KubeCon | CloudNativeCon

Europe 2020

Virtual

Kubernetes Leader Election for Fun and Profit

Nick Young, Staff Engineer, VMware

Leader Election for Fun and Profit

KubeCon CloudNativeCon Europe 2020 Virtual

What is Leader Election and how do you implement it?

Leader Election for Fun and Profit

What is Leader Election and how do you implement it?

My guidelines for using it

Leader Election for Fun and Profit

What is Leader Election and how do you implement it?

My guidelines for using it

Common patterns

What is Leader Election?

Leader election is a distributed lock, coordinated and managed for you by Kubernetes.

Lets your app elect one instance as a leader

Only one instance actually makes changes.

Leader Election for Fun and Profit

KubeCon | CloudNativeCon Europe 2020 Virtual

What can you use it for?

- One instance watches the Kubernetes API, and writes changes back.
- One instance watches the Kubernetes API, takes actions outside the Kubernetes cluster, then writes the result back.
- Each instance watches the Kubernetes API, and only one writes a status back.
- Other more complicated things

All of these things let you make your app more reliable by sprinkling a little HA on it.

KubeCon | CloudNativeCon Europe 2020 Virtual

How does Leader election work?

An election process updates a shared object that functions as the lock.

What's the shared object? It can be one of two options.

- < 1.14 ConfigMap only
- > 1.14 Lease or ConfigMap

For Configmaps, the info is stored in an annotation, for Lease, it's stored using the spec.

In either case, you'll need to ensure your app's role has the permissions to update the object.

The module to read the docs is `k8s.io/client-go/tools/leaderelection/resourcelock`

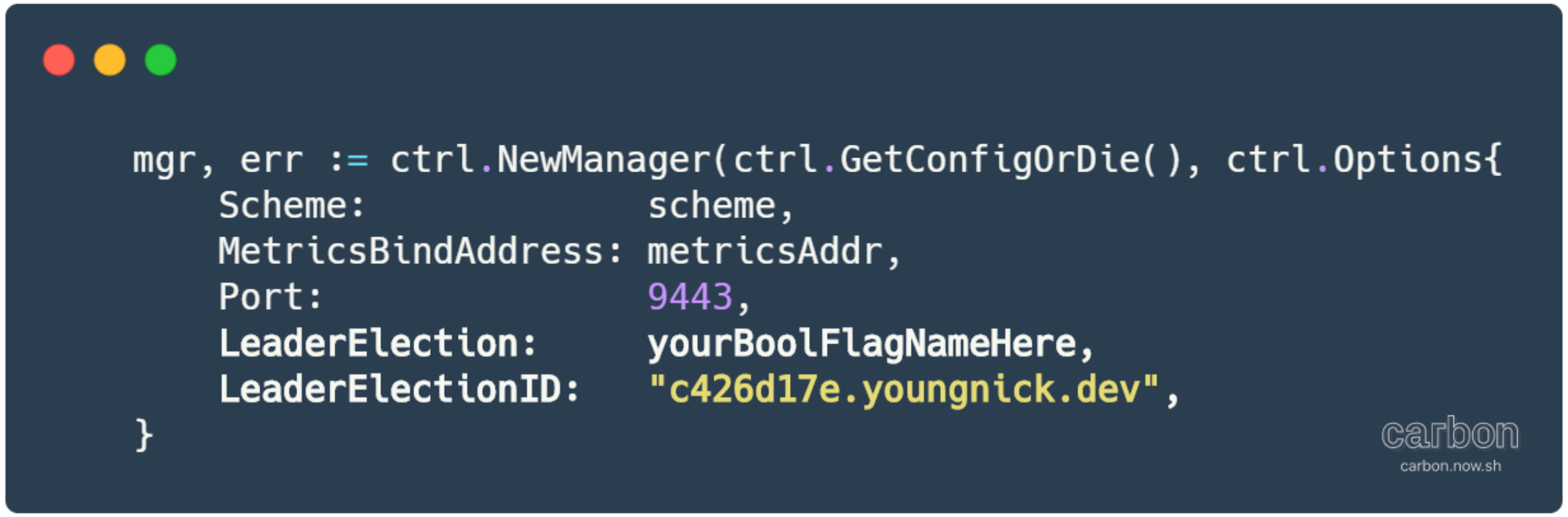
Leader Election for Fun and Profit

Implementing Leader Election - Kubebuilder

If you're using kubebuilder's generation tools, you already have it available! Turn it on using

`<progname> -enable-leader-election`

Otherwise, you just need to add the options in your call to `NewManager()`



```go
mgr, err := ctrl.NewManager(ctrl.GetConfigOrDie(), ctrl.Options{
    Scheme:             scheme,
    MetricsBindAddress: metricsAddr,
    Port:               9443,
    LeaderElection:     yourBoolFlagNameHere,
    LeaderElectionID:   "c426d17e.youngnick.dev",
}
```

Implementing Leader Election – Informers

Using informers means that you have two jobs, getting the leader election running, and then watching the channel that the leader election gives you.

In Contour, we:

- Create and run the Kubernetes leader elector, this returns the "I became leader" channel.
- Run another goroutine that triggers when we become leader by watching the channel.
- Have a number of other processes that watch the same channel to check if we're the leader.

The relevant Kubernetes package for this is `k8s.io/client-go/tools/leaderelection`

Leader Election for Fun and Profit

Usage Guidelines

Don’t try and clean up if your instance is deposed somehow. Use Kubernetes instead.

This is a human-scale lock, not a machine-scale one.

Be aware of what you’re using your distributed lock for.

Leader Election for Fun and Profit

KubeCon | CloudNativeCon Europe 2020 Virtual

Common patterns

Basic Controller or Operator

Watches Kubernetes objects, and does stuff based on it.
Does not present its own endpoints for anything other than health checking and metrics.

Use the leader lock to block the code that does stuff.

Leader Election for Fun and Profit

Common patterns

Basic Controller or Operator

Kubebuilder: the Reconcilers are leader-elected, so only one instance will actually do anything.

Informers: The easiest way is to block `main()` on becoming the leader. Once you become leader, you do stuff until you either

- shut down
- lose leadership, in which case you shut down

If you are going to do that, top tip: Make sure your health and metrics endpoints are initialized first. Ask me how I know!

Common patterns

Multi-read, leader writes

Watches Kubernetes objects, serves out endpoints of its own.
May need to take actions as well, and update objects in response.

Contour is an example:
- Watches Kubernetes objects, translates to Envoy xDS.
- Leader writes status updates back to Contour’s CRDs.

Kubebuilder’s webhooks are another example:
- Main Reconcilers are leader elected, only the leader actually does the Reconciling.
- Non-leader instances serve the defaulting, conversion, and validation webhooks.

Leader Election for Fun and Profit

Common patterns

Advanced pattern

Per-CRD leader election.

Knative breaks out the leader election by object type.

This allows the spreading of the reconciliation workload over instances of the controller.

Leader Election for Fun and Profit

Summary

Leader election is great and you should really use it for your Kubernetes controllers.

Kubebuilder makes it very easy, but it’s not super hard to DIY.

Let Kubernetes handle the deposed case, by restarting your app.

Think carefully about what you’re going to use the lock for.

KubeCon | CloudNativeCon

Europe 2020

Virtual

KEEP CLOUD NATIVE

CONNECTED