Streamlining Kubernetes Application CI/CD with Bazel

Gregg Donovan - Staff Software Engineer, Etsy
Christopher Love - Principal Architect for Project Helmsman, CNM Consulting - https://chrislovecnm.com

March 2019

Goals for the session

What?

Using Bazel to build and manage Containers and Kubernetes

Why?

Kubernetes is becoming the standard for container management. Using Bazel to build and deploy.

How?

Use Bazel rules to build containers and deploy them to Kubernetes.

Containers

Why do we need them?

01


Google has been developing and using containers to manage applications for **over 12 years.**
Images by Connie Zhou

Containers are about two capabilities

Image

A method of packaging an executable application and its dependencies (runtime, system tools, system libraries, configuration)

Runtime

Running the package as a set of resource-isolated processes

Google Cloud

Container Buzz Words

Lightweight

Containers contain only what is necessary, so the same host can run multiple containers.

Portable

Containers package all the dependencies into the image; therefore they do not rely on host to provide anything other than basic compute resources.

Fast

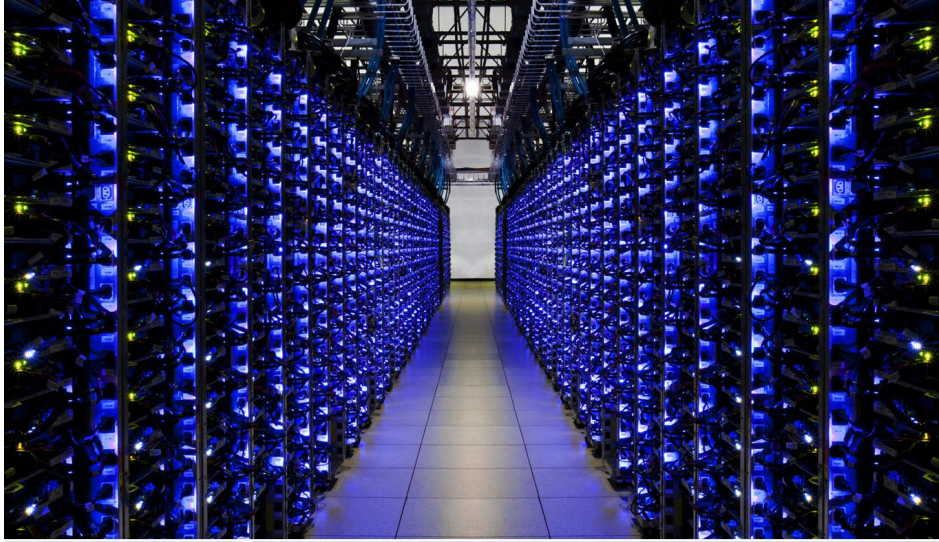
Containers (which run as processes) take less time to start up given that the host is already running and has the container image downloaded.

Google Cloud

But it's all so different!

- Deployment
- Management, monitoring
- Isolation (very complicated!)
- Rolling Updates
- Discovery
- Scaling, replication, sets

A **fundamentally different** way of managing applications requires different tooling and abstractions



Google Cloud

Containers do not solve everything

- Storage
- Load balancing
- Discovery
- Multiple Apps
- Security
- Failover
- QOS

Google Cloud

Shipping Containers At Clyde, by Steve Gibson

Kubernetes

What is all the buzz about?

02

Google Cloud

Kubernetes Open Source Project

- Manages container inside a cluster
- Inspired and informed by Google's experiences and the Borg
- Supports multiple cloud and bare-metal environments
- Solves the problems listed on the previous slide

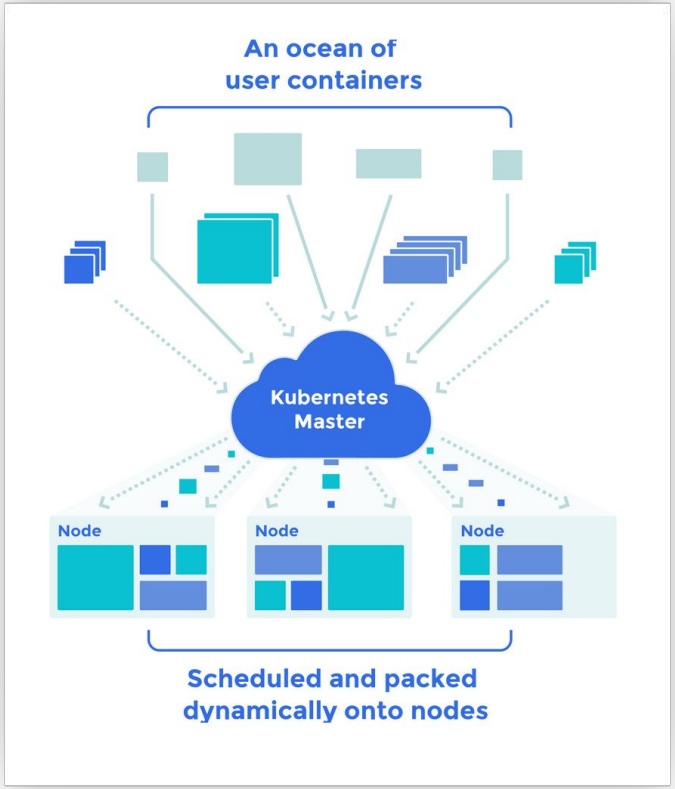
Think of Kubernetes as the OS for your compute fleet

Scheduling

Monitoring

Scaling

Self Healing



Google Cloud

Bazel

Using Bazel with your Containers

03

Google Cloud

BIG CODE

Google Cloud

Bazel: A Modern Build *and test* System

Bazel.build

Fast, reproducible build and test

Cloud accelerated

Google OSS

Google Cloud

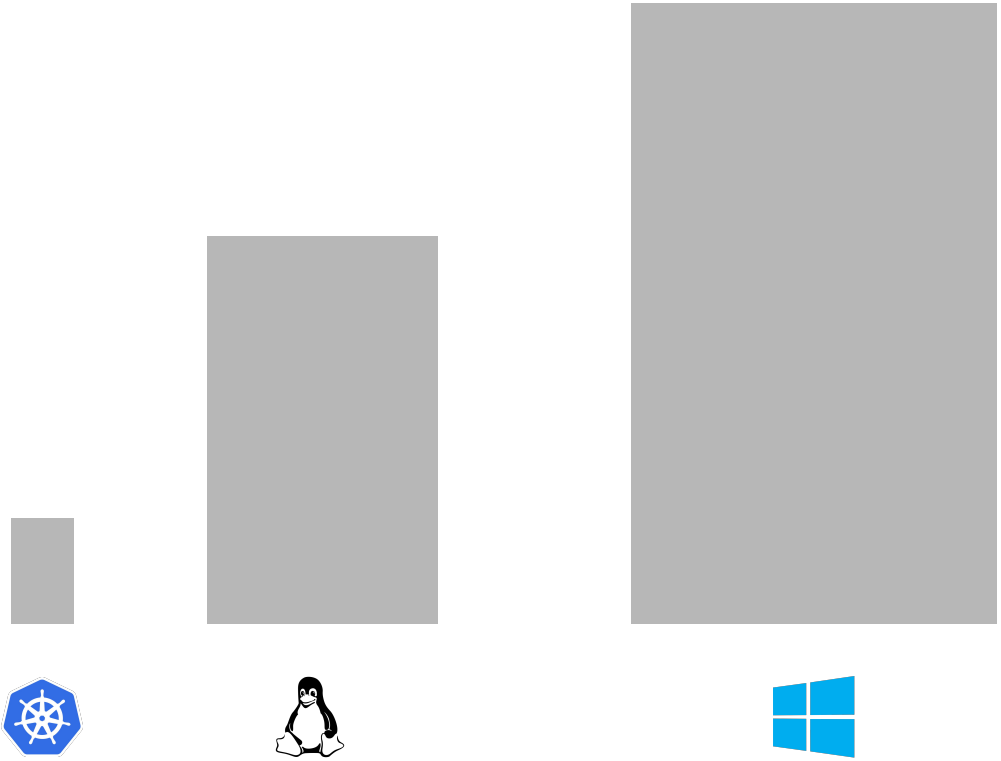
LoC

Google Cloud

LoC

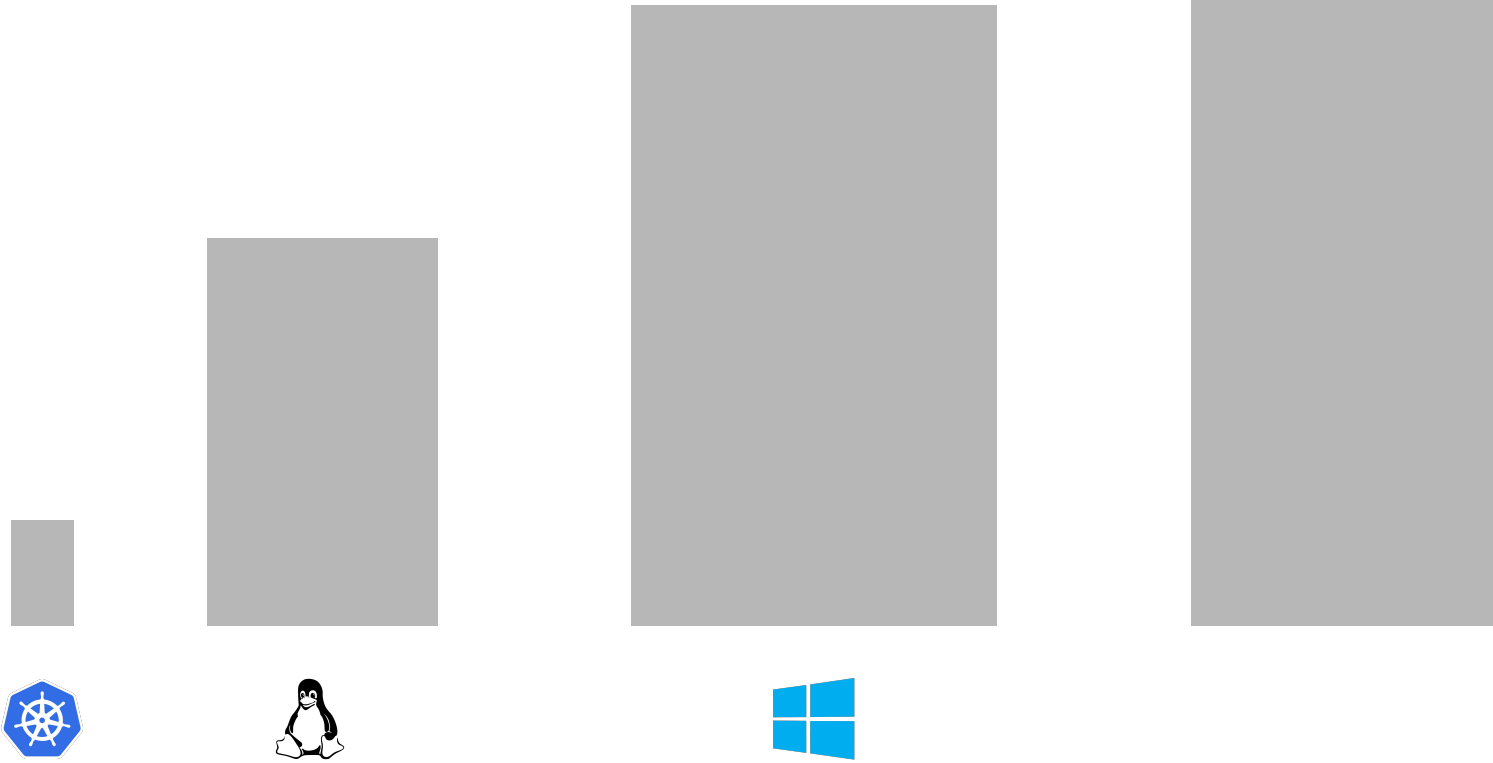
Google Cloud

LoC



Google Cloud

LoC

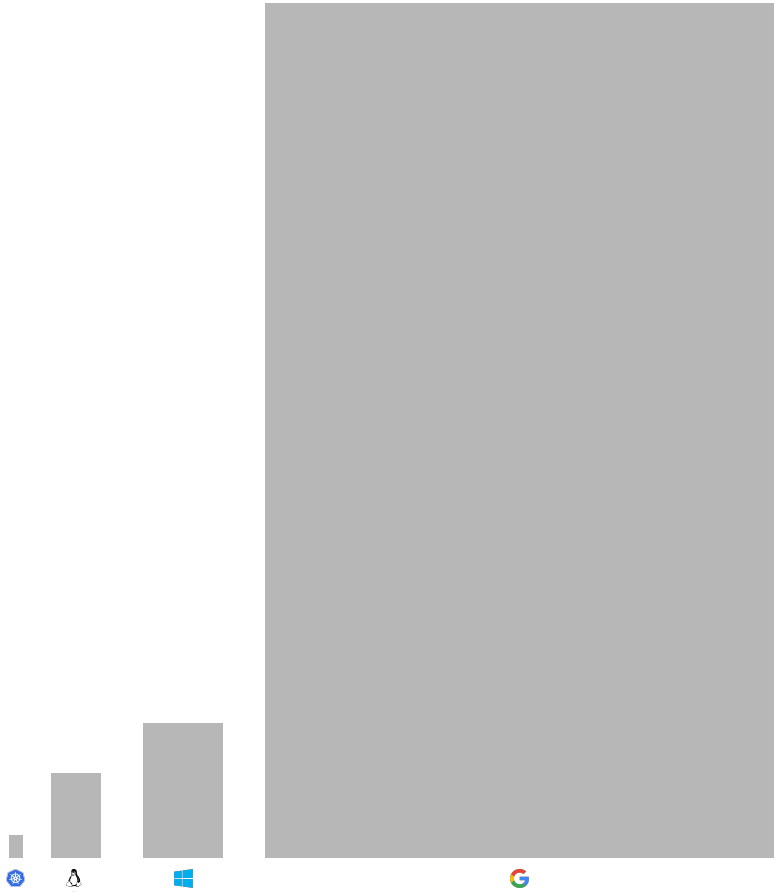


Google Cloud

LoC

Google Cloud

LoC



2,000,000,000

Google Cloud

BIG CODE →

BIG BUILD →

BIG TEST

Google Cloud

1. Building
2. Unit Tests
3. Dependency Management
4. Gazelle

Google Cloud

D.R.Y.

Only retest when necessary

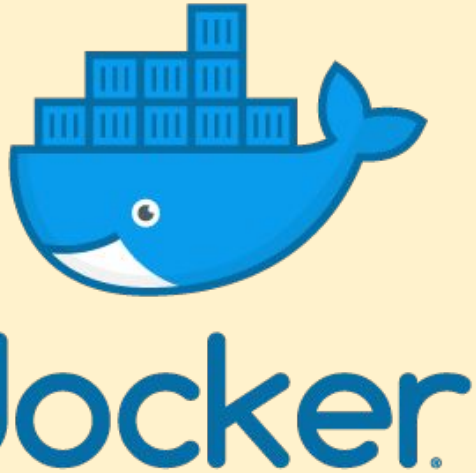
Fan out

Execute tests in parallel

Bazel builds ~all the things

Android	iOS	Python
C and C++	Java	Ruby
C#	JavaScript	Rust
D	Jsonnet	Sass
Docker	Objective C	Scala
Go	Perl	Shell
Groovy	PHP	Swift
Haskell	Protobuf	TypeScript
Kotlin		

A set of rules for pulling down base images, augmenting them with build artifacts and assets



Google Cloud

1. Authentication
2. Publish Containers
3. Mananage Container Digests
4. Manifest Templating
5. Deploying Manifests
6. Full Application CRUD

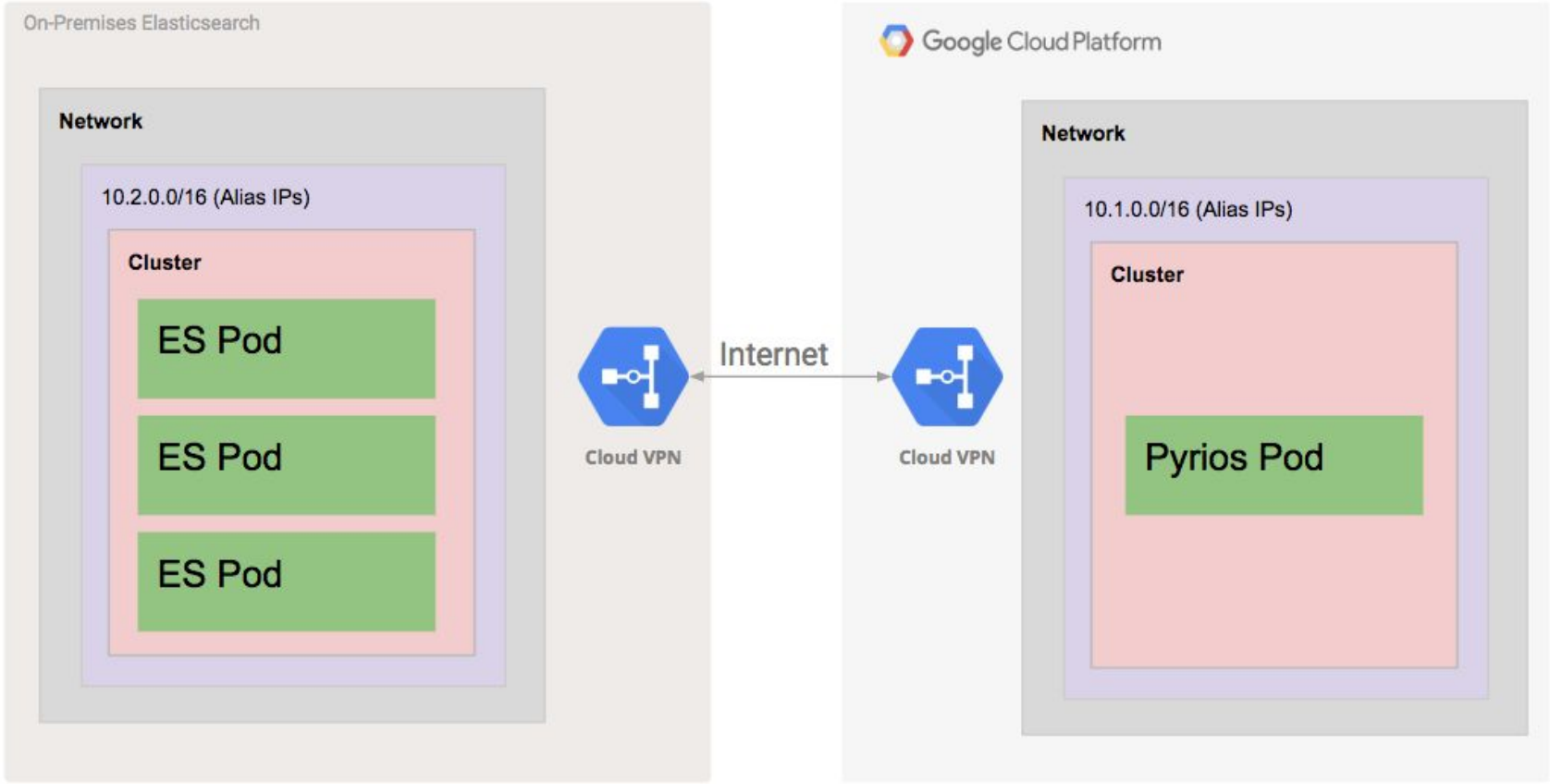
Google Cloud

Credits

Thanks to Eric Hole (@geojaz) for working on the demo.

Thanks to Shravani Dharam (@sdharam) for proofreading and formatting!

Demo


On-Premises Elasticsearch
Network
10.2.0.0/16 (Alias IPs)
Cluster
ES Pod
ES Pod
ES Pod
Cloud VPN
Internet
Cloud VPN
Google Cloud Platform
Network
10.1.0.0/16 (Alias IPs)
Cluster
Pyrios Pod

Demo

Professional Services: Project Helmsman

Project Helmsman

Workshops and matching open-source PoCs to guide customers and partners through using Kubernetes Engine in production

Helmsman is a project to build and release open-source examples of how to run common patterns in Google Kubernetes Engine along with workshops for Google's partners to deliver, to teach their customers how to move to a containerized world.

Shortlink to the code: **https://goo.gl/uD5sAM**

Google Cloud

Bazel and Kubernetes at Etsy

Etsy

The global marketplace for unique and creative goods

About Etsy

- 39.4m active buyers
- 2.1m sellers
- 60m+ listings
- $3.9b 2018 GMS
- 874 employees

etsy.com/shop/SimplyCreatedForYou6

Etsy

unicorn paintings | Search

Sell on Etsy | Register | Sign in | Discover | Cart

Jewelry & Accessories | Clothing & Shoes | Home & Living | Wedding & Party | Toys & Entertainment | Art & Collectibles | Craft Supplies | Vintage | Gifts

All categories › "unicorn paintings" (6,357 Results)

Sort by: Relevancy

Special offers
- On sale

All categories
- Art & Collectibles
- Craft Supplies & Tools
- Bath & Beauty
- Home & Living

+ Show more

Shipping
- Free shipping
- Ready to ship in 1 business day
- Ready to ship within 3 business days

Subject
- Abstract & geometric
- Animal
- Anime & cartoon
- Architecture & cityscape
- Beach & tropical

+ Show more

Orientation
- Horizontal

Ad

Unicorn Waterc

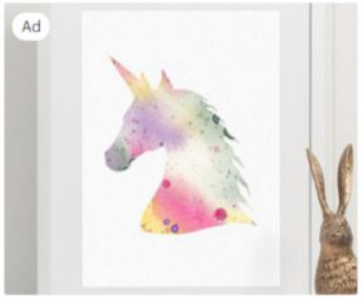
AudreyZombiesAr

★★★★★ (1)

$20.00

Ad

Ad



ıt, Cut...

Ad

Unicorn Watercolor Print Nursery ...

SuziBlueDesigns

★★★★★ (21)

$17.00

It's a fun problem

More colors

Unicorn canvas

Framed Magical Rainbow Haired U...

LoveBumble

Framed Magical Rainbow Haired U...

LoveBumble

Framed Magical Rainbow Haired, ...

LoveBumble

Headquartered in Brooklyn

Other offices in:

- San Francisco, CA
- Hudson, NY
- Berlin, Germany
- Dublin, Ireland
- London, UK
- New Delhi, India
- Paris, France
- Toronto, Canada

etsy.com/shop/JessicaThurstonArt

- Why and how Etsy adopted Bazel, **rules_k8s,and rules_docker**

- How they work to yield fast, correct deployments

- Bazel and Kubernetes learnings from our GKE migration

Search Monorepo

15+ services

One CI/CD pipeline

Bazel

rules_k8s

rules_docker

Python for YAML

Per k8s context config

etsy.com/shop/RossiVArt

Kubernetes: Hashing & Caching

SHA256

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: redis-master
  labels:
    app: redis
spec:
  selector:
    matchLabels:
      app: redis
      role: master
      tier: backend
  replicas: 1
  template:
    metadata:
      labels:
        app: redis
        role: master
        tier: backend
    spec:
      containers:
      - name: master
        image: k8s.gcr.io/redis:e2e
        resources:
          requests:
            cpu: 100m
            memory: 100Mi
        ports:
        - containerPort: 6379
```

rules_docker > Dockerfile

```
$ docker images
REPOSITORY                          TAG                       IMAGE ID       CREATED          SIZE
search/apps/mmx                     mmx_docker                e2a1d55be23d   48 years ago     932 MB
search_data_docker                  intermediate              cbefdae46002   48 years ago     460.4 MB
search/apps/spell_correction        spell_correction_docker   91653e8b5207   48 years ago     448.8 MB
search/apps/etsy-search1            etsy-search1_docker       167736f9b424   48 years ago     569.1 MB
search/apps/slv2                    slv2_docker               3aa5a41625c5   48 years ago     935.3 MB
search/apps/elastic2/kubernetes     elastic2_gke_docker       eb56b8285cad   48 years ago     125.9 MB
...
```

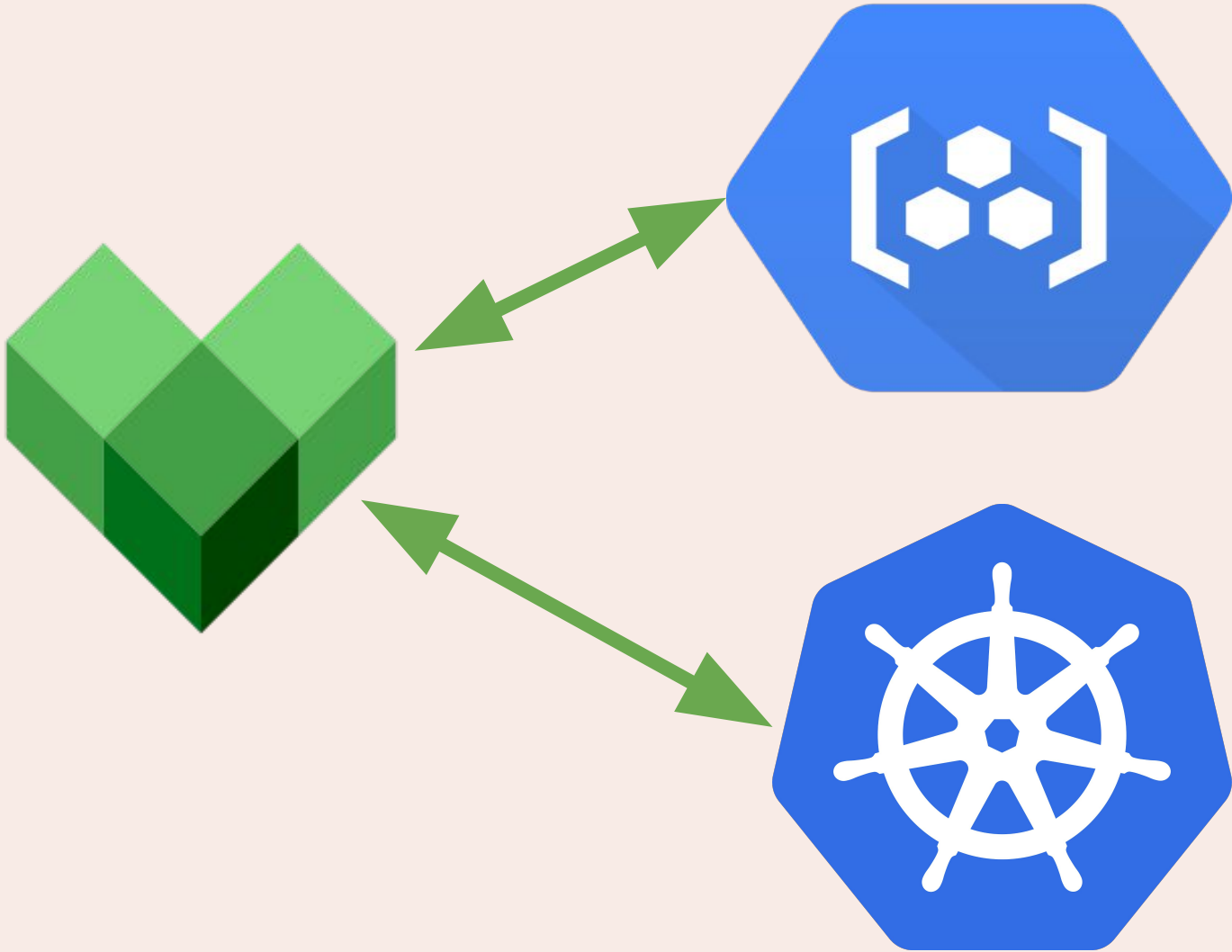
rules_k8s

```
load("@io_bazel_rules_k8s//k8s:object.bzl", "k8s_object")

k8s_object(
  name = "dev",
  kind = "deployment",

  # A template of a Kubernetes Deployment object yaml.
  template = ":deployment.yaml",

  # An optional collection of docker_build images to publish
when this target is bazel run. The digest of the published image
is substituted as a part of the resolution process.
  images = {
    "gcr.io/rules_k8s/server:dev": //server:image"
  },
)
```

Motivation: Monorepo

Deploy just the right amount, every time

Let Bazel work it out with the Container Registry and K8s



What is a Docker container?

```
$ docker inspect bb1efd443479
[
    {
        "Id": "sha256:bb1efd443479d95d959c990f268a6bb3d06bfafb82ce2200c45d0a24262e0c1d",
        "RepoTags": [ "bazel/grafana:grafana_docker" ],
        "Created": "1970-01-01T00:00:00Z",
        "Author": "Bazel",
        "Config": {
            "User": "grafana",
            "ExposedPorts": { "3000/tcp": {} },
            "Env": [

"PATH=/usr/share/grafana/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
                "GF_PATHS_CONFIG=/etc/grafana/grafana.ini",
            ],
            "Image": "sha256:ea9f0ca0dc5d538ab046a8618af1aaf0d3df05e89dc3a0420fabd9b46c4a0261",
            "WorkingDir": "/",
            "Entrypoint": ["/run.sh"],
        },
        "Architecture": "amd64",
        "Os": "linux",
        "Size": 238231783,
        "RootFS": {
            "Type": "layers",
            "Layers": [
                "sha256:d626a8ad97a1f9c1f2c4db3814751ada64f60aed927764a3f994fcd88363b659",
                "sha256:fe145ea19a267f67c106d3bf3df09a14d0d02c0f93e2c14df2f32f28562b954c",
                "sha256:d580759d14dac7f636711d0901258b1b22ae4c1bb046e06d1801c031192e52b5",
                "sha256:7d59735eaa9f4b2c5da8dc576540d1903a9db46fcbf867453cf95b6466f2ceab",
                "sha256:fd0c81ee3761fc31e63a56793e9baaa3744f1bc26077f63480bde878cc819b53",
                "sha256:f874fe8e2453b568a50fc6072edc1dd75c6ab568dbd658fe9978588411abad20",
                "sha256:9dd3209f58e05896460aac252bb068e1a59d107eabf7ffb7faf25f2cebae70cd"
            ]
        }
    }
]
```

`rules_docker`: Docker without docker or a Dockerfile

Container Registry v2 API

`HEAD /v2/<image-name>/manifests/<sha256>`

Check for the existence of an image manifest.

`HEAD /v2/<name>/blobs/<digest>`

Check for the existence of a layer.

Kubernetes pod-template-hash

SHA256

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: grafana
  labels:
    app: grafana
spec:
  selector:
    matchLabels:
      app: grafana
  replicas: 1
  template:
    metadata:
      labels:
        app: grafana
    spec:
      containers:
      - name: grafana
        image:
gcr.io/etsy-gcr/grafana@sha256:99b8c7ac7fdb1e04ccbd56090f91f3eeb0ed21a77abb5bb2a25532fca7026dbb
        resources:
          requests:
            cpu: 100m
            memory: 100Mi
        ports:
        - containerPort: 3000
```

Tip #1:
Use "Distroless" Containers

- Smaller size
- No package manager
- Fewer CVEs

github.com/GoogleContainerTools/distroless

```
load("@io_bazel_rules_docker//java:image.bzl",
"java_image")


java_image(
    name = "hello",
    srcs = ["HelloJava.java"],
    base = "//java:java8",
    main_class = "examples.HelloJava",
)
```

Tip #2:
Use SHA256 image references

Tip #3:
Build YAML with the K8s Client APIs

ip #4: ulumi

ulumi.io

```
// Canary ring. Replicate instrumented Pod 3 times.
const canary = new k8s.apps.v1beta1.Deployment(
    "canary-example-app",
    { spec: { replicas: 1, template: instrumentedPod } },
    { dependsOn: p8sDeployment }
);

// Staging ring. Replicate instrumented Pod 10 times.
const staging = new k8s.apps.v1beta1.Deployment("staging-example-app", {
    metadata: {
        annotations: {
// Check P90 latency is < 20,000 microseconds. Returns a `Promise<string>`
// with the P90 response time. It must resolve correctly before this
// deployment rolls out.
// In general any `Promise<T>` could go here.
            "example.com/p90ResponseTime": util.checkHttpLatency(canary,
containerName, {
                durationSeconds: 30,
                quantile: 0.9,
                thresholdMicroseconds: 20000,
                prometheusEndpoint: `localhost:${localPort}`,
            })
        }
    },
    spec: { replicas: 1, template: instrumentedPod }
});
```

ip #5:
ilt for dev
workflow

lt.dev

```
def bazel_build(image, target):
  custom_build(
    image,
    'bazel run ' + target,
    [],
    tag="image",
  )

k8s_yaml(bazel_k8s(":snack-server"))
bazel_build('bazel/snack', '//snack:image')
```

ip #6:
se CRDs to
odel cloud
esources

```
apiVersion: redis.cnrm.cloud.google.com/v1alpha2
kind: RedisInstance
metadata:
 name: redisinstance-sample
spec:
 displayName: Sample Redis Instance
 region: us-central1
 tier: BASIC
 memorySizeGb: 16
---------
apiVersion: service-operator.aws/v1alpha1
kind: ElastiCache
metadata:
 name: elasticache13
spec:
 cacheSubnetGroupName: "loadtest-cluster-k8s"
 vpcSecurityGroupIds: "sg-0581b94aa3c0db58c, sg-02b6d0034e8c2fa1b"
 autoMinorVersionUpgrade: true
 engine: redis
 engineVersion: 5.0.0
 numCacheNodes: 1
 port: 6379
 cacheNodeType: "cache.m4.large"
```

github.com/etsy/rules_grafana

```
# Picks up all *.json files in this directory:
json_dashboards(
    name = "json_dashboards",
    srcs = glob(["*.json"]),
)

# Picks up all *.py files in this directory:
py_dashboards(
    name = "py_dashboards",
    srcs = glob(["*.py"]),
)

# Built dashboards can be combined together in a filegroup for easy
access:
filegroup(
    name = "dashboards",
    srcs = [ ":json_dashboards", ":py_dashboards", ],
)

# Build the dashboards into a docker image:
grafana_image(
    name = "grafana",
    dashboards = [":dashboards"],
    datasources = [":datasources.yaml"],
)
```

Contact Us

@chrislovecnm
clove@google.com

@greggdonovan
gregg@etsy.com

etsy.com/shop/IrinaRedineArt

Thank you!

etsy.com/shop/CharlenePrecious