Kubernetes Overview

Presentation to MAGIC Group on Containers & Virtualization

Rick Wagner
rick@globus.org

February 7, 2018

Agenda

- **Kubernetes Overview**
- **Research CI Perspective**
- **More Information**

Kubernetes Overview

We’re Skipping Containers

- **Lots of other presenters on containers**
- **This is a 30,000’ view of Kubernetes**
- **See the *More Information* slides for… where to find more information**
- **The following slides are a good balance between quick & deep**
  Kubernetes: Container Orchestration and Micro-Services
  `https://courses.cs.washington.edu/courses/cse550/16au/notes/kubernetes.pdf`

  Some content from https://kubernetes.io/docs/ CC BY 4.0

What is Kubernetes

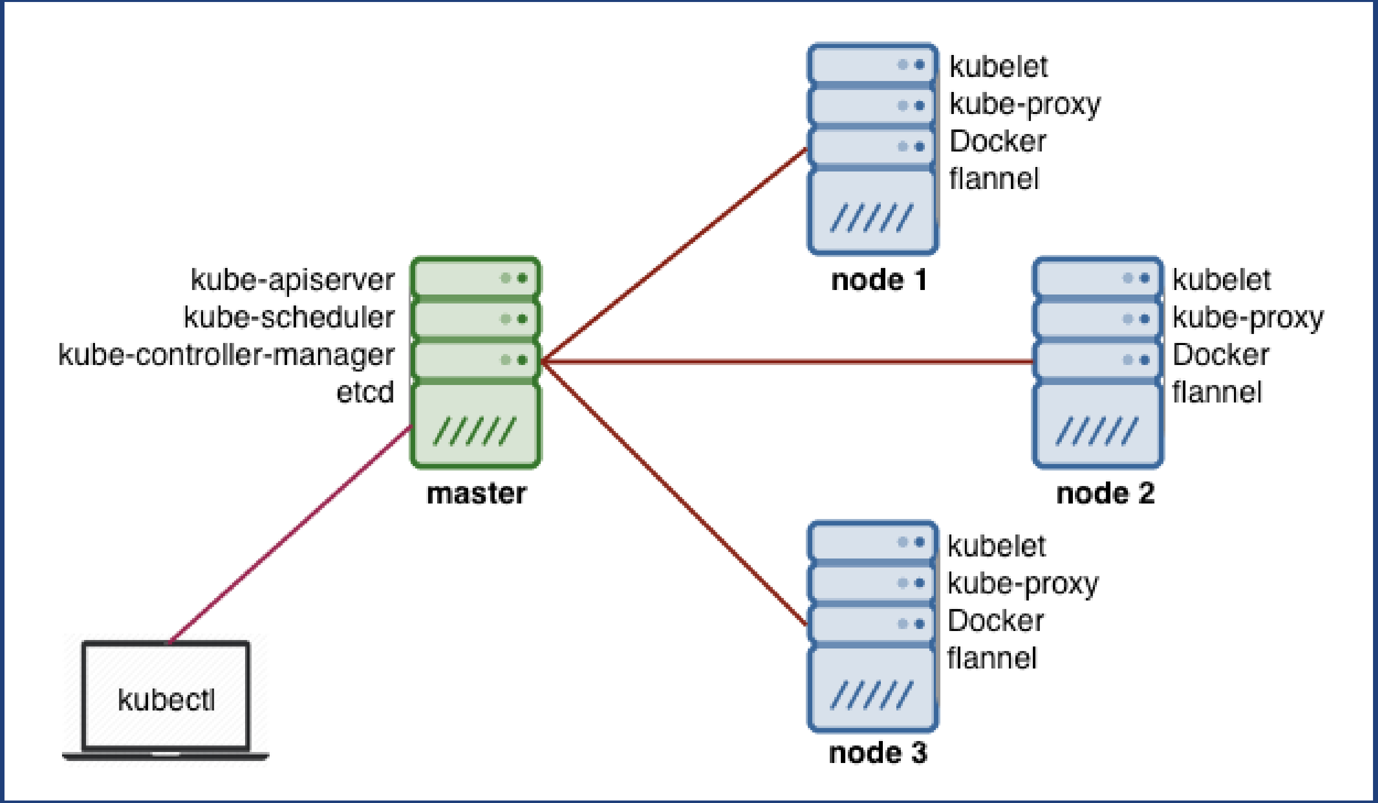
Kubernetes is an open-source platform designed to automate deploying, scaling, and operating application containers.

- **Portable:** public, private, hybrid, multi-cloud
- **Extensible:** modular, pluggable, hookable, composable
- **Self-healing:** auto-placement, auto-restart, auto-replication, auto-scaling

Currently in production or preview on Amazon, Google, and Azure

Kubernetes Architecture

- **Master**
  API server, scheduler, controller manager, and etcd (HA key-value store for config and service discovery)
- **Node**
  Docker or similar (e.g., Rocket) to run containers, kube-proxy (net access to apps), kubelet (takes k8s commands)



Kubernetes Concepts

Pods: Group of one or more containers, their storage, and config/run options; each **Pod** gets its own IP address

Labels: Key/value pairs that Kubernetes attaches to any object (e.g., a **Pod**)

Annotations: Key/value pairs for arbitrary non-queryable metadata

Services: Abstraction defining a logical set of **Pods** and a network access policy

Replication Controller: Manage number of pod replicas running

Secrets: Sensitive information (passwords, certificates, OAuth tokens, etc.)

ConfigMap: Mechanisms used to inject config into containers while keeping containers agnostic of Kubernetes

Putting it Together

Pods: Group of containers, basically an application

Pods run on **Nodes**

The (pluggable) **Scheduler** picks the **Nodes** based on the **Pods**’ needs

The **Replication Controller** makes sure that enough **Pods** are running, if they’re replicated (self-healing)

A virtual IP per **Service**, via a proxy on the **Node** (avoids port collisions)

Storage

Container: Ephemeral, tied to the lifecycle of a container

Volumes: Less ephemeral, tied to the lifecycle of a **Pod**

PersistentVolumes and PersistentVolumeClaims:
Not ephemeral; cluster operators define **PersistentVolume** objects, application developers define **PersistentVolumeClaim** objects

Research CI Perspective

Common Interface

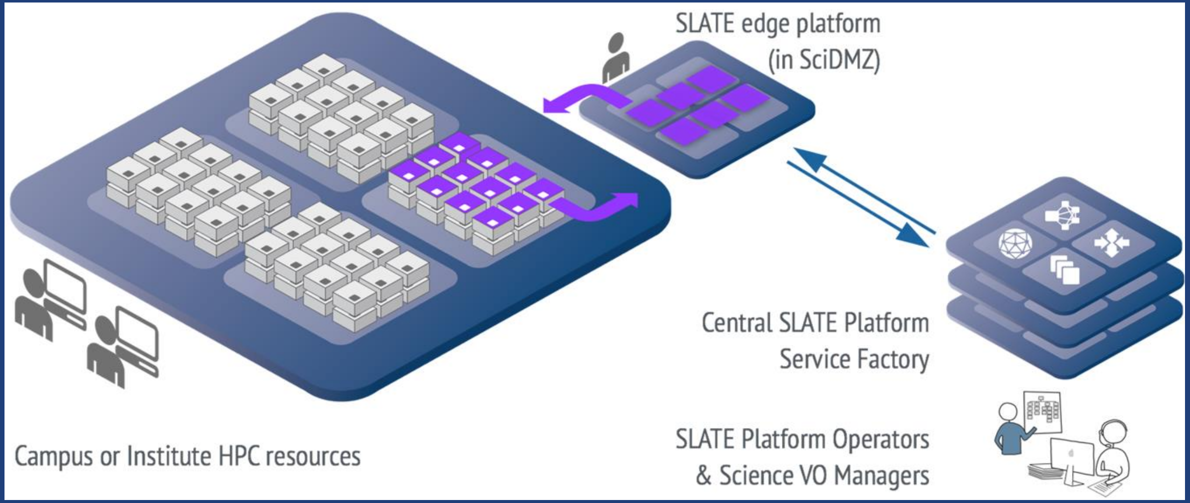
- **Aligns institutional & commercial solutions**
- **Single technology**
  - Familiarity
  - Less startup costs
- **Improves portability**
  - Greater chance of researchers and projects being able to leverage more resources
- **Workforce**
  - Staff understand on-prem and cloud solutions
  - Staff aren’t siloed into a narrow vertical market

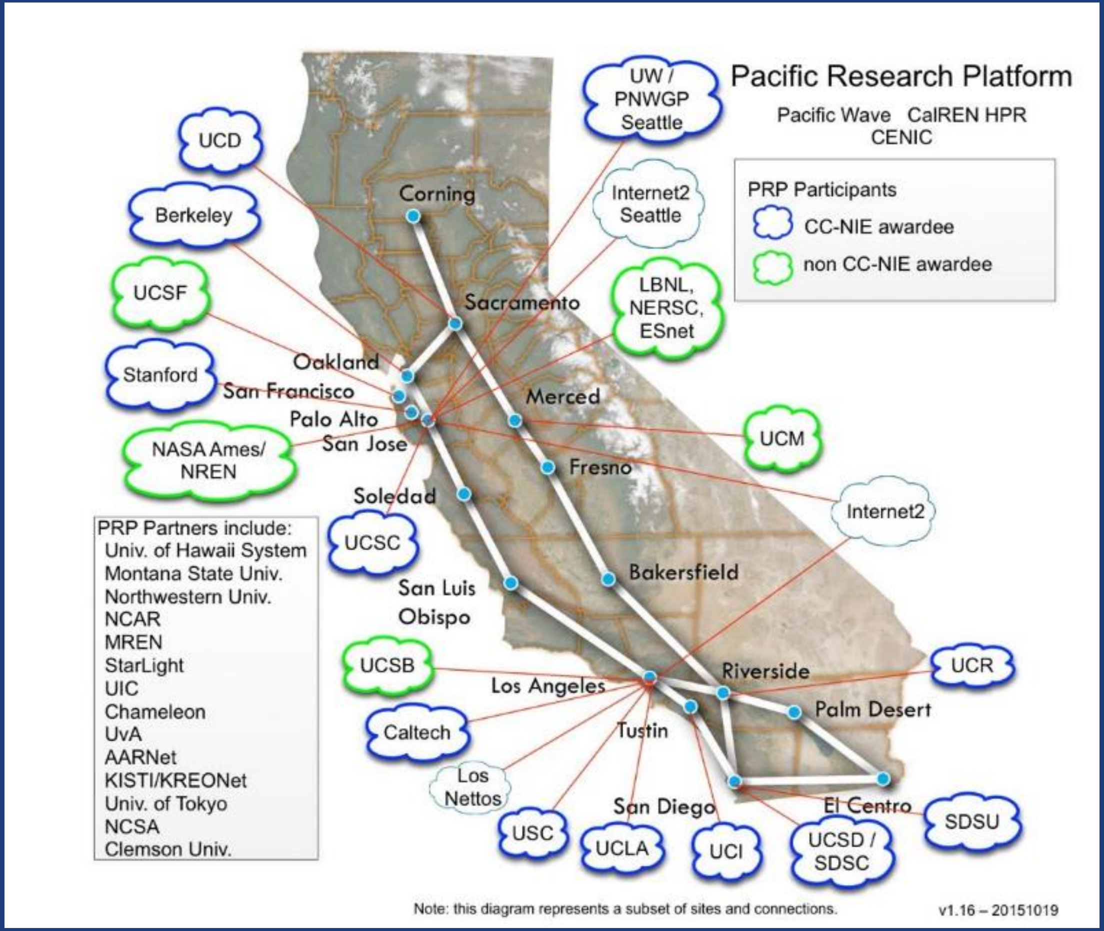
Who is it for?

- **Power users?**
  - Maybe
  - Research is full of those of us who need to try things
- **Projects**
  - More likely
  - Application control
  - Less need for multi-user environments
  - Data pipelines and workflows
  - Kubernetes inherently understands jobs and fault tolerance
  - Infrastructure may become less project-centric

Edge Models

SLATE CI and the PRP are using Kubernetes to decouple application and infrastructure support


SLATE edge platform (in SciDMZ)
Central SLATE Platform Service Factory
SLATE Platform Operators & Science VO Managers
Campus or Institute HPC resources



SLATE: Services Layer at the Edge and the Mobility of Capability

http://slateci.io

PRP: Pacific Research Platform

http://prp.ucsd.edu

Concerns

- **Service model is not familiar to current HPC & supercomputer centers**
  - I.e., running a backend REST API
  - Goal is ZERO interaction between application and infrastructure teams
  - Hard to acheive when teams may be part of different organizations
- **Major components tied to commercial interests**
  - E.g., Docker Hub as single point of failure
- **Federation**
  - Possible via on-prem deployments
  - But AWS and Azure credentials don’t translate
- **Least-common denominator processor instructions**
  - Workarounds require…work
- **Better for certain workloads**
  - We may be at that point, i.e., the long tail of HPC

More Information

Web Sites, Tutorials, Docs

- **Kubernetes:**
  `https://kubernetes.io/`
- **Kubernetes Basics:**
  `https://kubernetes.io/docs/tutorials/kubernetes-basics/`
- **Kubernetes the Hard Way:**
  `https://github.com/kelseyhightower/kubernetes-the-hard-way`
- **The Children's Illustrated Guide to Kubernetes**
  `https://deis.com/blog/2016/kubernetes-illustrated-guide/`
- **Anything by Kelsey Hightower**
  `Presentations, blogs, tutorials, etc.`

Cloud Providers

- **Google Cloud Platform Kubernetes Engine**
  `https://cloud.google.com/kubernetes-engine/`
- **Azure Container Service (AKS)**
  `https://azure.microsoft.com/en-us/services/container-service/`
- **In Preview: Amazon Elastic Container Service for Kubernetes (Amazon EKS)**
  `https://aws.amazon.com/eks/`

The Networking and Information Technology Research and Development (NITRD) Program

Mailing Address: NCO/NITRD, 2415 Eisenhower Avenue, Alexandria, VA 22314

Physical Address: 490 L'Enfant Plaza SW, Suite 8001, Washington, DC 20024, USA Tel: 202-459-9674, Fax: 202-459-9673, Email: nco@nitrd.gov, Website: https://www.nitrd.gov