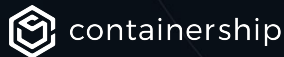



Kubernetes Manages More Than Containers

What are CCMs and how do you build one?

Ashley Schuett





About Me


What is a CCM?

Running a CCM

Features of a CCM

Creating your own

Questions?

A large, faint number '1' is visible on the left side of the slide. A vertical bar with a pink-to-purple gradient is positioned to the left of the 'About Me' title.

About Me

What is a CCM?

Running a CCM

Features of a CCM

Creating your own

Questions?

About me

- Engineer at **Containership**
- Cluster lifecycle management:
provisioning, plugins, upgrades, ...
- Controllers and anything running in-cluster



About Me

| **What is a CCM?**

Running a CCM

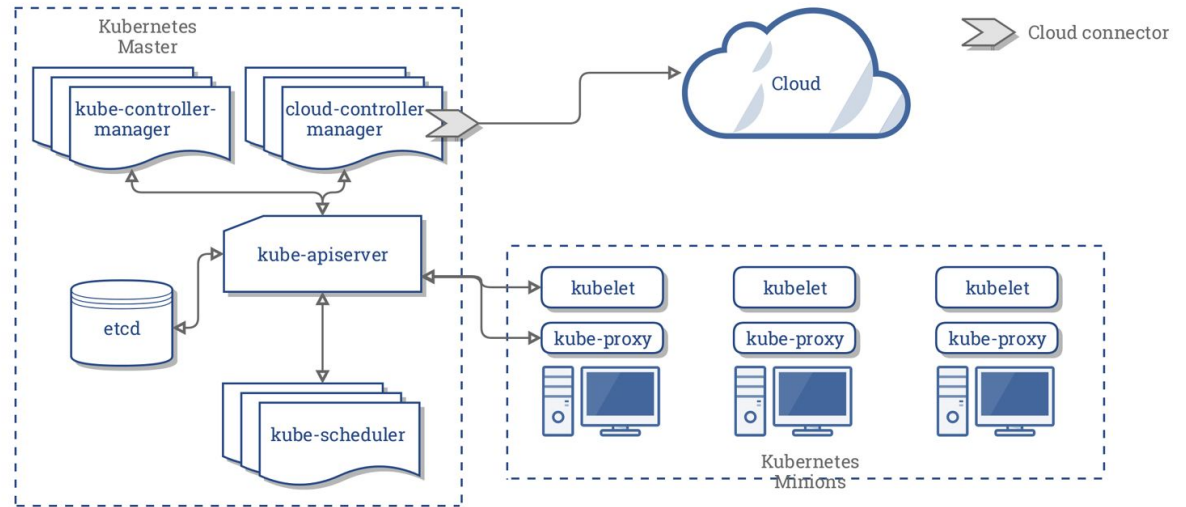
Features of a CCM

Creating your own

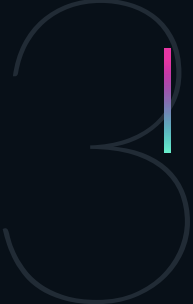
Questions?

What is a Cloud Controller Manager (CCM)?

- Daemon
- Cloud Specific Control loops
- Evolve independently of Kubernetes



Reference: [Concepts Underlying the Cloud Controller Manager](#)



About Me

What is a CCM?

Running a CCM

Features of a CCM

Creating your own

Questions?

Choose your adventure...

Add flag to kubelet (if in tree)

```
--cloud-provider=<provider>
```

Create a Deployment or DaemonSet

```
command: [  
  '/usr/local/bin/cloud-controller-manager',  
  '--cloud-provider=<provider>',  
],  
image:  
k8s.gcr.io/cloud-controller-manager:v1.12.3
```



When you specify a cloud-provider...

```
$ kubectl get nodes -o json | jq '.items[].spec'
{
  "taints": [{
    "effect": "NoSchedule",
    "key": "node.cloudprovider.kubernetes.io/uninitialized",
    "value": "true"
  } ]
}
```





About Me

What is a CCM?

Running a CCM

Features of a CCM

Creating your own

Questions?

Features of a CCM

- Kubernetes Node Management
- Load balancer Management
- Routing Management
- Any other feature you would like to add if you are running out-of-tree

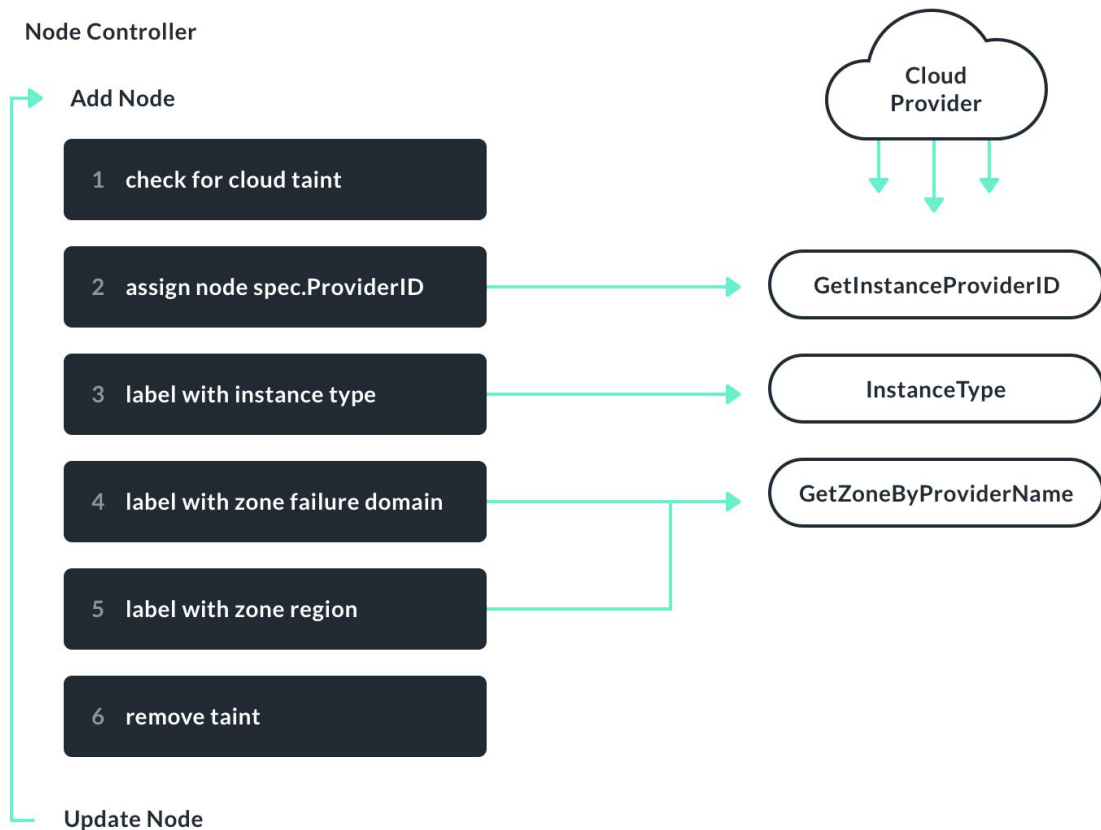


Kubernetes Node Management

- Responsible for initializing nodes
 - Adding providerID and removing taint
 - `node.spec.providerID: <provider>://<instance-id>`
 - Add zone/region labels
 - Writes the nodes network addresses and hostname to the node status
 - Node reconciliation

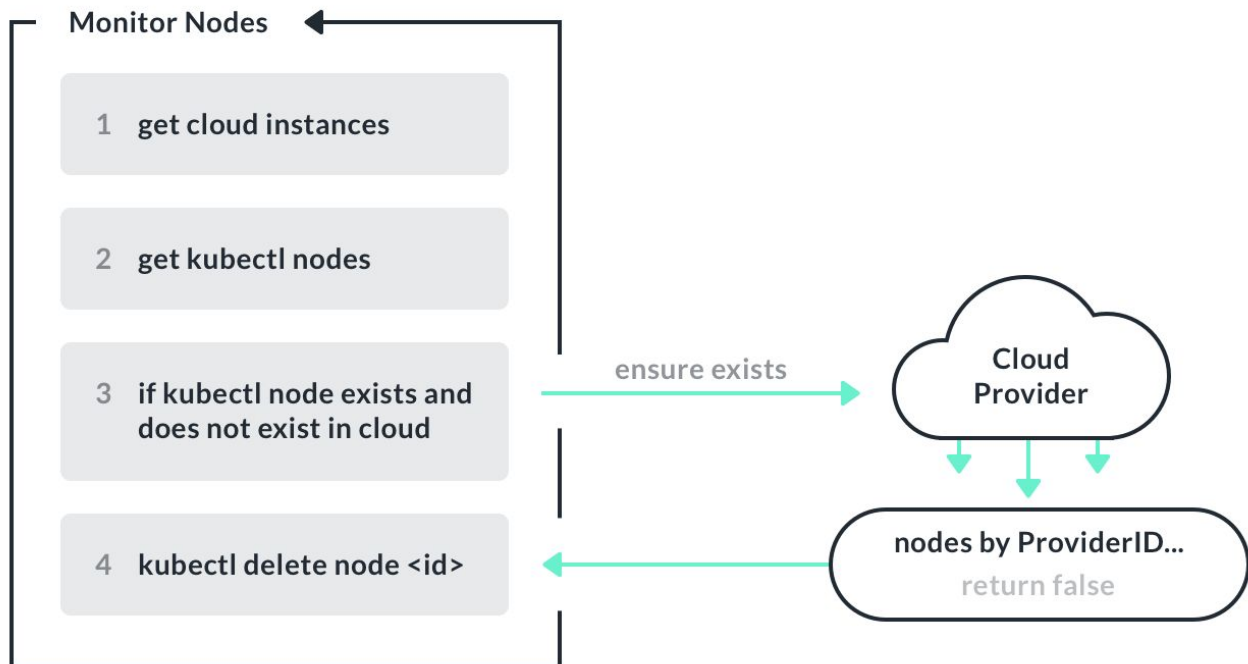


Node Controller

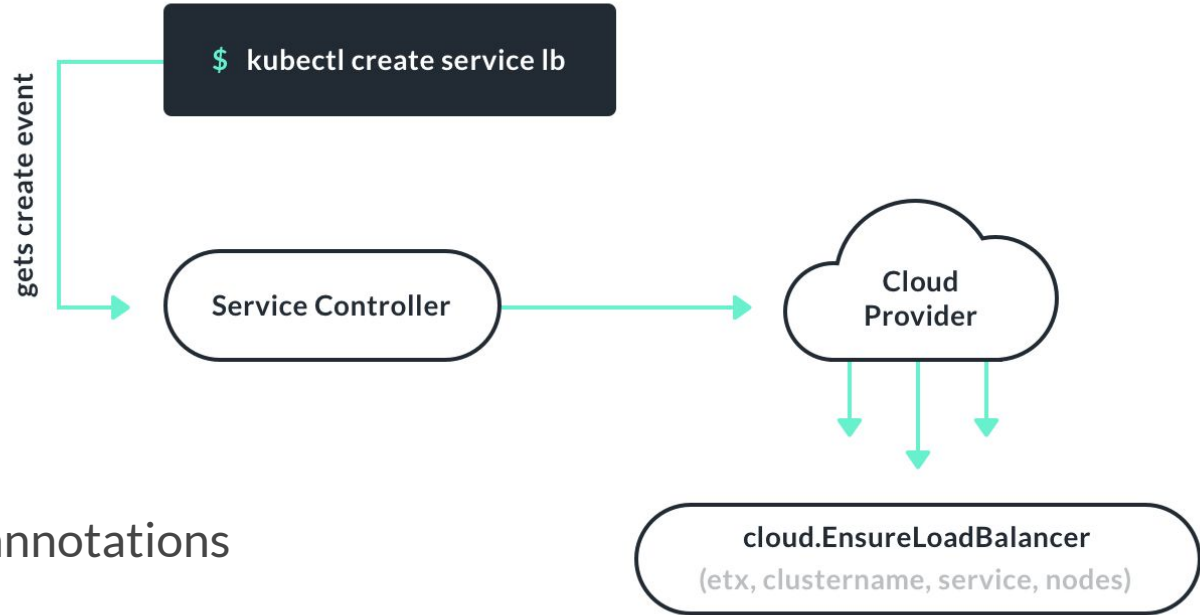


Node Reconciliation

Node Controller



Load balancer Management



- Services Controller
- Special features with annotations

Route Management

- Required for GCE
- Pod networking between nodes



Pain Points

- Documentation is sporadic
- Sometimes you have to dig into the code
- Configuration options not documented
- Hard to set up



Pain Points

```
// ConfigGlobal is the in memory representation of the gce.conf config data
// TODO: replace gcfg with json
type ConfigGlobal struct {
    TokenURL    string `gcfg:"token-url"`
    TokenBody  string `gcfg:"token-body"`
    // ProjectID and NetworkProjectID can either be the numeric or string-based
    // unique identifier that starts with [a-z].
    ProjectID   string `gcfg:"project-id"`
    // NetworkProjectID refers to the project which owns the network being used.
    NetworkProjectID string `gcfg:"network-project-id"`
    NetworkName string `gcfg:"network-name"`
    SubnetworkName string `gcfg:"subnetwork-name"`
    // SecondaryRangeName is the name of the secondary range to allocate IP
    // aliases. The secondary range must be present on the subnetwork the
    // cluster is attached to.
    SecondaryRangeName string `gcfg:"secondary-range-name"`
    NodeTags            []string `gcfg:"node-tags"`
    NodeInstancePrefix string `gcfg:"node-instance-prefix"`
    Regional            bool    `gcfg:"regional"`
    Multizone           bool    `gcfg:"multizone"`
    // APIEndpoint is the GCE compute API endpoint to use. If this is blank,
    // then the default endpoint is used.
    APIEndpoint string `gcfg:"api-endpoint"`
    // ContainerAPIEndpoint is the GCE container API endpoint to use. If this is blank,
    // then the default endpoint is used.
    ContainerAPIEndpoint string `gcfg:"container-api-endpoint"`
    // LocalZone specifies the GCE zone that gce cloud client instance is
    // located in (i.e. where the controller will be running). If this is
    // blank, then the local zone will be discovered via the metadata server.
    LocalZone string `gcfg:"local-zone"`
    // Default to none.
    // For example: MyFeatureFlag
    AlphaFeatures []string `gcfg:"alpha-features"`
}
```



Pain Points

- CCM's not standardized between providers
- CCM's not standardized when running from kubelet vs. deployment
- Container Storage Interface (CSI) / CCM discrepancy

command:

```
- /usr/local/bin/cloud-controller-manager
- --cloud-provider=<YOUR_CLOUD_PROVIDER>   # Add your own cloud provider here!
- --leader-elect=true
- --use-service-account-credentials
# these flags will vary for every cloud provider
- --allocate-node-cidrs=true
- --configure-cloud-routes=true
- --cluster-cidr=172.17.0.0/16
```



Past, Present, and Future of the CCM

- Cloud specific logic in core
 - api, controller and kubelet
- Pulled into own module in core
- Moving in-tree out of core

Read more: Kep [0002](#) & [0013](#)



About Me

What is a CCM?

Running a CCM

Features of a CCM

| Creating your own

Questions?

Building your own

- Implement the Cloud Interface
- Implement each features interface
- Why create your own?



Modules needed

Include core CCM

Import your API

```
import (  
    "k8s.io/kubernetes/cmd/cloud-controller-manager/app"  
    "k8s.io/kubernetes/cmd/cloud-controller-manager/app/options"  
  
    _ "github.com/organization/custom-ccm/custom" // your custom implementation  
)
```



Then run the CCM

```
func main() {
    s, err := options.NewCloudControllerManagerOptions()
    ...
    config, err := s.Config()
    ...
    if err := app.Run(config.Complete()); err != nil {
        fmt.Fprintf(os.Stderr, "%v\n", err)
        os.Exit(1)
    }
}
```



Interface

// Interface is an abstract, pluggable interface for cloud providers.

```
type Interface interface {  
    // Initialize provides the cloud with a kubernetes client builder and may spawn goroutines  
    // to perform housekeeping or run custom controllers specific to the cloud provider.  
    // Any tasks started here should be cleaned up when the stop channel closes.  
    Initialize(clientBuilder ControllerClientBuilder, stop <-chan struct{})  
    // LoadBalancer returns a balancer interface. Also returns true if the interface is supported, false otherwise.  
    LoadBalancer() (LoadBalancer, bool)  
    // Instances returns an instances interface. Also returns true if the interface is supported, false otherwise.  
    Instances() (Instances, bool)  
    // Zones returns a zones interface. Also returns true if the interface is supported, false otherwise.  
    Zones() (Zones, bool)  
    // Clusters returns a clusters interface. Also returns true if the interface is supported, false otherwise.  
    Clusters() (Clusters, bool)  
    // Routes returns a routes interface along with whether the interface is supported.  
    Routes() (Routes, bool)  
    // ProviderName returns the cloud provider ID.  
    ProviderName() string  
    // HasClusterID returns true if a ClusterID is required and set  
    HasClusterID() bool  
}
```

Reference: [kubernetes/cloud-provider/cloud.go](https://kubernetes.io/cloud-provider/cloud.go)



Initializing your provider

- Initialize provider
- Registering provider in `init()`

```
func init() {  
    cloudprovider.RegisterCloudProvider(  
        providerName,  
        func(config io.Reader),  
        (cloudprovider.Interface, error) {  
            return newCloud(config)  
        })  
}
```



Return the cloud provider interfaces

- Include the implemented interfaces

```
return &cloud{
    client:      client,
    instances:   newInstances(client, region),
    zones:       newZones(client, region),
    loadbalancers: newLoadbalancers(client, region),
}, nil
```

Examples:

- [packet-ccm](#)
- [digital-ocean-cloud-controller-manager](#)



Zones

- Used for syncing metadata
- Synced by the Node Controller
- Adds consistency

- **failure-domain.beta.kubernetes.io/region: nyc1**

```
// Zone represents the location of a particular machine.  
type Zone struct {  
    FailureDomain string  
    Region        string  
}
```



Implementing the Zone Interface

// Zones is an abstract, pluggable interface for zone enumeration. **Example:** [packet/packet-ccm/packet/facilities.go](https://github.com/packet/packet-ccm/blob/master/pkg/facilities.go)

```
type Zones interface {  
    // GetZone returns the Zone containing the current failure zone and locality region that the program is running in  
    // In most cases, this method is called from the kubelet querying a local metadata service to acquire its zone.  
    // For the case of external cloud providers, use GetZoneByProviderID or GetZoneByNodeName since GetZone  
    // can no longer be called from the kubelets.  
    GetZone(ctx context.Context) (Zone, error)  
  
    // GetZoneByProviderID returns the Zone containing the current zone and locality region of the node specified by providerID  
    // This method is particularly used in the context of external cloud providers where node initialization must be done  
    // outside the kubelets.  
    GetZoneByProviderID(ctx context.Context, providerID string) (Zone, error)  
  
    // GetZoneByNodeName returns the Zone containing the current zone and locality region of the node specified by node name  
    // This method is particularly used in the context of external cloud providers where node initialization must be done  
    // outside the kubelets.  
    GetZoneByNodeName(ctx context.Context, nodeName types.NodeName) (Zone, error)  
}
```



Implementing the Zone Interface

```
func newZones(client *packngo.Client, projectID, facility string) cloudprovider.Zones {  
    return zones{client, projectID, facility}  
}
```

```
// GetZone returns the Zone containing the current failure zone and locality region that the program is running in  
// In most cases, this method is called from the kubelet querying a local metadata service to acquire its zone.  
// For the case of external cloud providers, use GetZoneByProviderID or GetZoneByNodeName since GetZone  
// can no longer be called from the kubelets.
```

```
func (z zones) GetZone(_ context.Context) (cloudprovider.Zone, error) {  
    return cloudprovider.Zone{Region: z.facility}, nil  
}
```



Implementing the Zone Interface

```
// GetZoneByProviderID returns the Zone containing the current zone and locality region of the node specified by providerID
// This method is particularly used in the context of external cloud providers where node initialization must be done
// outside the kubelets.
func (z zones) GetZoneByProviderID(_ context.Context, providerID string) (cloudprovider.Zone, error) {
    id, err := deviceIDFromProviderID(providerID)
    if err != nil {
        return cloudprovider.Zone{}, err
    }

    device, err := deviceByID(z.client, id)
    if err != nil {
        return cloudprovider.Zone{}, err
    }

    return cloudprovider.Zone{Region: device.Facility.ID}, nil
}
```



Implementing the Zone Interface

```
// GetZoneByNodeName returns the Zone containing the current zone and locality region of the node specified by node name
// This method is particularly used in the context of external cloud providers where node initialization must be done
// outside the kubelets.
func (z zones) GetZoneByNodeName(_ context.Context, nodeName types.NodeName) (cloudprovider.Zone, error) {
    device, err := deviceByName(z.client, z.project, nodeName)
    if err != nil {
        return cloudprovider.Zone{}, err
    }


    return cloudprovider.Zone{Region: device.Facility.ID}, nil
}
```



CCM & You

- Contribute!
 - [sig cloud-provider](#)
- [Help out with docs!](#)
 - There is currently a plan waiting for approval with sig-docs
- Build and open source your own
- [In tree cloud providers](#)





About Me

What is a CCM?

Running a CCM

Features of a CCM

Creating your own

| Questions?

Thank you!

Ashley Schuett



@victimvillain



ashleyschuett



containership