Administering Clusters on Amazon Web Services (AWS) EKS

Integrating Cisco Container Platform with Amazon Web Services (AWS) allows you to deploy and run containerized applications across both Cisco-based on-prem environments and the AWS cloud.

This chapter contains the following topics:

Prerequisites for Configuring Clusters on AWS EKS

The prerequisites for configuring clusters on AWS EKS are as follows:

See also Adding Amazon Provider Profile.

Amazon Resource Requirements

The following table describes the default limits for the Amazon resources that you may need to increase depending on your Cisco Container Platform deployment requirements.

Note To increase the limits for a specific resource, you need to contact Amazon support.

Amazon Resource	Default Limit	Description
Network Address Translation (NAT) gateway for each AWS account	14	Each EKS cluster uses three NAT gateways. With the default setting, you are limited to four clusters.

Amazon Resource	Default Limit	Description
Amazon Virtual Private Cloud (Amazon VPC) for each AWS account	3	Each tenant cluster requires a separate Amazon VPC.
Amazon Elastic Container Service for Kubernetes (Amazon EKS) cluster for each AWS account	3	**Note** Changes to the Amazon EKS cluster limit are updated only on Thursdays.
Elastic IP address for each region	5	Each EKS cluster uses three elastic IP addresses. For more information, see Amazon VPC Limits.
Internet gateway for each region	5	Each EKS cluster uses one internet gateway.

Adding AMI Files to your Amazon Account

Cisco Container Platform generates a specific AMI (Amazon Machine Image) file with each product release. The AMI file ensures that compatible packages are available for successful tenant cluster creation.

To make the AMI file available to your Amazon account, you must submit a support case that includes your 12 digit Amazon account ID. You will be notified when the AMI is available within your Amazon account.

Creating AWS Roles

Step 1 Log in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.

Step 2 In the navigation pane of the IAM console, click **Roles**, and then click **Create role**.

Step 3 Under **Select type of trusted entity**, click **Another AWS account**.

Step 4 In the **Account ID** field, enter your **AWS Account ID**, and then click **Next**.

The AWS account number must be a trusted entity so that Cisco Container Platform can use the Role ARN during EKS cluster creation.

Step 5 Skip the screen to choose permission policies and permission boundary and click **Next**.

Step 6 Add metadata to the role by attaching tags of your choice as key–value pairs and click **Next**.

Step 7 In the **Role name** field, enter the name for the role as `k8s-ccp-user` or any other name of your choice.

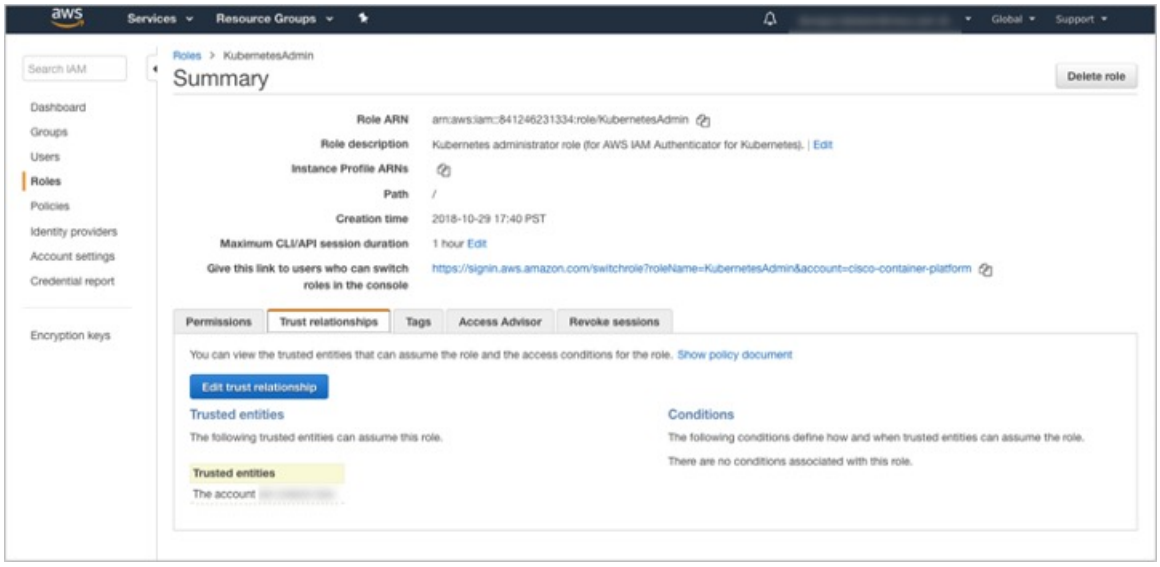
Step 8 In the **Description** field, enter a description of your choice and click **Create role**.

Step 9 After the role is created, navigate to the created role and verify the following details of the role:

a) Click the **Permissions** tab to verify that permissions are not set.

b) Click the **Trust Relationships** tab to verify that a trust relationship exists for the AWS account that you entered during creation of the Role ARN.

Figure 1: AWS Management Console-Trust Relationships Tab



Configuring Permissions for AWS Account

If the AWS provider account is not a root account, you must ensure that the account has the permissions needed to create the EKS and EC2 resources.

The following Sample aws-provider-policy.json File shows configuring the minimum permissions required for your AWS account. You need to create and import this file to configure the necessary permissions.

Sample aws-provider-policy.json File

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudformation:*",
                "elasticloadbalancing:*",
                "autoscaling:*",
                "ec2:*",
                "eks:*",
                "ecr:*",
                "ecs:*",
                "s3:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:List*",
                "iam:Get*",
                "iam:PassRole",
```

```
                "iam:AddRoleToInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile",
                "iam:CreateRole",
                "iam:CreateInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:DeleteRole",
                "iam:DeleteRolePolicy",
                "iam:AttachRolePolicy",
                "iam:DetachRolePolicy",
                "iam:PutRolePolicy",
                "iam:*AccessKey*",
                "iam:*MFA*"
            ],
            "Resource": "*"
        }
    ]
}
```

For more information on user privileges on AWS, see Minimum User Privileges on AWS.

Creating Access Keys

Access keys are required to authenticate your requests to the AWS Provider. An access key consists of two parts — an access key ID and a secret access key.

You can use the AWS IAM system in one of the following ways:

- Using a single user or personal account

  See Creating Access Keys for a Single User Account, on page 4, for creating access keys to allow access to AWS resources.

- Using a federated login account, for enterprises or corporate entities

  See Creating Access Keys for Federated Login Accounts, on page 5, for creating access keys to allow programmatic access to AWS resources.

Creating Access Keys for a Single User Account

Step 1 Log in to the AWS Management Console at https://console.aws.amazon.com.

Step 2 From the **Username** drop-down list on the top-right corner, choose **My Security Credentials**.
The **My Security Credentials** page appears.

Step 3 Expand the **Access keys** section.

Step 4 Click **Create New Access Key**.
A popup appears displaying the new access key ID and the secret access key.

Step 5 Click **Download Credentials**, and download the CSV file that contains the access keys and save it on your computer. You can use this access key while adding your Amazon provider profile. For more information, see Adding Amazon Provider Profile.

Creating Access Keys for Federated Login Accounts

Step 1 Log in to the AWS Management Console at https://console.aws.amazon.com.

Step 2 In the left pane, click **Add User** to create a new user, which Cisco Container Platform will use to login.

Step 3 In the **Set user details** section, enter a username in the **Username** field.

For example, **ccp-user**.

Step 4 In the **Select AWS access type** section, set **Programmatic Access** as the **Access Type**, and then click **Next**.

This setting provides the access key ID and secret access key for the Cisco Container Platform AWS provider.

Step 5 In the **Set Permissions** table, click **Add User to Group**, and in the lower section, select the group that you created previously (ccp-user), and then click **Next**.

Step 6 In the **Add tags** page, click **Next**.

In the next screen, the new access key ID and secret access key are displayed. Stay on this screen as it is only shown once.

Step 7 Click **Download Credentials**, and download the CSV file that contains the access keys and save it on your computer. You can use this access key while adding your Amazon provider profile. For more information, see Adding Amazon Provider Profile.

Amazon IAM Authentication

By default, the AWS IAM identity is used to authenticate and connect with clusters on EKS clusters. Cisco Container Platform uses AWS IAM Authenticator to authenticate on-prem cluster using the AWS IAM identity. This authentication provides a consistent, unified identity scheme across both on-premise and clusters on AWS EKS.

The AWS IAM Authenticator fulfills both a client and server function. On the client side, the authenticator generates, tokenizes and transmits a pre-signed URL to the server-side for identity validation. The client is a Go binary, installed on your workstation, which is transparently invoked by kubectl each time you interact with your Kubernetes cluster. The server-side is a containerized instance of AWS IAM Authenticator running as a DaemonSet on the Kubernetes master nodes. This interacts with the AWS Secure Token Service (STS) to perform identity validation. Cisco Container Platform takes care of the initial server-side configuration and provides a preconfigured `Kubeconfig` file for admin users to download.

Note You need to ensure that the AWS IAM Authenticator is available within your `$PATH` while using kubectl to interact with the clusters.

Enabling Common Identity

Within the Cisco Container Platform web interface, users are able to select a common identity scheme for clusters. After the clusters are provisioned, you can apply a shared RBAC policy.

Note The use of IAM Authentication is implicitly enabled for EKS clusters. Cisco Container Platform can map a user supplied IAM role to the EKS cluster and configuring IAM auth for on-premises clusters.

Configuring Control Plane Proxy for EKS Access

If your Control Plane VMs need proxy configuration to access the internet, specifically AWS API endpoints, you need to configure Cisco Container Platform application deployments with the proxy information.

Step 1 SSH to the Control Plane cluster master VM.

Step 2 Run the following commands to specify the proxy information:

Note You need to replace *<Proxy_URL_or_IPAddress:Port>* with the URL/IP address of your proxy server and port and the no_proxy list with a list of your internal IP addresses.

```
kubectl patch deployment kaas-api --patch
'{"spec":{"template":{"spec":{"initContainers":[{"name":"proxy-init"}],"containers":[{"name":"kaas-api","env":[{"name":"https_proxy","value":"<Proxy_URL_or_IPAddress:Port>"},{"name":"no_proxy","value":"kubernetes,localhost,127.0.0.1"}]}]}}}}'

kubectl patch deploy kaas-ccp-eks-operator --patch
'{"spec":{"template":{"spec":{"initContainers":[{"name":"proxy-init"}],"containers":[{"name":"kaas-ccp-eks-operator","env":[{"name":"https_proxy","value":"<Proxy_URL_or_IPAddress:Port>"},{"name":"no_proxy","value":"kubernetes,localhost,127.0.0.1"}]}]}}}}'

kubectl patch daemonset aws-iam-authenticator -n kube-system --patch
'{"spec":{"template":{"spec":{"initContainers":[{"name":"aws-iam-authenticator"}],"containers":[{"name":"aws-iam-authenticator","env":[{"name":"https_proxy","value":"<Proxy_URL_or_IPAddress:Port>"},{"name":"no_proxy","value":"kubernetes,localhost,127.0.0.1"}]}]}}}}'
```

Creating AWS EKS Clusters

Before you begin

- Ensure that you have added your Amazon provider profile. For more information, see Adding Amazon Provider Profile.
- Ensure that you have added the required AMI files to your account. For more information, see Adding AMI Files to your Amazon Account, on page 2.
- Ensure that you have created an AWS IAM Role for the Cisco Container Platform usage to create AWS EKS Clusters. For more information, see Creating AWS Roles, on page 2.

Step 1 In the left pane, click **Clusters**, and then click the **AWS** tab.

Step 2 Click **NEW CLUSTER**.

Step 3 In the **Basic Information** screen, enter the following information:

a) From the **INFRASTUCTURE PROVIDER** drop-down list, choose the provider related to the appropriate Amazon account.

b) From the **AWS REGION** drop-down list, choose an appropriate AWS region.

Note Not all regions support EKS. Ensure that you select a supported region. Currently, Cisco Container Platform supports the **ap-northeast-1, ap-northeast-2, ap-southeast-1, ap-southeast-2, eu-central-1, eu-north-1, eu-west-1, eu-west-2, eu-west-3, us-east-1, us-east-2,** and **us-west-2** regions.

c) In the **KUBERNETES CLUSTER NAME** field, enter a name for your cluster.
d) Click **NEXT**.

Step 4 In the **Node Configuration** screen, specify the following information:

a) From the **INSTANCE TYPE** drop-down list, choose an instance type for your cluster.
b) From the **MACHINE IMAGE** drop-down list, choose an appropriate Cisco Container Platform Amazon Machine Image (AMI) file.
To add AMI files to your Amazon account, see Adding AMI Files to your Amazon Account, on page 2.
c) In the **WORKER COUNT** field, enter an appropriate number of worker nodes.
d) In the **SSH PUBLIC KEY** drop-down field, choose an appropriate authentication key.
This field is optional. It is needed if you want to ssh to the worker nodes for troubleshooting purposes. Ensure that you use the Ed25519 or ECDSA format for the public key.

Note: As RSA and DSA are less secure formats, Cisco prevents the use of these formats.

e) In the **IAM ACCESS ROLE ARN** field, enter the Amazon Resource Name (ARN) information.

Note By default, the AWS credentials specified at the time of Amazon EKS cluster creation, that is the credentials configured in the Infrastructure Provider, are mapped to the `Kubernetes cluster-admin ClusterRole`. A default `ClusterRoleBinding` binds the credentials to the `system:masters` group, thereby granting super-user access to the holders of the IAM identity. The **IAM ACCESS ROLE ARN** field allows you to specify the ARN of an additional AWS IAM role or IAM user who is also granted administrative control of the cluster.

f) Click **NEXT**.

Step 5 In the **VPC Configuration** screen, specify the following information:

a) In the **SUBNET CIDR** field, enter a value of the overall subnet CIDR for your cluster.
b) In the **PUBLIC SUBNET CIDR** field, enter values for your cluster on separate lines.
c) In the **PRIVATE SUBNET CIDR** field, enter values for your cluster on separate lines.

Step 6 In the **Summary** screen, review the cluster information and then click **FINISH**.

Cluster creation can take up to 20 minutes. You can monitor the cluster creation status on the **Clusters** screen.

Note If you receive the Could not get token: AccessDenied error message, it indicates that the AWS account is not a trusted entity for the Role ARN.

For information on adding your AWS account as a trusted entity, see Creating AWS Roles, on page 2.

Scaling AWS EKS Clusters

You can scale EKS clusters by adding or removing worker nodes to them based on the demands of the workloads you want to run.

Step 1 In the right pane, click **EDIT**.
The **Edit Cluster** dialog box appears.

Step 2 From the **INSTANCE TYPE** drop-down list, choose an instance type for your cluster.

Step 3 From the **MACHINE IMAGE** drop-down list, choose an appropriate Cisco Container Platform Amazon Machine Image (AMI) file.
To add AMI files to your Amazon account, see Adding AMI Files to your Amazon Account, on page 2.

Step 4 In the **WORKER COUNT** field, change the number of work nodes as necessary.

Step 5 Click **UPDATE**.

Deleting AWS EKS Clusters

Before you begin

Ensure that the AWS EKS cluster that you want to delete is not currently in use, as deleting a cluster removes the containers and data associated with it.

Step 1 In the left pane, click **Clusters**, and then click the **EKS** tab.

Step 2 From the drop-down list displayed under the **ACTIONS** column, choose **Delete** for the cluster that you want to delete.

Step 3 Click **DELETE** in the confirmation dialog box.
Upon deleting an AWS EKS cluster, it takes about 10 minutes for the cluster resources to be released.