


KubeCon | CloudNativeCon

Europe 2020

Virtual

ComponentConfig: Technical Challenges

Mike Taufen (Google) & Alexander Knipping (Exaring AG)

What's WG-Component-Standard?

Develop a standard foundation (philosophy and libraries) for core Kubernetes components to build on top of.

Areas Include:

- Configuration (flags, ComponentConfig APIs, ...)
- Status Endpoints (healthz, configz, ...)
- Integration Points (delegated authn/z, ...)
- Logging / Metrics

Details in KEP 0032:
kubernetes/enhancements/keps/sig-cluster-lifecycle/wgs/0032-create-a-k8s-io-component-repo.md

Agenda

1. Introduction
2. Strict Decoding
3. Kube Controller Manager Component Config
4. Other work
5. Q&A

About me

- Alexander Knipping: Site Reliability Engineer @Exaring AG in Munich
- With wg-component-standard since **Mid 2019**
- Little prior open source experience

First Challenge

Problem: YAML files can have typos in field names, or duplicate fields. These errors may be silently ignored.

Implement CodecFactoryOptions allowing clients to opt-in to Pretty encoders and Strict Decoders #76805

Open with

Merged **k8s-ci-robot** merged 6 commits into `kubernetes:master` from `stealthybox:codecfactory-options` on 6 Aug 2019

Conversation 48 | Commits 6 | Checks 0 | Files changed 2 | +103 −30

stealthybox commented on 19 Apr 2019 • edited — Member

/kind api-change
@kubernetes/wg-component-standard

What this PR does / why we need it:
This patch introduces a new CodecFactory constructor that's capable of building different collections of Serializers.
Note that this diverges from the current design where an initialized CodecFactory is a kitchen sink holding many forms of useful Serilaizers in a non-customizeable way.

Reviewers

 neolit123

 logicalhan

 sttts

 liggitt

 luxas

 stewart-yu

First Challenge

Problem: YAML files can have typos in field names, or duplicate fields. These errors may be silently ignored.

Solution: Throw an error for unrecognized field names.

No problem! 

Loading YAML files

ComponentConfig?

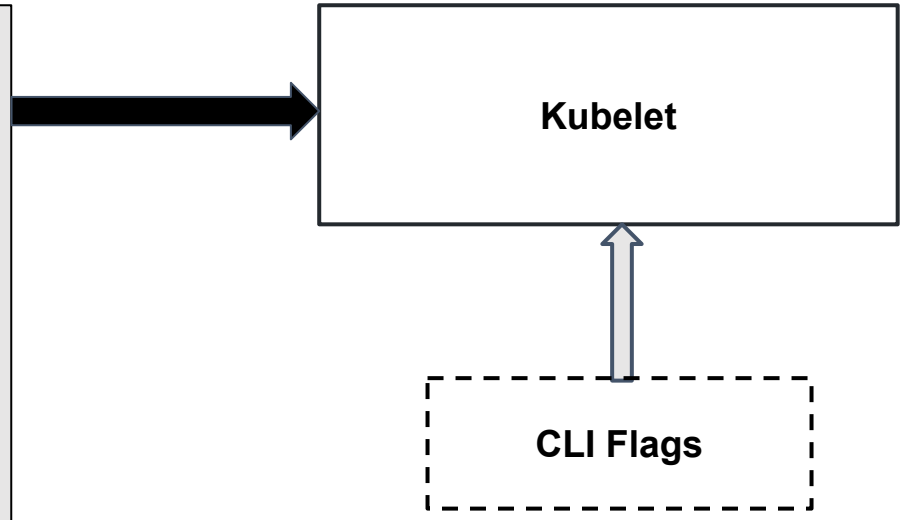
What's a Codec(Factory)?

Let's start at the beginning ...

Loading YAML Files

Node

```
kind: KubeletConfiguration
apiVersion: kubelet.config.k8s.io/v1beta1
authentication:
  anonymous:
    enabled: false
  webhook:
    enabled: true
  x509:
    clientCAFile: "/var/lib/kubernetes/ca.pem"
authorization:
  mode: Webhook
clusterDomain: "cluster.local"
clusterDNS:
  - "10.32.0.10"
```



Loading YAML Files

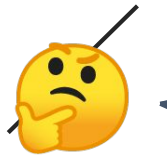

```
kind: KubeletConfiguration
apiVersion: kubelet.config.k8s.io/v1beta1
authentication:
  anonymous:
    enabled: false
  webhook:
    enabled: true
  x509:
    clientCAFile: "/var/lib/kubernetes/ca.pem"
authorization:
  mode: Webhook
clusterDomain: "cluster.local"
clusterDomain: "mycluster.local"
clusterDNS:
  - "10.32.0.10"
```

```
kind: KubeletConfiguration
apiVersion: kubelet.config.k8s.io/v1beta1
authentication:
  anonymous:
    enabled: false
  webhook:
    enabled: true
  x509:
    clientCAFile: "/var/lib/kubernetes/ca.pem"
authorization:
  mode: Webhook
clusterDomain: "cluster.local"
clusterDNS:
  - "10.32.0.10"
foo: "bar"
```

Loading YAML Files

configfiles.go

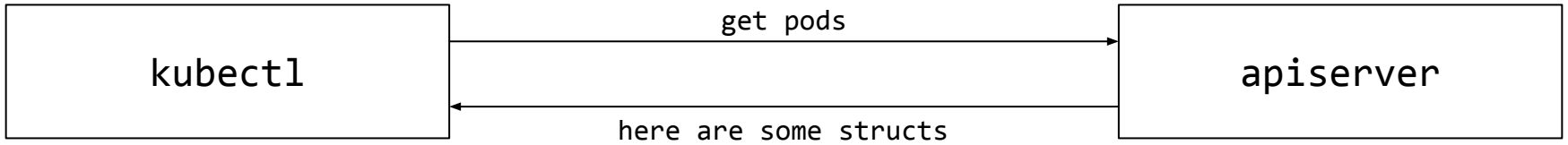
```go
// NewFsLoader returns a Loader that loads a KubeletConfiguration from the `kubeletFile`
func NewFsLoader(fs utilfs.Filesystem, kubeletFile string) (Loader, error) {
        _, kubeletCodecs, err := kubeletscheme.NewSchemeAndCodecs()
        if err != nil {
                return nil, err
        }

        return &fsLoader{
                fs:            fs,
                kubeletCodecs: kubeletCodecs,
                kubeletFile:   kubeletFile,
        }, nil
}
```



scheme.go

```go
// NewSchemeAndCodecs is a utility function that returns a Scheme and CodecFactory
// that understand the types in the kubeletconfig API group.
func NewSchemeAndCodecs() (*runtime.Scheme, *serializer.CodecFactory, error) {
        scheme := runtime.NewScheme()
        if err := kubeletconfig.AddToScheme(scheme); err != nil {
                return nil, nil, err
        }
        if err := kubeletconfigv1beta1.AddToScheme(scheme); err != nil {
                return nil, nil, err
        }
        codecs := serializer.NewCodecFactory(scheme)
        return scheme, &codecs, nil
}
```

Scheme & Versioning



- **Mapping** between *Object* (Golang struct of a Pod, for example) to a specific Go type (GVK - *GroupVersionKind*)
- *"Foundation of a versioned API and configuration over time"*

```
func (s *Scheme) ObjectKinds(obj Object) ([]schema.GroupVersionKind, bool, error)
```

```
scheme.AddKnownTypes(schema.GroupVersionKind{"", "v1", "Pod"}, &Pod{})
```

Scheme & Versioning

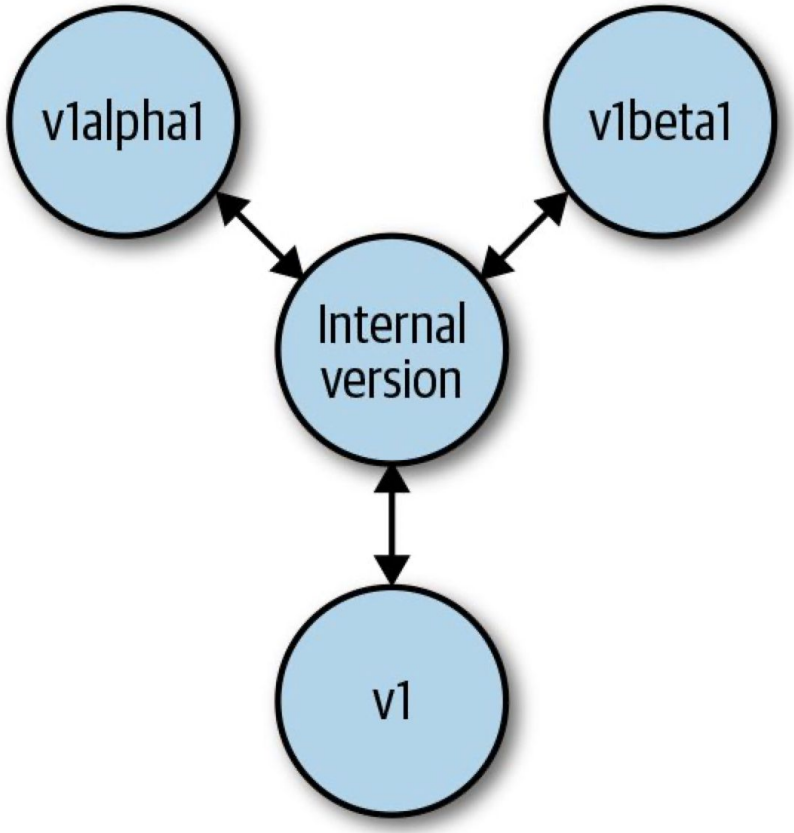
configfiles.go

```go
// NewFsLoader returns a Loader that loads a KubeletConfiguration from the `kubeletFile`
func NewFsLoader(fs utilfs.Filesystem, kubeletFile string) (Loader, error) {
        _, kubeletCodecs, err := kubeletscheme.NewSchemeAndCodecs()
        if err != nil {
                return nil, err
        }

        return &fsLoader{
                fs:            fs,
                kubeletCodecs: kubeletCodecs,
                kubeletFile:   kubeletFile,
        }, nil
}
```

scheme.go

```go
// NewSchemeAndCodecs is a utility function that returns a Scheme and CodecFactory
// that understand the types in the kubeletconfig API group.
func NewSchemeAndCodecs() (*runtime.Scheme, *serializer.CodecFactory, error) {
        scheme := runtime.NewScheme()
        if err := kubeletconfig.AddToScheme(scheme); err != nil {
                return nil, nil, err
        }
        if err := kubeletconfigv1beta1.AddToScheme(scheme); err != nil {
                return nil, nil, err
        }
        codecs := serializer.NewCodecFactory(scheme)
        return scheme, &codecs, nil
}
```

Scheme & Versioning



M Hausenblas, S Schimanski (2019): Programming Kubernetes

Scheme & Versioning

configfiles.go

```go
// NewFsLoader returns a Loader that loads a KubeletConfiguration from the `kubeletFile`
func NewFsLoader(fs utilfs.Filesystem, kubeletFile string) (Loader, error) {
        _, kubeletCodecs, err := kubeletscheme.NewSchemeAndCodecs()
        if err != nil {
                return nil, err
        }

        return &fsLoader{
                fs:            fs,
                kubeletCodecs: kubeletCodecs,
                kubeletFile:   kubeletFile,
        }, nil
}
```

scheme.go

```go
// NewSchemeAndCodecs is a utility function that returns a Scheme and CodecFactory
// that understand the types in the kubeletconfig API group.
func NewSchemeAndCodecs() (*runtime.Scheme, *serializer.CodecFactory, error) {
    ✔ scheme := runtime.NewScheme()
    ✔ if err := kubeletconfig.AddToScheme(scheme); err != nil {
            return nil, nil, err
        }
    ✔ if err := kubeletconfigv1beta1.AddToScheme(scheme); err != nil {
            return nil, nil, err
        }
        codecs := serializer.NewCodecFactory(scheme)
        return scheme, &codecs, nil
}
```

Strict Decoding


```
108  // CodecFactory provides methods for retrieving codecs and serializers for specific
109  // versions and content types.
110  type CodecFactory struct {
111          scheme      *runtime.Scheme
112          serializers []serializerType
113          universal   runtime.Decoder
114          accepts     []runtime.SerializerInfo
115
116          legacySerializer runtime.Serializer
117  }
118
119  // CodecFactoryOptions holds the options for configuring CodecFactory behavior
120  type CodecFactoryOptions struct {
121          // Strict configures all serializers in strict mode
122      ➡️  Strict bool
123          // Pretty includes a pretty serializer along with the non-pretty one
124          Pretty bool
125  }
```

[codec_factory.go]

Strict Decoding


```
28  // NewSchemeAndCodecs is a utility function that returns a Scheme and CodecFactory
29  // that understand the types in the kubeletconfig API group.
30  func NewSchemeAndCodecs() (*runtime.Scheme, *serializer.CodecFactory, error) {
31      ✔️ scheme := runtime.NewScheme()
32      ✔️ if err := kubeletconfig.AddToScheme(scheme); err != nil {
33              return nil, nil, err
34          }
35      ✔️ if err := kubeletconfigv1beta1.AddToScheme(scheme); err != nil {
36              return nil, nil, err
37          }
38      ✔️ codecs := serializer.NewCodecFactory(scheme)
39          return scheme, &codecs, nil                    ⬆️
40  }
```

So we just call NewCodecFactory with the Strict option enabled?

Easy!

But wait...

Strict Decoding

KubeCon CloudNativeCon Europe 2020 Virtual

```
kind: KubeletConfiguration
apiVersion: kubelet.config.k8s.io/v1beta1
authentication:
  anonymous:
    enabled: false
  webhook:
    enabled: true
  x509:
    clientCAFile: "/var/lib/kubernetes/ca.pem"
authorization:
  mode: Webhook
clusterDomain: "cluster.local"
clusterDomain: "foo"
clusterDNS:
  - "10.32.0.10"
```

```
kind: KubeletConfiguration
apiVersion: kubelet.config.k8s.io/v1beta1
authentication:
  anonymous:
    enabled: false
  webhook:
    enabled: true
  x509:
    clientCAFile: "/var/lib/kubernetes/ca.pem"
authorization:
  mode: Webhook
clusterDomain: "cluster.local"
clusterDNS:
  - "10.32.0.10"
foo: "bar"
```

Strict Decoding

- Only strengthen validation on version-boundary

→ have KubeletConfig/**v1beta1** decode with warnings, show errors on **v1beta2+**

```
+ // NewLenientSchemeAndCodecs constructs a CodecFactory with strict decoding
+ // disabled, that has only the Schemes registered into it which are passed
+ // and added via AddToScheme functions. This can be used to skip strict decoding
+ // a specific version only.
+ func NewLenientSchemeAndCodecs(addToSchemeFns ...func(s *runtime.Scheme) error) (*runtime.Scheme, *serializer.CodecFactory, error) {
+       lenientScheme := runtime.NewScheme()
+       for _, s := range addToSchemeFns {
+               if err := s(lenientScheme); err != nil {
+                       return nil, nil, fmt.Errorf("unable to add API to lenient scheme: %v", err)
+               }
+       }
+       lenientCodecs := serializer.NewCodecFactory(lenientScheme, serializer.DisableStrict)
+       return lenientScheme, &lenientCodecs, nil
+ }
```

Strict Decoding

- Right now we introduced strict decoding to:
  - kubelet
  - kube-proxy
  - kube-scheduler

Enable strict serializers in components by default #82924

Closed **obitech** opened this issue on 20 Sep 2019 · 4 comments

obitech commented on 20 Sep 2019 • edited — Member

What would you like to be added:
With #76805 merged we should enable strict serializers in Kubernetes components:

- [x] kube-proxy (#82927)
- [x] kube-scheduler (#83030)
- ~~kube-controller-manager~~ (not using a config file right now, see enhancements#786 and enhancements#1052)
- [x] kubelet (#83204)

Lenient path implemented:

- [x] kube-proxy (#84143)
- [x] kube-scheduler (#84129)
- [x] kubelet (#83204)

With strict serializers enabled, an error will be thrown on things such as duplicate or unknown fields when parsing a YAML file.

Why is this needed:
This will improve code quality of user's configuration files as it enforces a proper configuration file to be present when starting a component.

👍 2

Kube Controller Manager Config

What about a KubeControllerManager ComponentConfig?

Kube Controller Manager Config

kube-controller-manager

Synopsis

The Kubernetes controller manager is a daemon that embeds the core control loops shipped with Kubernetes. In applications of robotics and automation, a control loop is a non-terminating loop that regulates the state of the system. In Kubernetes, a controller is a control loop that watches the shared state of the cluster through the apiserver and makes changes attempting to move the current state towards the desired state. Examples of controllers that ship with Kubernetes today are the replication controller, endpoints controller, namespace controller, and serviceaccounts controller.

- Controller Manager is a "binary host" → starts other controllers
- Individual controllers are interchangable

github.com/kubernetes/kubernetes/blob/master/cmd/kube-controller-manager/app/controllermanager.go

```go
// NewControllerInitializers is a public map of named controller groups (you can start more than one in an init func)
// paired to their InitFunc.  This allows for structured downstream composition and subdivision.
func NewControllerInitializers(loopMode ControllerLoopMode) map[string]InitFunc {
	controllers := map[string]InitFunc{}
	controllers["endpoint"] = startEndpointController
	controllers["endpointslice"] = startEndpointSliceController
	controllers["replicationcontroller"] = startReplicationController
	controllers["podgc"] = startPodGCController
	controllers["resourcequota"] = startResourceQuotaController
	controllers["namespace"] = startNamespaceController
	controllers["serviceaccount"] = startServiceAccountController
	controllers["garbagecollector"] = startGarbageCollectorController
	controllers["daemonset"] = startDaemonSetController
	controllers["job"] = startJobController
	controllers["deployment"] = startDeploymentController
	controllers["replicaset"] = startReplicaSetController
	controllers["horizontalpodautoscaling"] = startHPAController
	controllers["disruption"] = startDisruptionController
	controllers["statefulset"] = startStatefulSetController
	controllers["cronjob"] = startCronJobController
	controllers["csrsigning"] = startCSRSigningController
	controllers["csrapproving"] = startCSRApprovingController
	controllers["csrcleaner"] = startCSRCleanerController
	controllers["ttl"] = startTTLController
	controllers["bootstrapsigner"] = startBootstrapSignerController
	controllers["tokencleaner"] = startTokenCleanerController
	controllers["nodeipam"] = startNodeIpamController
	controllers["nodelifecycle"] = startNodeLifecycleController
	if loopMode == IncludeCloudLoops {
		controllers["service"] = startServiceController
		controllers["route"] = startRouteController
		controllers["cloud-node-lifecycle"] = startCloudNodeLifecycleController
		// TODO: volume controller into the IncludeCloudLoops only set.
	}
	controllers["persistentvolume-binder"] = startPersistentVolumeBinderController
	controllers["attachdetach"] = startAttachDetachController
	controllers["persistentvolume-expander"] = startVolumeExpandController
	controllers["clusterrole-aggregation"] = startClusterRoleAggregrationController
	controllers["pvc-protection"] = startPVCProtectionController
	controllers["pv-protection"] = startPVProtectionController
	controllers["ttl-after-finished"] = startTTLAfterFinishedController
	controllers["root-ca-cert-publisher"] = startRootCACertPublisher

	return controllers
}
```

controllermanager.go

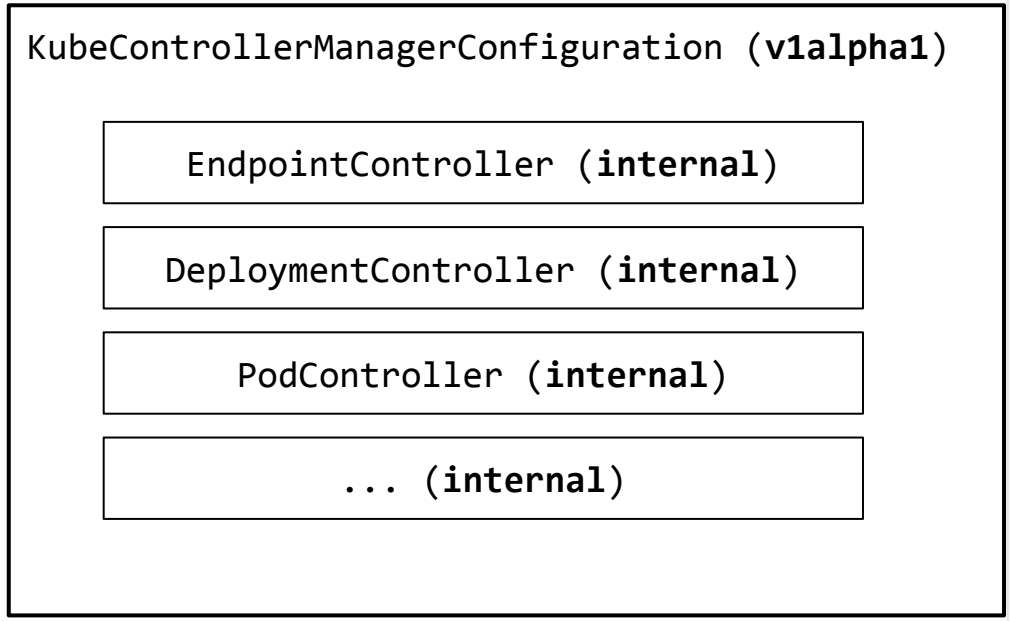
Kube Controller Manager Config

github.com/kubernetes/kubernetes/blob/master/pkg/controller/apis/config/types.go

```
47  // KubeControllerManagerConfiguration contains elements describing kube-controller manager.
48  type KubeControllerManagerConfiguration struct {
49      metav1.TypeMeta
50
51      // Generic holds configuration for a generic controller-manager
52      Generic GenericControllerManagerConfiguration
53      // KubeCloudSharedConfiguration holds configuration for shared related features
54      // both in cloud controller manager and kube-controller manager.
55      KubeCloudShared KubeCloudSharedConfiguration
56
57      // AttachDetachControllerConfiguration holds configuration for
58      // AttachDetachController related features.
59      AttachDetachController attachdetachconfig.AttachDetachControllerConfiguration
```

controller/apis/config/types.go

Kube Controller Manager Config

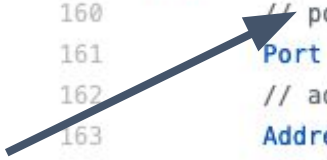


github.com/kubernetes/kubernetes/blob/master/pkg/controller/apis/config/types.go

```
16
17  package config
18
19  import (
20          metav1 "k8s.io/apimachinery/pkg/apis/meta/v1"
21          componentbaseconfig "k8s.io/component-base/config"
22          csrsigningconfig "k8s.io/kubernetes/pkg/controller/certificates/signer/config"
23          daemonconfig "k8s.io/kubernetes/pkg/controller/daemon/config"
24          deploymentconfig "k8s.io/kubernetes/pkg/controller/deployment/config"
25          endpointconfig "k8s.io/kubernetes/pkg/controller/endpoint/config"
26          endpointsliceconfig "k8s.io/kubernetes/pkg/controller/endpointslice/config"
27          garbagecollectorconfig "k8s.io/kubernetes/pkg/controller/garbagecollector/config"
28          jobconfig "k8s.io/kubernetes/pkg/controller/job/config"
29          namespaceconfig "k8s.io/kubernetes/pkg/controller/namespace/config"
30          nodeipamconfig "k8s.io/kubernetes/pkg/controller/nodeipam/config"
31          nodelifecycleconfig "k8s.io/kubernetes/pkg/controller/nodelifecycle/config"
32          poautosclerconfig "k8s.io/kubernetes/pkg/controller/podautoscaler/config"
33          podgcconfig "k8s.io/kubernetes/pkg/controller/podgc/config"
34          replicasetconfig "k8s.io/kubernetes/pkg/controller/replicaset/config"
35          replicationconfig "k8s.io/kubernetes/pkg/controller/replication/config"
36          resourcequotaconfig "k8s.io/kubernetes/pkg/controller/resourcequota/config"
37          serviceconfig "k8s.io/kubernetes/pkg/controller/service/config"
38          serviceaccountconfig "k8s.io/kubernetes/pkg/controller/serviceaccount/config"
39          statefulsetconfig "k8s.io/kubernetes/pkg/controller/statefulset/config"
40          ttlafterfinishedconfig "k8s.io/kubernetes/pkg/controller/ttlafterfinished/config"
41          attachdetachconfig "k8s.io/kubernetes/pkg/controller/volume/attachdetach/config"
42          persistentvolumeconfig "k8s.io/kubernetes/pkg/controller/volume/persistentvolume/config"
43  )
```

[controller/apis/config/types.go]

Kube Controller Manager Config



```
158  // GenericControllerManagerConfiguration holds configuration for a generic controller-manager.
159  type GenericControllerManagerConfiguration struct {
160          // port is the port that the controller-manager's http service runs on.
161          Port int32
162          // address is the IP address to serve on (set to 0.0.0.0 for all interfaces).
163          Address string
164          // minResyncPeriod is the resync period in reflectors; will be random between
165          // minResyncPeriod and 2*minResyncPeriod.
166          MinResyncPeriod metav1.Duration
167          // ClientConnection specifies the kubeconfig file and client connection
168          // settings for the proxy server to use when communicating with the apiserver.
169          ClientConnection componentbaseconfigv1alpha1.ClientConnectionConfiguration
170          // How long to wait between starting controller managers
171          ControllerStartInterval metav1.Duration
172          // leaderElection defines the configuration of leader election client.
173          LeaderElection componentbaseconfigv1alpha1.LeaderElectionConfiguration
174          // Controllers is the list of controllers to enable or disable
175          // '*' means "all enabled by default controllers"
176          // 'foo' means "enable 'foo'"
177          // '-foo' means "disable 'foo'"
178          // first item for a particular name wins
179          Controllers []string
180          // DebuggingConfiguration holds configuration for Debugging related features.
181          Debugging componentbaseconfigv1alpha1.DebuggingConfiguration
182  }
```

[kube-controller-manager/config/v1alpha1/types.go]

Kube Controller Manager Config


```
kind: KubeControllerManagerConfiguration
apiVersion: kubecontrollermanager.config.k8s.io/v1alpha1
Generic:
  Port: 1234
  Address: 0.0.0.0
  ClientConnection:
    burst: 10
ResourceQuotaController:
  ConcurrentResourceQuotaSyncs: 15
# ...
```

Kube Controller Manager Config

Why is it so difficult to implement a ComponentConfig for KubeControllerManager?

Kube Controller Manager Config

What happens if...

```
kind: KubeControllerManagerConfiguration
apiVersion: kubecontrollermanager.config.k8s.io/v1alpha1
Generic:
  Port: 1234
  Address: 0.0.0.0
  ClientConnection:
    burst: 10
ResourceQuotaController:
  ConcurrentResourceQuotaSyncs: 15
# …
```

… a controller moves out-of-tree (like cloud-controller)?

… individual controllers share config fields ?

… the owner of a controller doesn't want (or can't) follow the container versioning?

… users implement their own controller manager?

Kube Controller Manager Config


```
kind: KubeControllerManagerConfiguration
apiVersion: kubecontrollermanager.config.k8s.io/v1alpha1
controllers:
  resourceQuota:
    kind: ResourceQuotaControllerConfiguration
    apiVersion: kubecontrollermanager.config.k8s.io/v1alpha2
    ConcurrentResourceQuotaSyncs: 15
# ...
```

github.com/kubernetes/kubernetes/blob/master/cmd/kube-controller-manager/app/controllermanager.go

```
384  // NewControllerInitializers is a public map of named controller groups (you can start more than one in an init func)
385  // paired to their InitFunc.  This allows for structured downstream composition and subdivision.
386  func NewControllerInitializers(loopMode ControllerLoopMode) map[string]InitFunc {
387          controllers := map[string]InitFunc{}
388          controllers["endpoint"] = startEndpointController
389          controllers["endpointslice"] = startEndpointSliceController
390          controllers["replicationcontroller"] = startReplicationController
391          controllers["podgc"] = startPodGCController
392          controllers["resourcequota"] = startResourceQuotaController
393          controllers["namespace"] = startNamespaceController
394          controllers["serviceaccount"] = startServiceAccountController
395          controllers["garbagecollector"] = startGarbageCollectorController
396          controllers["daemonset"] = startDaemonSetController
397          controllers["job"] = startJobController
398          controllers["deployment"] = startDeploymentController
399          controllers["replicaset"] = startReplicaSetController
400          controllers["horizontalpodautoscaling"] = startHPAController
401          controllers["disruption"] = startDisruptionController
402          controllers["statefulset"] = startStatefulSetController
403          controllers["cronjob"] = startCronJobController
404          controllers["csrsigning"] = startCSRSigningController
405          controllers["csrapproving"] = startCSRApprovingController
406          controllers["csrcleaner"] = startCSRCleanerController
407          controllers["ttl"] = startTTLController
408          controllers["bootstrapsigner"] = startBootstrapSignerController
409          controllers["tokencleaner"] = startTokenCleanerController
410          controllers["nodeipam"] = startNodeIpamController
411          controllers["nodelifecycle"] = startNodeLifecycleController
412          if loopMode == IncludeCloudLoops {
413                  controllers["service"] = startServiceController
414                  controllers["route"] = startRouteController
415                  controllers["cloud-node-lifecycle"] = startCloudNodeLifecycleController
416                  // TODO: volume controller into the IncludeCloudLoops only set.
417          }
418          controllers["persistentvolume-binder"] = startPersistentVolumeBinderController
419          controllers["attachdetach"] = startAttachDetachController
420          controllers["persistentvolume-expander"] = startVolumeExpandController
421          controllers["clusterrole-aggregation"] = startClusterRoleAggregrationController
422          controllers["pvc-protection"] = startPVCProtectionController
423          controllers["pv-protection"] = startPVProtectionController
424          controllers["ttl-after-finished"] = startTTLAfterFinishedController
425          controllers["root-ca-cert-publisher"] = startRootCACertPublisher
426
427          return controllers
428  }
429
```

[controllermanager.go]

Kube Controller Manager Config

KubeCon CloudNativeCon Europe 2020 Virtual

kubernetes/pull/70359

Add support for configuring kube-controller manager via a config file instead of command line flags. #70359

Open with

Closed **stewart-yu** wants to merge 3 commits into `kubernetes:master` from `stewart-yu:stewart-kube-controller-manager-config-flag`

Conversation 9 · Commits 3 · Checks 0 · Files changed 8 · +394 −4

stewart-yu commented on 29 Oct 2018 • edited · Member

What type of PR is this?
/kind feature

What this PR does / why we need it:
Now that we make `kube-controller manager` api configuration in v1.12 #67469, we should also add support for configuring kube-controller manager via a config file instead of command line flags.
Like `kube-proxy` and `kube-scheduler`

Which issue(s) this PR fixes *(optional, ;*
in `fixes #<issue number>(, fixes #<issue_number>, ...)` format, will close the issue(s) when PR gets merged):
Fixes #

Special notes for your reviewer:

Does this PR introduce a user-facing change?:

```
NONE
```

Reviewers
- cheftako
- j3ffml
- wojtek-t

Assignees
- sttts
- luxas

Labels
cncf-cla: yes · kind/api-change · kind/feature · lifecycle/rotten · needs-priority · release-note-none · sig/api-machinery · sig/apps · sig/auth · sig/cloud-provider · size/L

Kube Controller Manager Config

KubeCon CloudNativeCon Europe 2020 Virtual

[enhancements/issues/786]

[design doc by @luxas]

kube-controller-manager ComponentConfig #786

Open **luxas** opened this issue on 30 Jan 2019 · 24 comments

luxas commented on 30 Jan 2019 • edited — Member

Enhancement Description

- One-line enhancement description (can be used as a release note): Usage of the kube-controller-manager configuration file is experimental, as the API version now is v1alpha1
- Primary contact (assignee): **@luxas**
- Responsible SIGs: **@kubernetes/sig-api-machinery-api-reviews @kubernetes/wg-component-standard**
- Design proposal link (community repo): N/A
- Link to e2e and/or unit tests:
- Reviewer(s) - (for LGTM) recommend having 2+ reviewers (at least one from code-area OWNERS file) agreed to review. Reviewers from multiple companies preferred: **@liggitt @deads2k**
- Approver (likely from SIG/area to which enhancement belongs): **@liggitt @deads2k**
- Enhancement target (which target equals to which milestone):
  - Alpha release target (x.y) v1.14
  - Beta release target (x.y) v1.15
  - Stable release target (x.y) v1.16

The kube-controller-manager ComponentConfig is currently in v1alpha1 and unserializable. It needs to become serializable, and then the spec needs to be graduated to v1beta1 and beyond in order to be usable widely.
/assign **@liggitt @deads2k**

Controller Manager ComponentConfig

WARNING: this totally WIP. Don't expect logical text flow or proper sentences. It's a sketch for now.

Proposal

We propose to introduce a controller manager component config container kind with

- manager-global configuration
- shared controller configuration (which serves as a default if not specified differently by controllers
- one config object per controller.

Kube Controller Manager Config

KubeCon CloudNativeCon Europe 2020 Virtual

[enhancements/pull/1052]

Add KEP for controller manager config #1052

Open with

Closed **stewart-yu** wants to merge 1 commit into `kubernetes:master` from `stewart-yu:master`

Conversation 21 | Commits 1 | Checks 0 | Files changed 1 | +537 −0

stewart-yu commented on 8 May 2019 • edited — Member

controller manager doesn't consume a config file currently, we aim to add controller manager config file support. In this way, we can configure controller manager via a config file instead of command line flags.

part of #786

Reviewers

- lavalamp
- sttts
- LalatenduMohanty
- deads2k

Kube Controller Manager Config

Current plan:

write a KEP to restart the discussion ✍️

Other Projects

- ComponentConfig
  - Flags to Config Migrations
  - Instance Specific Config
  - API Documentation
  - Testing Config APIs
- Command building
  - Standard way to deal with flags that works well for migration to config
  - Standardized logging
- Component servers
  - Standardized endpoints
  - Standardized authn/z implementations

Contributor shout outs

KubeCon CloudNativeCon Europe 2020 Virtual

@RainbowMango
@dims
@logicalhan
@liggitt
@serathius
@draveness
@tahsinrahman
@jpbetz
@yuzhiquan
@yastij
@rahulchheda
@pohly
@jennybuckley
@wenjiaswe
@tanjunchen
@sambdavidson
@ricky1993
@pjbgf
@odinuge
@ingvagabund
@deads2k
@conwaychriscosmo
@codenrhoden
@cjcullen
@cblecker
@carlosedp
@SataQiu
@apelisse
@danielqsj
@zhouya0
@yue9944882
@voor
@tapih
@tallclair
@saad-ali
@roycaihw
@prameshj
@obitech
@mrbobbytables
@iobuf
@gongguan
@gnufied
@giuseppe
@farah
@enj
@dprotaso
@dashpole
@borgerli
@aojea
@andyzhangx
@ahg-g
@Zhuzhenghao
@NicolasT
@andrewsykim
@savitharaghunathan
@liggitt
@phenixblue
@knabben
@vincent178
@irbull
@wojtek-t
@marosset
@BenTheElder
@afrouzMashaykhi
@stealthybox
@mtaufen
@sttts

Individuals who have merged code related to WG projects, including contributions to various parts of k/k and k/component-base, in the past year.

How you can get involved

Weekly meeting: Tuesdays 8:30am-9:00am PT
Weekly office hours: Tuesdays 10:00am-11:00am PT

Mailing list:
kubernetes-wg-component-standard@googlegroups.com
Join for meeting invites!

GitHub:
kubernetes/community/tree/master/wg-component-standard

wg/component-standard

Slack:
Chairs: @mtaufen, @stealthybox, @sttts
#wg-component-standard #wg-component-standard-mentorship