OpenShift Commons Briefing:
Kubernetes 1.9 Features and Future

Derek Carr - Lead Engineer, Kubernetes
What’s new this time around?
RELEASE STATS

- Shorter release (end of year)
- 6000+ pull requests merged across org
- 75,000+ comments
- Focus on fixes, targeted feature enhancements
- ~18 features across 29 SIGs and 5 WG
Focus on Stability
STABILITY IS A FEATURE

● Strong focus on fixing bugs
● Mature existing features to beta or stable
● Production matters
  Refine, polish, scale, tighten
SIG/WG Highlights
APPS
(Workload APIs)

- Workloads API is now apps/v1
  - DaemonSet
  - Deployment
  - ReplicaSet
  - StatefulSet
- Batch Workloads API have separate path to v1
  - Job
  - CronJob
APPs
(MIGRATION TO V1)

● Things to remember when migrating to apps/v1
  ○ Default selectors are deprecated
  ○ Selectors are immutable
  ○ RollingUpdate strategy is default
● Bi-directional auto-conversion is supported with older API versions
API MACHINERY
(ADMISSION CONTROL)

- Mutating and validating webhooks (beta)
- Metrics for monitoring webhook latency
- Support intra and extra cluster webhooks
API MACHINERY
(ADMISSION CONTROL)

- Ecosystem enablement
  - Istio
  - Service Catalog
  - OpenShift

- Webhook configuration stored in server
  - Rules control what operations against what resources are intercepted
  - Failure policy controls what happens if the webhook admission server is unavailable.

- Example
  - https://github.com/openshift/kubernetes-namespaces-namespace-reservation
• Fetch large number of resources in chunks to improve latency (beta)
• Reliability and latency improvements for dense clusters

```
kubectl get --all-namespaces secrets

LIST Secrets (90k)
```

```
kubectl get --all-namespaces --chunk-size=500 secrets

500 500 500
500 ...

LIST Secrets
Results returned in chunks of 500
```
API MACHINERY
(CUSTOM RESOURCES)

- Custom resource definitions support validation
  - Specification allows optional validation
  - OpenAPI v3 schema can be defined in spec
- Custom resource instances are then validated against associated schema in CREATE and UPDATE handlers
API MACHINERY
(CUSTOM RESOURCES)

- Example
  - Spec.version must be “v1.0.0” or “v1.0.1”
  - Spec.replicas must be between 1 and 10

```yaml
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata: ...
spec: ...
  validation:
    openAPIV3Schema:
      properties:
        spec:
          properties:
            version:
              type: string
              enum:
                - "v1.0.0"
                - "v1.0.1"
            replicas:
              type: integer
              minimum: 1
              maximum: 10
```
API MACHINERY
(CUSTOM RESOURCES)

apiVersion: mygroup.example.com/v1
kind: App
metadata:
  name: example-app
spec:
  version: "v1.0.2"
  replicas: 15

$ kubectl create -f app.yaml
The App "example-app" is invalid: []: Invalid value:
map[string]interface {}{"apiVersion":"mygroup.example.com/v1",
"kind":"App", "metadata":map[string]interface 
{}{"creationTimestamp":"2017-08-31T20:52:54Z",
"uid":"5c67f651-9e8e-11e7-96ad-f0761cb232d1", "selfLink":"
, "clusterName":", "name":"example-app", "namespace":"default",
"deletionTimestamp":interface {}(nil),
"deletionGracePeriodSeconds":(*int64)(nil)},
"spec":map[string]interface {}{"replicas":15,
"version":"v1.0.2"}};
validation failure list:
spec.replicas in body should be less than or equal to 10
spec.version in body should be one of [v1.0.0 v1.0.1]
AUTH

- Audit
  - Audit events provide better timestamps: RequestReceived and StageTimestamp

- RBAC
  - Aggregated cluster roles union the rules of matching ClusterRoles by label
  - Useful for integrating default cluster roles with custom resource definitions

```yaml
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: monitoring
aggregationRule:
  clusterRoleSelectors:
    - matchLabels:
        rbac.example.com/aggregate-to-monitoring: "true"
rules: [] # Rules are automatically filled in by the controller manager.
```
CLI

- Added support for field selectors!
  - Find all pods scheduled to node1 in the namespace
    - `kubectl get pods --field-selector=spec.nodeName=node1`
  - Find all pods running in the namespace
    - `kubectl get pods --field-selector=status.phase=Running`
  - Find all pods not running across all namespaces
    - `kubectl get pods --field-selector=status.phase!=Running --all-namespaces`
  - Find all events sourced from node-controller
    - `kubectl get events --field-selector=source=node-controller --all-namespaces`
CLOUD PROVIDERS

- AWS
  - Nodes can now use instance types that use NVMe volumes (i.e. C5 types)
  - Nodes are tainted if volumes are stuck attaching
    - Operators are encouraged to monitor and remedy as appropriate

- Azure
  - Improvements to Azure Load Balancer implementation
  - Stability improvements in cloud provider

- OpenStack
  - Block storage (Cinder) V3 is supported
  - Load Balancer (Octavia) V2 is supported, in addition to Neutron LBaaS V2
  - Neutron LBaaS V1 support removed
NETWORKING

- Support for ipv6 (Alpha)
- IPVS mode for kube-proxy (Beta)
  - Available for evaluation
  - Potential Benefits
    - Performance improvement in dense clusters (hashing vs chains)
    - More load balancing algorithms (least-load, least connections, etc.)
    - Health checks and connection retries
Numerous performance and reliability improvements

Container Runtime Ecosystem
- cri-o is stable, minikube integration, try it out!
- Other runtimes evolve (containerd beta, frakti stable, rktlet alpha)
- Debugging tools for cri implementations (cri-tools)

Resource Management
- Device plugin reliability improvements (accelerators)
- Static CPU pinning works across kubelet restarts (latency workloads)
- HugePages no longer tied to QoS

Metrics improvements
- Support for accelerator stats (make, model, memory total, memory used, duty cycle)
- Ephemeral pod storage (how much local storage is used by a pod)
- Pod level usage stats (previously just container only)
RESOURCE MANAGEMENT (QUOTA)

- Improvements to quota
  - Object count quota on all standard namespaced resources (count/resource.group=?)
  - Ability to quota hugepages
- Examples
  - `kubectl create quota object-counts --hard=count/pods=10,count/jobs.batch=10`
SCHEDULING

- Improvements in pod priority and preemption
  - Handles pod disruption budget
  - Integrated with kubelet eviction logic (usage > requests, priority, usage-requests)
- New priority function to prefer nodes that satisfy resource limits (alpha)
  - Useful tie-breaker to optimize scheduler prefers nodes for max burst of resources
STORAGE

- Container Storage Interface (CSI) implementation (Alpha)
  - Effort across multiple storage orchestrators
  - Enable new volume plugins outside of Kubernetes core
  - Enables volume plugins to support containerized deployment in future
- Raw block device support (Alpha)
  - Fibre channel implementation available
- Improvements to volume resizing (Alpha)
  - Supports GCE PD, Ceph RBD, AWS EBS, OpenStack Cinder
WINDOWS

- kubelet and kube-proxy support on Windows Server 2016+ (beta)
- Control plane components still run on Linux only
- Notable improvements
  - Shared network namespace
  - Reduced network complexity with single endpoint per pod
  - Kernel based load-balancing with Virtual Filtering Platform (VFP) analogous to iptables
  - CRI pod and node level usage stats integration
- Evaluate usage and provide feedback to community
Kubernetes 1.10
(SOME) 1.10 GOALS

- Stability and bug fixes (obv!)
- Everything is extensible
- Scaling improvements
- Descheduler
- Priority and preemption to beta
- Device plugins to beta
- CPU pinning to beta
- Hugepages to beta
- Get volume snapshots and resizing to beta
- Metrics used in the scheduler
- Better Prometheus integration into metrics
- Broader block device support
Questions?

1.10 is underway!

Derek Carr - @derekwaynecarr

For upcoming briefings & events, visit: https://commons.openshift.org/events.html