Building and Running Applications on Kubernetes

A Practical Hello World

Vallery Lancey
Lead DevOps Engineer, Checkfront

@vllry

Did Anyone Forget To Set Up Their Demo?

https://github.com/vllry/startupslam-2018

@vllry

The Agenda

- Meet our app
- Basic concepts of Kubernetes
- Setting up our app
- Basics beyond our app

Meet Our App

App Specs

- “True” stateless.
  - No data is stored by the app → The app can be replicated without storage/sharding concerns.
- HTTP API in a container.

Basic Concepts of Kubernetes

@vllry

Primer / Reminder: What Does Kubernetes Do?

- Automatically provisions and manages deployed software.
- Allows for automatic scaling - reduces downtime and hardware overspend.
- Creates a clear(er) divide between system and app, allowing respective teams to focus.

Containers

- Lightweight process abstraction.
- Restricts process interaction and filesystem to a closed environment.
  - Containers are not inherently secure! Special care is needed for untrusted workloads.
- Process resources are controllable (CPU, memory, etc).
- Typically use a pre-built “image” (filesystem archive) to boot a container.

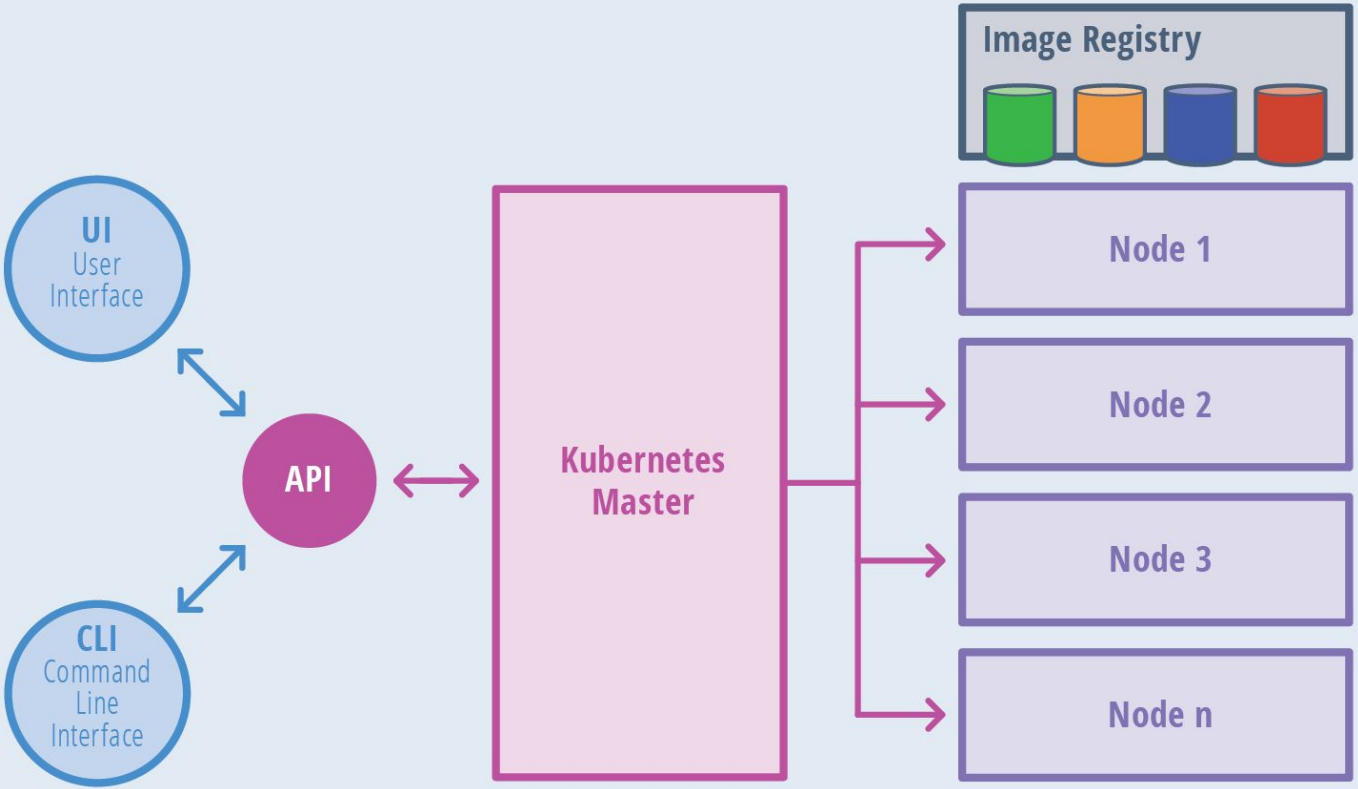
Container Orchestration

- Operators use a declarative config (EG replicas=4) to specify how many copies of a container set to run.
- Kubernetes automatically creates and reaps containers to stick to this spec.
- New containers are launched if a container crashes or a Kubernetes node fails.

Container Networking

- Traditional deployments have a 1:1, long-lived relationship between a machine and a service.
- Containers are randomly scattered across machines on the cluster.
- Kubernetes translates abstract services (EG “api”) to a concrete container in the cluster.

Kubernetes Architecture

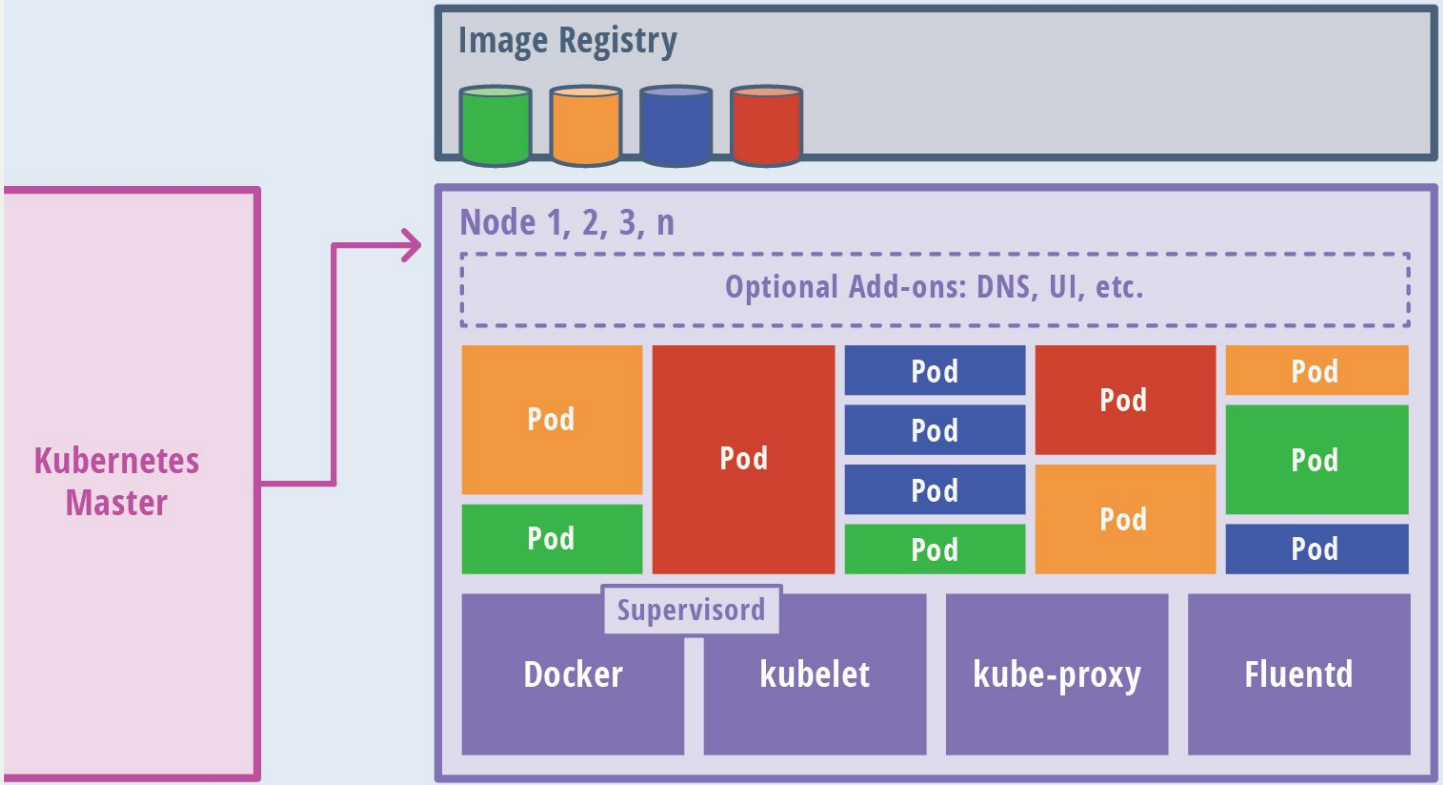


Source: Janakiram MSV

THENEWSTACK

@vllry

Kubernetes Node



Source: Janakiram MSV

THENEWSTACK

@vllry

Special Considerations vs Traditional Deployment

- Individual instances are ephemeral and mortal.
  - Persistent data must be shifted to dedicated data stores.
  - Session pinning is weak.
  - Local telemetry data (logs, metrics, etc) must be *aggregated* in a central location.

- Shameless plug: Challenges To Writing Cloud Native Applications, https://www.youtube.com/watch?v=di6oFceM_CQ

Setting Up Our App

 ← Indicates a hands-on step (terminal, web UI, etc)

https://github.com/vllry/startupslam-2018

@vllry

Checklist

We need…

- Container images
- A Kubernetes cluster
- Deployment to Kubernetes
- Network access to app
- Deployment auto-scaling

Container Images

Container Builds and Registry

- Dockerfile in repo.
- Want to build image, tag it, and push it to a registry.
- We'll use the Google Container Registry.

- https://console.cloud.google.com
  - Free trial available for new Google Cloud users, otherwise this demo should be <1$.

A Cluster

Google Kubernetes Engine - GKE

- Manages Kubernetes nodes (servers running Kubernetes itself).
- This allows us to focus on managing apps and Kubernetes.

Deploying The App

The Deployment Resource

- *Deployments* are the typical way to run an app.
- Specifies a group of containers, their config, and target # of replicas.
- Kubernetes ensures that the target number of replicas exist, by creating or reaping pods.

- https://kubernetes.io/docs/concepts/workloads/controllers/deployment/

Deploying With kubectl

- kubectl is the “default” command-line tool for interacting with Kubernetes.
  - There are other CLI tools, some GUIs, and the Kubernetes API itself.
- Once kubectl is authenticated with our cluster, we can use it to create and verify our deployment.
- We won’t yet be able to connect to our app.

Adding Network Access

Services

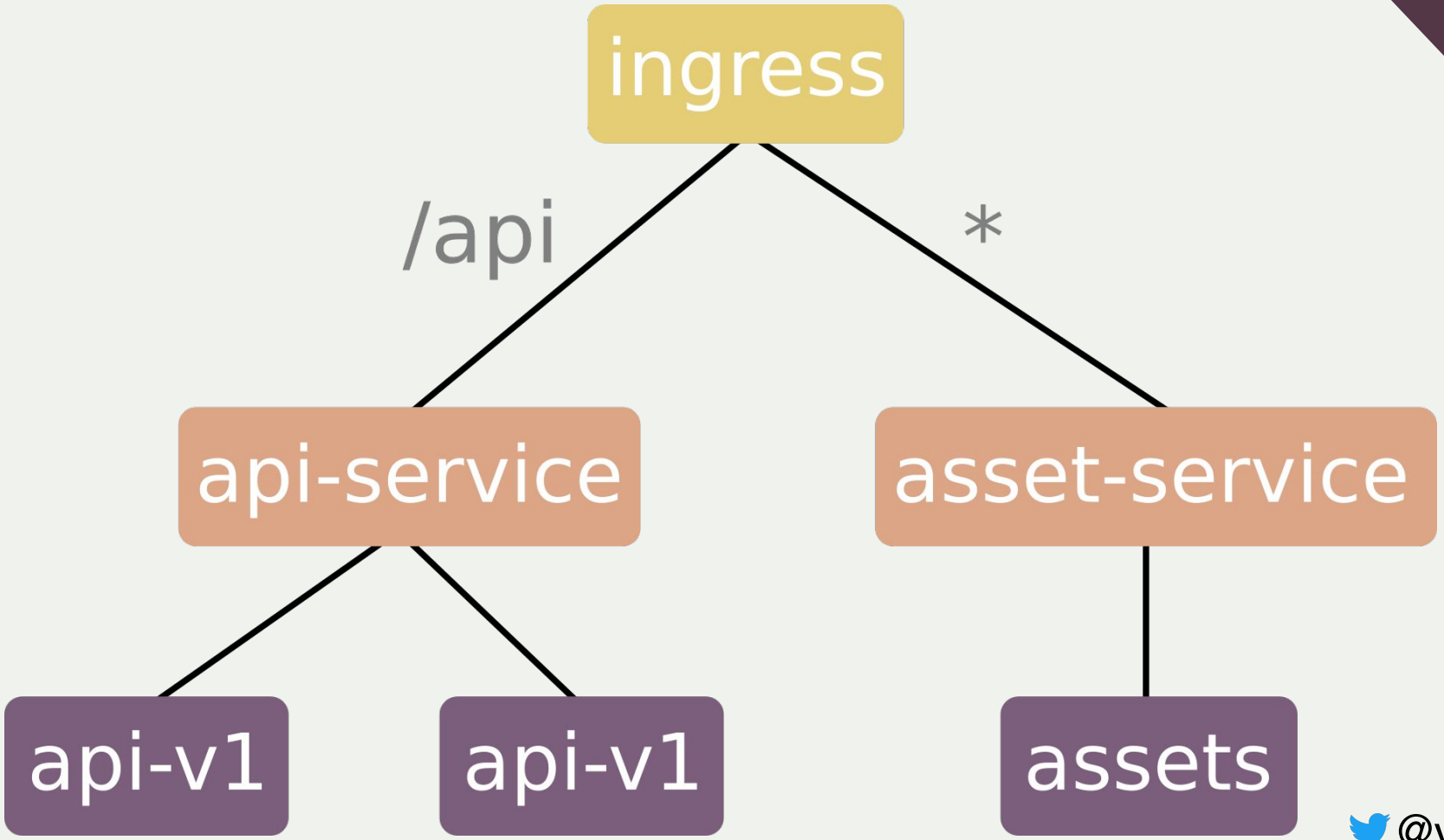
- *Services* map an abstract service identifier to an appropriate pod.
- Network call: myapp → myapp-blue-2334:443
- Used in both intra-cluster and inbound networking.

- https://kubernetes.io/docs/concepts/services-networking/service/

Ingress

- Maps on to services based on host & path rules.
- Use an ingress controller to handline inbound traffic from the outside world.
  - Ingress controllers differ, and are chosen for specific features and IaaS/baremetal compatibility.

- https://kubernetes.io/docs/concepts/services-networking/ingress/



@vllry

Deploying the Service and Ingress

- Again, kubectl will allow us to deploy our resources. Once these are deployed, we can access our app from the outside world.
- Our ingress config will create a Google Load Balancer to funnel traffic in to our Kubernetes VMs.

Scaling

So, Does it Autoscale?

- Sorry, not quite yet...
- Deployments have a precise target (EG 4), no ranges or logic.
- We scale by using a changing that target - manually, with our own tool, or with the horizontal pod autoscaler.

Horizontal Pod Autoscaler

- The horizontal pod autoscaler updates the deployment replica target, based on metrics.
- Default metric is CPU-use base, but custom ones can be added.

- https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale-walkthrough/

Beyond Our App

Other key things to learn

@vllry

Outsourcing Persistence

- A common/easy way to run data stores is to run them outside of Kubernetes.
- May use a managed service, or traditional nodes.
- A tunnel or shared network should be used to prevent over-exposing the data store.

In-Cluster Storage With StatefulSets

- Essentially deployments, with a twist.
- Statefulsets support stable storage/network attachment. IE a disk/interface is always re-assigned to the “same” pod.
- Can be created and terminated in order.

- https://kubernetes.io/docs/concepts/workloads/controllers/statefulset/

Misc Things To Know - App Devs

- If your ops team isn't using an e2e encryption system (EG Istio), you need to encrypt comms within the cluster (nginx, stunnel, etc).
- There's a lot you can do with deployment/service config. You should be writing these, not ops.
- API is highly usable - use API integrations for CD, admin tools, monitoring, etc.
- CRDs (custom resource definitions) - way to add new concepts and abstractions to Kubernetes itself.

Misc Things To Know - Cluster Admin

- Running clusters is hard - you probably want a managed service like GKE… especially at Victoria scale. ;)
- You probably want “a system person” before adopting.
- Clusters can get hacked, and it’s *really* bad - isolate network access to the control plane and intra-cluster networking.
- Use RBAC profiles for developers and service accounts.
- https://kubernetes.io/docs has many explanations and common use cases. It’s great for finding the best way to solve a problem.

And Finally...

@vllry

Don’t Forget To Clean Up Your Demo!

- Delete the cluster.
- Delete any container images you’ve uploaded.

@vllry

Questions?

This content? Kubernetes?
Cloud platforms in general?

Repo:
https://github.com/vllry/startupslam-2018

@vllry