Setting Up Airbyte in EKS Using Helm

This guide will walk you through setting up Airbyte on Amazon Elastic Kubernetes Service (EKS) using Helm. By following these steps, you'll have Airbyte running in a cloud-native environment with persistent storage provided by Amazon EBS.

Prerequisites

Before proceeding, ensure that you have the following tools installed and properly configured:

- **Amazon EKS**: Ensure your EKS cluster is up and running.
- **Helm**: Helm is used to manage Kubernetes packages. You can install it from Helm's official website.
- **kubectl**: The Kubernetes command-line tool, configured to communicate with your EKS cluster.

Steps to Set Up Airbyte in EKS

1. Set Up EKS Cluster

First, ensure your EKS cluster is configured and ready. If you haven't set up an EKS cluster, you can do so using the `eksctl` tool:

```
eksctl create cluster --name my-cluster --region <region> --nodes 2
--node-type t3.medium
```

2. Attach EBS CSI Driver Policy to EKS Node IAM Instance Profile

To enable persistent storage using Amazon EBS, you need to attach the EBS CSI driver policy to your EKS node's IAM instance profile.

Locate the IAM role associated with your node group. You can list the instance profiles using the AWS CLI:

```
aws iam list-instance-profiles
```

Once you've identified the instance profile, attach the Amazon EBS CSI driver policy using the following command:
bash

Copy code

```
aws iam attach-role-policy --role-name <NodeInstanceRole>
--policy-arn
arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy
```

3. Add EBS CSI Driver in EKS Console

After attaching the EBS CSI policy, go to your EKS console and install the **EBS CSI Driver** using the "Add-ons" section of the EKS Console:

1. Navigate to **Amazon EKS** > **Clusters**.
2. Select your cluster.
3. Under **Add-ons**, choose **Amazon EBS CSI driver** and install it.

This driver will allow your pods to use Amazon EBS volumes for persistent storage.

4. Set Default Storage Class to `gp2`

Ensure that `gp2` is the default storage class for your EKS cluster:

```
kubectl patch storageclass gp2 -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"
}}}'
```

This step ensures that persistent volumes use the `gp2` storage class unless otherwise specified.

5. Add Airbyte Helm Chart Repository

Add the Airbyte Helm chart repository to your local Helm configuration:

```
helm repo add airbyte https://airbytehq.github.io/helm-charts
helm repo update
```

6. Create a Namespace for Airbyte

To keep things organized, create a new namespace for Airbyte:

```
kubectl create namespace airbyte
```

7. Install Airbyte Using Helm

Now, install Airbyte in the `airbyte` namespace using Helm:

```
helm install airbyte airbyte/airbyte -n airbyte --timeout 10m
```

This command will install Airbyte and set a timeout of 10 minutes to ensure the pods are ready. You can monitor the pod statuses:

```
kubectl get pods -n airbyte
```

Wait until all Airbyte pods are up and running. This may take a few minutes.

8. Update Service Type to LoadBalancer

By default, Airbyte will be installed with the `ClusterIP` service type. To make it accessible from outside the cluster, change the service type to `LoadBalancer`:

```
kubectl patch svc airbyte-webapp -n airbyte -p '{"spec": {"type": "LoadBalancer"}}'
```

9. Access Airbyte

After updating the service type, it may take a few minutes for AWS to provision an external load balancer. You can check the status and get the external IP using:

bash
Copy code

```
kubectl get svc -n airbyte
```

Once you have the external IP or DNS name of the load balancer, open it in your browser to access the Airbyte dashboard.

Summary

In this document, you've learned how to:

- Set up an EKS cluster.
- Attach the EBS CSI driver policy to your node IAM instance profile.
- Install and configure the EBS CSI driver in EKS.

- Add the Airbyte Helm chart and install Airbyte.
- Change the service type to `LoadBalancer` for external access.

By following these steps, you will have a fully functioning Airbyte instance running on EKS with persistent storage enabled.