Google Container Background

- Google knows Containers
  - Been doing this for ~10 years.
  - Everything from Search to GCE VMs run in containers.
  - We launch 2B containers every week
- Deep understanding of what it takes to launch, run and manage containers at scale.
  - We have lots of experience/scars
- Core contributor to linux cgroups and Docker libcontainer.
  - In some ways, we invented the core technology for containers on Linux
- Containers are a keystone technology for cluster management systems.
Containers Outside of Google

Customers look for 3 things

• Packaging and portability
  • Easy to capture and move systems between environments

• Density
  • Use every part of the "animal." Drive utilization up. Both in VM clouds and on bare metal.

• Security
  • !! Note: We don't think that container technology is there yet as a hard security boundary. Surface area to secure is too big. But, when combined with other mitigations, it is doable.
Benefits to users:

- Even higher utilization and efficiency
  - Larger pool of workloads and compute allows for more complementary workloads to be scheduled.
- Easy platform for horizontal scaling
  - Running, tracking, managing arrays processes is easy.
- Self healing
  - Workloads can be dynamically rescheduled in the face of individual machine failure.
- Allows for specialization of ops roles. Cluster ops vs. individual app ops.
  - Internal Example: GMail SRE rarely has to talk to Borg SRE
  - Insights into performance and metrics at the app server level vs. machine level.
- Pairs well with microservices
  - Reducing the management burden allows you to manage more things
  - Keep deployed/managed items small enough for efficient teams.
Kubernetes

Open Source Container Cluster
• Inspired by internal systems
• Focused on portability -- run anywhere
• Launched in June
• Active developer community
  • 100+ contributors, 5100+ GitHub stars, active IRC channel
• Open roadmap: roadmap.md on GitHub
• Built on Docker
Kubernetes

Current status

• Releasing every ~2 weeks
• Most moving parts are in place
• Road to v1:
  • Improve usability
  • Introspectability
  • Production reliability
  • Cluster upgrade
  • Finalize API with deprecation policy
  • Pluggable auth model
Kubernetes Concepts

- Dynamic Container Placement
  - Includes replacement on machine failure
  - Groups of containers that must be co-scheduled: **Pods**
- Native support for sets of containers
  - **Labels** on containers with query/selection syntax
  - Horizontal scaling based on template: **ReplicationController**
  - Critical for visualization, logging and monitoring
- **Services**: connections between containers and beyond
  - Find containers based on name/labels
  - Easily communicate with a set of containers
  - Import/export services from cluster
Google Container Engine

Managed and Enhanced Kubernetes Cluster

Goal: Google is your "cluster ops"

Now:
  • One click/API call to spin up cluster
  • Builds on Kubernetes API/tools for using the cluster.

Coming:
  • Automatic scaling of cluster pool.
  • Integration with GCP around storage, logging, monitoring, LB, auth.
  • Regional/Global solutions.
Beyond Kubernetes and GKE

Kubernetes/GKE is not a PaaS!

It is a building block for compute workload management

Missing parts:
- Application description and deployments, upgrade
- HTTP router
- "Just bring the code"

There will be many solutions to these problems both on GCP and beyond.

App Engine is will be built on GKE -- Navneet to detail later.