Kubernetes 101

Doug Davis, STSM

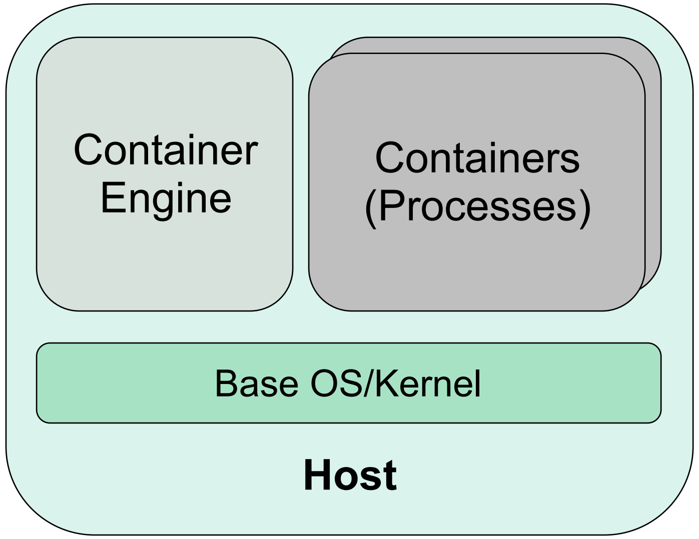
September, 2017

Today's Agenda

- What is Kubernetes?
- How was Kubernetes created?
- Where is the Kubernetes community?
- Technical overview
- What's the current status of Kubernetes?
- What is next for the Kubernetes community?
- Kubernetes at IBM
- Kubernetes in the Enterprise
- Let's Get Started

What are Containers?

- Container: a set of processes run in isolation
- Each container gets its own:
  - PID, User, UTS, Mount Points, Network Stack, etc...
  - And its own view of the filesystem
- Very similar to VMs
  - But process based – run just the app itself, nothing else
  - No operating system, just the Linux kernel files are available
- Benefits:
  - Smaller footprint – just the application's files
  - **Faster start-up times – just starting the exe – not even the OS**
    - **Milliseconds vs minutes**
  - All adds up to better resource utilization at faster scale


Container Engine
Containers (Processes)
Base OS/Kernel
Host

What is Kubernetes?

Enterprise Level Container Orchestration

- Provision, manage, scale applications (containers) across a cluster

- Manage infrastructure resources needed by applications
  - Volumes
  - Networks
  - Secrets
  - And many many many more...

- Declarative model
  - Provide the "desired state" and Kubernetes will make it happen

- What's in a name?
  - Kubernetes (K8s/Kube): "Helmsman" in ancient Greek

How was Kubernetes created?

Google Bringing its Cloud-Scale Expertise to OSS

- Based on Google's internal "Borg" project
- Not just Open Source, but Open Governance!
  - Working very hard to ensure that it is not a "Google" project
  - For example, it was the reason the CNCF was created
  - Pushing for non-Googlers to be in key leadership roles
- Quickly attracted the attention & support of others
  - Looking for alternatives to Docker
  - E.g. RedHat, CoreOS, Deis/EngineYard (now Microsoft), IBM
- One of the fastest growing OSS projects
  - Which has brought many scaling challenges to deal with
  - E.g. had to have "stability" releases to deal with the rapid pace of changes

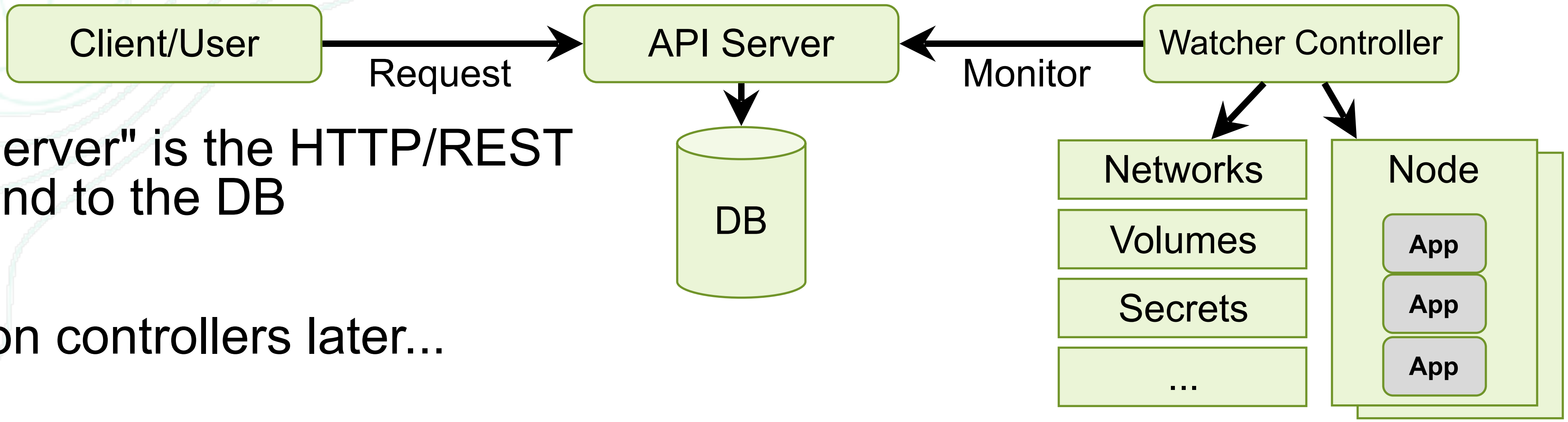
Where is the Kubernetes community?

- Main source entry point: https://github.com/kubernetes/
- Communications
  - https://github.com/kubernetes/community/blob/master/communication.md
- Mailing list / google groups:
  - Devs: kubernetes-dev@googlegroups.com
  - Users: kubernetes-users@googlegroups.com
  - Weekly Community Meeting
  - Conferences (CNCF-con & KubeCon)
- Lots of SIGs (special interest groups) for focused areas/functionality
  - https://github.com/kubernetes/community/blob/master/sig-list.md
  - Each with their own slack channel, mailing lists, regular calls, ...
- Leaders: Google, RedHat, CoreOS, Microsoft (Deis), IBM, Huawei

Kubernetes: Technical Overview

- At its core, Kubernetes is a database (etcd).
  With "watchers" & "controllers" that react to changes in the DB.
  The controllers are what make it Kubernetes.
  This pluggability and extensibility is part of its "secret sauce".

- DB represents the user's desired state.
  Watchers attempt to make reality match the desired state



"API Server" is the HTTP/REST front-end to the DB

More on controllers later...

Kubernetes: Resource Model

A resource for just about any purpose

- Config Maps
- Daemon Sets
- Deployments
- Events
- Endpoints
- Ingress
- Jobs
- Nodes
- Namespaces
- **Pods**
- Persistent Volumes
- Replica Sets
- Secrets
- Service Accounts
- Services
- Stateful Sets, and more...

- Kubernetes aims to have the building blocks on which you build a cloud native platform.

- Therefore, the internal resource model **is** the same as the end user resource model.

Key Resources

- Pod: set of co-located containers
  - Smallest unit of "code" deployment

- Application: undefined, but is a set of pods
  - Several types of resources to help manage them
  - Replica Sets, Deployments, Stateful Sets, ...

- Services & Endpoints
  - Define how to expose your app
  - Query based selector to choose which pods apply

Kubernetes: Technical Overview

- The user directly manipulates resources via json/yaml

```
$ kubectl (create|get|apply|delete) -f myResource.yaml
```

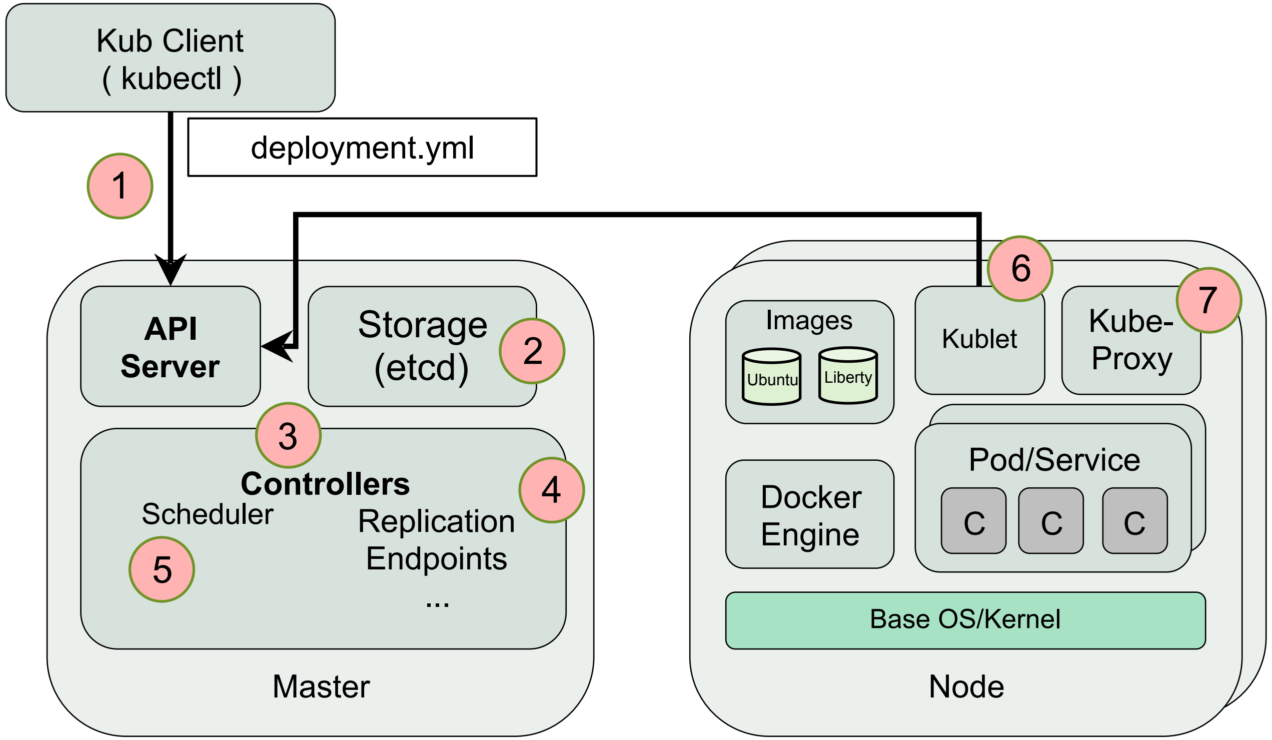
- Some attempts to soften the UX:

```
$ kubectl scale ...
$ kubectl run ...
$ kubectl annotate ...
```

  - But those are limited and not the norm

Kubernetes: Putting it all together...

1. User via "kubectl" deploys a new application
2. API server receives the request and stores it in the DB (etcd)
3. Watchers/controllers detect the resource changes and act upon it
4. ReplicaSet watcher/controller detects the new app and creates new pods to match the desired # of instances
5. Scheduler assigns new pods to a kubelet
6. Kubelet detects pods and deploys them via the container runing (e.g. Docker)
7. Kubeproxy manages network traffic for the pods – including service discovery and load-balancing



What is the current status of Kubernetes?

- V1.8 - celebrated its two year birthday (July 21, 2017) since v1.0
- Some newer features
  - Custom Resource Definitions replaces Third Party Resources
  - Network Policy API is "stable" – control for pod->pod communication
  - Encryption for data at rest in Secrets
  - "Local Storage" persistent volume type was added (alpha)
  - API Aggregation
  - External admission controllers
  - Role Based Access Control (RBAC) is in beta
  - Kubeadm (tool for deploying kube) is in beta
  - Node affinity/anti-affinity, taints, tolerations for scheduling

What's next for the Kubernetes community?

Explosive Growth – a good problem to have

- Kubernetes is going through some growing pains
  - Rapid growth (code & community)
  - Finds itself needing to "slow down" at times to ensure stability
    - Stability releases
    - Lock down amount and rate of changes

- Aside from being a CN platform, K8s is a promoter of interop

• Container Storage Interface (CSI)	w/ Docker, CF
• Container Networking Interface (CNI)	w/ Docker, CF
• Open Service Broker API (SIG-ServiceCatalog)	w/ CF
• Istio	w/ CF

Kubernetes at IBM

Preferred Container Orchestration

- Offerings / Plans
  - Bluemix Kubernetes Service – Docker containers orchestrated by K8s
  - ICp – IBM Cloud Private
  - Watson is leveraging Kubernetes to hosting its infrastructure

- Key Development Activities
  - Service Catalog (co-lead)
  - Contributor Experience
  - Networking & Istio (co-lead)
  - ContainerD integration (co-lead)
  - Storage
  - Performance

Kubernetes in the Enterprise

- Line of business web applications using expected resources:
  - Ingress
  - Load-Balancers
  - Persistent Volumes

- Customer feedback is along the lines of:
  - Kubernetes is a great way to get new app up and running **really** fast!

Kubernetes: Let's Get Started

- Development Guide
  - Getting Started Guide from IBM's Mike Brown
  - Kubernetes Source Code Tour from IBM's Brad Topol
  - Help: Ask on the appropriate SIG Slack channel

- Looking for work:
  - Backlog of issues – many many open issues
  - Contributor experience, testing, "process" related activities
  - Find us on the Slack channel and ask questions!

- Journeys
  - https://developer.ibm.com/code/journey/run-gitlab-kubernetes/
  - https://developer.ibm.com/code/journey/deploy-microprofile-java-microservices-on-kubernetes/
  - https://developer.ibm.com/code/events/manage-microservices-traffic-using-istio/

Future Kubernetes Classes

- Start K8s Journey - K8s concepts and kubectl
- Connect to the world - K8s service catalog
- What makes Kubernetes smart? Scheduling in Kubernetes
- Everything is well connected and located - Kubernetes network
- Stateless vs Stateful? Kubernetes storage
- Keep healthy - Kubernetes logging and monitoring
- Play with applications - Kubernetes Helm and Charts
- What we can do for you - IBM ICp

Thank you!
Q & A

Backup

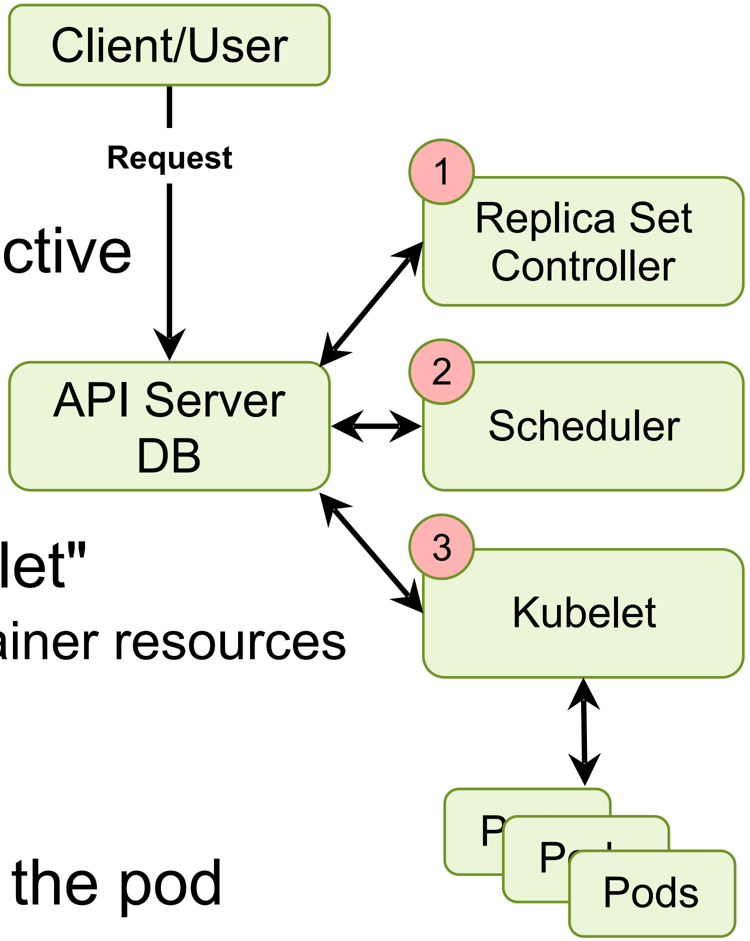
Watchers and Controllers

Some examples:

1. Replica Set Controller
   - Verifies the correct number of "pod" instances are active
     - ReplicaSet is a scalable set of pods

2. Scheduler
   - Watches for new pods and assigns them to a "kubelet"
     - Pod is a group of containers that share lifecycle and container resources

3. Kubelet
   - Watches for pods to be assigned to it, then deploys the pod
     - Kubelet manages the pods/containers running on a host


Client/User
Request
API Server
DB
1
Replica Set
Controller
2
Scheduler
3
Kubelet
P
Pod
Pods