
Spotify
Kubeflow
Kubeflow @ Spotify:
Building & Managing a Centralized Platform
Ryan Clough
Keshi Dai
KubeCon
CloudNativeCon
North America 2019

Ryan Clough

@fnord2vec

Keshi Dai

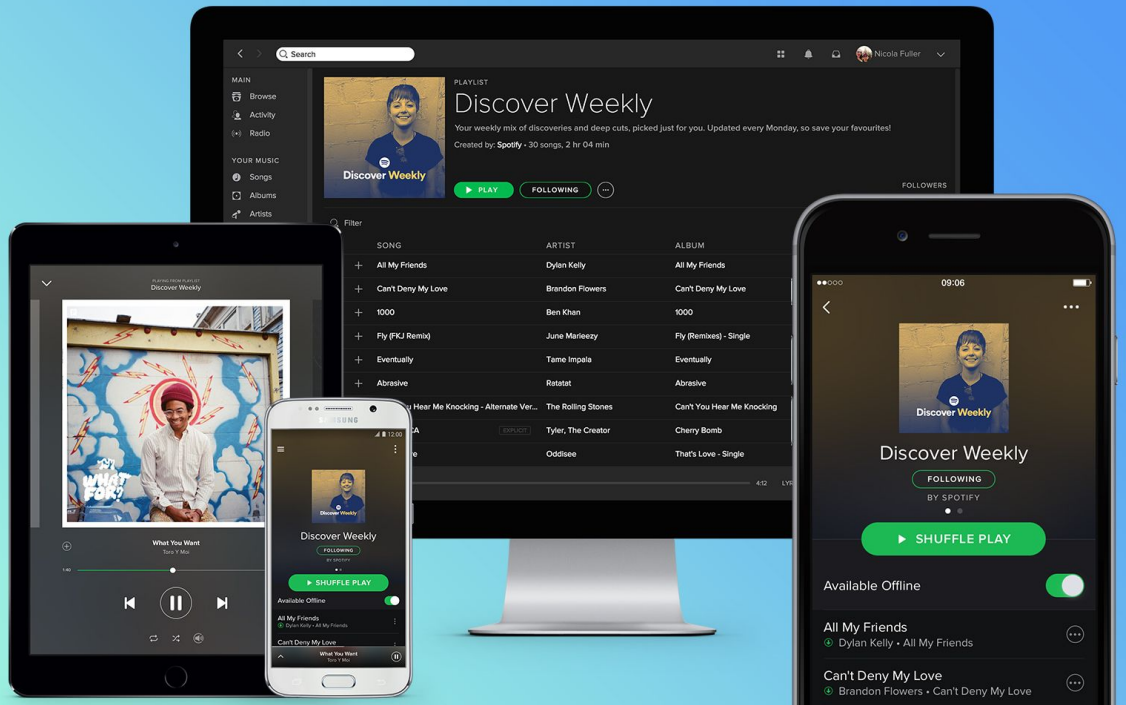
@daikeshi

Agenda

- Introduction to Spotify
- Why Make a Centralized Platform?
- Optimizing Development with Centralized Resources
- Building a Kubeflow Platform
- Managing the Cluster
- Lessons Learned


Spotify®

Music Streaming Service
Launched in 2008
248M Active Users
50M Tracks
79 Countries


Search
Nicola Fuller
MAIN
Browse
Activity
Radio
YOUR MUSIC
Songs
Albums
Artists
PLAYLIST
Discover Weekly
Your weekly mix of discoveries and deep cuts, picked just for you. Updated every Monday, so save your favourites!
Created by: Spotify • 30 songs, 2 hr 04 min
PLAY
FOLLOWING
FOLLOWERS
Filter
SONG
ARTIST
ALBUM
All My Friends
Dylan Kelly
All My Friends
Can't Deny My Love
Brandon Flowers
Can't Deny My Love
1000
Ben Khan
1000
Fly (FKJ Remix)
June Marieezy
Fly (Remixes) - Single
Eventually
Tame Impala
Eventually
Abrasive
Ratatat
Abrasive
Tyler, The Creator
Cherry Bomb
Oddisee
That's Love - Single
What You Want
Toro Y Moi
09:06
Discover Weekly
FOLLOWING
BY SPOTIFY
SHUFFLE PLAY
Available Offline
All My Friends
Dylan Kelly • All My Friends
Can't Deny My Love
Brandon Flowers • Can't Deny My Love
1000
Ben Khan • 1000

Kubeflow

Kubeflow is an open-source framework for running **ML pipelines** on **Kubernetes** by turning individual components of an ML workflow into Docker containers.

Organizational Structure

“Organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations.”

- Conway’s Law

You

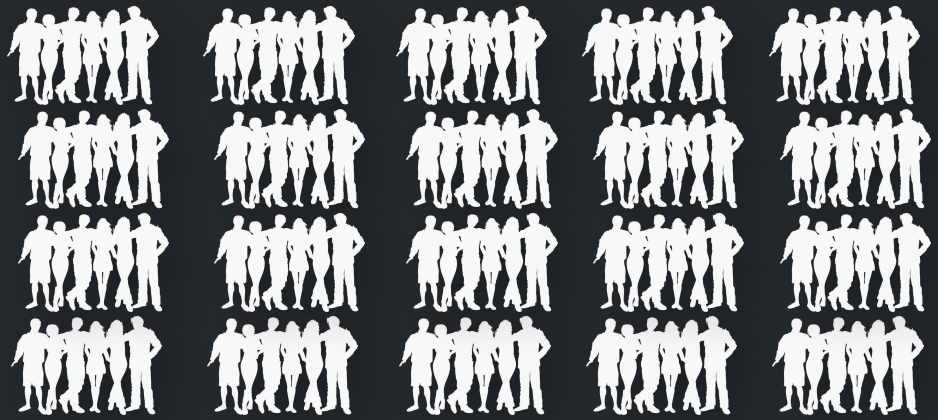
Your Squad

280+ Squads

Your Squad

280+ Squads

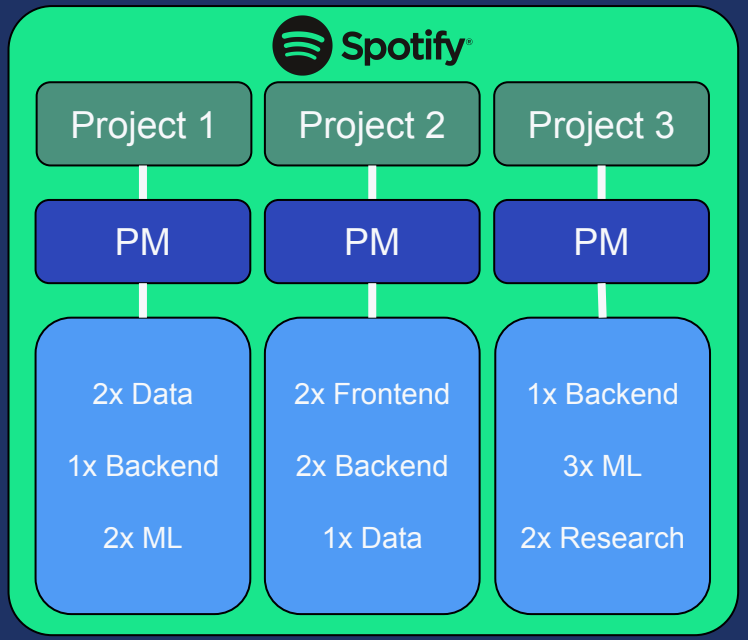
Your Squad



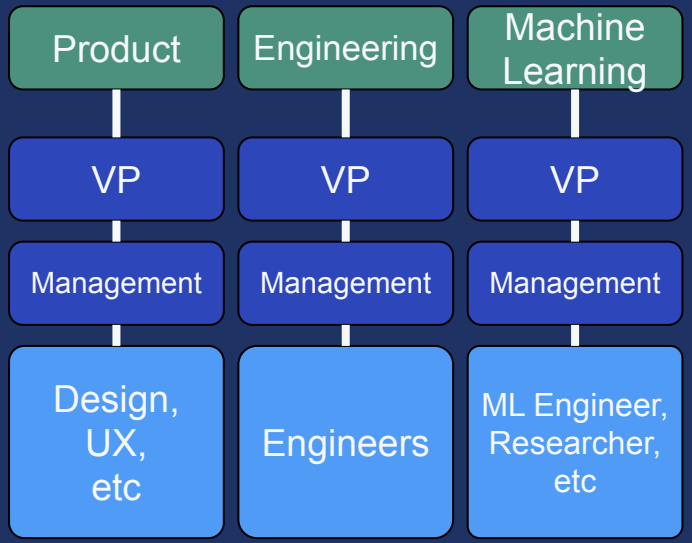
Example Team Structure

- 5-8 people
- Cross-functional
- Responsibly Autonomous

Example Team Structure



Horizontal/Distributed



Vertical/Centralized

Autonomy & Decentralization

Pros:

- Teams move faster
- Not blocked on other teams

Cons:

- Lack of standardization
- Information/Experience silos

Centralization

vs

Self-Deployment

Pros: Self-Deployment

- Natural choice for autonomy
- Let teams decide
- No fighting over resources

Self-Deploy: Your Job?

- Make deployment easy
- Provide components

“The Best Engineers Are Lazy”

-Ancient Engineering Proverb

Cons: Self-Deployment

Cons: Self-Deployment

- Kubernetes expertise

Cons: Self-Deployment

- Kubernetes expertise
- Exacerbates information silos

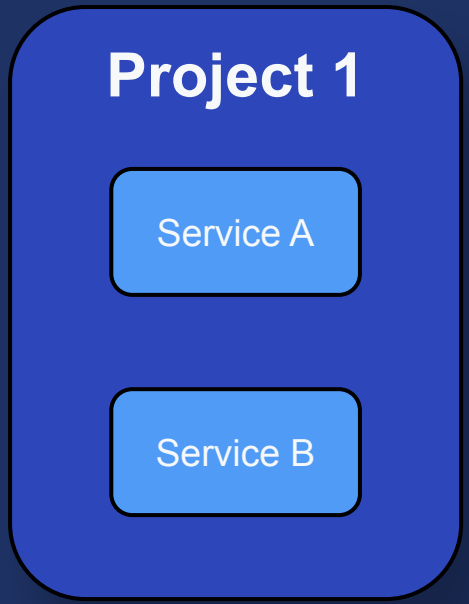
Cons: Self-Deployment

- Kubernetes expertise
- Exacerbates information silos
- Upgrade issues

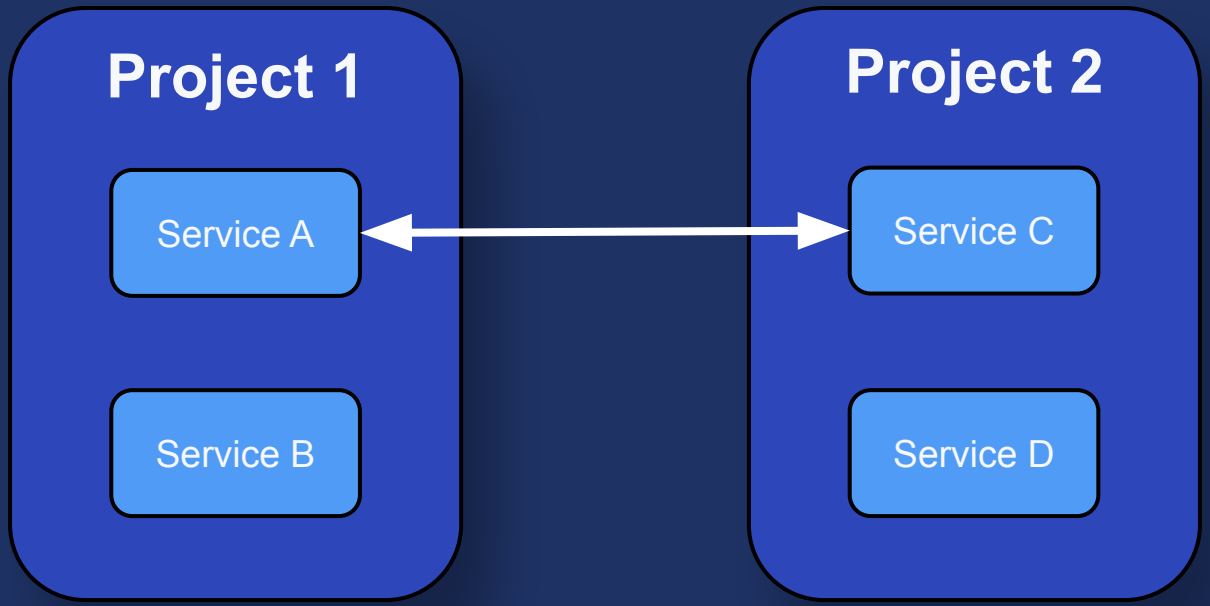
Cons: Self-Deployment

- Kubernetes expertise
- Exacerbates information silos
- Upgrade issues
- Technical limitation- Shared VPC

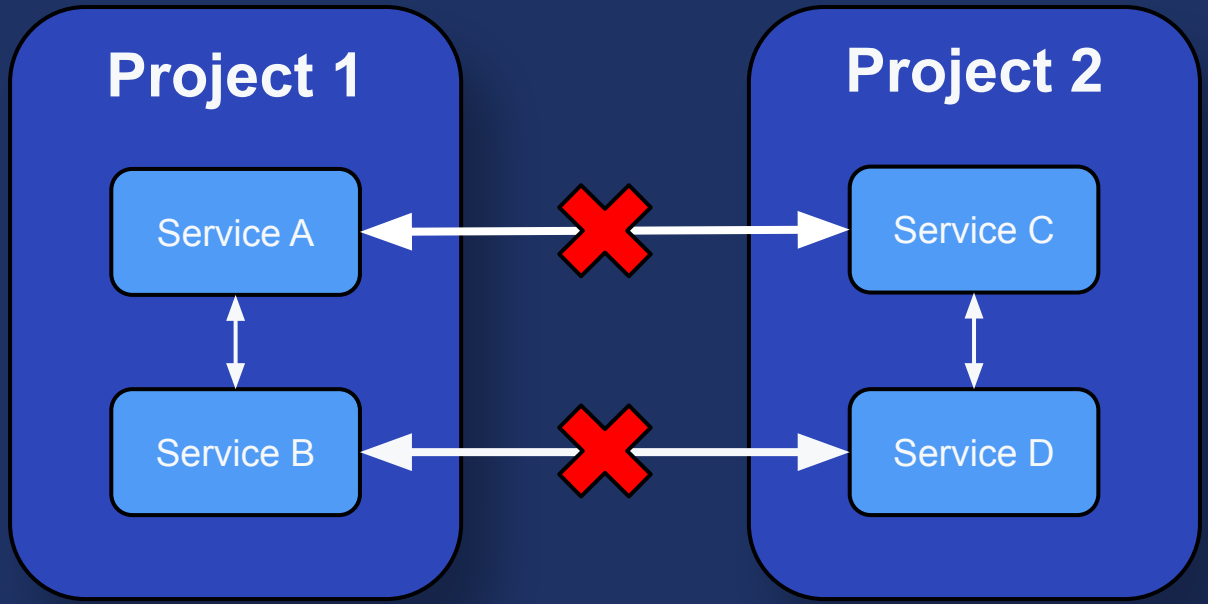
Shared VPC



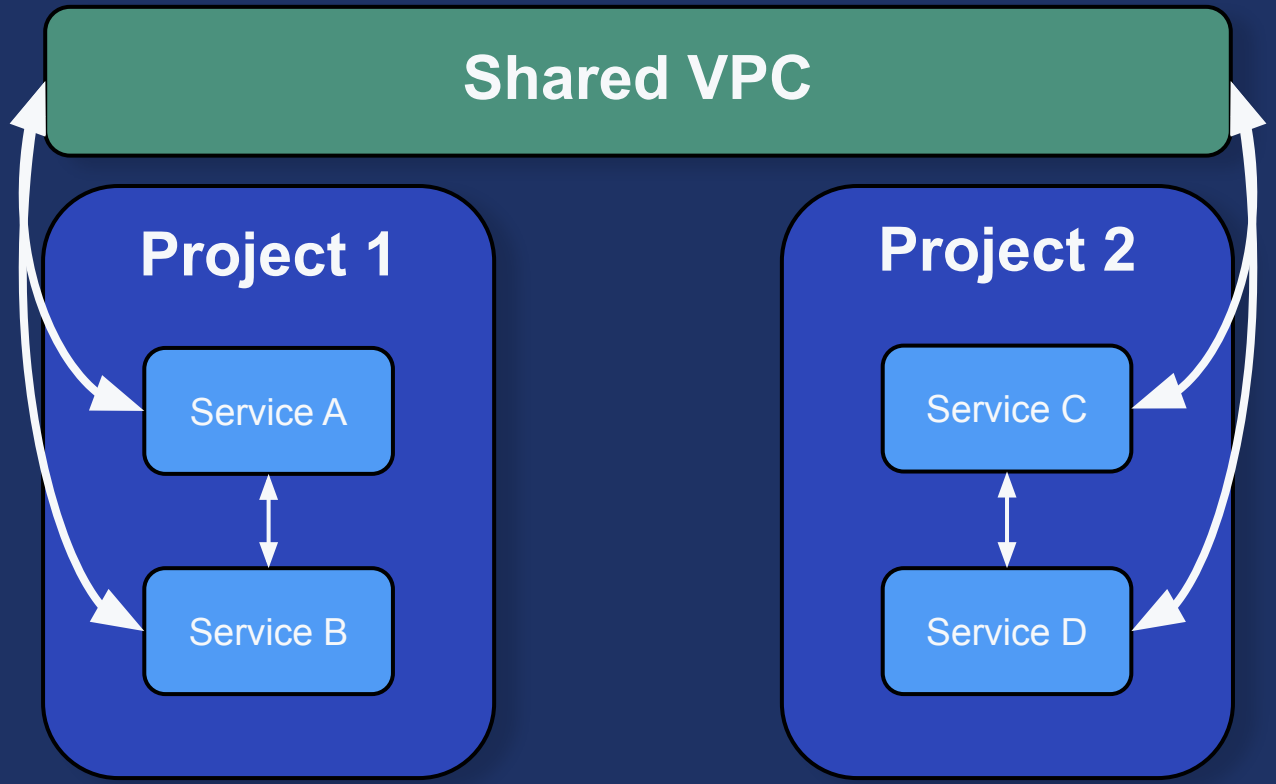
Shared VPC



Shared VPC



Shared VPC


Shared VPC
Project 1
Service A
Service B
Project 2
Service C
Service D

Shared VPC

- Major friction points:
  - Consultation with security
  - Blocks deployment on networking team
  - Limited IP resources
  - Special configuration for GKE

Pros: Centralization

- Shared VPC is solved once
- Centralization of Kubernetes expertise
- Teams focus on ML not infrastructure
- Easier adoption
- Centralized metadata

Cons: Centralization

- On call/SLOs
- Larger risk surface- upgrades, tenants
- Kubeflow is not yet fully Multi-tenant

Optimizing Pipeline Development

Provide Common Components

- Based on TFX (Tensorflow Extended)
- Covers common tasks
- Access to existing systems (ex: data)

Build beyond Kubeflow

- Template repo
- Reduces boilerplate
- CLI tool
- Skaffold for automated Docker builds
- Make upgrades easier

Building Kubeflow Cluster

Everything We Do is on Google Cloud

Default Deployment

Google Deployment Manager creates GCP resources
kfctl deploys jsonnet files

HashiCorp
Terraform

Terraform + ksonnet

Terraform creates GCP resources
kfctl deploys jsonnet files

HashiCorp
Terraform

Terraform + Kustomize

Terraform creates GCP resources
kfctl deploys Kustomize manifest files

Default Deployment

UI tool + deployment cli

Default Deployment
(May - June 2019)

Google Deployment Manager
creates GCP resources
kfctl deploys jsonnet files

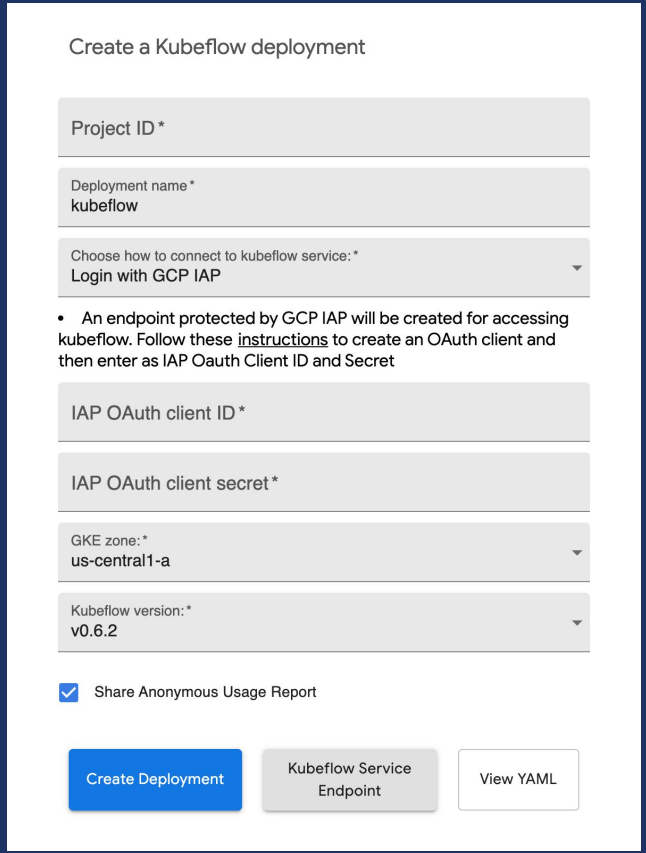
HashiCorp
Terraform

Kustomize.io

UI Deployment Tool

If you are using Google Cloud, this tool works as a charm.

- GKE Cluster + Kubeflow Installation
- Google Cloud Endpoints
- Google IAP (Identity-Aware Proxy)

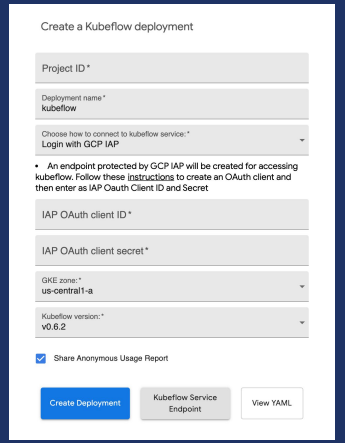


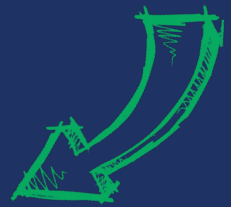
Things We Like

Really simple!

Problems

- Black box
- No customization
- No support for shared VPC
- No option to upgrade





Deployment CLI

Use kfctl to create a cluster and deploy kubeflow

- Kfctl generates config files
- Update GCP config for shared VPC
- Update pipelines to use Cloud SQL and named PD
- Kfctl creates GCP resources and installs Kubeflow apps

```
# If using Cloud IAP, create environment variables from the
# OAuth client ID and secret that you obtained earlier:
export CLIENT_ID=<CLIENT_ID from OAuth page>
export CLIENT_SECRET=<CLIENT_SECRET from OAuth page>

# The following command is optional, to make kfctl binary easier to use.
export PATH=$PATH:<path to kfctl in your kubeflow installation>
export ZONE=<your target zone> #where the deployment will be created

export PROJECT=<your GCP project>
export KFAPP=<your choice of application directory name>
# Default uses Cloud IAP:
kfctl init ${KFAPP} --platform gcp --project ${PROJECT}

cd ${KFAPP}
kfctl generate all -V --zone ${ZONE}
kfctl apply all -V
```

Problems

- Customization is manual
- Upgrade is still hard
- Replica is hard
- Can’t specify context in kfctl

Problems

- Customization is manual
- Upgrade is still hard
- Replica is hard
- Can’t specify context in kfctl

```
~/Desktop 5:45:11 (⎈ kubeflow-platform|kf-test:default)
$ kubectl get pod -n kubeflow
```

Terraform + ksonnet

Terraform is introduced to create GCP resources

Terraform + ksonnet

(June - September 2019)

Terraform creates GCP resources

kfctl deploys jsonnet files

What is Terraform

“Terraform is a tool for building, changing, and versioning infrastructure safely and efficiently.”

Terraform for GCP Resources

- kfctl is no longer in charge of managing GCP resources
- Define the entire stack in a module
  - GKE cluster, shared VPC, dns, node pools, etc
  - Cloud SQL instance, PDs, DB users
  - Service accounts, k8s secrets, RBAC roles, etc
- Multiple instances based on the same module

Benefits of using Terraform

- Learn from existing examples (backend GKE clusters)
- Easily rebuild, modify, and track changes
- Easily replicate the entire kubeflow deployment
- Integrate with our git workflow

```
keshi@local (master)$ terraform state list
module.kf-test-xpn.data.google_client_config.de
module.kf-test-xpn.google_compute_disk.artifact
module.kf-test-xpn.google_container_cluster.spo
module.kf-test-xpn.google_container_node_pool.m
module.kf-test-xpn.google_project_iam_member.cl
module.kf-test-xpn.google_project_iam_member.cl
module.kf-test-xpn.google_project_iam_member.cl
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_project_iam_member.ku
module.kf-test-xpn.google_service_account.cloud
module.kf-test-xpn.google_service_account.kubef
module.kf-test-xpn.google_service_account.kubef
module.kf-test-xpn.google_service_account.kubef
module.kf-test-xpn.google_service_account_key.c
module.kf-test-xpn.google_service_account_key.k
module.kf-test-xpn.google_service_account_key.k
module.kf-test-xpn.google_sql_database_instance
module.kf-test-xpn.google_sql_user.root_user
module.kf-test-xpn.kubernetes_config_map.kube-d
module.kf-test-xpn.kubernetes_namespace.kubeflo
module.kf-test-xpn.kubernetes_secret.admin-gcp-
module.kf-test-xpn.kubernetes_secret.cloudsql-i
module.kf-test-xpn.kubernetes_secret.user-gcp-s
module.kf-test-xpn.null_resource.delete_default
module.kf-test-xpn.random_id.db_name_suffix
keshi@local (master)$ terraform state list | wc
      38
```

feedback-service replied 19 days ago — Member

`terraform plan` output for `13be0c3`

```
------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create
  - destroy
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # module.kf-test-xpn.google_project_iam_member.cloudsql_proxy-cloudsql_client w
  + resource "google_project_iam_member" "cloudsql_proxy-cloudsql_client" {
      + etag    = (known after apply)
      + id      = (known after apply)
      + member  = "serviceAccount:kf-test-cloudsqlproxy@kubeflow-platform.iam.gse
      + project = "kubeflow-platform"
      + role    = "roles/cloudsql.client"
    }

  # module.kf-test-xpn.google_project_iam_member.cloudsql_proxy-cloudsql_viewer w
  - resource "google_project_iam_member" "cloudsql_proxy-cloudsql_viewer" {
      - etag    = "BwWL2e69m5g=" -> null
      - id      = "kubeflow-platform/roles/cloudsql.viewer/serviceAccount:kf-test
      - member  = "serviceAccount:kf-test-cloudsqlproxy@kubeflow-platform.iam.gse
      - project = "kubeflow-platform" -> null
      - role    = "roles/cloudsql.viewer" -> null
    }

  # module.kf-test-xpn.google_sql_user.root_user must be replaced
-/+ resource "google_sql_user" "root_user" {
      + host     = "%" # forces replacement
      ~ id       = "root//kf-test-41ba207a" -> (known after apply)
        instance = "kf-test-41ba207a"
        name     = "root"
      ~ project  = "kubeflow-platform" -> (known after apply)
    }

Plan: 2 to add, 0 to change, 2 to destroy.

------------------------------------------------------------------------

This plan was saved to: tf.plan

To perform exactly these actions, run the following command to apply:
    terraform apply "tf.plan"
```

```
        }

        timeouts {
            create = "30m"
            delete = "30m"
            update = "30m"
        }
    }

Plan: 1 to add, 0 to change, 1 to destroy.

------------------------------------------------------------------------

This plan was saved to: tf.plan

To perform exactly these actions, run the following command to apply:
    terraform apply "tf.plan"
```

feedback-service replied 8 days ago — Member

❌ **Build failed for 'Add oauth scopes for node pool'**

- Terraform failed to apply changes
- Command not found in /var/jenkins_home/workspace/tingle_3.8985/workspace/check-pl
  8: curl

View log | Re-trigger | View Backstage

Ksonnet Deployment

- Kfctl generates Kubernetes resources only
- Parameterize the deployments for different envs
  - host name for ingress, Cloud SQL instance, PD
- Kfctl installs Kubeflow apps

Terraform + Kustomize

Since v0.6, Kubeflow has started using Kustomize for deployment

HashiCorp
Terraform

Kustomize.io

Terraform + Kustomize
(September 2019 - Present)

Terraform creates GCP resources
kfctl deploys Kustomize manifest files

Kustomize Deployment

- Kfctl generates Kustomize manifests
- Overlays for customized deployment
- kfctl apply deploys manifests

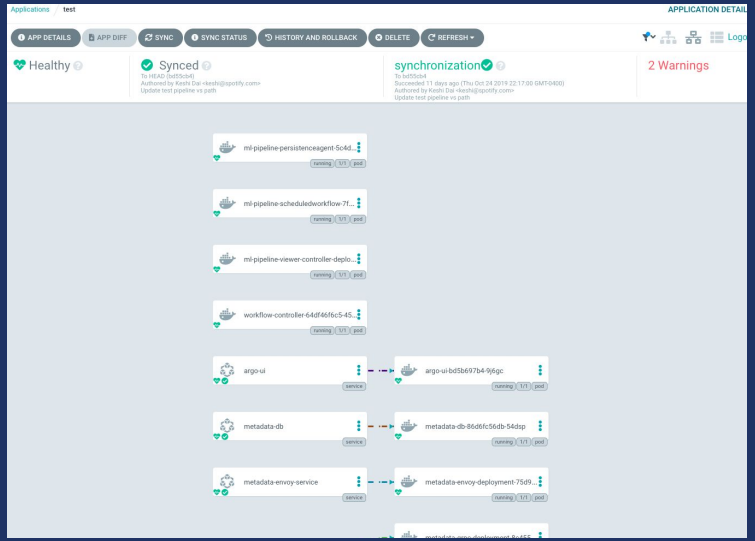
```
+-- components
|   +-- api-service
|   +-- argo
|   +-- metadata
|   +-- minio
|   +-- ...
+-- kf-test
|   +-- kustomization.yaml
|   +-- params.yaml
+-- kf-dev
|   +-- kustomization.yaml
|   +-- params.yaml
+-- kf-prod
    +-- kustomization.yaml
    +-- params.yaml
```

Ideal World

- Automatically track Kubeflow deployment changes
- Convert manifests generated by kfctl to our own Kustomize layout

Ideal World

- Use gitops to deploy to multiple envs e.g. argo-cd



Managing Kubeflow Cluster

User Access

- ~100 Spotifiers
- Two ways to interact with our cluster
  - Python SDK
  - Web UI

Secure Access

Provide secure access to our clusters

- Web UI access protected by Google IAP
- Python SDK access protected by
  - Google IAM (project viewer)
  - Kubernetes RBAC (more granular permissions on APIs)

Service Accounts

Manage service accounts for different teams

- Different teams using different GCP projects
- Store service accounts as k8s secret (not ideal)
- Switch to use workload identity (future)
- Use Velero to backup secrets hourly

Resource Management

Strategy for managing workload resource on our platform to meet requirements for various Machine Learning tasks

ML Job Resource Config

- Provide multiple node pools for different types of jobs
  - standard, high-memory, gpu
- Allow users to request custom resource
  - set resource request/limit in the pipeline job

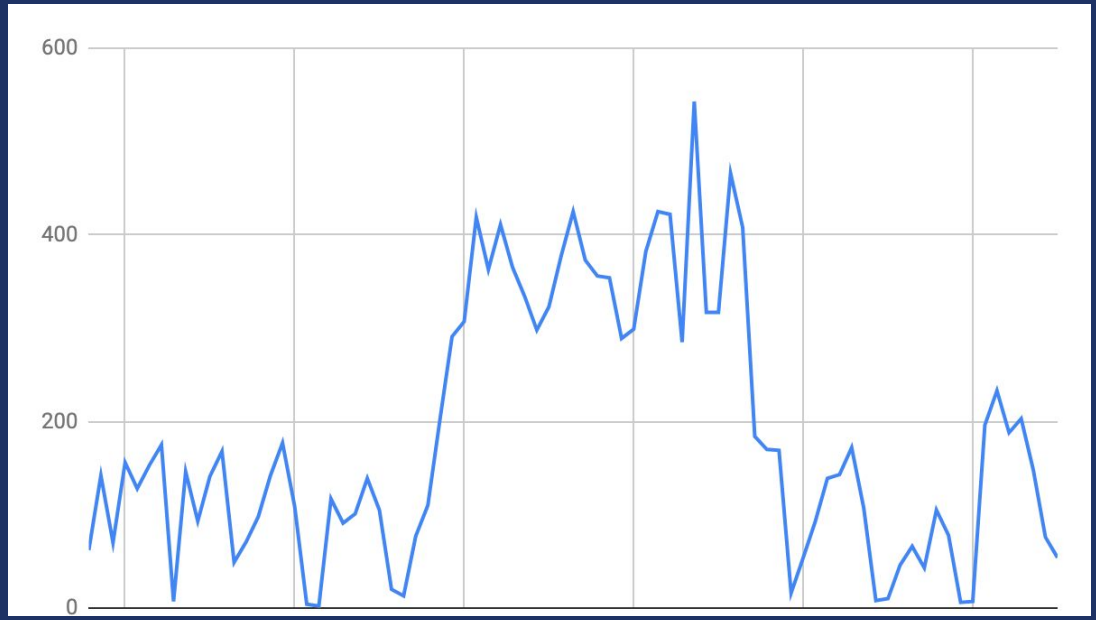
Service Resource Config

Default resource config is not sufficient

- Isitio-policy, istio-telemetry
- Metrics server

Platform Usage Stats

- 100 Spotifiers
- Over 15,000 Pipeline Runs!



Benefits of Centralized Platform

- More ML, less infra
- Shorter iteration cycles
- Faster time to production
- Better ML in our products

Lessons Learned

“Even with a handful of machine learning/data engineers, we are successfully able to manage multiple Kubernetes clusters and machine learning workloads at scale.”

- from our talk proposal submission

“Even with a handful of machine learning/data engineers, we are successfully able to manage multiple Kubernetes clusters and machine learning workloads at scale.”

- from our talk proposal

Lessons Learned

- **Lean on k8s expertise of others: Spotify platform team**
  - Steep learning curve
  - Networking, security, deployment, cluster management, etc

Lessons Learned

- **Kubeflow is too big to chew all at once**
  - Kubeflow Pipelines, Metadata, Istio, Kustomize, etc
  - Infra team takes the pain

Lessons Learned

- **Nothing is small in terms of security**
  - Initiate the conversation as early as possible
  - Keep them happy!

Thank You!

Sound interesting? **Join the band:** spotifyjobs.com

Questions?

@fnord2vec

@daikeshi