1. Create a new resource group

As I said before, a resource group is a way for Azure to keep all related resources together so that you can make templates, share permissions and policies, or clean out everything by simply deleting the resource group. So run the following command to create it:

az group create --name coolapp --location eastus

2. Create the AKS cluster

Run the following command to create the cluster:

az aks create --resource-group coolapp --name coolk8s --node-count 2 --node-vm-size Standard\_A2\_v2 --generate-ssh-keys

So what exactly did we just do? Let’s explore the parameters and values we entered here:

* **–resource-group** is the name of the resource group we just created.
* **–name** is the name of the cluster to identify it.
* **–node-count** is the number of nodes we want for our cluster.
* **–node-vm-size** is the name of the instance type we choose. It’s better to be specific here to avoid potential problems with the limitations of our subscription.
* **–generate-ssh-keys** will generate SSH keys on your local machine so it’s easier for you to connect to any node if need be.

Specifying only those arguments means that Azure will use the default values for things like networking or monitoring that we saw when creating the cluster in the portal. Let’s keep it simple for now, but in case you want to explore the other arguments, you can take a look at the [docs](https://docs.microsoft.com/en-us/cli/azure/aks?view=azure-cli-latest#az-aks-create).

In the meantime, Azure will create the cluster. Mine took twenty minutes or so to finish. And that’s it! You created the cluster with just two commands. Now let’s make sure the cluster is actually working.

Accessing the Kubernetes UI locally

Whether you created the cluster using the portal, the command line, or both, the following instructions will work to access the Kubernetes dashboard. This is where you **need to have the latest version of**[**Azure CLI**](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli?view=azure-cli-latest)**and** **[kubetcl](https://kubernetes.io/docs/tasks/tools/install-kubectl/" \t "_blank)** installed and configured.

1. Download cluster credentials

Start by downloading the cluster credentials to your computer by running this command:

az aks get-credentials --resource-group=coolapp --name=coolk8s

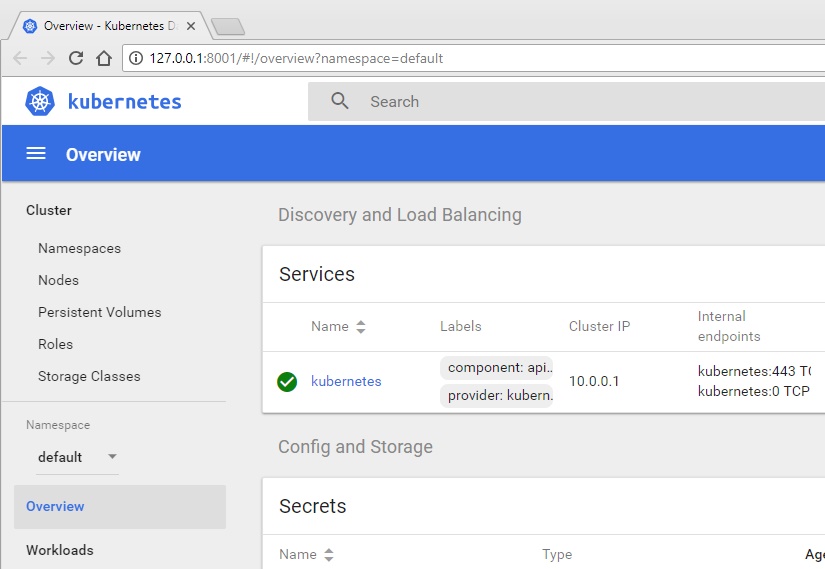
Specify the resource group with the **–resource-group** parameter and the name of the cluster with the**–name** parameter. Doing this makes it easy to toggle between different Kubernetes clusters that you’ve connected previously (for example, a local version of Kubernetes).

2. Browse the cluster

Run the following command. A new browser tab or window will open with the Kubernetes dashboard automatically.

az aks browse --resource-group coolapp --name coolk8s

Wait a few more seconds, and the Kubernetes dashboard will appear:



If your screen looks like this one, congratulations! That means your cluster works and you can connect to it.

Clean up your resources

When you’re done, you can delete the AKS cluster by running the following command and then confirming that you want to delete it:

az aks delete --resource-group coolapp --name coolk8s

It will take some time to finish, but please keep an eye on the results of the command. The cluster might not be deleted, and you’ll end up [paying](https://azure.microsoft.com/en-us/pricing/details/container-service/) until you delete it. If you want to delete the resource group, run this command:

az group delete -n coolapp

Let the cloud manage the Kubernetes cluster for you

Even though this was a long post, you can see that it’s really easy to create a Kubernetes cluster in Azure. The best part is that you only pay for the nodes you use *and* you don’t have to worry about administering the master nodes. If for some reason the master nodes are down, don’t worry! Remember that Kubernetes is just an orchestrator. The worst thing that can happen is that you won’t be able to do deployments, scale out, or down the application.

You can also connect to the Kubernetes cluster to do deployments, but I would recommend you use [Visual Studio Team Services](https://stackify.com/what-is-visual-studio-team-services/) (VSTS) to create CI/CD pipelines. Or you can just simply access the dashboard to take a look. What’s important here is that you don’t have to worry about installing, configuring, and managing Kubernetes. You simply use it; it will be transparent for your deployments because in the end, it will continue to be a Kubernetes cluster.