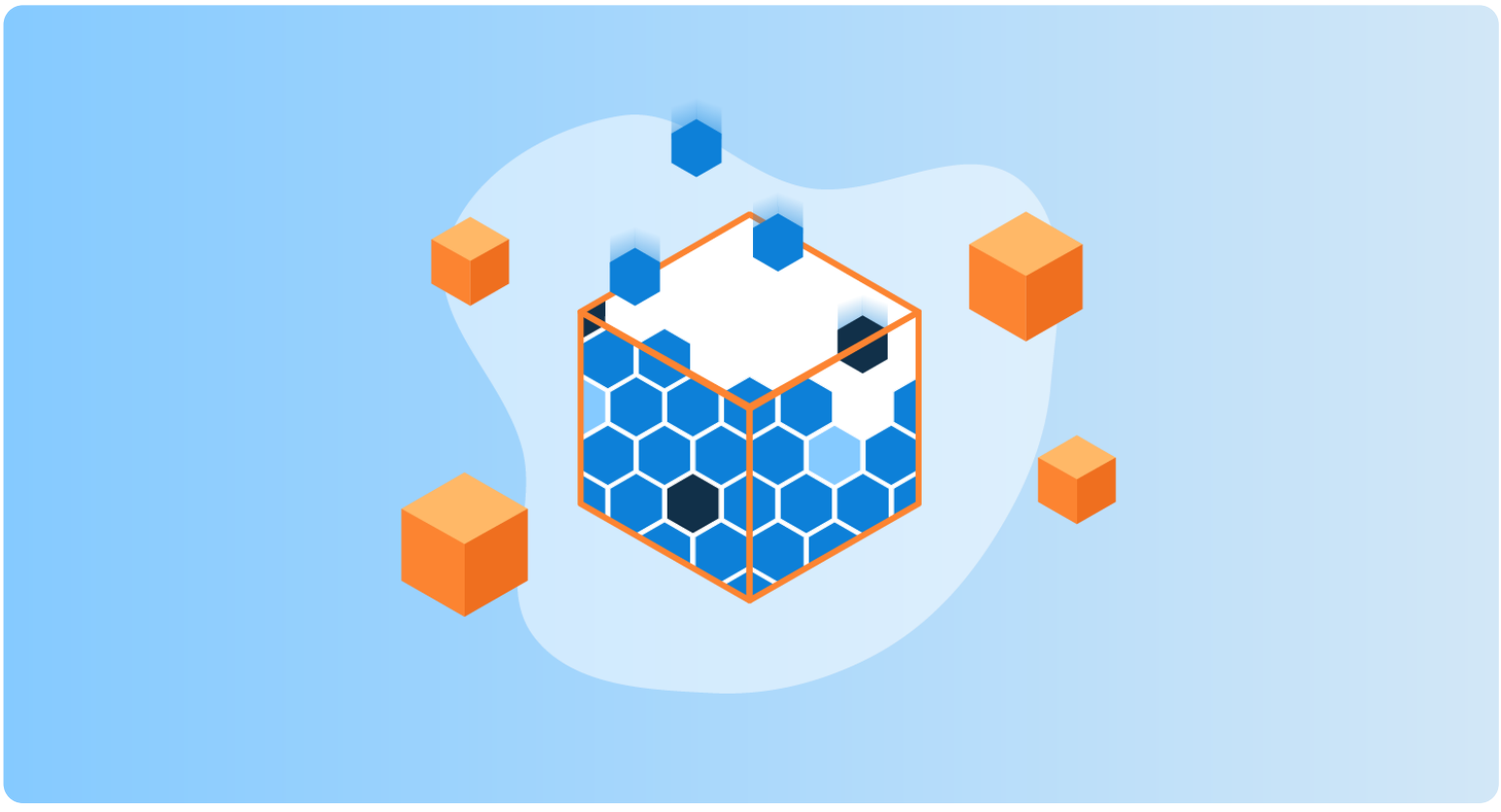


Creating an EKS cluster in AWS

Terence Wong

February 9, 2022 • 3 mins

In this post, you learn how to set up an Elastic Kubernetes Service (EKS) cluster in Amazon Web Services (AWS).

EKS is a managed container service to run and scale Kubernetes in the cloud or on-premises. Kubernetes provides a scalable, distributed way to manage workloads. It does this by containerizing applications. Containers ensure replicability across different environments and cloud infrastructures.

The clusters you create in this post will be used in later posts in our Continuous Integration series, to set up web applications and as part of workflows. We'll add links to the relevant posts as they become available:

- Building a Docker image in Jenkinsfile and publishing to ECR
- Deploying to Amazon EKS with Docker and Jenkins

Prerequisites

To follow along with this post, you need:

- An AWS account
- A terminal with kubectl, eksctl, and aws-iam-authenticator installed
- IAM permissions

There are two ways to set up a cluster in EKS:

- The command-line interface (CLI)
- The console

Before this though, you need to set up some access keys and user accounts.

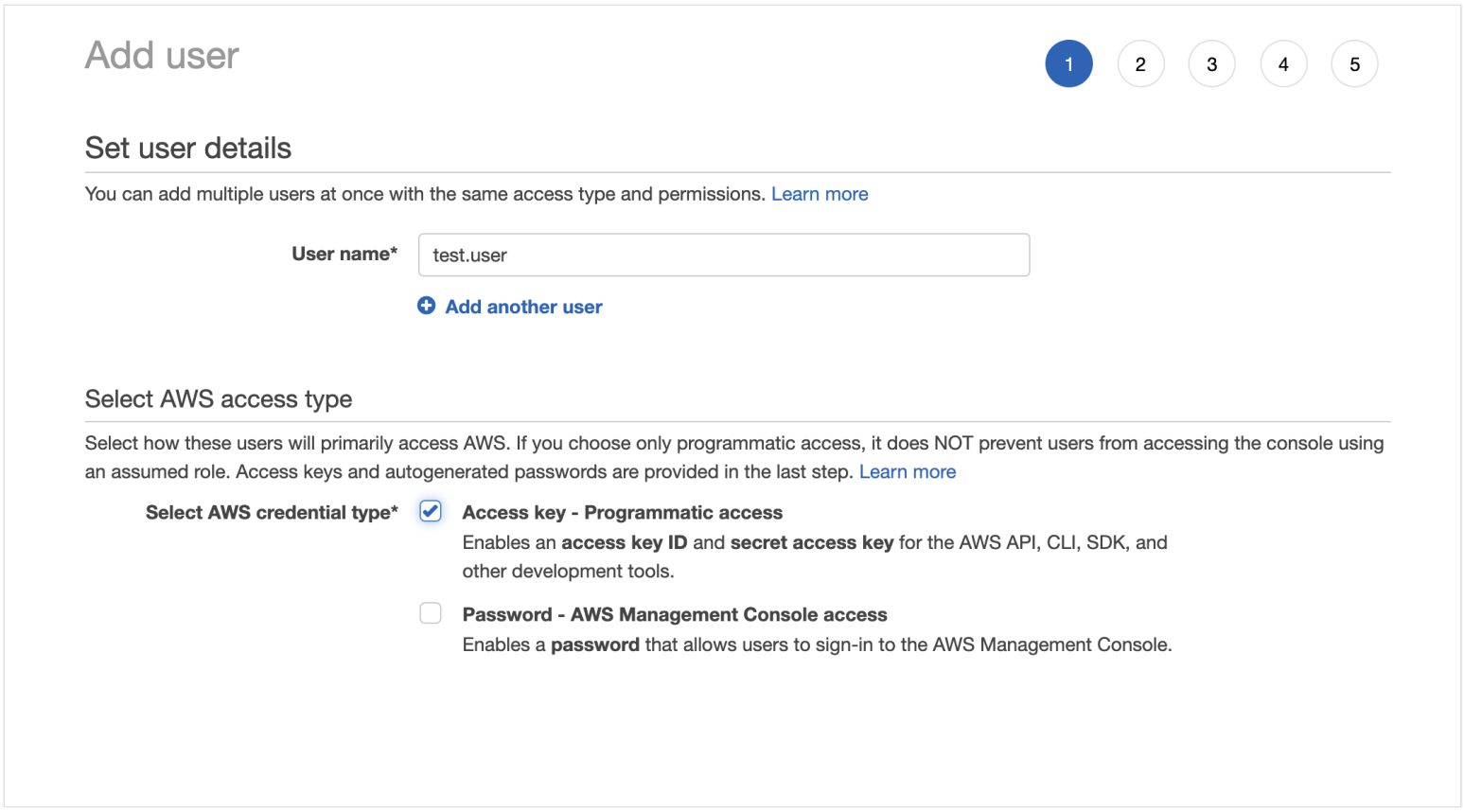
Preliminary setup

In AWS, you need to configure access policies. These policies determine what kind of user can access the cluster.

Typically, it's useful to follow the principle of least privilege. This means you give users the minimum set of privileges required to carry out their role. This supports the security of the cloud infrastructure by not granting users more permissions than they need.

Follow these steps to set up your access keys and user accounts:

- Go to **AWS Console**, then **IAM**, then **Users**, and **Add Users**
- Give the user a name, and tick **Access Key - Programmatic access**
- Click **Next**



- Select **Attach existing policies directly**, then **Create policy**

The IAM policy allows you to create an EKS cluster from the command-line. The actions in this policy are the minimum policies required by eksctl.

- Click **Next** and give the policy a name
- Click **Add Policy**

Amazon then shows you the **Access Key ID** and **Secret Access Key**. Download this file for reference later.

Command-line interface

Log in to the AWS command-line using `aws login`.

Run `aws configure`.

Enter your Access Key ID and Secret Access Key earlier. Set the zone to `us-east-2` and accept the defaults.

Now you can create your cluster.

```
eksctl create cluster \
--name my-cluster \
--region us-east-2 \
--fargate
```

AWS Fargate lets you run containers without managing servers or clusters of Amazon EC2 instances. The profile provides a simple way to spin up a test cluster. Test the cluster configuration:

```
kubectl get svc
```

Output

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   25h
```

Run the following command to view your cluster nodes:

```
kubectl get nodes -o wide
```

Output

```
NAME                                                STATUS   ROLES    AGE   VERSION
fargate-ip-192-168-129-76.us-east-2.compute.internal    Ready    <none>   25h   v1.20.7-eks-1353
fargate-ip-192-168-165-146.us-east-2.compute.internal   Ready    <none>   25h   v1.20.7-eks-1353
```

View the workloads that are running on your cluster:

```
kubectl get pods --all-namespaces -o wide
```

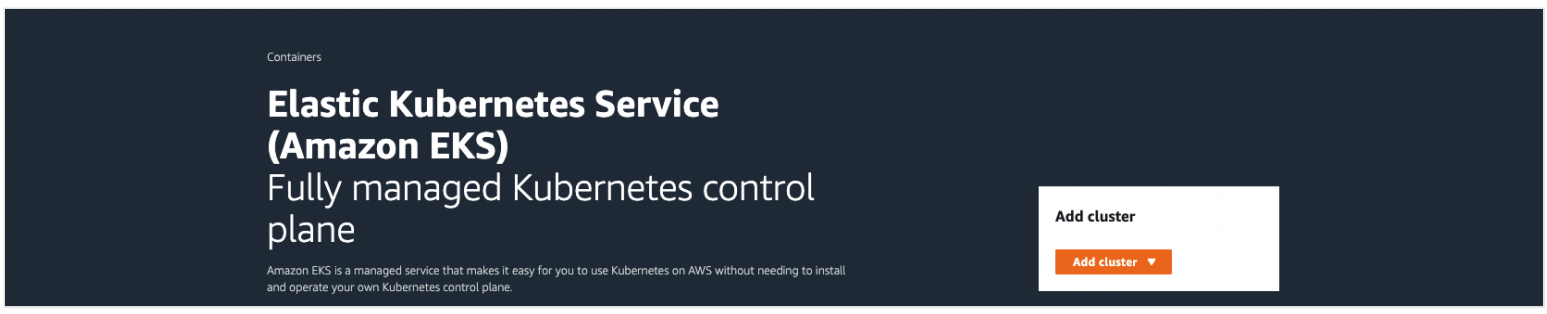
Output

```
NAMESPACE     NAME                       READY   STATUS    RESTARTS   AGE   IP               NO
kube-system   coredns-85f9f6cd8b-2n8wr   1/1     Running   0          25h   192.168.129.76   fa
kube-system   coredns-85f9f6cd8b-c4jfk   1/1     Running   0          25h   192.168.165.146  fa
```

Console interface

You can also create a cluster from the AWS console.

Go to **EKS**, then **Add Cluster**, then **Create**.



- **Name**: give your cluster a name
- **Kubernetes Version**: select the latest Kubernetes version
- **Cluster Service Role**: re-use the service role created from the CLI cluster

Accept all other defaults to create your cluster.

Checking clusters

Check the state of your two clusters by going to **EKS**, then **Clusters**.

Now that your clusters are live, you can perform operations on them, such as deploying an application or configuring resources. For this example, we'll delete them.

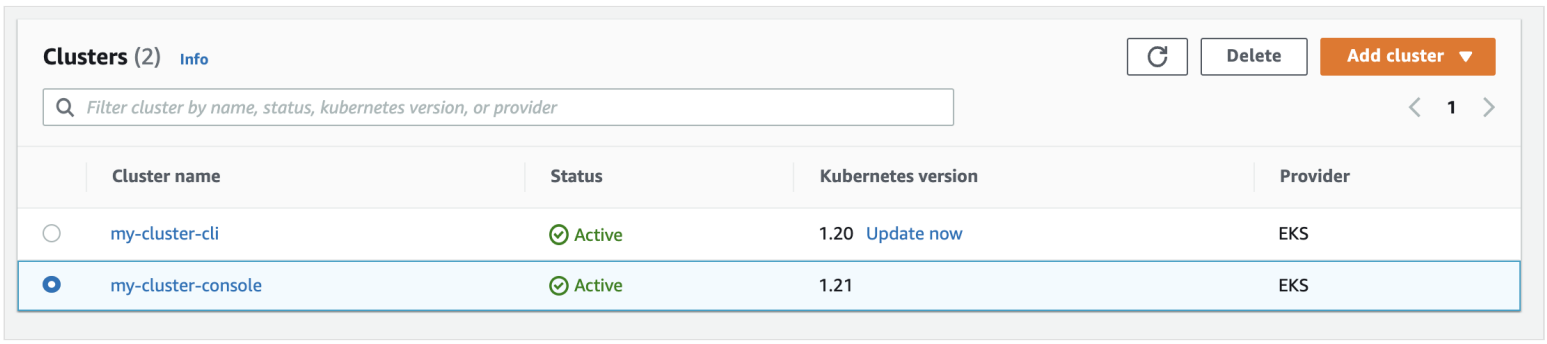
Deleting the clusters

You can delete a cluster using the CLI by running this command. Replace the cluster name and region with your values.

```
eksctl delete cluster --name my-cluster --region us-east-2
```

You can delete a cluster using the console by selecting the cluster, clicking **Delete**, and typing the name of the cluster to delete.

You can only delete a resource the same way you created it. This means clusters created through the CLI can only be deleted through the CLI. Clusters created through the console can only be deleted through the console.



Conclusion

In this post, you set up IAM permissions in AWS. You used the CLI and console to create, inspect, and delete an EKS cluster.

EKS on AWS allows you to provision Kubernetes services in the cloud to deploy and scale workloads.

The clusters you created can be used to set up web applications and as part of workflows in upcoming posts. We'll add links to the relevant posts as they become available:

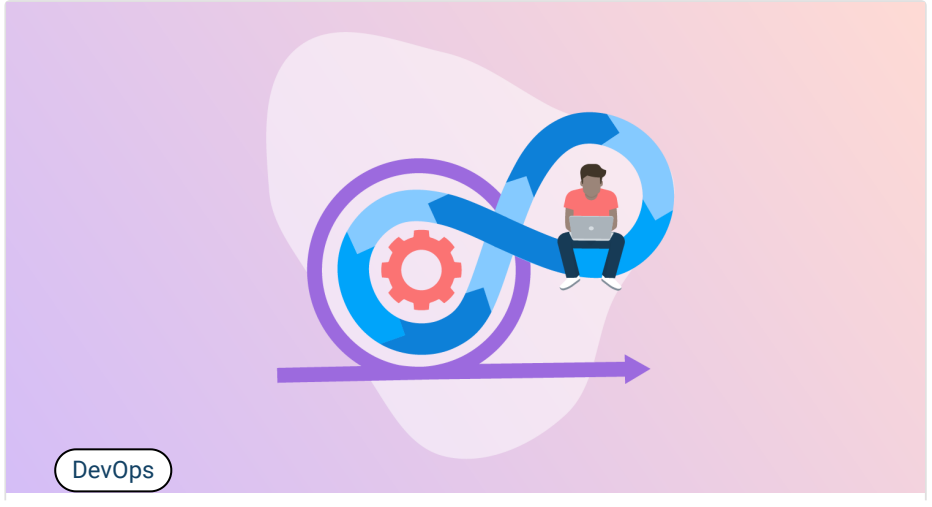
- Building a Docker image in Jenkinsfile and publishing to ECR
- Deploying to Amazon EKS with Docker and Jenkins

Read the rest of our Continuous Integration series.

Happy deployments!

Tagged with: DevOps CI Series Continuous Integration AWS Kubernetes

Related posts

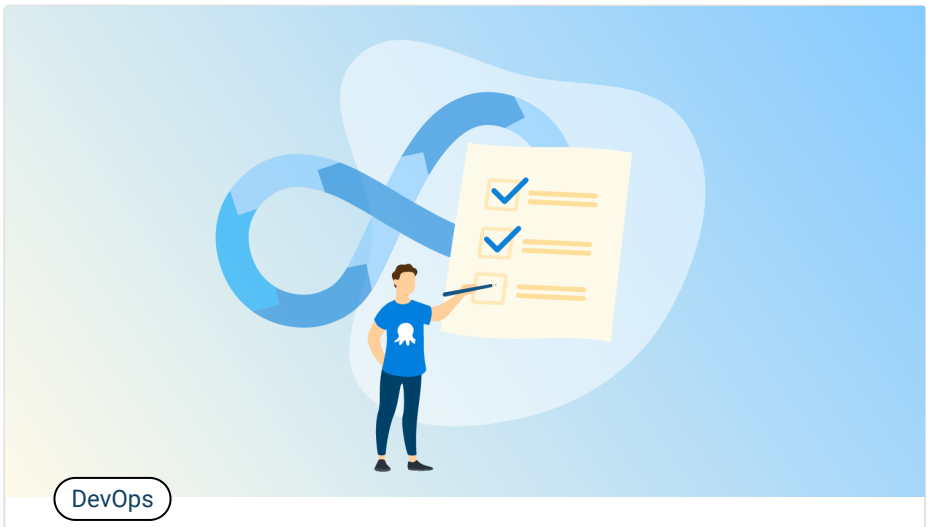


Common mistakes in DevOps metrics

Steve Fenton
December 7, 2022 • 6 mins

Comparing Lean, Agile, and Continuous Delivery

Steve Fenton
December 5, 2022 • 6 mins



DevOps

Best practices for CI/CD

Terence Wong
November 30, 2022 • 5 mins

Newsletter

Logged in as Terence Wong (terence.wong@octopus.com)

Join ~48,000 DevOps professionals and sign up for the latest Octopus news, events, and opinions. No spam. Unsubscribe at any time.

Subscribe

Your privacy is important to us. Read more in our Privacy Policy.