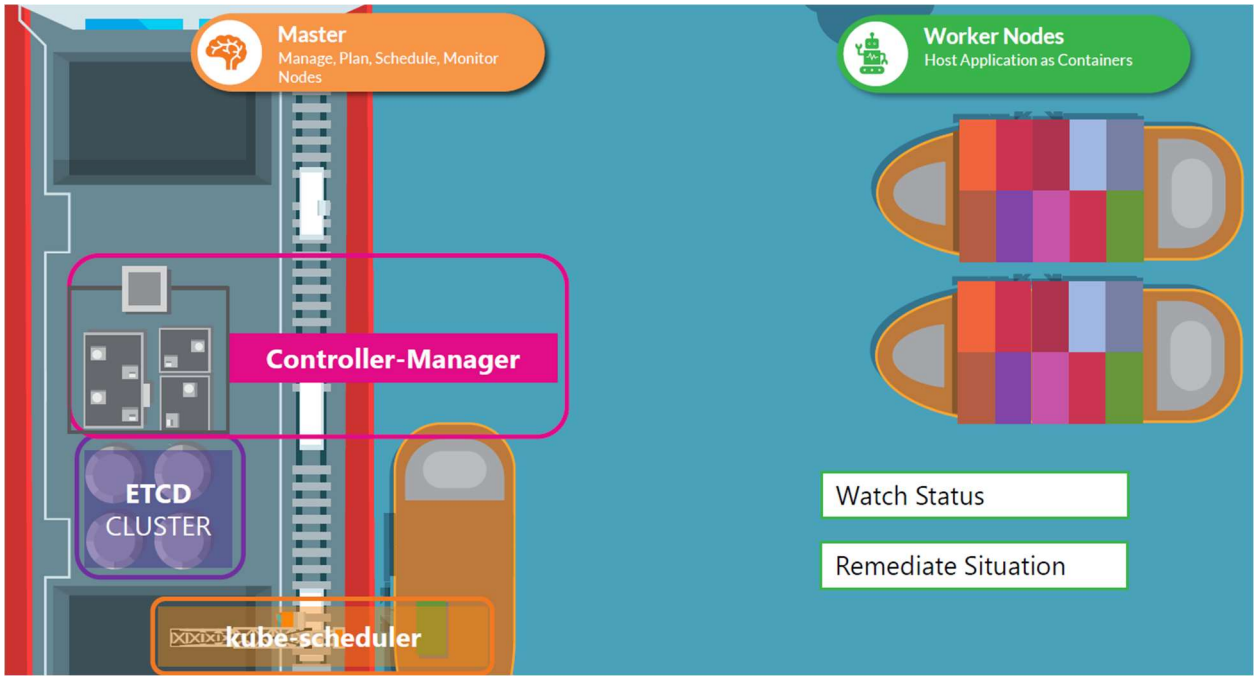
Kube Controller Manager

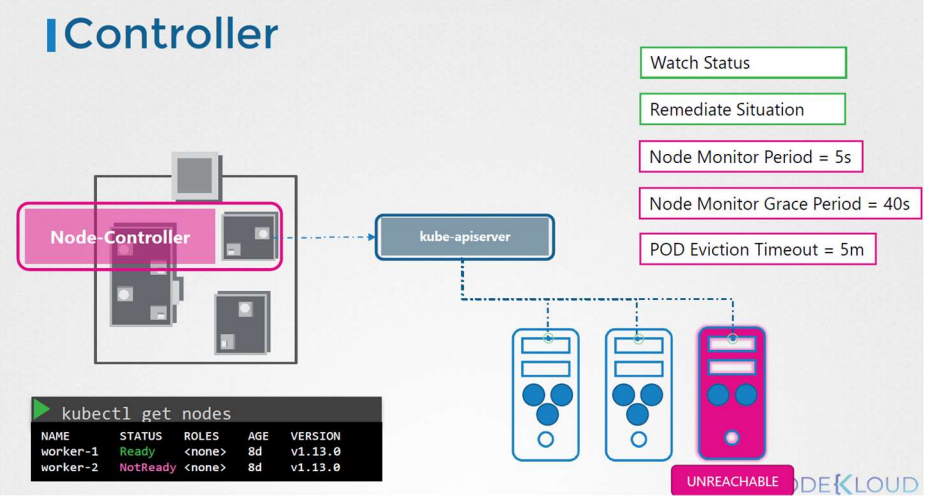
In this lecture, we will talk about Kube Controller Manager. As we discussed earlier, the Kube controller manager manages various controllers in Kubernetes. A controller is like an office or department within the mastership, that has its own set of responsibilities, such as an office for the ships would be responsible for monitoring and taking necessary actions about the ships. Whenever a new ship arrives or when a ship leaves or gets destroyed.

Another office could be one that manages the containers on the ships. They take care of containers that are damaged or fall off ships. So these offices are number one, continuously on the lookout for the status of the ships, and two, take necessary actions to remediate the situation.

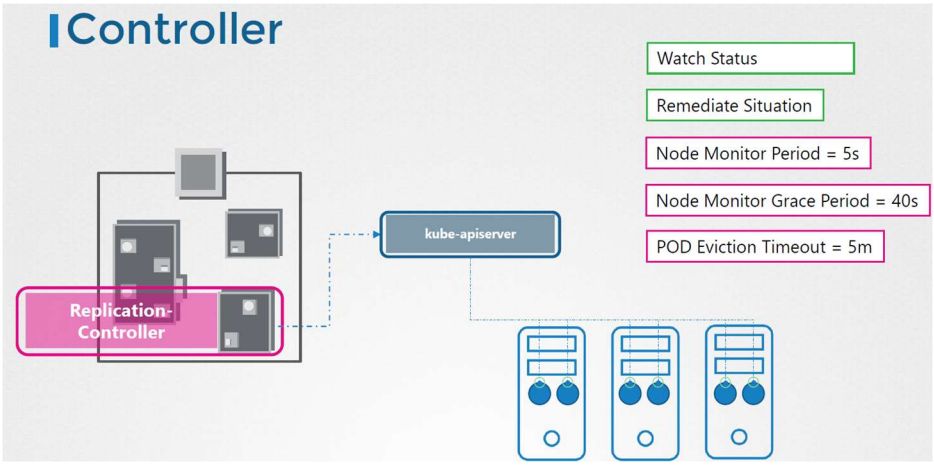


In Kubernetes terms, a controller is a process that continuously monitors the state of various components within the system and works towards bringing the whole system to the desired functioning state. For example, the node controller is responsible for monitoring the status of the nodes and taking necessary actions to keep the applications running. It does that through the Kube API server.

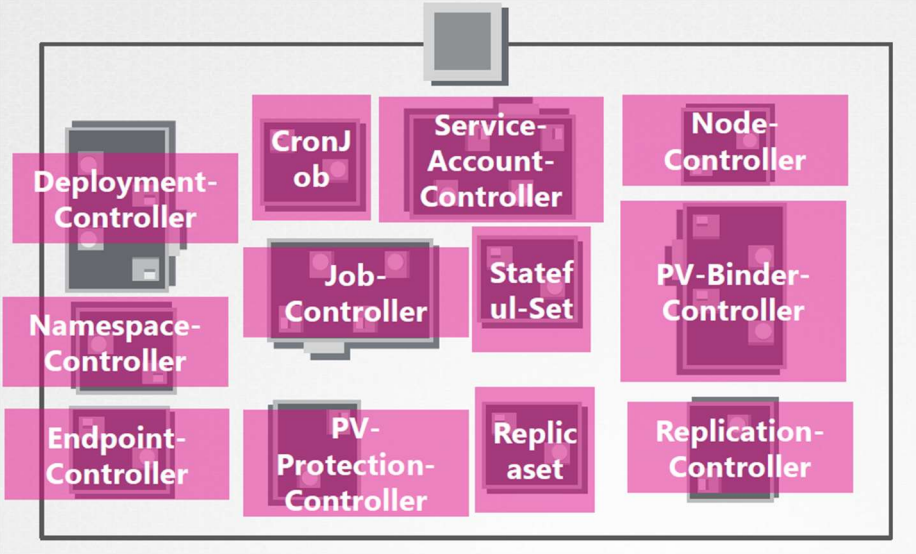
The node controller tests the status of the nodes every five seconds. That way, the node controller can monitor the health of the nodes. If it stops receiving a heartbeat from a node, the node is marked as unreachable, but it waits for 40 seconds before marking it unreachable. After a node is marked unreachable, it gives it five minutes to come back up. If it doesn't, it removes the PODs assigned to that node and provisions them on the healthy ones if the PODs are part of a replica set.



The next controller is the replication controller. It is responsible for monitoring the status of replica sets and ensuring that the desired number of PODs are available at all times within the set. If a POD dies, it creates another one.



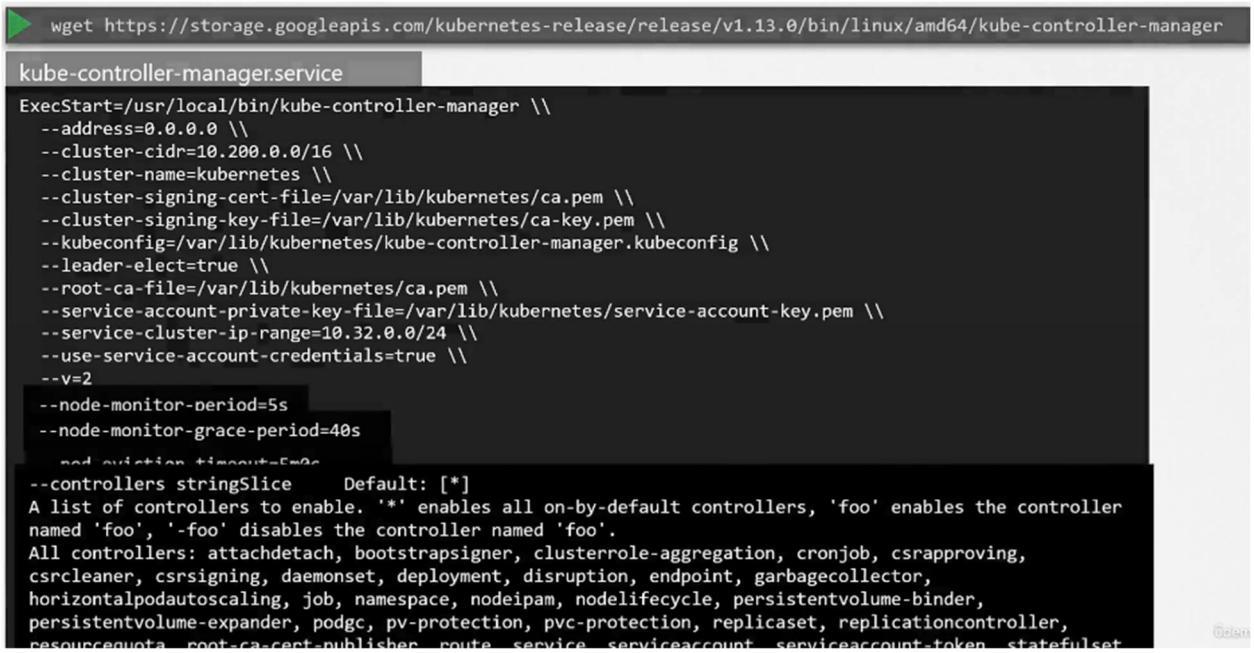
Now, those were just two examples of controllers. There are many more such controllers available within Kubernetes. Whatever concepts we have seen so far in Kubernetes, such as deployment, services, namespaces, or persistent volumes, and whatever intelligence is built into these constructs, it is implemented through these various controllers. As you can imagine, this is kind of the brain behind a lot of things in Kubernetes.



Now, how do you see these controllers and where are they located in your cluster? They're all packaged into a single process known as the Kubernetes Controller Manager. When you install the Kubernetes controller manager, the different controllers get installed as well.

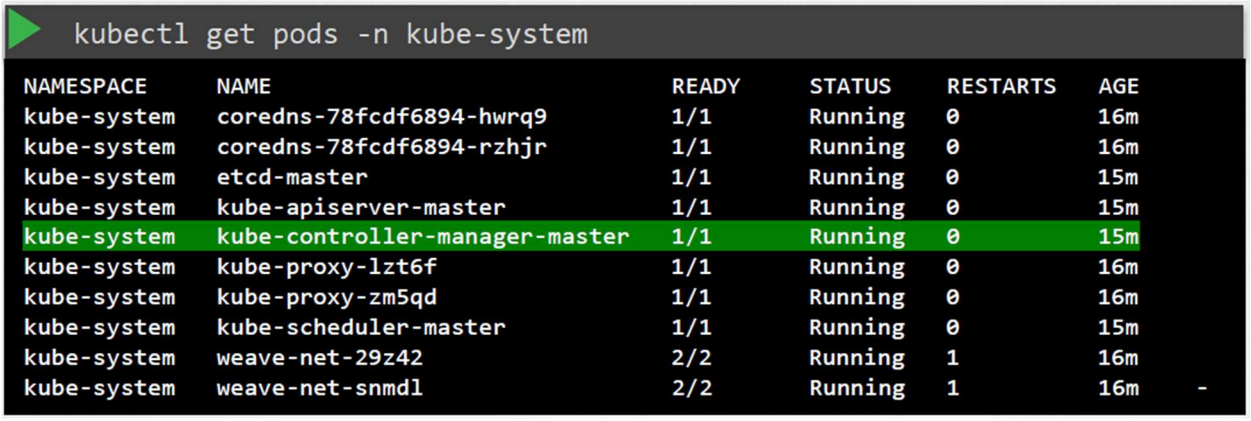
So how do you install and view the Kubernetes controller manager? Download the Kube controller manager from the Kubernetes release page, extract it, and run it as a service. When you run it, as you can see, there is a list of options provided. This is where you provide additional options to customize your controller.

Remember some of the default settings for the node controller we discussed earlier, such as the node monitor period, the grace period, and the eviction timeout. These go in here as options. There is an additional option called `controllers` that you can use to specify which controllers to enable. By default, all of them are enabled, but you can choose to enable and select a few.



So in case any of your controllers don't seem to work or exist, this would be a good starting point to look at.

So how do you view the Kube controller manager's server options? Again, it depends on how you set up your cluster. If you set it up with the Kube admin tool, Kube admin deploys the Kube controller manager as a POD in the Kube system namespace on the master node. You can see the options within the POD definition file located at Etsy Kubernetes Manifest folder.



```
cat /etc/kubernetes/manifests/kube-controller-manager.yaml
```

```
spec:
  containers:
  - command:
    - kube-controller-manager
    - --address=127.0.0.1
    - --cluster-signing-cert-file=/etc/kubernetes/pki/ca.crt
    - --cluster-signing-key-file=/etc/kubernetes/pki/ca.key
    - --controllers=*,bootstrapsigner,tokencleaner
    - --kubeconfig=/etc/kubernetes/controller-manager.conf
    - --leader-elect=true
    - --root-ca-file=/etc/kubernetes/pki/ca.crt
    - --service-account-private-key-file=/etc/kubernetes/pki/sa.key
    - --use-service-account-credentials=true
```

In a non-Kube admin setup, you can inspect the options by viewing the Kube Controller Manager's service located in the services directory. You can also see the running process and the effective options by listing the process on the master node and searching for Kube Controller Manager.

```
cat /etc/systemd/system/kube-controller-manager.service
```

```
[Service]
ExecStart=/usr/local/bin/kube-controller-manager \\
  --address=0.0.0.0 \\
  --cluster-cidr=10.200.0.0/16 \\
  --cluster-name=kubernetes \\
  --cluster-signing-cert-file=/var/lib/kubernetes/ca.pem \\
  --cluster-signing-key-file=/var/lib/kubernetes/ca-key.pem \\
  --kubeconfig=/var/lib/kubernetes/kube-controller-manager.kubeconfig \\
  --leader-elect=true \\
  --root-ca-file=/var/lib/kubernetes/ca.pem \\
  --service-account-private-key-file=/var/lib/kubernetes/service-account-key.pem \\
  --service-cluster-ip-range=10.32.0.0/24 \\
  --use-service-account-credentials=true \\
  --v=2
Restart=on-failure
RestartSec=5
```

```
ps -aux | grep kube-controller-manager
```

```
root      1994  2.7  5.1 154360 105024 ?       Ssl  06:45   1:25 kube-controller-manager --address=127.0.0.1 --cluster-signing-cert-file=/etc/kubernetes/pki/ca.crt --cluster-signing-key-file=/etc/kubernetes/pki/ca.key --controllers=*,bootstrapsigner,tokencleaner --kubeconfig=/etc/kubernetes/controller-manager.conf --leader-elect=true --root-ca-file=/etc/kubernetes/pki/ca.crt --service-account-private-key-file=/etc/kubernetes/pki/sa.key --use-service-account-credentials=true
```