluster with all the components you need to create and run an Istio service mesh, in a single step. Once installed, your Istio control plane components are automatically kept up-to-date, with no need for you to worry about upgrading to new versions. You can also use the add-on to install Istio on an existing cluster.

Old Way

- Install helm
- Git clone istio
- kubectl apply crds
- kubectl apply service-account
- helm init
- helm install istio

Easy Way

gcloud beta container clusters create istio-demo
--addons=Istio

https://cloud.google.com/istio/docs/istio-on-gke/overview
**Architectural components**

**Pilot**: Control plane to configure and push service communication policies.

**Envoy**: Network proxy to intercept communication and apply policies.

**Mixer**: Policy enforcement with a flexible plugin model for providers for a policy.

**Istio Auth**: Service-to-service auth[n,z] using mutual TLS, with built-in identity and credential management.
Bookinfo Application

This example deploys a sample application composed of four separate microservices used to demonstrate various Istio features. The application displays information about a book, similar to a single catalog entry of an online book store. Displayed on the page is a description of the book, book details (ISBN, number of pages, and so on), and a few book reviews.

The Bookinfo application is broken into four separate microservices:
Review Bookinfo architecture
Multicluster Istio

Istio Multicluster

Instructions for the installation of Istio multicluster.

Prerequisites

- Two or more Kubernetes clusters with 1.9 or newer.
- The ability to deploy the Istio control plane on one Kubernetes cluster.
- The usage of an RFC1918 network, VPN, or alternative more advanced network techniques to meet the following requirements:
  - Individual cluster Pod CIDR ranges and service CIDR ranges must be unique across the multicluster environment and may not overlap.
  - All pod CIDRs in every cluster must be routable to each other.
  - All Kubernetes control plane API servers must be routable to each other.
- Helm 2.7.2 or newer. The use of Tiller is optional.

Google Kubernetes Engine

This example demonstrates how to use istio's multicluster feature to join 2 Google Kubernetes Engine clusters together, using the multicluster installation instructions.

Before you begin

In addition to the prerequisites for installing Istio the following setup is required for this example:

- This sample requires a valid Google Cloud Platform project with billing enabled. If you are not an existing GCP user, you may be able to enroll for a $300 US Free Trial credit.
  - Create a Google Cloud Project to host your GKE clusters.
- Install and initialize the Google Cloud SDK.
Multicluster Patterns

Multicluster Installation

Configure an Istio mesh spanning multiple Kubernetes clusters.

**Gateway Connectivity**
Install an istio mesh across multiple Kubernetes clusters using Istio Gateway to reach remote pods.

**VPN Connectivity**
Install an istio mesh across multiple Kubernetes clusters with direct network access to remote pods.

Refer to the multicluster service mesh concept documentation for more information.

https://preliminary.istio.io/docs/setup/kubernetes/multicluster-install.html
Our Objective - Single Mesh

Google Cloud Platform

us-west1-a

Pod IP Range 10.48.0.0/14

Java

Reviews-v3

⭐⭐⭐⭐

us-central1 (regional)

Pod IP Range 10.12.0.0/14

Java

Reviews-v1

Java

Reviews-v2

⭐⭐⭐⭐

Product Page

Ruby

Details

Pod IP Range 10.4.0.0/14

node

Ratings
Istio will discover the endpoints for a service name to route requests
Routing Resources

- Ingress
- gateway
- virtual service
- Ext service
- DestRule
  - Subset 1
  - Subset 2
  - Subset 3

- service1
- deployment
- pods

- service2
- deployment
- pods
kind: **VirtualService**

metadata:
  name: echo-vsvc

spec:
  hosts:
    - "echo-svc.default.svc.cluster.local"
  http:
    - match:
        - uri:
            prefix: "/v1"
      route:
        - destination:
            host: echo-svc.default.svc.cluster.local
              subset: v1
              weight: 90
        - route:
            - destination:
                host: echo-svc.default.svc.cluster.local
                  subset: v2
                  weight: 10

kind: **DestinationRule**

metadata:
  name: echo-destination

spec:
  host: echo-svc.default.svc.cluster.local
  subsets:
    - name: v1
      labels:
        version: v0.1
    - name: v2
      labels:
        version: v0.2
Routing Resources

- Ingress
- gateway
- virtual service
- Ext service
- DestRule
  - Subset 1
  - Subset 2
  - Subset 3
- service1
  - deployment
  - pods
- service2
  - deployment
  - pods

Google Cloud
Istio will discover the endpoints for a service name to route requests

How do the endpoints get registered?
Pilot has references to the K8s service entries which in turn reference the actual endpoints.
Multicluster entries share the same service name but reference separate endpoints.
Pilot Service Discovery

Client’s request for svcB is matched against service names before returning destination details. **Endpoints must be unique & routable**

Config data to Envoys

```
svcB
svcB.my-namespace
svcB.my-namespace.svc.cluster.local
{
  10.48.3.7:443
  10.4.0.6:443
}
```
Demo: Multi Cluster Istio
1. Welcome

Thanks for joining us at the Istio Multi Cloud Burst codelab by Google. This codelab requires beginner-level hands-on experience with Kubernetes, Node and Go.

What you will need

- Google Cloud Platform account (use existing, or we will give free accounts)
- Your laptop (install "kubectl", "gcloud" etc) or you can use Google Cloud Shell.
Rate today’s session

Effective enterprise architecture

Who is this presentation for?
- Architects of all stripes, tech leads, senior developers, and managers

Prerequisite knowledge
- Experience as a senior developer or architect working on software projects

What you’ll learn
- Learn a holistic approach to architecture that explains how to bring business architecture, information architecture, data architecture, application architecture, and infrastructure architecture together into a comprehensive design.
- You’ll also learn how to incorporate design thinking principles and work effectively with Agile teams.

Session page on oreillysacon.com/ny

O’Reilly Events App
That’s a Wrap