Building a Kubernetes Powered Central Modules Repository

CNCF Webinar, August 22nd 2019

@jfrog @rimusz

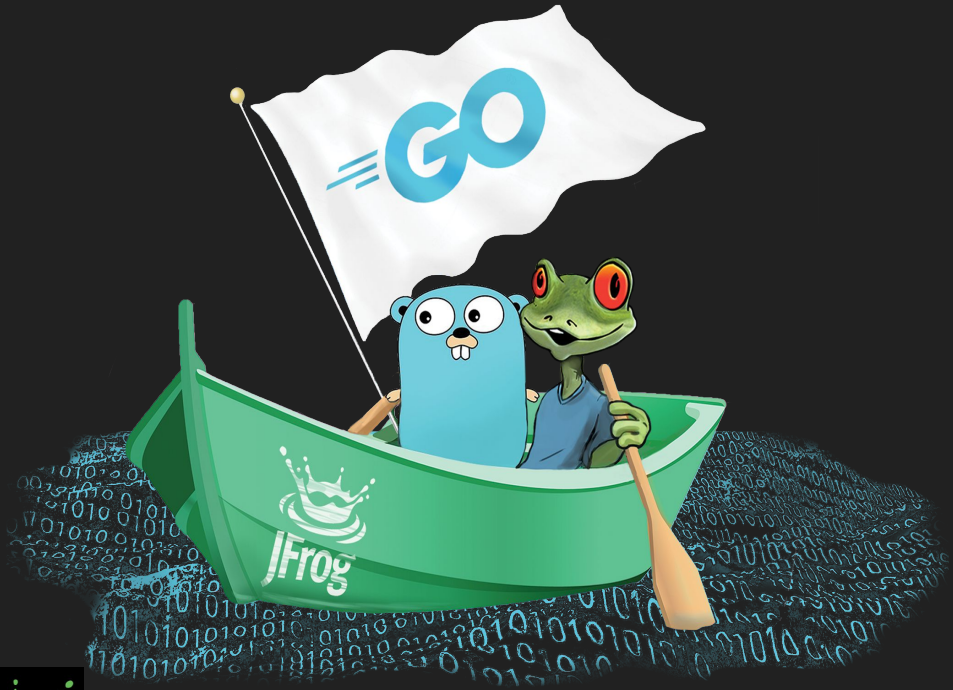
Who Am I?

- Rimantas Mocevicius
- Senior Developer
- Big supporter of Open Source Software, Co-founder of Helm, author of CoreOS Essentials book
- Love coffee, green tea, halva and cheesecake

JFrog

@jfrog @rimusz

We ♥ Golang and Kubernetes



@jfrog @rimusz

JFrog

GO Central Modules Repository

https://gocenter.io

@jfrog @rimusz

GO, GO, Kubernetes

Lessons learned and best practices

JFrog

@jfrog @rimusz

Running on Google Kubernetes Engine

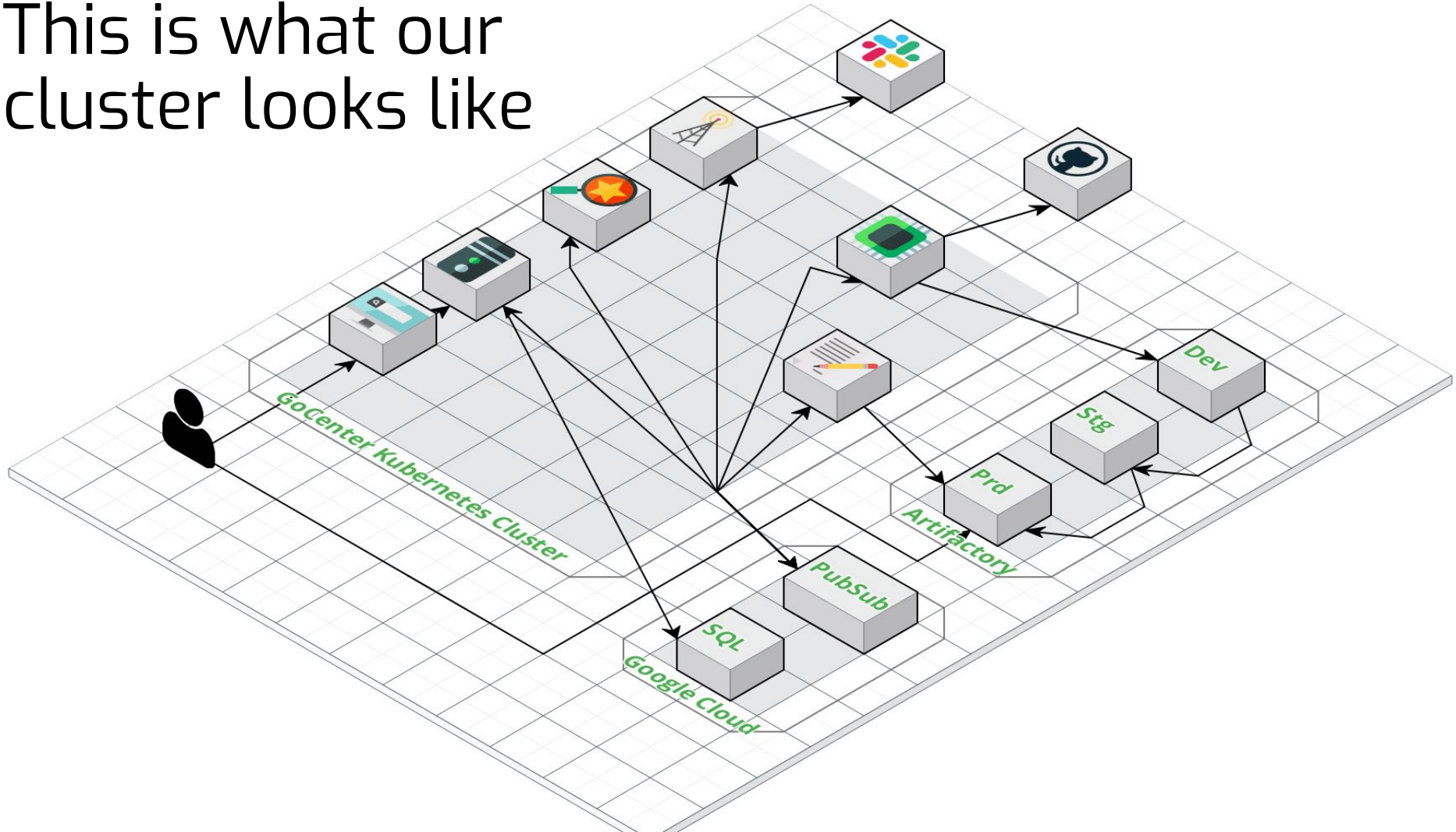
- Mature Kubernetes offering
- Choice of managed services
- Choice of troubleshooting tools like Stackdriver, BigQuery and etc
- Regional GKE clusters for Staging and Production

Number of clusters currently running

5

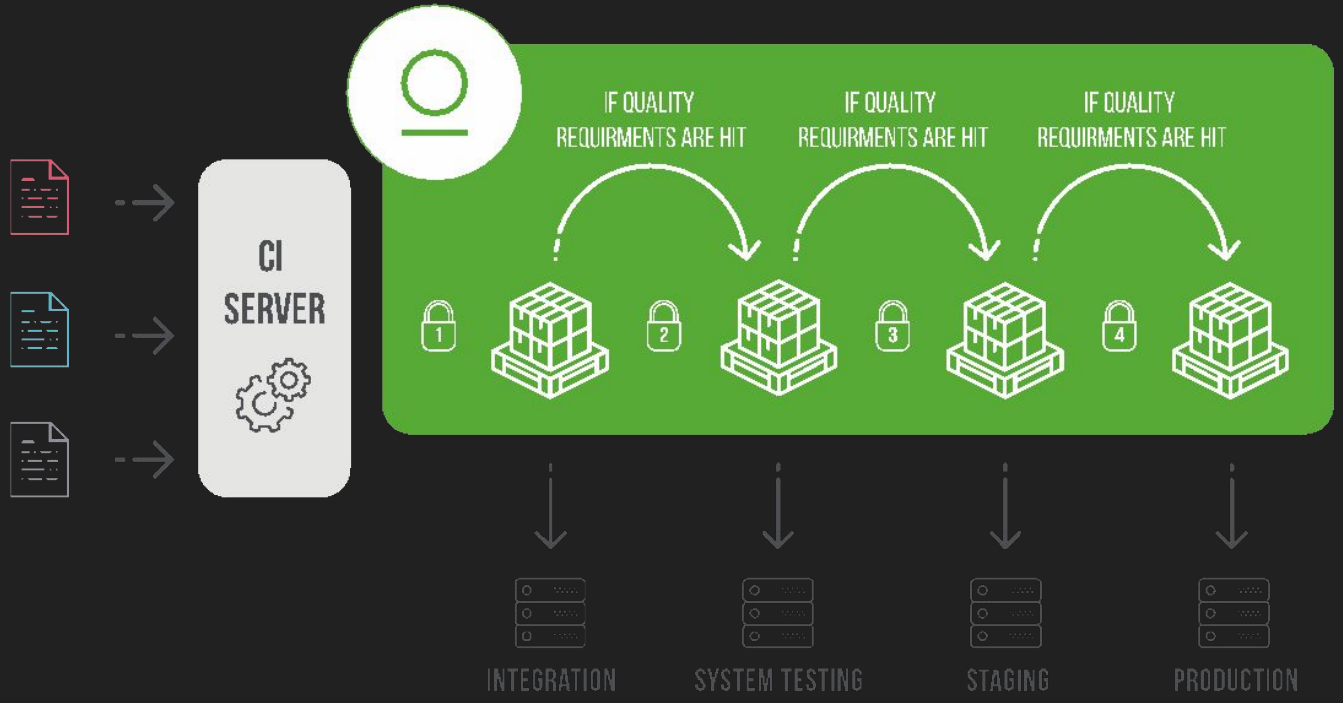
@jfrog @rimusz

This is what our cluster looks like



We're big on those pipelines and promotions concepts

Feature branch --->



Implementing with best practices

- Tillerless Helm v2
- Secrets
- Namespaces
- Persistence
- Probes
- Resource Limits
- Node/pode affinity
- Node pools

@jfrog @rimusz

Deployments using Helm

- Tillerless Helm v2 is used
- Helm Charts are versioned
- Separate values files per environment
- Even secrets are Helm Charts
- Namespaces

Deployments using Helm

```
resources:
  limits:
    cpu: 800m
    memory: 2Gi
  requests:
    cpu: 800m
    memory: 1Gi
```

```
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: node-pool
            operator: In
            values:
            - pvm-pool
    podAntiAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
      - labelSelector:
          matchExpressions:
          - key: app
            operator: In
            values:
            - discovery
        topologyKey: kubernetes.io/hostname
```

JFrog

Deployments using Helm

```
readinessProbe:
  failureThreshold: 3
  httpGet:
    path: /ping
    port: http
    scheme: HTTP
  periodSeconds: 10
  successThreshold: 1
  timeoutSeconds: 1
```

```
# Create a StorageClass for the regional disk
kubectl apply -f - <<EOF
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: repd-${GCP_REGION}-b-c
provisioner: kubernetes.io/gce-pd
parameters:
  type: pd-standard
  replication-type: regional-pd
  zones: ${GCP_REGION}-b, ${GCP_REGION}-c
allowVolumeExpansion: true
reclaimPolicy: Delete
EOF
```

JFrog

Scaling on many levels

Node

Provision nodes when needed

Compute

Run more microservices

Messaging

Scaling based on queue depth

Up and Down

What goes up must come down… usually

@jfrog @rimusz

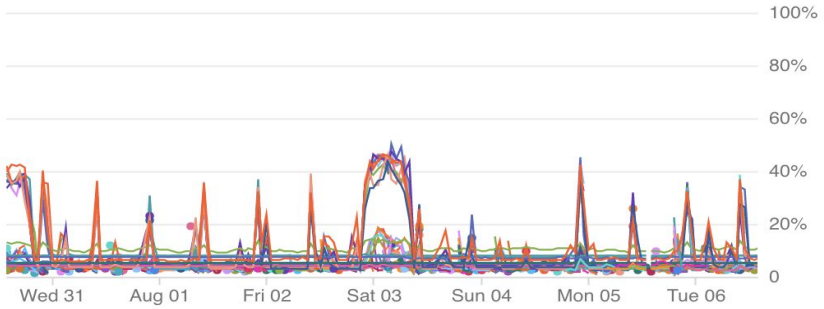
GoCenter

Off TIME 1h 6h 1d 1w 1m 6w custom Add Chart

Filter...

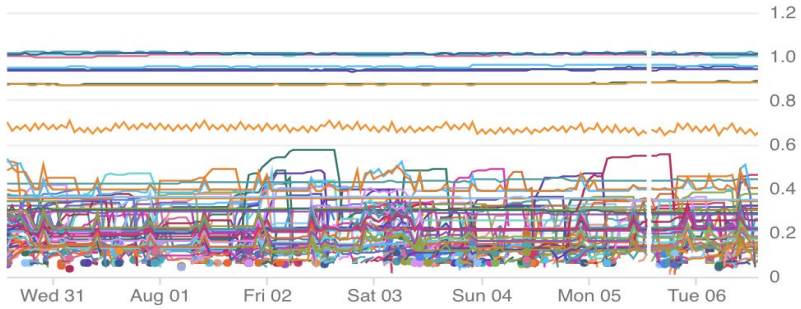
Kubernetes Nodes - CPU utilization

1 1 hr interval (mean)



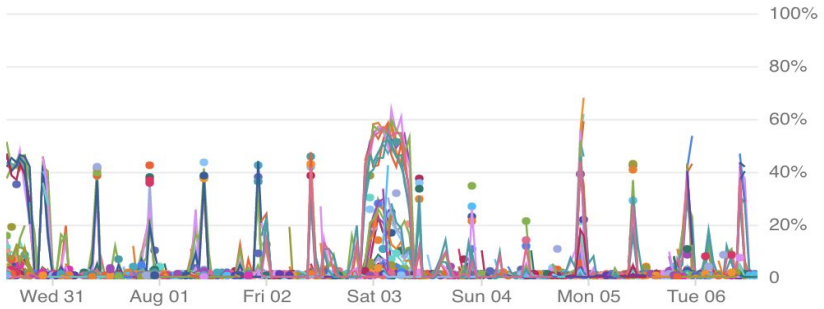
Kubernetes Nodes - Memory allocatable utilization

1 hr interval (mean)



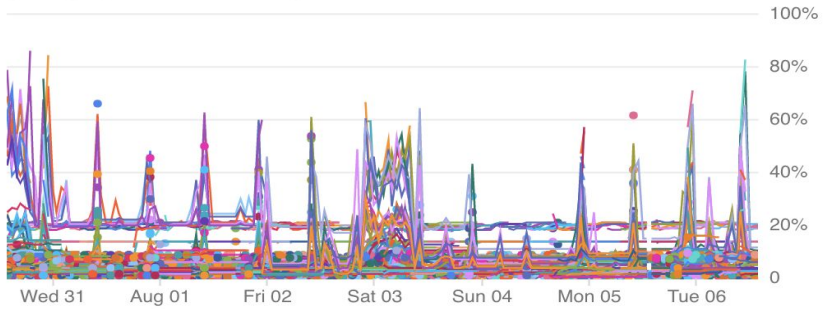
Container - CPU limit utilization for caddyshack.*

1 1 hr interval (mean)



Container - Memory limit utilization for caddyshack.*

1 1 hr interval (mean)

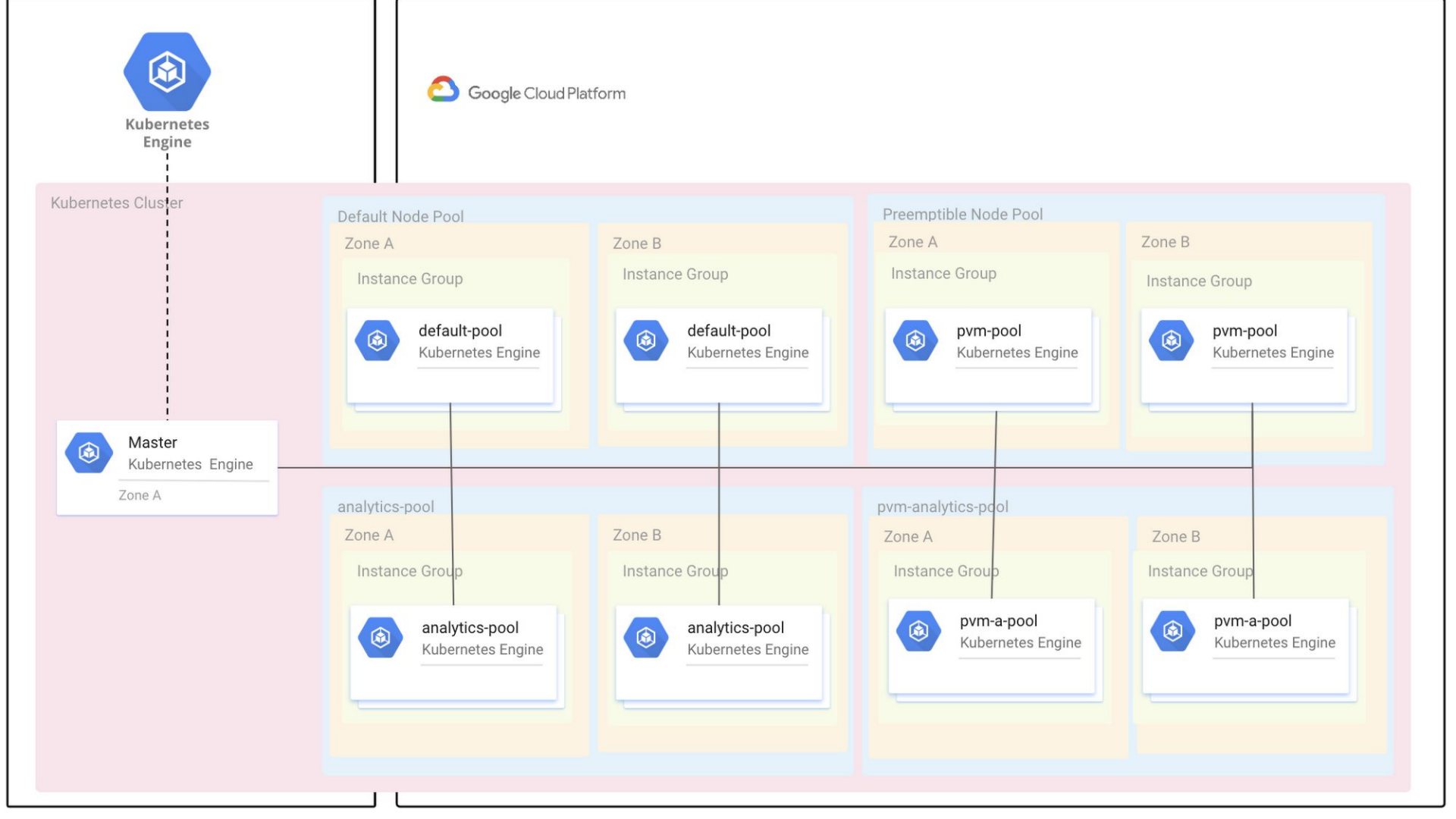


Scaling on many levels

15 pods ---> 3 Kubernetes nodes

25 pods ---> 5 Kubernetes nodes

Node-pools


Kubernetes Engine
Google Cloud Platform
Kubernetes Cluster
Default Node Pool
Preemptible Node Pool
Zone A
Zone B
Instance Group
default-pool
Kubernetes Engine
pvm-pool
Kubernetes Engine
Master
Kubernetes Engine
Zone A
analytics-pool
pvm-analytics-pool
pvm-a-pool
Kubernetes Engine

Node-pools

Node pools

Filter node pools

Columns

Name	Status	Number of nodes	Machine type	Image type	Autoscaling	
default-pool	OK	8 (2 - 3 per zone)	custom-4-12288	Container-Optimized OS (cos)	2 - 15 nodes per zone	
elastic-data-pool	OK	6 (2 per zone)	custom-2-6144	Container-Optimized OS (cos)	1 - 5 nodes per zone	
elastic-master-pool	OK	3 (1 per zone)	custom-1-4096	Container-Optimized OS (cos)	1 - 5 nodes per zone	
pvm-pool-1	OK	10 (3 - 4 per zone)	custom-4-12288	Container-Optimized OS (cos)	2 - 15 nodes per zone	

@jfrog @rimusz

Keeping a close eye on cluster and container health

- Stackdriver
- Prometheus
- Grafana

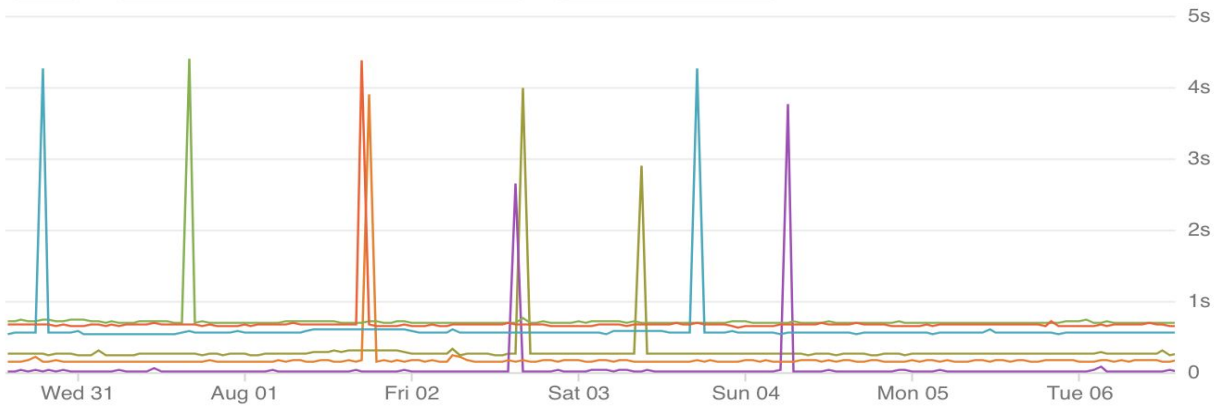
Stackdriver

- Monitoring Overview
- Resources
- Alerting
- Uptime Checks
- Groups
- Dashboards
- Debug
- Trace
- Logging
- Error Reporting
- Profiler

Uptime Checks / search.gocenter.io check BETA

Uptime Check latency

1 | by checker location, instance id (mean) | 1 hr interval (mean)



Stackdriver
Logging

CREATE METRIC CREATE EXPORT

- Logs Viewer
- Logs-based metrics
- Exports
- Logs ingestion

Filter by label or text search

Kubernetes Container, caddyshack-prod1, cad... | All logs | Any log level | Last hour | Jump to now

Showing logs from **12:53 PM** to **now** (BST) Download logs View Options

```
2019-08-06 13:50:34.231 BST Request Received: GET /ping
2019-08-06 13:50:34.231 BST Sending 200 response in default format: Success: true, Message: Pong
2019-08-06 13:50:35.152 BST Request Received: GET /ping
2019-08-06 13:50:35.152 BST Sending 200 response in default format: Success: true, Message: Pong
2019-08-06 13:50:35.293 BST Request Received: GET /ping
2019-08-06 13:50:35.293 BST Sending 200 response in default format: Success: true, Message: Pong
2019-08-06 13:50:35.307 BST Request Received: GET /ping
2019-08-06 13:50:35.307 BST Sending 200 response in default format: Success: true, Message: Pong
2019-08-06 13:50:35.423 BST No message available. Going to sleep...
2019-08-06 13:50:36.245 BST Running job job-process-module...
2019-08-06 13:50:36.286 BST Receiving message from subscription processing
2019-08-06 13:50:36.503 BST Request Received: GET /ping
2019-08-06 13:50:36.503 BST Sending 200 response in default format: Success: true, Message: Pong
2019-08-06 13:50:39.576 BST Request Received: GET /ping
2019-08-06 13:50:39.576 BST Sending 200 response in default format: Success: true, Message: Pong
2019-08-06 13:50:39.665 BST Request Received: GET /ping
2019-08-06 13:50:39.665 BST Sending 200 response in default format: Success: true, Message: Pong
2019-08-06 13:50:39.824 BST Running job job-process-module...
2019-08-06 13:50:39.863 BST Receiving message from subscription processing
2019-08-06 13:50:40.591 BST Request Received: GET /ping
```

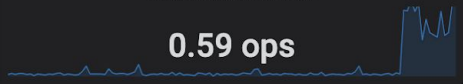
NGINX Ingress controller

Last 1 hour Refresh every 5s

Namespace: All | Controller Class: All | Controller: All | Ingress: All | Config Reloads

Controller Request Volume

0.59 ops

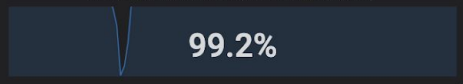


Controller Connections

16

Controller Success Rate (non-4|5xx responses)

99.2%



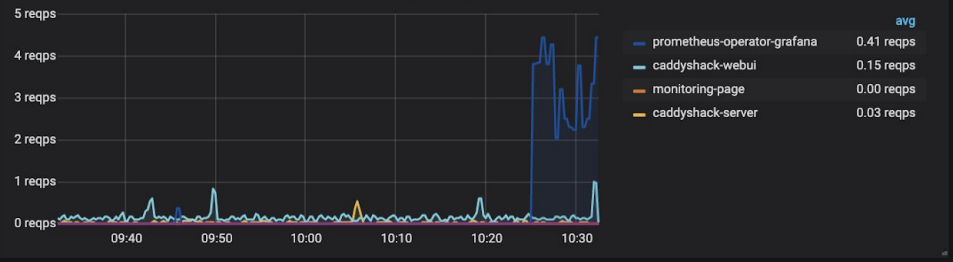
Config Reloads

33

Last Config Failed

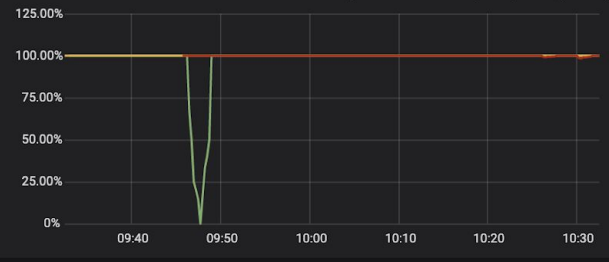
N/A

Ingress Request Volume



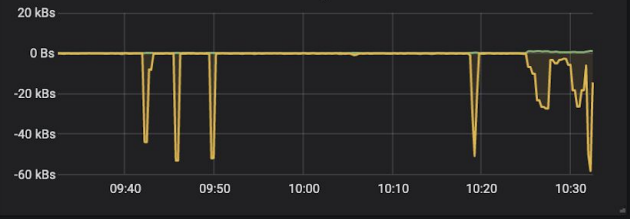
	avg
prometheus-operator-grafana	0.41 reqps
caddyshack-webui	0.15 reqps
monitoring-page	0.00 reqps
caddyshack-server	0.03 reqps

Ingress Success Rate (non-4|5xx responses)

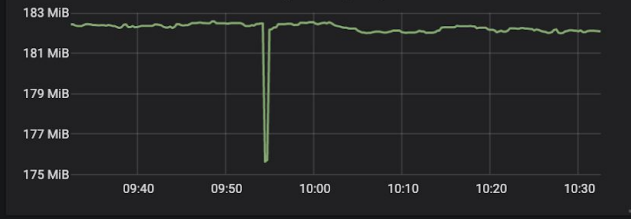


	avg
monitoring-page	100.00%
caddyshack-webui	100.00%
prometheus-operator-grafana	99.70%
caddyshack-server	96.86%

Network I/O pressure



Average Memory Usage

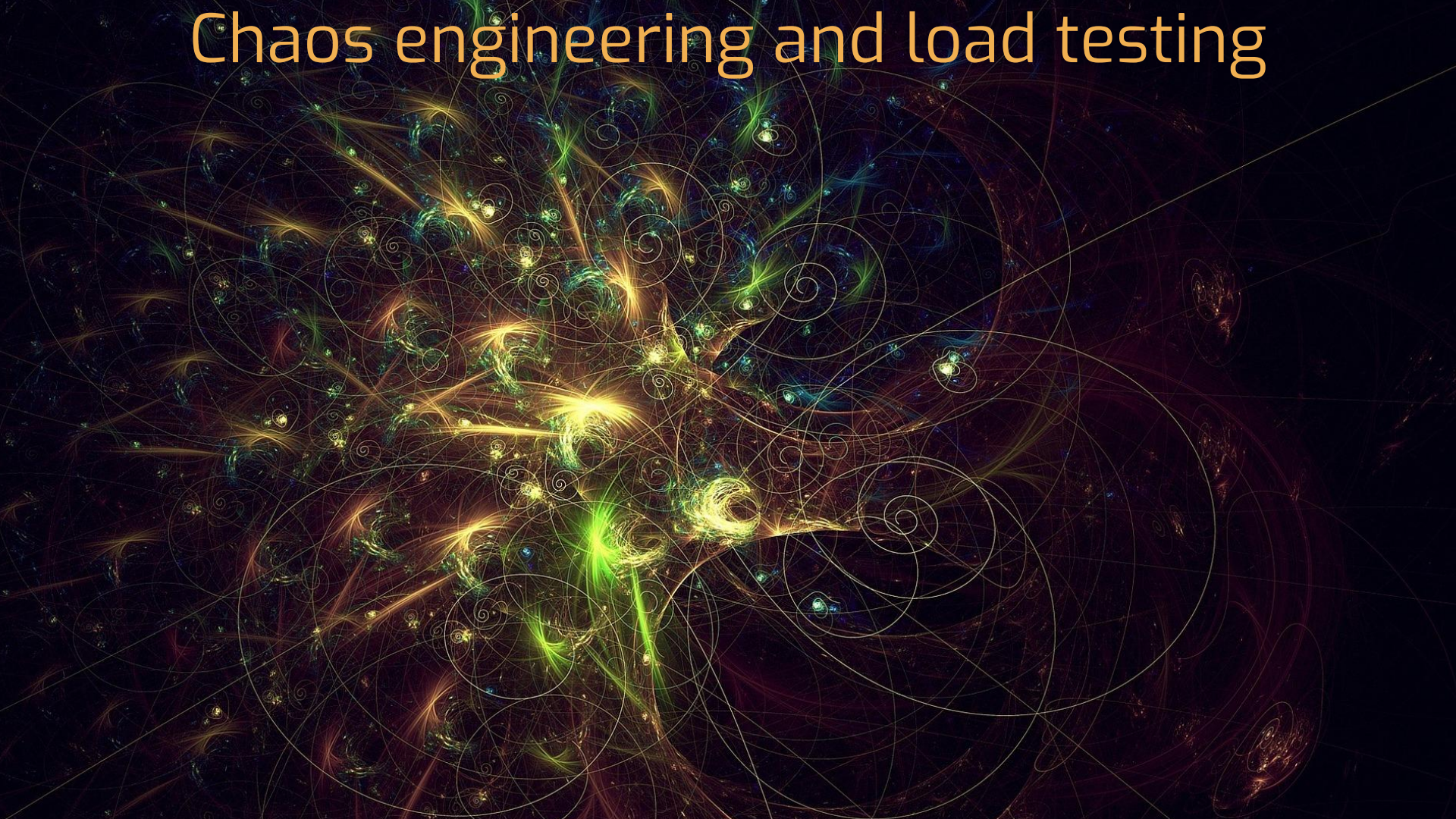


Average CPU Usage

Ingress Percentile Response Times and Transfer Rates

Ingress	P50 Latency	P90 Latency	P99 Latency	IN	OUT
prometheus-operator-prometheus				0 Bs	0 Bs
prometheus-operator-grafana	60 milliseconds	485 milliseconds	2 seconds	1.03 kBs	14.46 kBs
prometheus-operator-alertmanager				0 Bs	0 Bs
monitoring-page				0 Bs	0 Bs
kubernetes-dashboard				0 Bs	0 Bs
kibana-caddyshack-kibana				0 Bs	0 Bs
elasticsearch-data				0 Bs	0 Bs

Chaos engineering and load testing



We keep in touch with our nodes

Kube-Slack

jenkins APP 7:24 AM
Started Job 'SolDev/caddyshack_deploy_prod' to deploy '0.1.147-2' on prod.

gocenter APP 7:25 AM
containers with unready status: [caddyshack-proxy]

caddyshack/caddyshack-webui-85d58dd497-t66lk: ContainersNotReady
containers with unready status: [caddyshack-proxy]

Pod is ready

caddyshack/caddyshack-webui-85d58dd497-t66lk: Pod is ready
Pod is ready

containers with unready status: [caddyshack-ui caddyshack-proxy]

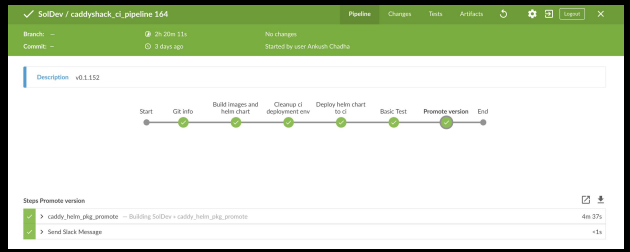
caddyshack/caddyshack-webui-85d58dd497-kjbs8: ContainersNotReady
containers with unready status: [caddyshack-ui caddyshack-proxy]

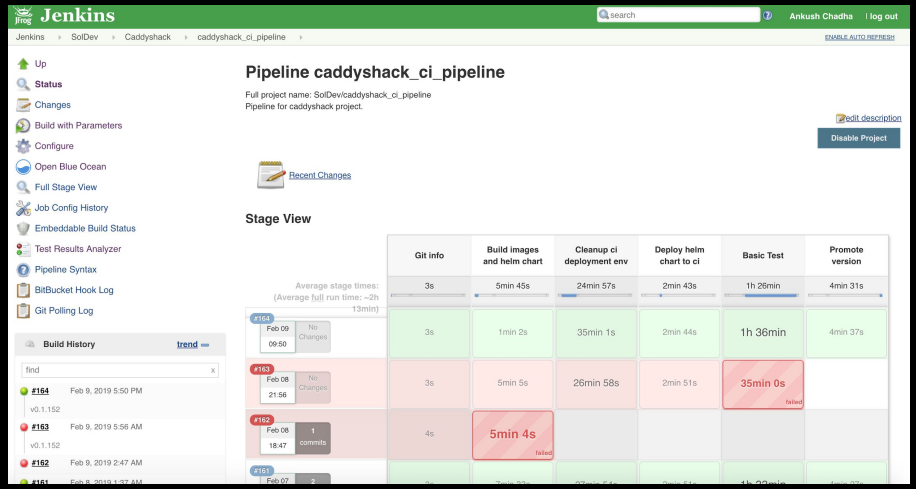
Pod is ready

caddyshack/caddyshack-webui-85d58dd497-kjbs8: Pod is ready
Pod is ready

jenkins APP 7:28 AM
Finished Job 'SolDev/caddyshack_deploy_prod'. Deployment '0.1.147-2' on prod.

Jenkins





To use GoCenter: `export GOPROXY=https://gocenter.io`

Set Me Up

github.com/goreleaser/goreleaser

github.com/goreleaser/goreleaser Version: v0.113.0

README | Mod File | Dependencies (25) | Used By

GoReleaser

Deliver Go binaries as fast and easily as possible.

GoReleaser builds Go binaries for several platforms, creates a GitHub release and then pushes a Homebrew formula to a tap repository. All that wrapped in your favorite CI.

This project adheres to the Contributor Covenant code of conduct. By participating, you are expected to uphold this code. We appreciate your contribution. Please refer to our contributing guidelines for further information.

For questions join the #goreleaser channel in the Gophers Slack.

Get GoReleaser

Questions?

@jfrog @rimusz

Thank you!

@jfrog @rimusz