Cloud Provider:AWS
Server:Ubuntu 18.04 x6
Server List: 2 Controller(Master), 2(Worker), 1 Nginx(Reverse Proxy), 1 Ubuntu 18.04(Local)

Worker Node Architecture Overview
Kubernetes Remote Access and kubectl

// We will set up kubectl to allow remote access from our machine in order to manage the cluster remotely.
To do this, we will generate a local kubeconfig that will authenticate as the admin user and access the Kubernetes API through the load balancer.

// There are a few steps to configuring a local kubectl installation for managing a remote cluster. This lesson will guide you through that process. After completing this lesson, you should have a local kubectl installation that is capable of running kubectl commands against your remote Kubernetes cluster.

//In a separate shell, open up an ssh tunnel to port 6443 on your Kubernetes API load balancer:

```
$ ssh -L 6443:localhost:6443 cloud_user@<your Load balancer cloud server public IP>
```

// You can configure your local kubectl in your main shell like so.

```
$ cd ~/kthw
$ kubectl config set-cluster kubernetes-the-hard-way \
  --certificate-authority=ca.pem \
  --embed-certs=true \
  --server=https://localhost:6443
$ kubectl config set-credentials admin \
  --client-certificate=admin.pem \
  --client-key=admin-key.pem
$ kubectl config set-context kubernetes-the-hard-way \
  --cluster=kubernetes-the-hard-way \
  --user=admin
$ kubectl config use-context kubernetes-the-hard-way
```

// Verify that everything is working with:

```
$ kubectl get pods
// should return no resources since no pods are running
$ kubectl get nodes
$ kubectl version
```