SAS® Viya® Workbench: Administrator's Guide

SAS® Documentation
September 3, 2026

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2024. *SAS® Viya® Workbench: Administrator's Guide*. Cary, NC: SAS Institute Inc.

SAS® Viya® Workbench: Administrator's Guide

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

September 2026

v_001-P1:workbenchag

Contents

1

Overview of SAS Viya Workbench

What Is SAS Viya Workbench?

SAS Viya Workbench is a developer canvas for experimentation and exploration. SAS Viya Workbench enables developers and models to work in the language of their choice (SAS, Python, or R). SAS Viya Workbench is a flexible, scalable, and efficient development environment that is on-demand, self-provisioning, and self-terminating with minimal IT support.

Where Is SAS Viya Workbench Available?

You can subscribe and deploy SAS Viya Workbench from AWS or Microsoft Marketplace.

For more information, see these resources:

- Chapter 3, “SAS Viya Workbench on AWS,” on page 15
- Chapter 2, “SAS Viya Workbench on Microsoft Marketplace,” on page 3

Status of SAS Viya Workbench

To view the health status and maintenance schedule for SAS Viya Workbench, see https://status.workbench.sas.com. To receive updates and alerts by email, click **Subscribe**.

2

SAS Viya Workbench on Microsoft Marketplace

About SAS Viya Workbench on Microsoft Marketplace

In Azure, SAS Viya Workbench is split into these components:

- A SaaS subscription is used for access to the control plane for SAS Viya Workbench.
- An Azure Managed Application enables provisioning the data plane deployment per customer within the customer's Azure tenant. SAS Viya Workbench is a publisher managed application. For more information, see Azure Managed Applications overview.

  The Service Principal enables SAS to upgrade infrastructure like Kubernetes automatically. The Azure Marketplace grants the principal owner of the SAS Azure Tenant Service permission over the managed resource group (MRG) that is created in the customer's Azure tenant. For more information about service principal accounts on Azure, see and Application and service principal objects in Microsoft Entra ID by Microsoft Ignite.

  In future releases, users will be able to scan images prior to accepting updates for SAS Viya Workbench.

To purchase and deploy SAS Viya Workbench:

- You must have purchase authority at your site to purchase SAS Viya Workbench. You might need to contact your IT administrator to purchase.
- SAS Viya Workbench uses your organization's SSO for logins. Contact your IT department for this information.

For next steps, see "Before You Begin: Requirements for SAS Viya Workbench" on page 5.

If you encounter any issues or if you have questions or feedback, please reply directly in this SAS Communities Discussion Topic. Do not include any company-specific information in your post.

About the Split Plane Configuration

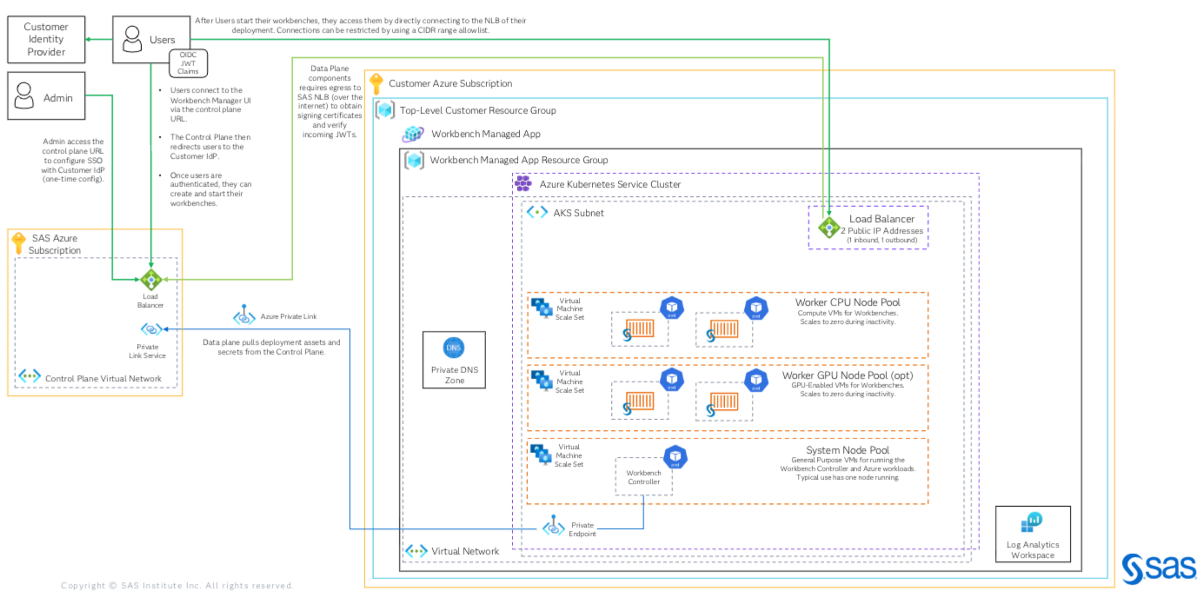
In a split plane deployment, the control plane and the data plane are deployed into separate AKS clusters. These AKS clusters can be in separate Azure accounts but they must be deployed in the same Azure region. The clusters are connected using only Azure PrivateLink to provide secure communication between them.

The control plane for SAS Viya Workbench is multi-tenant and hosted in an Azure account owned by SAS. The control plane for SAS Viya Workbench supports numerous customer data planes. The components of the data plane for SAS Viya Workbench connect to and communicate with the SAS Viya Workbench controller via Azure PrivateLink.

The control plane is half of a split plane deployment in the SAS Viya Workbench architecture, which isolates administrative system components from SAS Viya Workbench compute components. You use the control plane to administer your SAS Viya Workbench tenant and to create and manage workbenches and storage locations in their data plane.

The data plane is half of a split plane deployment in the SAS Viya Workbench architecture, which isolates the compute components that interact with and process customer data from SAS Viya Workbench system components.

Figure 2.1 *Diagram of Azure Architecture*



Before You Begin: Requirements for SAS Viya Workbench

Register Azure Resource Providers

Ensure that the required Azure resource providers are registered in the Azure subscription that will be used for the data plane deployment.

You must register these resource providers:

- Microsoft.Capacity
- Microsoft.CloudShell
- Microsoft.Compute
- Microsoft.ContainerInstance
- Microsoft.ContainerService
- Microsoft.Diagnostics
- Microsoft.Insights
- Microsoft.ManagedIdentity
- Microsoft.MarketplaceOrdering
- Microsoft.Network
- Microsoft.OperationalInsights
- Microsoft.OperationsManagement
- Microsoft.PolicyInsights
- Microsoft.Portal
- Microsoft.Quota
- Microsoft.ResourceHealth
- Microsoft.Resources
- Microsoft.SaaS
- Microsoft.Security
- Microsoft.Solutions
- Microsoft.Storage

For more information, see Azure resource providers and types from Microsoft Ignite.

Azure User Account: Subscription Owner or Contributor

Ensure that you have an Azure user account with subscription owner or contributor permissions to make a purchase. Azure operates on the Azure role-based access control (Azure RBAC) model. For more information, see Assign Azure roles using the Azure portal.

Update the Subscription to Register the Encryption Resource Provider

For the specific subscription, run the following command before deploying where the data plane will reside.

```
az feature register --namespace "Microsoft.Compute" --name "EncryptionAtHost" --subscription "<
```

If you do not run this command, you will receive an error from Terraform. For more information, see “Encryption Error from Terraform” on page 12.

OpenID Connect (OIDC)-compliant Identity Provider

Ensure use of an OpenID Connect (OIDC)-compliant Identity Provider. For more information, see Using Microsoft Entra ID (Azure Active Directory) as the Identity Provider for SAS Viya Workbench.

Sufficient Azure Quota

Ensure that you have the appropriate Azure quota for the CPUs that you want to provision. For more information, see Azure Quotas from Microsoft Ignite.

Run the Check Eligibility Tool from Microsoft

To ensure that you have everything you need, run the Check Eligibility tool from Microsoft. For more information, see Use the private offers eligibility report and Using the Microsoft Marketplace Check Eligibility tool (video).

CIDR Blocks

You must provide a CIDR block that grants users access to their deployments. IP addresses outside the CIDR block will not be able to access SAS Viya Workbench. You can use the CIDR block 0.0.0.0/0. While 0.0.0.0/0 opens your data plane from a networking perspective, access is gated by your SSO authorization.

Some products, such as ZScaler, might make it difficult to determine which CIDR block to allow because the user IP that is seen by the data plane could be the egress IP of that product.

Supported Azure Regions

Here are the regions that SAS Viya Workbench on Microsoft Marketplace supports.

Table 2.1 Supported Regions for Azure

Region for the Data Plane	Connects to the Control Plane in This Region
■ East US2 ■ East US ■ Central US ■ West US ■ Canada Central	East US2
■ West Europe ■ Germany West Central ■ North Europe ■ UK South ■ France Central ■ Sweden Central	West Europe
■ Australia East	Australia East
■ Brazil South	Brazil South
■ Japan East ■ Southeast Asia ■ Central India	Japan East
■ South Africa	South Africa

Note: Your site is responsible for ensuring that the specified control plane complies with your security practices.

Process for Onboarding SAS Viya Workbench on Microsoft Azure

Step 1: Subscribe to the SAS Viya Workbench Offer in the Azure Marketplace

1. Review the requirements for SAS Viya Workbench. For more information, see "Before You Begin: Requirements for SAS Viya Workbench" on page 5.
2. Decide on these values:
   - the compute instance type
   - the maximum number of instances
   - whether you want to use GPU compute
   - the CIDR IP ranges that should be on the allowlist
3. Log on to Azure Marketplace.
4. Select **SAS Viya Workbench** .
5. From the **Subscription** drop-down list, select the subscription where you want to deploy SAS Viya Workbench.
6. From the **Plan** drop-down list, select **Public Preview** and then click **Subscribe**.
7. In the Subscribe to SAS Viya Workbench page, complete these steps:
   - Select or create a resource group for the subscription. A resource group is a container that holds related resources for an Azure solution.
   - Under the **SaaS details** heading, enter the name for your plan.
   - Click **Review + subscribe**.
8. Review the terms of use and contact information, and click **Subscribe**.

   A message appears that your SaaS subscription is in progress.
9. To configure your account on the publisher's website, click **Configure account now**.
10. Enter the following information for your subscription:
    - the Azure region for the deployment of the data plane

Note: You must select the same region for the SaaS subscription and your SAS Viya Workbench deployment. If you select two different regions, your deployment fails.

- the primary contact's name and email
- the administrator's name and email

11 Click **Submit**.

Now, the deployment automation processes run and could take several minutes to complete.

Step 2: Review Subscription Email from SAS

After the SAS deployment automation completes, you receive an email.

The Subscription page includes the additional customer keys that are required for deploying the SAS Viya Workbench data plane. You cannot proceed with the data plane deployment without these keys.

Step 3: Subscribe to the SAS Viya Workbench Deployment Offer through Azure Marketplace

After your Subscription page updates with the customer keys, you can deploy SAS Viya Workbench.

1. In Azure Marketplace, select **SAS Viya Workbench Deployment**.
2. From the SAS Viya Workbench Deployment page, select your plan and click **Create**.
3. On the **Basics** tab of the Create SAS Viya Workbench Deployment page, specify these values:
   - Enter the name of the resource group for the deployment. You do not have to specify the same resource group that you used for the subscription.
   - From the **Region** drop-down list, select the same region that you used when creating the subscription.

     Note: You must select the same region for the subscription and the deployment. If you do not, the deployment fails.

   - Enter the application name.
4. On the **Configuration** tab, specify these values:
   - Enter your linking code. This value is on your Subscription page.

Note: This is a one-time code. After you use it to create the deployment, this value disappears from the Subscription page.

- Select the size for the AKS Worker VM. Remember to make sure that you have the quota for the VM sizes that you select. For more information, see Azure Quotas from Microsoft Ignite.
- Select **Authorized IP Ranges** and specify the CIDR range.

Note: Contact your site administrator for the value to use for the CIDR range.

- (Optional) Select **GPU support**.

5 On the **Review + create** tab, accept the terms and conditions and click **Create**.

Step 4: Review Deployment Email from SAS

After the deployment completes, a message appears on the Landing page. You will also receive an email. After the deployment completes, you can create your admin password and configure SSO.

Step 5: Set Up Your SSO Provider

After you receive an email that your deployment is complete, you need to set up your SSO provider.

1. Open the Landing page.
2. Click the link for the tenant admin one-time login.

Step 6: Share SAS Viya Workbench with Your Users

After your administrative work is complete, share the login for SAS Viya Workbench with users at your site.

Additional administrative steps are available in these topics:

- Chapter 4, “User Management,” on page 53
- Chapter 5, “Resource Management,” on page 57
- Chapter 6, “Additional Administrative Tasks,” on page 61

Troubleshooting Tips

Common Errors

If your Azure deployment does not succeed, you should receive an email letting you know about the error.

Here are some common errors:

Missing Registered Providers
: For more information, see “Register Azure Resource Providers” on page 5.

Account Does Not Have Sufficient Quota
: For more information, see Quotas overview.

Region Unavailable
: Microsoft has advised SAS to avoid US East when able due to known limitations. Unless you have existing infrastructure in East US, SAS recommends that you use East US 2 or another option.

Encryption Error from Terraform

If you do not update your subscription to register the encryption resource provider, you see the following error.


```
{
  "@level": "error",
  "@timestamp": "2026-03-04T21:52:11.139868Z",
  "@module": "terraform.ui",
  "@message": "Error: creating Kubernetes Cluster (Subscription:
\"<Subscription ID>\" Resource Group Name: \"<ENV Name>-rg\"
Kubernetes Cluster Name: \"<ENV Name>-aks\"): performing
CreateOrUpdate: unexpected status 400 (400 Bad Request)",
  "type": "diagnostic",
  "diagnostic": {
    "severity": "error",
    "address": "module.aks.azurerm_kubernetes_cluster.aks",
    "response": {
      "code": "SubscriptionNotEnabledEncryptionAtHost",
      "message": "Subscription does not enable EncryptionAtHost.",
      "details": null,
      "subcode": ""
    },
    "range": {
      "filename": "modules/azure_aks/main.tf",
```

```
        "start": {
          "line": 5,
          "column": 45
        }
      },
      "snippet": {
        "context": "resource \"azurerm_kubernetes_cluster\" \"aks\"",
        "code": "resource \"azurerm_kubernetes_cluster\" \"aks\" {"
      }
    }
  }
```


For more information, see “Update the Subscription to Register the Encryption Resource Provider” on page 7.

View the Log for Your Deployment

To view the log for your deployment:

1 Open the resource group for your managed application object.

2 Click **apiCall**.

Support and Feedback

If you encounter any issues or if you have questions or feedback, please reply directly in this SAS Communities Discussion Topic. Do not include any company-specific information in your post.

3

SAS Viya Workbench on AWS

About the Split Plane Configuration

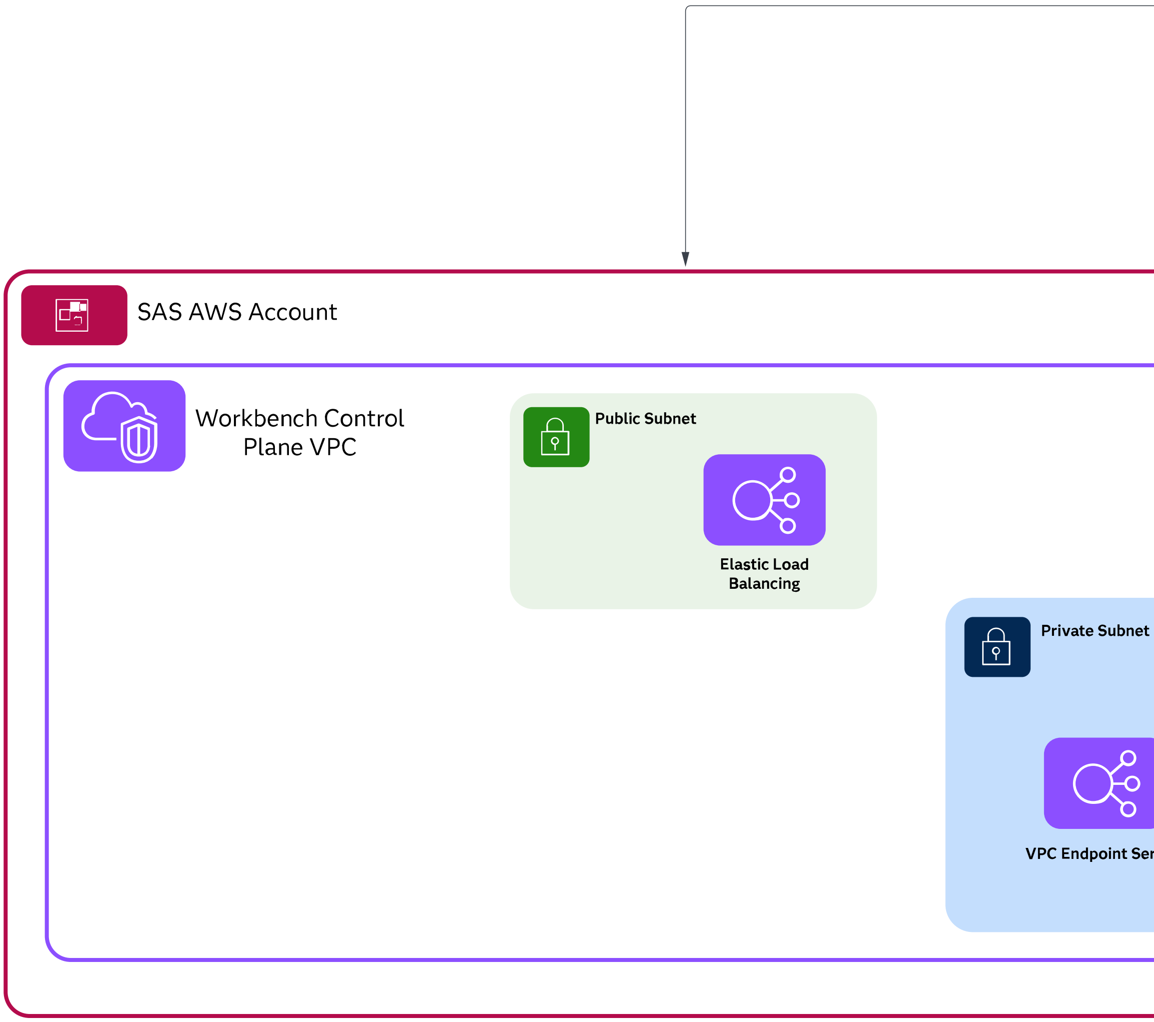
In a split plane deployment, the control plane and the data plane are deployed into separate EKS clusters. These EKS clusters can be in separate AWS accounts but they must be deployed in the same AWS region. The clusters are connected using only AWS PrivateLink to provide secure communication between them. For more information, see What is AWS PrivateLink? in *Amazon Virtual Private Cloud: User's Guide*.

The control plane for SAS Viya Workbench is multi-tenant and hosted in an AWS account owned by SAS. The control plane for SAS Viya Workbench supports numerous customer data planes. The components of the data plane for SAS Viya Workbench connect to and communicate with the SAS Viya Workbench controller via AWS PrivateLink.

The control plane is half of a split plane deployment in the SAS Viya Workbench architecture, which isolates administrative system components from SAS Viya Workbench compute components. You use the control plane to administrator your SAS Viya Workbench tenant and to create and manage workbenches and storage locations in their data plane.

The data plane is half of a split plane deployment in the SAS Viya Workbench architecture, which isolates the compute components that interact with and process customer data from SAS Viya Workbench system components.

> **IMPORTANT** The data plane for SAS Viya Workbench requires connectivity to the control plane for SAS Viya Workbench. This connectivity establishes secure connections, enables the application's functionality, and enables the administration of the product. This connectivity is not used to transport user data to the control plane.


SAS AWS Account
Workbench Control Plane VPC
Public Subnet
Elastic Load Balancing
Private Subnet
VPC Endpoint Ser

Understanding the Deployment of the Data Plane

The data plane for SAS Viya Workbench is deployed as part of an AWS Marketplace Private Offer. A Private Offer is required to enable the customer account to have the correctly licensed software sizing and pricing.

AWS Marketplace Private Offer enables SAS to deliver software using cloud native tooling and deeper integration with AWS Services and Marketplace Billing. This enables automated deployment methods that are not possible with traditional SAS offerings.

The infrastructure and application tiers for the data plane in SAS Viya Workbench are deployed using an Amazon Machine Image (AMI) created by SAS. The data plane for SAS Viya Workbench cannot be deployed into an existing EKS cluster. This AMI runs as an EC2 instance inside a VPC that is designated for a SAS Viya Workbench data plane deployment. The AMI requires permissions to deploy EKS clusters and other AWS resources for the SAS Viya Workbench data plane.

You must configure the Amazon Virtual Private Cloud (VPC) and networking resources that are needed for the data plane deployment.

For more information, see these resources:

- Amazon Machine Images (AMI) in *Amazon Elastic Compute Cloud: User Guide for Linux Instances*
- What Is Amazon VPC? in *Amazon Virtual Private Cloud: User's Guide*.

Understanding the Roles and Policies for AWS

About the EC2 Instance Provided by Amazon Machine Image (AMI)

An EC2 instance created from a SAS provided Amazon Machine Image (AMI) is used as a bootstrap host. The EC2 creates and manages a SAS Workbench data plane in a customer AWS account. The IAM model uses one role

(`SASPoiBuilderRolè) plus several attached managed policies. Each policy is scoped to a functional area:

- network and VPC operations
- EC2 and launch template operations
- IAM role/policy life cycle and pass-role operations
- EKS cluster and nodegroup life cycle
- EFS life cycle
- AWS Resource Groups life cycle
- CloudWatch Logs Write access for bootstrap logs

SASPoiBuilderRole

SASPoiBuilderRole is the primary instance role assumed by EC2 (`ec2.amazonaws.com̀) for the bootstrap host. Here is the purpose of this role:

- Act as the execution identity for all builder automation.
- Aggregate the functional managed policies.
- Include `AmazonSSMManagedInstanceCorè so that the instance can be managed through SSM without requiring SSH ingress.

Managed Policies

Table 3.1 Managed Policies for AWS

Policy Name	Description	Purpose
SASPoiBuilderVpcPolicy	Grants the networking permissions that are needed to prepare and maintain the data plane VPC infrastructure.	■ Read VPC and network inventory. ■ Create and update networking components used by the deployment ■ Manage internet and VPC wiring operations ■ Manage public addressing workflow used by infrastructure components
SASPoiBuilderEC2Policy	Grants compute and launch-template permissions for provisioning and operating cluster-related instances.	■ Read EC2 account, instance, type, and image metadata needed for placement and sizing decisions. ■ Create and manipulate volumes and attach or detach them from instances.

Policy Name	Description	Purpose
		■ Create and manage launch templates and template versions used by node groups. ■ Run instances from permitted launch templates and supporting EC2 resources. ■ Start, stop, modify, pr terminate instances and delete volumes.
SASPoiBuilderIamPolicy	Grants IAM life cycle permissions that are required to create and wire roles and policies used by EKS and related integrations.	■ Pass approved roles to AWS services (`iam:PassRolè) for cluster and node components. ■ Create, tag, update, attach, detach, and delete customer-scoped IAM roles/ policies. ■ Create required service-linked roles for EKS, autoscaling, ELB, and EFS integrations. ■ Read IAM state (get/list policy and role metadata) to make idempotent decisions.
SASPoiBuilderEksPolicy	Grants life cycle permissions to the EKS control plane and nodegroup.	■ Inspect EKS clusters, nodegroups, addons, and updates. ■ Create cluster, nodegroups, and addons. ■ Update cluster configuration and version, nodegroup configuration and version, and addon state. ■ Delete cluster or nodegroup resources during teardown workflows.
SASPoiBuilderEfsPolicy	Grants EFS permissions for persistent shared storage used by workloads.	■ Discover and inspect file systems and mount target configuration. ■ Create file systems and mount targets. ■ Tag or untag EFS resources for life cycle ownership. ■ Delete file systems and mount targets during cleanup.
SASPoiBuilderRgPolicy	Grants life cycle permissions for AWS Resource Groups.	■ Create, query, tag, update query definitions, and delete resource groups. ■ Enable grouping or tag-based organization of deployment resources.

Policy Name	Description	Purpose
SASPoiBuilderCloudWatchPolicy	Grants Write access to specific CloudWatch Logs groups that are used by bootstrap execution.	■ Create log groups and streams, and publish events ■ Provide deployment observability and troubleshooting data without broad CloudWatch scope.

How These Resources Work Together

At stack creation time, the EC2 builder instance receives `SASPoiBuilderRolè through an instance profile. The attached managed policies provide least-privilege-by-domain access, enabling the host to complete these steps:

1. Prepare networking.
2. Create IAM dependencies.
3. Provision and manage EKS components.
4. Provision EFS backing storage.
5. Group and tag resources for management.
6. Stream operational logs to CloudWatch for supportability.

Required Permissions and Resources for AWS

You must have these resources and permissions to successfully onboard SAS Viya Workbench.

Supplemental CFTs might apply additional permissions. For more information, see "Supplemental CFTs" on page 69.

Table 3.2 AWS Required Resources and Permissions

Resource Name	Unconditional Permissions
*	■ - autoscaling:DescribeAutoScalingGroups ■ - ec2:CreateTags

Resource Name	Unconditional Permissions
	- ec2:DeleteTags - ec2:DescribeAccountAttributes - ec2:DescribeAddresses - ec2:DescribeAvailabilityZones - ec2:DescribeImages - ec2:DescribeInstanceAttribute - ec2:DescribeInstanceCreditSpecifications - ec2:DescribeInstances - ec2:DescribeInstanceTypes - ec2:DescribeInternetGateways - ec2:DescribeKeyPairs - ec2:DescribeLaunchTemplates - ec2:DescribeLaunchTemplateVersions - ec2:DescribeNatGateways - ec2:DescribeNetworkAcls - ec2:DescribeNetworkInterfaceAttribute - ec2:DescribeNetworkInterfaces - ec2:DescribePrefixLists - ec2:DescribeRouteTables - ec2:DescribeSecurityGroupRules - ec2:DescribeSecurityGroups - ec2:DescribeSubnets - ec2:DescribeTags - ec2:DescribeVolumes - ec2:DescribeVpcAttribute - ec2:DescribeVpcEndpoints - ec2:DescribeVpcs - ec2:ImportKeyPair - eks:DescribeAddon - eks:DescribeAddonVersions - eks:DescribeCluster - eks:DescribeNodegroup - eks:DescribeUpdate - eks:ListNodegroups - elasticfilesystem:DescribeFileSystems - elasticfilesystem:DescribeLifecycleConfiguration

Resource Name	Unconditional Permissions
	■ - elasticfilesystem:DescribeMountTargets ■ - elasticfilesystem:DescribeMountTargetSecurityGroup
arn:aws:autoscaling:${AWS::Region}:${AWS::AccountId}:autoScalingGroup:*:autoScalingGroupName/*	- autoscaling:PutWarmPool
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:elastic-ip/*	■ - ec2:AllocateAddress ■ - ec2:AssociateAddress ■ - ec2:DisassociateAddress ■ - ec2:ReleaseAddress
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:instance/*	■ - ec2:AssociateAddress ■ - ec2:AttachVolume ■ - ec2:DetachVolume ■ - ec2:DisassociateAddress ■ - ec2:GetLaunchTemplateData ■ - ec2:ModifyVolume ■ - ec2:ReleaseAddress ■ - ec2:RunInstances
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:internet-gateway/*	■ - ec2:AttachInternetGateway ■ - ec2:DetachInternetGateway ■ - ec2:ModifyVpcAttribute
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:key-pair/*	- ec2:RunInstances
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:launch-template/*	■ - ec2:CreateLaunchTemplate ■ - ec2:CreateLaunchTemplateVersion
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:network-acl/*	■ - ec2:CreateNetworkAcl ■ - ec2:CreateNetworkAclEntry ■ - ec2:CreateNetworkInterface ■ - ec2:CreateSecurityGroup ■ - ec2:DeleteNetworkAcl

Resource Name	Unconditional Permissions
	■ - ec2:DeleteSecurityGroup
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:network-interface/*	■ - ec2:AssociateAddress ■ - ec2:CreateNetworkAcl ■ - ec2:CreateNetworkAclEntry ■ - ec2:CreateNetworkInterface ■ - ec2:CreateSecurityGroup ■ - ec2:DisassociateAddress ■ - ec2:ReleaseAddress ■ - ec2:RunInstances
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:route-table/*	■ - ec2:AssociateRouteTable ■ - ec2:CreateVpcEndpoint ■ - ec2:DisassociateRouteTable ■ - ec2:ModifyVpcEndpoint
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:security-group/*	■ - ec2:AuthorizeSecurityGroupEgress ■ - ec2:AuthorizeSecurityGroupIngress ■ - ec2:CreateNetworkAcl ■ - ec2:CreateNetworkAclEntry ■ - ec2:CreateNetworkInterface ■ - ec2:CreateSecurityGroup ■ - ec2:CreateVpcEndpoint ■ - ec2:DeleteNetworkAcl ■ - ec2:DeleteSecurityGroup ■ - ec2:ModifyVpcEndpoint ■ - ec2:RevokeSecurityGroupEgress ■ - ec2:RevokeSecurityGroupIngress ■ - ec2:RunInstances
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:subnet/*	■ - ec2:AssociateRouteTable ■ - ec2:CreateVpcEndpoint ■ - ec2:DisassociateRouteTable ■ - ec2:ModifyVpcEndpoint ■ - ec2:RunInstances
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:volume/*	■ - ec2:AttachVolume ■ - ec2:CreateVolume ■ - ec2:DetachVolume

Resource Name	Unconditional Permissions
	- ec2:ModifyVolume - ec2:RunInstances
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:vpc-endpoint/*	- ec2:CreateVpcEndpoint - ec2:ModifyVpcEndpoint
arn:aws:ec2:${AWS::Region}:${AWS::AccountId}:vpc/${SASWorkbenchDataPlaneVpcId}	- ec2:AttachInternetGateway - ec2:CreateNetworkAcl - ec2:CreateNetworkAclEntry - ec2:CreateNetworkInterface - ec2:CreateSecurityGroup - ec2:CreateVpcEndpoint - ec2:DetachInternetGateway - ec2:ModifyVpcAttribute - ec2:ModifyVpcEndpoint
arn:aws:ec2:${AWS::Region}::image/ami-*	- ec2:RunInstances
arn:aws:eks:${AWS::Region}:${AWS::AccountId}:addon/*${SASWorkbenchCustomerID}*/*/*	- eks:CreateAddon - eks:CreateCluster - eks:CreateNodegroup - eks:DeleteAddon - eks:TagResource - eks:UntagResource - eks:UpdateAddon - eks:UpdateClusterConfig - eks:UpdateClusterVersion - eks:UpdateNodegroupConfig - eks:UpdateNodegroupVersion
arn:aws:eks:${AWS::Region}:${AWS::AccountId}:cluster/*${SASWorkbenchCustomerID}*	- eks:CreateAddon - eks:CreateCluster - eks:CreateNodegroup - eks:DeleteAddon - eks:DeleteCluster - eks:DeleteNodegroup - eks:TagResource

Resource Name	Unconditional Permissions
	- eks:UntagResource - eks:UpdateAddon - eks:UpdateClusterConfig - eks:UpdateClusterVersion - eks:UpdateNodegroupConfig - eks:UpdateNodegroupVersion
arn:aws:eks:${AWS::Region}:${AWS::AccountId}:nodegroup/*${SASWorkbenchCustomerID}*/*/*	- eks:CreateAddon - eks:CreateCluster - eks:CreateNodegroup - eks:DeleteAddon - eks:DeleteCluster - eks:DeleteNodegroup - eks:TagResource - eks:UntagResource - eks:UpdateAddon - eks:UpdateClusterConfig - eks:UpdateClusterVersion - eks:UpdateNodegroupConfig - eks:UpdateNodegroupVersion
arn:aws:elasticfilesystem:${AWS::Region}:${AWS::AccountId}:file-system/*	- elasticfilesystem:CreateFileSystem - elasticfilesystem:CreateMountTarget - elasticfilesystem:DeleteFileSystem - elasticfilesystem:DeleteMountTarget - elasticfilesystem:TagResource - elasticfilesystem:UntagResource
arn:aws:iam::${AWS::AccountId}:oidc-provider/*	- iam:CreateOpenIDConnectProvider - iam:GetOpenIDConnectProvider - iam:TagOpenIDConnectProvider
arn:aws:iam::${AWS::AccountId}:policy/*${SASWorkbenchCustomerID}*	- iam:CreatePolicy - iam:CreatePolicyVersion - iam:CreateRole - iam:DeletePolicy

Resource Name	Unconditional Permissions
	- iam:DeletePolicyVersion - iam:DetachRolePolicy - iam:DetachUserPolicy - iam:GetPolicy - iam:GetPolicyVersion - iam:GetUser - iam:ListAttachedRolePolicies - iam:ListAttachedUserPolicies - iam:ListPolicyVersions - iam:ListRolePolicies - iam:TagPolicy - iam:TagRole - iam:UntagPolicy - iam:UntagRole
arn:aws:iam::${AWS::AccountId}:role/*${SASWorkbenchCustomerID}*	- iam:CreatePolicy - iam:CreatePolicyVersion - iam:CreateRole - iam:DeletePolicy - iam:DeletePolicyVersion - iam:DetachRolePolicy - iam:DetachUserPolicy - iam:GetPolicy - iam:GetPolicyVersion - iam:GetRole - iam:GetUser - iam:ListAttachedRolePolicies - iam:ListAttachedUserPolicies - iam:ListInstanceProfilesForRole - iam:ListPolicyVersions - iam:ListRolePolicies - iam:TagPolicy - iam:TagRole - iam:UntagPolicy - iam:UntagRole

Resource Name	Unconditional Permissions
arn:aws:iam::${AWS::AccountId}:role/*${SASWorkbenchCustomerID}-*-eks-node-group	- iam:PassRole
arn:aws:iam::${AWS::AccountId}:role/*${SASWorkbenchCustomerID}-ebs-csi-role	- iam:PassRole
arn:aws:iam::${AWS::AccountId}:role/*${SASWorkbenchCustomerID}-efs-csi-role	- iam:PassRole
arn:aws:iam::${AWS::AccountId}:role/*${SASWorkbenchCustomerID}-eks-cluster-*	- iam:PassRole
arn:aws:iam::${AWS::AccountId}:role/aws-service-role/autoscaling.amazonaws.com/*	- iam:GetRole
arn:aws:iam::${AWS::AccountId}:role/aws-service-role/eks-nodegroup.amazonaws.com/*	- iam:GetRole
arn:aws:iam::${AWS::AccountId}:role/aws-service-role/eks.amazonaws.com/*	- iam:GetRole
arn:aws:iam::${AWS::AccountId}:role/aws-service-role/elasticfilesystem.amazonaws.com/*	- iam:GetRole
arn:aws:iam::${AWS::AccountId}:role/aws-service-role/elasticloadbalancing.amazonaws.com/*	- iam:GetRole

Resource Name	Unconditional Permissions
arn:aws:iam::${AWS::AccountId}:user/*	- iam:GetPolicy - iam:GetPolicyVersion - iam:GetUser - iam:ListAttachedRolePolicies - iam:ListAttachedUserPolicies - iam:ListPolicyVersions - iam:ListRolePolicies
arn:aws:logs:${AWS::Region}:${AWS::AccountId}:log-group:sas-workbench-${SASWorkbenchCustomerID}-cloud-init-output:*	- logs:CreateLogGroup - logs:CreateLogStream - logs:DescribeLogStreams - logs:PutLogEvents
arn:aws:logs:${AWS::Region}:${AWS::AccountId}:log-group:sas-workbench-${SASWorkbenchCustomerID}-cloud-init:*	- logs:CreateLogGroup - logs:CreateLogStream - logs:PutLogEvents
arn:aws:logs:${AWS::Region}:${AWS::AccountId}:log-group:sas-workbench-${SASWorkbenchCustomerID}-poi:*	- logs:CreateLogGroup - logs:CreateLogStream - logs:DescribeLogStreams - logs:PutLogEvents
arn:aws:resource-groups:${AWS::Region}:${AWS::AccountId}:group/*	- resource-groups:CreateGroup - resource-groups:DeleteGroup - resource-groups:GetGroup - resource-groups:GetGroupConfiguration - resource-groups:GetGroupQuery - resource-groups:GetTags - resource-groups:Tag - resource-groups:Untag - resource-groups:UpdateGroupQuery

Overview of the VPC Configuration

Before you request access to SAS Viya Workbench, you must create a VPC for the data plane.

SAS Viya Workbench is deployed in a split plane architecture in AWS.

- The data plane is deployed within a VPC inside a customer AWS account.
- The control plane is deployed within a VPC inside an AWS account controlled by SAS.

The two VPCs are connected through AWS PrivateLink. The use of the AWS PrivateLink service requires that Service Providers create VPC Endpoint Services. SAS is the Service Provider for the SAS Viya Workbench control plane. You must create the VPC Endpoint to establish the connection from the VPC for the data plane to the control plane. SAS provides the connection information for the control plane during the tenant onboarding process after an AWS Marketplace Private Officer has been published for the customer.

> **IMPORTANT** This documentation assumes that the administrator has knowledge of AWS.

Supported AWS Regions

In AWS, you select the data plane region for your installation of SAS Viya Workbench. For more information, see AWS PrivateLink supports cross-region connectivity.

Here are the regions that SAS Viya Workbench supports. If you deploy to any of these regions, you do not need to provide the VPC endpoint name when running the CFT.

Region for the Data Plane	Connects to the Control Plane in This Region	VPC Endpoint Service Name
■ us-east-1 (N. Virginia) ■ us-east-2 (Ohio) ■ us-west-1 (N. Cali)	us-east-1 (N. Virginia)	"com.amazonaws.vpce.us-east-1.vpce-svc-00a479c21df7e4013"

Region for the Data Plane	Connects to the Control Plane in This Region	VPC Endpoint Service Name
■ us-west-2 (Oregon)		
■ eu-west-3 (Paris) ■ eu-central-1 (Frankfurt) ■ eu-west-1 (Ireland) ■ eu-west-2 (London) ■ eu-south-1 (Milan) ■ eu-north-1 (Stockholm)	eu-west-3 (Paris)	"com.amazonaws.vpce.eu-west-3.vpce-svc-0c16cca670e80c572"
■ ap-northeast-1 (Tokyo) ■ ap-northeast-2 (Seoul) ■ ap-northeast-3 (Osaka)	ap-northeast-1 (Tokyo)	"com.amazonaws.vpce.ap-northeast-1.vpce-svc-03e10fe5e1d066ce2"
sa-east-1 (Brazil)	sa-east-1 (Brazil)	"com.amazonaws.vpce.sa-east-1.vpce-svc-060a91a8a0084c557"
ap-southeast-2 (Sydney)	ap-southeast-2 (Sydney)	"com.amazonaws.vpce.ap-southeast-2.vpce-svc-08bbca399bafba205"

Note: Your site is responsible for ensuring that the specified control plane complies with your security practices.

Maintenance Windows for Database Instances in AWS

The database instances in AWS occasionally require maintenance to perform these tasks:

- update underlying hardware
- update underlying operating system (OS)
- update database engine version

For more information, see Amazon RDS maintenance window in *Amazon Relational Database Service: User Guide*.

The maintenance window for SAS Viya Workbench is every Wednesday from 3:00–3:30 a.m. standard time local to the AWS region of the control plane.

Note: The maintenance update could result in downtime for SAS Viya Workbench.

Prerequisites for SAS Viya Workbench on AWS

Components and Information Provided by SAS

SAS provides many of the required components for SAS Viya Workbench. The split plane configuration and architecture provides a turn-key environment, making it simple for you.

SAS provides a CloudFormation Template that you can use to create the VPC. For more information, see "CloudFormation Template Provided by SAS" on page 64.

Note: The CloudFormation Template provided by SAS is an example. Your deployment configuration might warrant additional changes for a successful deployment based on your user count or security standards for a VPC.

The following information is provided by SAS via email:

- your SAS Viya Workbench Customer ID

  Note: In the CloudFormation Template, use this ID as the SAS Technical Site ID.

- the vault access token for SAS Viya Workbench
- the endpoint service name for the control plane for SAS Viya Workbench

If you are missing any of this information, contact your SAS representative.

Components Provided by the Customer

You are responsible for these components:

- the VPC
- the VPC endpoint that connects to the control plane for SAS Viya Workbench
- the private subnets for EKS control plane

- the private subnets for the EKS nodes
- the public subnet for the NAT gateway
- (optional) public subnet for the optional Public Load Balancer

Before completing the CloudFormation Template, the administrator should review the access limitations for the data plane. See “Considerations for Access to the Data Plane” on page 39.

AWS DNS Resolution

As a prerequisite for SAS Viya Workbench on AWS, you must use the native AWS DNS resolution. Dynamic DNS resolution products, such as BlueCat, are not supported. The data planes in SAS Viya Workbench access the control plane via PrivateLink.

Validation scripts check that your VPC has the AWS DNS features available.

The following settings are required so that workloads in the data plane VPC can reliably use AWS-native DNS resolution (AmazonProvidedDNS / Route 53 Resolver) for AWS-managed private DNS names (such as EFS DNS names and PrivateLink-related names).

CIDR Blocks

You must provide a CIDR block that grants users access to their deployments. IP addresses outside the CIDR block will not be able to access SAS Viya Workbench. You can use the CIDR block 0.0.0.0/0. Although 0.0.0.0/0 opens your data plane from a networking perspective, access is gated by your SSO authorization.

Some products, such as ZScaler, might make it difficult to determine which CIDR block to allow because the user IP that is seen by the data plane could be the egress IP of that product.

The VPC for the Data Plane

Requirements for VPC

To satisfy the EKS VPC requirements and to use the VPC with an AWS PrivateLink, these attributes are required:

- Enables DNS host names

- Enables DNS support

The VPC must have at least a /23 IPv4 CIDR range worth of IP addresses available.

SAS provides a CloudFormation Template that creates a VPC that meets these requirements.

For more information, see these resources in the Amazon documentation:

- Amazon EKS VPC and subnet requirements and considerations in the *Amazon EKS: User's Guide*
- What is AWS PrivateLink? in *Amazon Virtual Private Cloud: User's Guide*
- DNS hostnames in *Amazon Virtual Private Cloud: User's Guide*

CAUTION
The egress for the VPC from the CloudFormation template is publicly open. Your administrator needs to configure the egress to meet the security standards at your site.

Subnets

Subnet Recommendations

SAS recommends deploying the subnets used for the SAS Viya Workbench data plane into separate AWS Availability Zones within the same AWS Region.

For more information, see these topics in the *Amazon Elastic Compute Cloud: User Guide for Linux Instances*:

- Availability Zones
- Regions
- Global Infrastructure Regions & AZs

IPv4 Address Requirements

The data planes for SAS Viya Workbench currently support only IPv4 addresses.

The VPC must be configured with a /23 IPv4 CIDR range worth of IP addresses to satisfy the requirements for the EKS cluster and the SAS Viya Workbench data plane components and coder workbenches. These workbenches run as pods inside the EKS cluster.

Note: These requirements assume that your system has less than 50 users. If you have additional users, more IP addresses might be required because each workbench session is a unique pod. To support a larger number of workbenches, allocate additional IP addresses in the private subsets for EKS. Contact SAS for more information about sizing considerations.

AWS Elastic Kubernetes Services (EKS)

Overview

The data plane uses AWS Elastic Kubernetes Services (EKS). The VPC configuration requirements use a mixture of public and private subnets and follow the recommendations for EKS. The EKS control plane is deployed into its own private subnets. The EC2 instances that make up the worker nodes for the EKS cluster are deployed into a separate set of private subnets.

The Public and Private Subnets for the Data Plane

Table 3.3 *Subnets for the Data Plane*

Subnet	Description
Private subnets for EKS control plane	Two private subnets for the EKS control plane. Each subnet should have at least a /28 IPv4 CIDR range worth of IPv4 addresses. For more information, see Amazon EKS VPC and subet requirements and considerations in *Amazon EKS: User's Guide*.
Private subnet for EKS nodes	A private subnet that is dedicated for the EKS nodes. This subnet holds all the EKS nodes and pods. Placing the EKS nodes in a subnet that is separate from the EKS control plane provides greater security and network isolation. ■ The private subnet for the EKS nodes should have at least a /24 IPv4 CIDR range worth of IP addresses. ■ The private subnet for the EKS nodes must have the required AWS tag for internal load balancers. `kubernetes.io/role/internal-elb: 1`
Public subnet for NAT Gateway	At least one public subnet must be included in the VPC for use by the public NAT Gateway. Each public subnet should have at least a /28 IPv4 CIDR range worth of IP addresses.
(Optional) Public subnet for load balancers	An optional public subnet that is dedicated for deploying the public load balancer. ■ The public subnet should have at least a /28 IPv4 CIDR range worth of IP addresses.

Subnet	Description
	■ The public subnet must have the required AWS tag for internet-facing load balancers. `kubernetes.io/role/elb: 1`

NAT Gateway

The NAT Gateway deployed in the VPC must be a Public NAT Gateway, which means that the gateway includes an ENI deployed into a public subnet within the VPC. The NAT Gateway enables egress traffic from launched workbenches in the data plane. This is required for the workbenches to enforce authorization via the SSO that is configured in the SAS Viya Workbench control plane at the end of data plane onboarding.

The NAT Gateway must be associated with all private subnets used by the EKS control plane and with the private subnets where the worker nodes run workbenches. The NAT Gateway does not need to be associated with the public subnets.

The VPC Endpoints

VPC Endpoints for AWS Services

In order for a SAS Viya Workbench data plane to communicate with AWS services securely, the VPC for the data plane must have VPC Endpoints.

These VPC Endpoints for the AWS services can be automatically provisioned when you deploy SAS Viya Workbench.

- `s3 - com.amazonaws.`*`region`*`.s3`

  Note: The s3 endpoint must be a Gateway type.

- `logs - com.amazonaws.`*`region`*`.logs`
- `ECR (api) - com.amazonaws.`*`region`*`.ecr.api`
- `autoscaling - com.amazonaws.`*`region`*`.autoscaling`
- `elasticloadbalancing - com.amazonaws.`*`region`*`.elasticloadbalancing`
- `sts - com.amazonaws.`*`region`*`.sts`

- dkr - com.amazonaws.*region*.ecr.dkr
- ec2 - com.amazonaws.*region*.ec2

For the *region* value, enter the AWS region that you want to use. For more information, see “Supported AWS Regions” on page 30.

> **IMPORTANT** If you already have VPC Endpoints in your VPC for any of these AWS services, you must create the VPC Endpoints for the rest of the AWS services. If you create any of these VPC Endpoints, you must select **false** when the CloudFormation Template runs, or the installation fails.

For more information, see these topics in the AWS documentation:

- What are VPC endpoints?
- What is AWS PrivateLink?

Prerequisites for VPC Endpoint for the Control Plane

> **TIP** The CloudFormation Template provided by SAS creates the VPC endpoint. If you use this template, you do not need to create the VPC endpoint that connects to the control plane. For more information, see “CloudFormation Template Provided by SAS” on page 64.

To create the VPC Endpoint to the control plane, you need the following information:

Table 3.4 Requirements for VPC Configuration

Requirement	Description
VPC ID	This ID should be for the VPC where the data plane is deployed.
VPC Endpoint Service Name	SAS provides the service name to enable the VPC Endpoint to connect to the control plane. The service name is in the email that you received from SAS about SAS Viya Workbench.
Private Subnet ID	The ID of private subnet that is used for EKS worker nodes.
Security Group IDs	A security group must be assigned to the VPC endpoint that allows all traffic to itself. This is used by the EC2 instance to communicate with the VPC endpoint. An example of a security

Requirement	Description
	group that meets this requirement is the default security group that is created when a VPC is created.
Tags	You must add these AWS tags to the VPC Endpoint: ■ Key: `Name` Value: `sas-private-endpoint-workbench` ■ Key: `workbench.sas.com/access-enabled` Value: `1` **Note:** These AWS tags are required.

Create the VPC Endpoint for the SAS Viya Workbench Control Plane

1 Copy this command.

```
# https://docs.aws.amazon.com/cli/latest/reference/ec2/create-vpc-
endpoint.html
  aws ec2 create-vpc-endpoint \
    --vpc-id "Customer_VPC_ID" \
    --service-name "SAS_Endpoint_Service_Name" \
    --service-region "SAS_Control_Plane_Region" \
    --vpc-endpoint-type "Interface" \
    --private-dns-enabled \
    --subnet-ids "Customer_Worker_Node_Subnet"  \
    --security-group-ids "Customer_Security_Group_ID" \
    --tag-specifications "ResourceType=vpc-
endpoint,Tags=[{Key=Name,Value=sas-private-endpoint-workbench},
{Key=workbench.sas.com/access-enabled,Value=1}]"
```

2 Specify these values for your site:

- *Customer_VPC_ID*
- *SAS_Endpoint_Service_Name*
- *SAS_Control_Plane_Region*

Note: To determine the value of *SAS_Control_Plane_Region*, review the matrix in “Supported AWS Regions” on page 30 to find the control plane that is associated with the region for your data plane.

- *Customer_Worker_Node_Subnet*
- *Customer_Security_Group_ID*

3 Run this command in AWS.

Considerations for Access to the Data Plane

End-User Access to Workbenches

The end user can access the workbenches in the data plane via the AWSNetworkLoadBalancer (NLB), which is created during the deployment and is configured using the values from your SAS offer.

The administrator must choose whether to make the data plane NLB accessible on a public subnet (which is public facing) or a private subnet (which is accessible only to the VPC).

- In the CloudFormation Template, the subnet type is specified in SASWorkbenchDataPlaneLoadBalancerSubnetType field.
- The administrator should use their corporate strategy to decide whether to use a private or public subnet.
  - Choosing a private subnet requires all end-user access to originate from within the VPC, which requires a more complex AWS networking configuration. You must enable this configuration. It is not enabled by SAS. If you are uncertain about which IP address to use, `0.0.0.0/0` is used at the start. You can update this list after the data plane is onboarded and before any data is loaded into the environment. Using `0.0.0.0/0` for the IP allowlist is secured by the authentication access requirement and does not indicate that anyone can access the data plane.
  - You can use an open subnet because all endpoints require authentication with the configured SSO provider for the data plane. Regardless of the networking configurations, authenticated access is required. Selecting a limited set of IP address, instead of using `0.0.0.0/0`, provides an additional layer of security.
- This choice is independent of the type of access to EKS or the Dataplane Builder EC2 instance.

When completing the CloudFormation Template, the administrator must specify the CIDR ranges in the allowlist to access the load balancer.

- In the CloudFormation Template, the CIDR range is specified in SASWorkbenchDataPlaneLoadBalancerCIDRRange field.
- SAS sets the customer's network to public by default or set the default allowlist for CIDR to `0.0.0.0/0` (which is open to any incoming connection).

Access for the EKS Cluster

By default, private access to the EKS cluster in the VPC is enabled.

Data Plane Access to the Dataplane Builder EC2 Instance

- Private access is available only via SSM.
- Public access and SSH are not supported.
- The customer administrator needs access to the Dataplane Builder EC2 instance when working with SAS support for debugging, infrastructure updates, and more.

Connecting to External Services

If your workbench is connecting external endpoints, downloading external code on the deployment, or exporting data, your deployment requires external dependencies. These dependencies must follow product usage policies and disclosure. For more information, see Product usage policies in the AWS documentation.

Collecting or Exporting Customer Data

If you are collecting or exporting customer data while using SAS Viya Workbench on AWS, you must follow the customer information policies specified by AWS. For more information, see Customer information policies in the AWS documentation.

Onboarding Requirements for SAS Viya Workbench

CAUTION
When deploying SAS Viya Workbench, you should create new AWS accounts. Permissions in the CFT enable SAS Viya Workbench to create administrative roles, users, or groups.

1. After SAS publishes the private offer to the AWS Customer Marketplace, you receive an email from SAS that contains the following information:
   - SAS Viya Workbench Customer ID
   - the URL for the SAS Viya Workbench control plane
   - the URL that SAS created for the data plane
   - URL for the Private Offer in AWS Marketplace
   - the vault access token for the SAS Viya Workbench control plane
   - VPC Endpoint Service Name for the SAS Viya Workbench control plane
2. The administrator at your site must add the URL for the SAS Viya Workbench control plane and the URL for the customer data plane to their corporate allowlist, if appropriate.
3. Create a VPC in your AWS account for the data plane. For more information, see "The VPC for the Data Plane" on page 33.

   Note: SAS provides an example CloudFormation Template that you can use. For more information, see "CloudFormation Template Provided by SAS" on page 64.
4. Accept the Private Offer in the AWS Marketplace.
5. Complete the CloudFormation Template. SAS builds all deployment assets.
6. The Amazon Machine Image (AMI) starts and waits while SAS builds your deployment assets.

   The AMI creates a set of log groups in AWS CloudWatch. Review these logs for the process of the deployment of the data plane and to see any errors.
7. In the CloudFormation Template, you specified the administrator at your site. This individual receives an email from SAS with the following information:

- a link that takes the customer administrator to the SAS Viya Workbench Organization Administration page where you can connect your SAS Viya Workbench tenant to your identity provider.

  Note: This link is valid for only 72 hours and can be used only once.

- the URL for SAS Viya Workbench
- user name of the administrator

Upgrading Kubernetes Cluster

Upgrading Kubernetes Cluster from 1.30 to 1.31

About the Upgrade from 1.30 to 1.31

To run SAS Viya Workbench, you must upgrade your Kubernetes cluster from 1.30 to 1.31 for an existing data plane deployment. The upgrade process involves updating configuration files, modifying AWS permissions, and applying infrastructure changes in the SAS Viya Workbench platform.

Prerequisites

- An existing data plane deployed with Kubernetes 1.30
- Active control plane infrastructure
- AWS administrator access to the subscription
- Access to the data plane AMI instance

Updated CFT for Kubernetes 1.31 Upgrade

To complete the Kubernetes steps, you need to download and save this CFT to a `upgrade-k8s-cft.yaml` file.

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
```

```
   AttachPolicyToBuilderRole:
      Type: "AWS::IAM::Policy"
      Properties:
         PolicyName: "sas-workbench-update-iam-policies"
         Roles:
           - !Sub "${BuilderRole}"
         PolicyDocument:
            Version: "2012-10-17"
            Statement:
               - Effect: "Allow"
                 Action:
                   - eks:UpdateClusterVersion
                   - eks:UpdateNodegroupVersion
                 Resource: "*"
Parameters:
   BuilderRole:
      Type: String
      Description: "The full name of the IAM role ending with -
dataplane-builder-role. The prefix
        is the name of your already-created Workbench CFT stack."
```

Step 1: Update Amazon Web Services Permissions

To deploy the permission updates:

1. In the AWS Console, navigate to CloudFormation.
2. Select **Create Stack** ⇨ **With new resources (standard)**. Click **Next**.
3. Select **Upload a template file** and select the downloaded template. (See "Updated CFT for Kubernetes 1.31 Upgrade" on page 42.)
4. Enter a unique stack name (for example, `k8supdate131`) .
5. In the **BuilderRole** field, enter the following command:

   `<dataplane-cft-name>-dataplane-builder-role`

   Click **Next**.

   > **TIP** To find the name of the builder role, see the **Resource** tab for your data plane CFT stack.

6. Select the acknowledgment check box. Click **Next** ⇨ **Create**.

Before moving to the next step, wait for the stack to complete successfully.

Step 2: Prepare the Data Plane Amazon Machine Image

Locate the Data Plane Instance

1. Navigate to the AWS Console.
2. Go to CloudFormation and select your data plane CFT stack.
3. Open the **Resource** tab and locate the EC2 instance for the AMI. Record the instance ID for your connection.

Connect to the Amazon Image Instance

1. Connect to the EC2 instance using AWS Session Manager.
2. After you are connected, use this command to switch to the ec2-user account:

   ```
   sudo su ec2-user
   ```
3. Enter this command to go to the home directory:

   ```
   cd ~
   ```
4. Enter this command to move to the system-poi directory:

   ```
   cd system-poi
   ```

Step 3: Update Kubernetes Configuration

1. Enter this command to open the data plane system configuration file:

   ```
   vi config/<dataplane-name>-system.yaml
   ```
2. In the yaml file, locate the Kubernetes version and change the value from `1.30` to `1.31`.
3. Save the file and close the code editor.
4. To confirm the changes, enter the following code at the command line:

   ```
   less config/<dataplane-name>-system.yaml
   ```

   In the output, the Kubernetes version should be **`1.31`**.

Step 4: Apply Infrastructure Changes

1. To update the credentials, run the following command:

   ```
   /data-plane-setup/bin/fetch-credentials "${POI_HOME}/.private"
   ```
2. To start the development container, run the following command:

```
poi-exec
```

3 Run the Kubernetes upgrade command from the AMI instance:

```
stagezero aws infrastructure-apply
statelock state upsert
```

> **TIP** To monitor the progress of the update, see the operation logs in the CloudWatch log stream.

4 To check the Kubernetes version before and after the upgrade, run the following command:

```
kubectl version
```

If the Kubernetes version is now 1.31, the upgrade has completed successfully.

Upgrading Kubernetes Cluster from 1.31 to 1.32

About the Upgrade from 1.31 to 1.32

Use these steps to upgrade a Kubernetes cluster from version 1.31 to 1.32 in an existing data plane deployment. The upgrade process involves updating the AWS CFT and applying the infrastructure.

Prerequisites

- An existing data plane deployed with Kubernetes 1.31
- AWS access to the AMI EC2 instance
- AWS access to update CFTs

Updated CFT for Kubernetes 1.32 Upgrade

To complete the Kubernetes steps, you need to download and save this CFT to a `upgrade-k8s-cft.yaml` file.

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  AttachPolicyToBuilderRole:
    Type: "AWS::IAM::Policy"
    Properties:
      PolicyName: "2025-11-sas-workbench-update-iam-policies"
```

```
          Roles:
            - !Sub "${BuilderRole}"
          PolicyDocument:
             Version: "2012-10-17"
             Statement:
                - Effect: "Allow"
                  Action:
                    - "iam:CreatePolicyVersion"
                    - "iam:DeletePolicyVersion"
                    - "eks:UpdateAddon"
                    - "eks:UpdateClusterVersion"
                    - "eks:UpdateNodegroupVersion"
                    - "iam:UpdateOpenIDConnectProviderThumbprint"
                    - "ec2:ModifyLaunchTemplate"
                    - "resource-groups:UpdateGroupQuery"
                  Resource: "*"
Parameters:
   BuilderRole:
      Type: String
      Description: "The full name of the IAM role ending with -
dataplane-builder-role. The prefix
        is the name of your already created Workbench CFT stack."
```

Step 1: Update Amazon Web Services Permissions

To deploy the permission updates:

1 In the AWS Console, navigate to CloudFormation.

2 Select **Create Stack** ⇨ **With new resources (standard)**. Click **Next**.

3 On the Create stack page, complete these steps:

   a For the **Prepare template** option, select **Choose an existing template**.

   b For the **Template source** option, select **Upload a template file** and select template for the upgrade to 1.32. (See “Updated CFT for Kubernetes 1.32 Upgrade” on page 45.)

4 On the Specify stack details page, complete these steps:

   a Enter a unique stack name (for example, `k8supdate132)`.

   b In the **BuilderRole** field, enter the following command:

   ```
   <dataplane-cft-name>-dataplane-builder-role
   ```

   Click **Next**.

   > **TIP** To find the name of the builder role, see the **Resource** tab for your data plane CFT stack.

5 Select the acknowledgment check box. Click **Next** ⇨ **Create**.

Before moving to the next step, wait for the stack to complete successfully.

Step 2: Upgrade Kubernetes in the EC2 Data Plane Builder Instance

1. In the AWS console, select **EC2 service**.
2. Locate your dataplane-builder EC2 instance.

   Note: This instance ends with the suffix "-dataplane-builder."

3. Right-click the name of the instance and select **Connect**.
4. On the next screen, select the **Session Manager** tab and select **Connect**. A shell opens to your EC2 instance.
5. After you are logged in, run the following command:

   ```
   sudo su ec2-user
   ```

6. Run the following command:

   ```
   /data-plane-setup/bin/fetch-credentials "${POI_HOME}/.private"
   ```

7. Run the following command:

   ```
   poi-exec
   ```

   Note: If you are prompted for an `oh-my-zsh` upgrade, enter `n` for No.

8. Run the folllowing command:

   ```
   sed -i.bak "s/kubernetes_version: \"1.31\"/kubernetes_version:
   \"1.32\"/" ${IAC_VAR_SYSTEM_CONFIG}
   ```

9. Run the following command:

   ```
   sz aws infrastructure-apply
   ```

   Note: The infrastructure-apply script takes about 20-30 minutes to complete. After Infrastructure-apply runs, the cluster is upgraded and in a usable state.

10. Run the following command:

    ```
    statelock state upsert
    ```

Upgrading Kubernetes Cluster from 1.32 to 1.33

About the Upgrade from 1.32 to 1.33

Use these steps to upgrade a Kubernetes cluster from version 1.32 to 1.33 in an existing data plane deployment. The upgrade process involves updating the AWS CFT and applying the infrastructure.

Prerequisites

- An existing data plane deployed with Kubernetes 1.32
- AWS access to the AMI EC2 instance
- AWS access to update CFTs

Updated CFT for Kubernetes 1.33 Upgrade

To complete the Kubernetes steps, you need to download and save this CFT to a `upgrade-k8s-cft.yaml` file.

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
   AttachPolicyToBuilderRole:
      Type: "AWS::IAM::Policy"
      Properties:
         PolicyName: "sas-workbench-update-iam-policies"
         Roles:
           - !Sub "${BuilderRole}"
         PolicyDocument:
            Version: "2012-10-17"
            Statement:
               - Effect: "Allow"
                 Action:
                   - "iam:CreatePolicyVersion"
                   - "iam:DeletePolicyVersion"
                   - "iam:ListInstanceProfilesForRole"
                   - "iam:DeleteRole"
                   - "eks:UpdateAddon"
                   - "eks:UpdateClusterVersion"
                   - "eks:UpdateNodegroupVersion"
                   - "eks:UpdateNodegroupConfig"
                   - "eks:ListNodegroups"
                   - "iam:UpdateOpenIDConnectProviderThumbprint"
                   - "ec2:ModifyLaunchTemplate"
```

```
                    - "resource-groups:UpdateGroupQuery"
                  Resource: "*"
Parameters:
   BuilderRole:
      Type: String
      Description: "The full name of the IAM role ending with -
dataplane-builder-role. The prefix is the name of your already-created
Workbench CFT stack."
```

Step 1: Update Amazon Web Services Permissions

To deploy the permission updates:

1. In the AWS Console, navigate to CloudFormation.
2. Select **Create Stack** ⇨ **With new resources (standard)**. Click **Next**.
3. On the Create stack page, complete these steps:
   a. For the **Prepare template** option, select **Choose an existing template**.
   b. For the **Template source** option, select **Upload a template file** and select template for the upgrade to 1.33. (See “Updated CFT for Kubernetes 1.33 Upgrade” on page 48.)
4. On the Specify stack details page, complete these steps:
   a. Enter a unique stack name (for example, `k8supdate133`) .
   b. In the **BuilderRole** field, enter the following command:

      ```
      <dataplane-cft-name>-dataplane-builder-role
      ```

      Click **Next**.

      > **TIP** To find the name of the builder role, see the **Resource** tab for your data plane CFT stack.

5. Select the acknowledgment check box. Click **Next** ⇨ **Create**.

Before moving to the next step, wait for the stack to complete successfully.

Step 2: Upgrade Kubernetes in the EC2 Data Plane Builder Instance

Note: You must complete steps 1-7 every time that a user enters the EC2 instance.

1. In the AWS console, select **EC2 service**.
2. Locate your dataplane-builder EC2 instance.

Note: This instance ends with the suffix "-dataplane-builder."

3 Right-click the name of the instance and select **Connect**.

4 On the next screen, select the **Session Manager** tab and select **Connect**. A shell opens to your EC2 instance.

5 After you are logged in, run the following command:

```
sudo su ec2-user
```

6 Run the following command:

```
/data-plane-setup/bin/fetch-credentials "${POI_HOME}/.private"
```

7 Run the following command:

```
poi-exec
```

Note: If you are prompted for an `oh-my-zsh` upgrade, enter `n` for No.

8 Run

```
git pull
cd config
git stash
git pull
cd ..
```

9 Run the following command:

```
sz aws infrastructure-apply
```

Note: The infrastructure-apply script takes about 20-30 minutes to complete. After Infrastructure-apply runs, the cluster is upgraded and in a usable state.

10 Run the following command:

```
statelock state upsert
```

Upgrading Kubernetes Cluster from 1.33 to 1.34

About the Upgrade from 1.33 to 1.34

Use these steps to upgrade a Kubernetes cluster from version 1.33 to 1.34 in an existing data plane deployment.

Prerequisites

- An existing data plane deployed with Kubernetes 1.33
- AWS access to the AMI EC2 instance
- AWS access to update CFTs

Upgrade Kubernetes in the EC2 Data Plane Builder Instance

Note: You must complete steps 1-7 every time that a user enters the EC2 instance.

1. In the AWS console, select **EC2 service**.
2. Locate your dataplane-builder EC2 instance.

   Note: This instance ends with the suffix "-dataplane-builder."

3. Right-click the name of the instance and select **Connect**.
4. On the next screen, select the **Session Manager** tab and select **Connect**. A shell opens to your EC2 instance.
5. After you are logged in, run the following command:

   ```
   sudo su ec2-user
   ```

6. Run the following command:

   ```
   /data-plane-setup/bin/fetch-credentials "${POI_HOME}/.private"
   ```

7. Run the following command:

   ```
   poi-exec
   ```

   Note: If you are prompted for an `oh-my-zsh` upgrade, enter `n` for No.

8. Run

   ```
   git pull
   cd config
   git stash
   git pull
   cd ..
   ```

9. Run the following command:

   ```
   sz aws infrastructure-apply
   ```

Note: The infrastructure-apply script takes about 20-30 minutes to complete. After infrastructure-apply runs, the cluster is upgraded and in a usable state.

10 Run the following command:

```
statelock state upsert
```

Schedule Data Plane Cluster Shutdown and Start-Up

To save costs on AWS, you might want to schedule times to shutdown and start up the data plane cluster on a cron schedule. SAS provides a script that you can customize to meet the needs of your site. The script enables to you to specify the cluster and the times for shutdown and start-up. The enablement script must be run from the EC2 Data Plane Builder that was deployed as part of the AWS private offer.

Here are some important considerations when using this script:

- The script creates an AWS Lambda that starts and stops the cluster based on the specified schedule.
- Your configuration for node scaling is preserved between the start-up and shutdown.
- You can update the script to meet your needs. Rerun this script to apply your changes.
- Workbenches cannot be used when a cluster is shutdown.
- When a cluster is shutdown, a banner appears in the user interface to let you know that the data plane is not available.

For more information, see the README file in GitHub: Schedule Cluster Shutdown and Start Up for AWS Infrastructure.

4 User Management

Understanding User Roles

Roles Available in SAS Viya Workbench

In SAS Viya Workbench, you have the following roles:

Table 4.1 Roles in SAS Viya Workbench

Role	Description
Workbench Admin	Enables the user to perform administration tasks.
Workbench User	Enables the user to access SAS Viya Workbench, create workbenches, and create storage locations.
Organization Admin	Enables the user to perform these administrative tasks for the organization: ■ manager identity providers

Role	Description
	■ invite new members; add and delete SAS Viya Workbench users; manage permissions for users ■ add or delete single-sign on connections or domains for the organization

Note: SAS Data Maker roles do not impact SAS Viya Workbench and can be ignored.

Manage User Roles

To manage your user permissions, click . The **Users** page lists all users and their assigned roles. To add or remove roles for a user, click .

Managing Your SSO Connections

Create an SSO Connection

1 In the navigation pane for SAS Viya Workbench, click .

2 Click the **Authentication** tab. Then click **Request New Connection**.

3 Specify the name of the connection.

4 Complete the steps in the Auth0 IDP setup wizard.

Edit an SSO Connection

1 In the navigation pane for SAS Viya Workbench, click .

2 Click the **Authentication** tab. Then click **Request Edit**.

Note: You can edit only IDPs that you created.

3 Use the Auth0 IDP setup wizard to make your changes.

Delete an SSO Connection

To remove a connection, click **Delete**. If you do not have permission to delete an IDP connection, the **Delete** button is disabled.

Working with Domains

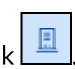
Domains direct users to your organization based on the user's email address. Domains are not required with Auth0.

To add a domain, you must use the Auth0 IDP wizard. You can use SAS Viya Workbench to remove domains.

5

Resource Management

Managing Workbenches

To view a list of workbenches at your site, click .

From the **Workbenches** page, you can complete these tasks:

- view the status of a workbench, such as running or stopped
- view whether a workbench has an error. To view the specific error for a workbench, click **Error**. From the pop-up, you can then choose to view the details of the error in the Activity log.
- start, stop, or delete a workbench. Select the check box of the workbench from the table and then select the operation that you want to perform: **Start**, **Stop**, or **Delete**.
- edit the settings for an individual workbench. You can modify the workbench name, the maximum compute, the storage location, or the mounting folder.

Use the filter to subset the workbenches by compute size or by time period.

Reviewing the Activity Log

As an administrator, you can view all the activity for workbenches at your site.

From the Home page, click .

By default, this activity log shows events for all workbenches, all storage locations, and all time periods. Use the filter to subset the results.

- Status values include normal, warnings, and errors.
- Event types include:
  - **App**
  - **Ingress** shows the event for the APISIX ingress controller.
  - **Pod** shows the events associated with a pod in a workbench. Examples of events include stopping the VScode container, stopping the SAS compute container, successfully assigning a storage location to a pod, and starting a container.
  - **Storage** shows the storage locations created, updated, or deleted during the selected time period. Use the **Storage** drop-down list to view the activity for a specific storage location.
  - **Workbench** shows the activity for all active workbenches. Use the **Workbench** drop-down list to view the activity for a specific workbench.

Configuring SAS Viya Workbench per User

Configuration Settings for a User

By default, each user has the same configuration settings for storage limits, number of concurrent workbenches, and time-out. As an administrator, you can set these configuration settings for individual users.

To specify the configuration settings for a specific user:

1 In the left navigation pane, click .

2 From the Workbench Administration page, click the **Configuration** tab.

3 From the Configurations pane, select a user. You define users on the Users page. For more information, see “Understanding User Roles” on page 53.

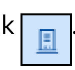
4 For the selected user, specify new values for one or more of these options:

- **Overall limit (GB)** specifies the amount of storage per user. The default value is `50`.
- **EBS limit (GB)** specifies the limit for EBS. The default value is `40`.
- **EFS limit (GB)** specifies the limit for EFS. By default, no limit is specified.
- **Concurrent workbenches limit** specifies the number of workbenches that a user can have at one time. The default value is `3`.
- **Default inactivity timeout (minutes)** specifies how long a workbench must be idle before automatically shutting down. The default value is `30`.
- **Maximum inactivity timeout (minutes)** specifies the maximum time-out that a user can specify for a workbench. The default value is `45`.

5 Click **Save** for the new configuration settings to take effect.

Reset Configuration Settings

To reset a user’s configuration settings to the default values:

1 In the left navigation pane, click .

2 From the Workbench Administration page, click the **Configuration** tab.

3 From the Configurations pane, select a user. You can define users from the Users page. For more information, see “Understanding User Roles” on page 53.

4 To return a customized setting to the default value, click **Reset** by that option. You can reset one value or all the values. If an option does not have the **Reset** option, the default value is used.

5 Click **Save** for the new configuration settings to take effect.

6

Additional Administrative Tasks

Telemetry Level for OpenVSCode Server

To protect your data, SAS Viya Workbench runs the OpenVSCode server with telemetry disabled.

However, as a user, you might see the telemetry level set to `all`. VSCode has multiple locations for storage settings: remote (or machine) settings; user settings; and workspace settings. All of these settings are configured in different locations. If you specify a machine setting and that setting is also valid in user settings, VSCode ignores the setting. When VSCode is accessed in a browser, the user settings are stored in a local, browser-specific location. In this case, SAS Viya Workbench cannot change the values of the local settings. SAS Viya Workbench can control the machine settings and the workspace settings.

Because telemetry is considered a user setting, SAS Viya Workbench cannot set the telemetry level for users. However, in VSCode, the telemetry setting can be disabled in another way besides the user setting. Using this other method, SAS Viya Workbench has disabled telemetry for OpenVSCode server, first-party extensions, and participating third-party extensions.

Appendix 1

CloudFormation Template and VPC Validation Script Provided by SAS

Overview

You can use the CloudFormation Template and VPC validation script provided by SAS to ensure a seamless deployment of the SAS Viya Workbench data plane in AWS.

- The VPC validation script can determine whether the VPC is configured correctly for a SAS Viya Workbench data plane.
- A CloudFormation Template is provided that can create a VPC and subnets that meet the requirements for a SAS Viya Workbench data plane.

Note: The CloudFormation Template provided by SAS is an example. Your deployment configuration might warrant additional changes for a successful deployment based on your user count or security standards for a VPC.

CloudFormation Template Provided by SAS

You can save this code as a yaml file and use it to create your VPC that meets the requirements for SAS Viya Workbench. This template requires an input parameter for the VPC Endpoint Service name. SAS emails this name to you after the AWS Private Officer is accepted.

TIP If you use this file, you do not need to create the VPC endpoint as described in "Prerequisites for VPC Endpoint for the Control Plane" on page 37.

```
AWSTemplateFormatVersion: '2010-09-09'
Description: CloudFormation Template to create VPC that adheres to SAS
Viya Workbench data plane requirements

Parameters:
  Prefix:
    Default: sas
    Type: String
    Description: The prefix to add to your resource names that will be
created.
  ControlPlaneVPCEndpointServiceName:
    Type: String
    Description: The VPC Endpoint Service Name for the Controlplane
VPC Endpoint.
  ControlPlaneServiceRegion:
    Type: String
    Description: The AWS Region where the Controlplane VPC Endpoint
Service is located.

Resources:
  WorkbenchVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.0.0.0/16
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Sub ${Prefix}-workbench-dataplane-vpc
```

```
##########################################################
# Public Subnets
##########################################################
PublicSubnet1a:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref WorkbenchVPC
    CidrBlock: 10.0.1.0/24
    AvailabilityZone: !Sub ${AWS::Region}a
    MapPublicIpOnLaunch: true
    Tags:
      - Key: Name
        Value: !Sub ${Prefix}-workbench-dataplane-public-subnet-1a
      - Key: kubernetes.io/role/elb
        Value: "1"

PublicSubnet1b:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref WorkbenchVPC
    CidrBlock: 10.0.4.0/24
    AvailabilityZone: !Sub ${AWS::Region}b
    MapPublicIpOnLaunch: true
    Tags:
      - Key: Name
        Value: !Sub ${Prefix}-workbench-dataplane-public-subnet-1b
      - Key: kubernetes.io/role/elb
        Value: "1"

##########################################################
# NAT Gateway and attachment to
# PublicSubnet1a and InternetGateway
##########################################################
NatGateway:
  Type: AWS::EC2::NatGateway
  Properties:
    AllocationId: !GetAtt NATGatewayEIP.AllocationId
    ConnectivityType: public
    SubnetId: !Ref PublicSubnet1a
    Tags:
      - Key: Name
        Value: !Sub ${Prefix}-workbench-dataplane-nat-gateway

NATGatewayEIP:
  Type: AWS::EC2::EIP
  Properties:
    Domain: vpc
    Tags:
      - Key: Name
        Value: !Sub ${Prefix}-workbench-dataplane-eip

InternetGateway:
  Type: AWS::EC2::InternetGateway
  Properties:
    Tags:
```

```
        - Key: Name
          Value: !Sub ${Prefix}-workbench-dataplane-internet-gateway

  AttachGateway:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      VpcId: !Ref WorkbenchVPC
      InternetGatewayId: !Ref InternetGateway

  ##########################################################
  # Private Subnets for EKS Control plane
  ##########################################################
  PrivateEksControlplaneSubnet1a:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref WorkbenchVPC
      CidrBlock: 10.0.2.0/24
      AvailabilityZone: !Sub ${AWS::Region}a
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
          Value: !Sub ${Prefix}-workbench-dataplane-private-eks-
subnet-1a

  PrivateEksControlplaneSubnet1b:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref WorkbenchVPC
      CidrBlock: 10.0.5.0/24
      AvailabilityZone: !Sub ${AWS::Region}b
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
          Value: !Sub ${Prefix}-workbench-dataplane-private-eks-
subnet-1b

  ##########################################################
  # Private Subnets for EKS Worker nodes(EC2)
  ##########################################################
  PrivateWorkbenchSubnet1a:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref WorkbenchVPC
      CidrBlock: 10.0.64.0/18
      AvailabilityZone: !Sub ${AWS::Region}a
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
          Value: !Sub ${Prefix}-workbench-dataplane-private-worker-
node-subnet-1a
        - Key: kubernetes.io/role/internal-elb
          Value: "1"

  ##########################################################
  # PublicRouteTable configuration
  ##########################################################
```

```
PublicRouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId: !Ref WorkbenchVPC
    Tags:
      - Key: Name
        Value: !Sub ${Prefix}-workbench-dataplane-rtb-public

PublicRoute:
  Type: AWS::EC2::Route
  DependsOn: AttachGateway
  Properties:
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway

PublicSubnet1aRouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    SubnetId: !Ref PublicSubnet1a
    RouteTableId: !Ref PublicRouteTable

PublicSubnet1bRouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    SubnetId: !Ref PublicSubnet1b
    RouteTableId: !Ref PublicRouteTable

##########################################################
# PrivateRouteTable configuration
##########################################################
PrivateRouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId: !Ref WorkbenchVPC
    Tags:
      - Key: Name
        Value: !Sub ${Prefix}-workbench-dataplane-rtb-private

RouteNatGateway:
  Type: AWS::EC2::Route
  Properties:
      RouteTableId: !Ref PrivateRouteTable
      DestinationCidrBlock: '0.0.0.0/0'
      NatGatewayId: !Ref NatGateway

PrivateEksControlplaneSubnet1aRouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    SubnetId: !Ref PrivateEksControlplaneSubnet1a
    RouteTableId: !Ref PrivateRouteTable

PrivateWorkbenchSubnet1aRouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    SubnetId: !Ref PrivateWorkbenchSubnet1a
```

```
      RouteTableId: !Ref PrivateRouteTable

  PrivateEksControlplaneSubnet1bRouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId: !Ref PrivateEksControlplaneSubnet1b
      RouteTableId: !Ref PrivateRouteTable

  ##########################################################
  # Security Group
  ##########################################################
  WorkbenchControlplaneAccessSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Security Group for SAS Viya Workbench Data
Plane
      VpcId: !Ref WorkbenchVPC
      GroupName: !Sub ${Prefix}-workbench-controlplane-access-sg
      Tags:
        - Key: Name
          Value: !Sub ${Prefix}-workbench-controlplane-access-sg

  WorkbenchControlplaneAccessSecurityGroupEgress:
    Type: AWS::EC2::SecurityGroupEgress
    Properties:
      GroupId: !Ref WorkbenchControlplaneAccessSecurityGroup
      IpProtocol: -1
      DestinationSecurityGroupId: !Ref
WorkbenchControlplaneAccessSecurityGroup
      Description: Allow all egress traffic to resources in the same
security group

  WorkbenchControlplaneAccessSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !Ref WorkbenchControlplaneAccessSecurityGroup
      IpProtocol: -1
      SourceSecurityGroupId: !Ref
WorkbenchControlplaneAccessSecurityGroup
      Description: Allow all ingress traffic from resources in the
same security group

  ##########################################################
  # VPC Endpoint to Controlplane
  ##########################################################
  WorkbenchVpcEndpointToControlplane:
    Type: AWS::EC2::VPCEndpoint
    DependsOn:
      - WorkbenchControlplaneAccessSecurityGroup
      - WorkbenchControlplaneAccessSecurityGroupIngress
      - WorkbenchControlplaneAccessSecurityGroupEgress
    Properties:
      IpAddressType: ipv4
      PrivateDnsEnabled: true
      SecurityGroupIds:
        - !Ref WorkbenchControlplaneAccessSecurityGroup
```

```
      ServiceName: !Sub ${ControlPlaneVPCEndpointServiceName}
      ServiceRegion: !Sub ${ControlPlaneServiceRegion}
      SubnetIds:
        - !Ref PrivateWorkbenchSubnet1a
      Tags:
        - Key: Name
          Value: sas-private-endpoint-workbench
        - Key: workbench.sas.com/access-enabled
          Value: "1"
      VpcEndpointType: Interface
      VpcId: !Ref WorkbenchVPC

Outputs:
  VPCId:
    Description: VPC Id
    Value: !Ref WorkbenchVPC
  PublicSubnet1aId:
    Description: Public Subnet Id 1a
    Value: !Ref PublicSubnet1a
  PublicSubnet1bId:
    Description: Public Subnet Id 1b
    Value: !Ref PublicSubnet1b
  PrivateEksControlplaneSubnet1aId:
    Description: Private EKS Controlplane Subnet 1a Id
    Value: !Ref PrivateEksControlplaneSubnet1a
  PrivateEksControlplaneSubnet1bId:
    Description: Private EKS Controlplane Subnet 1b Id
    Value: !Ref PrivateEksControlplaneSubnet1b
  PrivateWorkbenchSubnet1aId:
    Description: Private Worker Node Subnet 1a Id
    Value: !Ref PrivateWorkbenchSubnet1a
  WorkbenchVpcEndpointToControlplaneId:
    Description: VPC Endpoint to Controlplane Id
    Value: !Ref WorkbenchVpcEndpointToControlplane
  WorkbenchControlplaneAccessSecurityGroupId:
    Description: Workbench Security Group Id
    Value: !Ref WorkbenchControlplaneAccessSecurityGroup
  WorkbenchControlplaneAccessSecurityGroupName:
    Description: Workbench Security Group Name
    Value: !Sub ${Prefix}-workbench-controlplane-access-sg
```

Note: If the size of the PrivateWorkbenchSubnet1a CIDR block is too large, reduce the value of `CidrBlock` to `10.0.3.0/24` to accommodate about 50 users.

Supplemental CFTs

The supplemental CFTs apply policies at a broader level than the primary CFT.

- CFT for SAS IAM Policy on page 70 applies these policies:

```
- Effect: "Allow"
                  Action:
                    - "iam:CreatePolicyVersion"
                    - "iam:DeletePolicyVersion"
                    - "eks:UpdateAddon"
                  Resource: "*"
```

- CFT for Kubernetes 1.31 Upgrade on page 42

```
 - Effect: "Allow"
   Action:
      - eks:UpdateClusterVersion
      - eks:UpdateNodegroupVersion
   Resource: "*"
```

- CFT for Kubernetes 1.32 Upgrade on page 45

```
- Effect: "Allow"
                  Action:
                    - "iam:CreatePolicyVersion"
                    - "iam:DeletePolicyVersion"
                    - "eks:UpdateAddon"
                    - "eks:UpdateClusterVersion"
                    - "eks:UpdateNodegroupVersion"
                    - "iam:UpdateOpenIDConnectProviderThumbprint"
                    - "ec2:ModifyLaunchTemplate"
                    - "resource-groups:UpdateGroupQuery"
                  Resource: "*"
```

Update SAS IAM Policy

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
   AttachPolicyToBuilderRole:
      Type: "AWS::IAM::Policy"
      Properties:
         PolicyName: "sas-workbench-update-iam-policies"
         Roles:
           - !Sub "${BuilderRole}"
         PolicyDocument:
            Version: "2012-10-17"
            Statement:
               - Effect: "Allow"
                 Action:
                   - "iam:CreatePolicyVersion"
                   - "iam:DeletePolicyVersion"
                   - "eks:UpdateAddon"
                 Resource: "*"
Parameters:
   BuilderRole:
      Type: String
```

```
      Description: "The full name of the IAM role ending with -
dataplane-builder-role. The prefix is the name of your already-created
Workbench CFT stack."
```

Getting Started

Configure AWS CLI

Option 1: Using an Access Key and Secret Access Key

Set the AWS_PROFILE using the following command. You can set this environment variable to anything you would like.

````
```sh
export AWS_PROFILE="awesome_profile"
```
````

Use AWS Access Key and Secret Key to authenticate with the AWS CLI.

````
```sh
aws configure set aws_access_key_id "XXXXXXXX"
aws configure set aws_secret_access_key "XXXXXXXX"
```
````

Option 2: Using SSO (IAM Identity Center)

To use SSO to authenticate with the AWS CLI, see Configure the AWS CLI to use AWS IAM Identity Center authentication in *AWS Command Line Interface: User Guide for Version 2*.

After completing the wizard, set the AWS_PROFILE environment variable to the profile displayed during the SSO process.

````
```sh
export AWS_PROFILE="AdministratorAccess-12345678910"
```
````

Configure AWS Region

Define the AWS region. The region must be the same as the region for the VPC, or the validation script will not run as expected.

````
```sh
export AWS_DEFAULT_REGION="us-east-1"
```
````

The VPC Validation Script

VPC Validation Script Provided by SAS

Copy the following code and save it as **./validate-vpc.sh**.

```
#!/usr/bin/env bash
set -e

# Define colors for formatting
GREEN='\033[0;32m'
RED='\033[91m'
NC='\033[0m' # No Color

valid_eks_control_plane_private_subnets="[]"
valid_sas_viya_workbench_private_subnets="[]"
valid_sas_viya_workbench_public_subnets="[]"

function validate_vpc() {
    vpc_id="${1}"

    # Check if CidrBlock of VPC is at least /23
    echo "Validate VPC: ${vpc_id}"
    vpc="$(aws ec2 describe-vpcs --filters "Name=vpc-id,Values=$
{vpc_id}" --query "Vpcs[0]")"
    cidr_block="$(echo "${vpc}" | jq -r '.CidrBlock')"
    tenancy="$(echo "${vpc}" | jq -r '.InstanceTenancy')"

    if [[ $cidr_block == "null" ]]; then
        echo -e "  ${RED} VPC ${vpc_id} unable to be found${NC}"
        echo -e "Review the following section in the Administrator's
Guide: The VPC for the Data Plane${NC}"
        exit 1
    fi

    subnet_mask="${cidr_block##*/}"
```

```
    if [[ "${subnet_mask}" -le 23 ]]; then
        echo -e "  ${GREEN} • IPv4 CIDR subnet mask: ${subnet_mask}$
{NC}"
    else
        echo -e "  ${RED} • IPv4 CIDR subnet mask: ${subnet_mask}"
        echo -e "  • The CidrBlock subnet mask should be less than or
equal to 23${NC}"
        echo -e "Review the following section in the Administrator's
Guide: The VPC for the Data Plane"
        exit 1
    fi

    if [[ "${tenancy}" == "default" ]]; then
        echo -e "  ${GREEN} • Tenancy: ${tenancy}${NC}"
    else
        echo " • Tenancy: ${tenancy}"
        echo -e "  ${RED} • The Tenancy should be set to default${NC}"
        exit 1
    fi

    # Check if DNS Support is enabled for VPC
    enable_dns_support="$(aws ec2 describe-vpc-attribute --vpc-id "$
{vpc_id}" --attribute enableDnsSupport | jq '.EnableDnsSupport.Value')"
    if [[ "${enable_dns_support}" == "true" ]]; then
        echo -e "  ${GREEN} • Enabled DNS Support: $
{enable_dns_support}${NC}"
    else
        echo " • Enabled DNS Support: ${enable_dns_support}"
        echo -e "  ${RED} • EnableDnsSupport value should be true${NC}"
        echo -e "Review the following section in the Administrator's
Guide: The VPC for the Data Plane${NC}"
        exit 1
    fi

    # Check if DNS Hostnames are enabled for VPC
    enable_dns_hostnames="$(aws ec2 describe-vpc-attribute --vpc-id "$
{vpc_id}" --attribute enableDnsHostnames | jq
'.EnableDnsHostnames.Value')"
    if [[ "${enable_dns_hostnames}" == "true" ]]; then
        echo -e "  ${GREEN} • Enabled DNS Hostname: $
{enable_dns_hostnames}${NC}"
    else
        echo " • Enabled DNS Hostname: ${enable_dns_hostnames}"
        echo -e "  ${RED} • EnableDnsHostnames value should be true$
{NC}"
        echo -e "Review the following section in the Administrator's
Guide: The VPC for the Data Plane${NC}"
        exit 1
    fi
    echo
}

function validate_subnets() {
    vpc_id="${1}"
    subnets="$(aws ec2 describe-subnets --filters "Name=vpc-id,Values=$
{vpc_id}" --query "Subnets[*].
```

```
{SubnetId:SubnetId,AvailabilityZoneId:AvailabilityZoneId,CidrBlock:Cidr
Block,Tags:Tags}")"
    private_subnets="[]"
    public_subnets="[]"

    # Iterate subnets and organize into public and private
    # If a subnet is tied to a route table, which contains a route
linked to an internet gateway (igw), it is public
    # Docs: https://docs.aws.amazon.com/vpc/latest/userguide/configure-
subnets.html
    echo "Validate Subnets"
    while read -r subnet; do
        subnet_id=$(echo "${subnet}" | jq -r '.SubnetId')
        is_public="$(aws ec2 describe-route-tables --filter
"Name=association.subnet-id,Values=${subnet_id}" --query
"RouteTables[].Routes[]" | jq '.[] | select(.GatewayId | strings |
startswith("igw")) | length > 0')"

        if [[ -n "${is_public}" ]]; then
            public_subnets="$(echo "${public_subnets}" | jq --argjson
"subnet" "${subnet}" '. + [$subnet]')"
        else
            private_subnets="$(echo "${private_subnets}" | jq --
argjson "subnet" "${subnet}" '. + [$subnet]')"
        fi
    done <<<"$(echo "${subnets}" | jq -c '.[]')"
    private_subnets_count="$(echo ${private_subnets} | jq -r 'length')"
    public_subnets_count="$(echo ${public_subnets} | jq -r 'length')"

    # We need at least 3 private subnets, to handle all possible
deployments
    if [[ "${private_subnets_count}" -ge 3 ]]; then
        echo -e "  ${GREEN} • Number of private subnets: $
{private_subnets_count}${NC}"
    else
        echo " • Number of private subnets: ${private_subnets_count}"
        echo -e "  ${RED} • VPC has fewer than 3 private subnets ($
{private_subnets_count})${NC}"
        echo -e "Review the following section in the Administrator's
Guide: The VPC for the Data Plane${NC}"
        exit 1
    fi

    # We need at least 1 public subnets, to handle all possible
deployments
    if [[ "${public_subnets_count}" -ge 1 ]]; then
        echo -e "  ${GREEN} • Number of public subnets: $
{public_subnets_count}${NC}"
    else
        echo " • Number of public subnets: ${public_subnets_count}"
        echo -e "  ${RED} • VPC has less than 1 public subnets ($
{public_subnets_count})${NC}"
        echo -e "Review the following section in the Administrator's
Guide: The VPC for the Data Plane${NC}"
        exit 1
    fi
```

```
    echo

    # Validate private subnets
    while read -r subnet; do
        cidr_block=$(echo "${subnet}" | jq -r '.CidrBlock')
        subnet_mask="${cidr_block##*/}"
        tags=$(echo "${subnet}" | jq -r '.Tags')
        subnet_id=$(echo "${subnet}" | jq -r '.SubnetId')
        internal_elb_tag_exists="$(echo "${tags}" | jq '.[] |
select(.Key == "kubernetes.io/role/internal-elb" and .Value == "1") |
length > 0')"

        # Private subnets are either for the EKS Control Plane Private
Subnets or SAS Viya Workbench Private Subnets
        echo "Validate Private Subnet: ${subnet_id}"
        if [ -n "${internal_elb_tag_exists}" ]; then
            echo -e "  ${GREEN} • Expected Tag: kubernetes.io/role/
internal-elb: 1${NC}"
            # Validate subnet for the SAS Viya Workbench Private
Subnet, which requires that CidrBlock is at least /24 and the tags
kubernetes.io/role/internal-elb:1
            if [[ "${subnet_mask}" -le 24 ]]; then
                echo -e "  ${GREEN} • IPv4 CIDR subnet mask: $
{subnet_mask}${NC}"
                valid_sas_viya_workbench_private_subnets="$(echo "$
{valid_sas_viya_workbench_private_subnets}" | jq --argjson "subnet" "$
{subnet}" '. + [$subnet]')"
            else
                echo " • IPv4 CIDR subnet mask: ${subnet_mask}"
                echo -e "  ${RED} • The CidrBlock subnet mask should
be less than or equal to 24${NC}"
                echo -e "Review the following section in the
Administrator's Guide: The VPC for the Data Plane"
            fi
        else
            # Validate subnet for the EKS Control Plane Private
Subnets, which requires proper AZ
            # EKS Control Plane Private Subnet must not have an
availability zone that matches use1-az3, usw1-az2, or cac1-az3
            # https://docs.aws.amazon.com/eks/latest/userguide/
network_reqs.html#network-requirements-subnets
            availability_zone_id="$(echo "${subnet}" | jq
'.AvailabilityZoneId')"
            if [[ "${availability_zone_id}" != "use1-az3" ]] && [[ "$
{availability_zone_id}" != "usw1-az2" ]] && [[ "$
{availability_zone_id}" != "cac1-az3" ]]; then
                echo -e "  ${GREEN} • The private subnet is located in
a supported Availability Zone (${availability_zone_id})${NC}"
                # Validate subnet for the EKS Control Plane Private
Subnets, which requires that CidrBlock is at least /28
                if [[ "${subnet_mask}" -le 28 ]]; then
                    echo -e "  ${GREEN} • IPv4 CIDR subnet mask: $
{subnet_mask}${NC}"
                    valid_eks_control_plane_private_subnets="$(echo "$
{valid_eks_control_plane_private_subnets}" | jq --argjson "subnet" "$
{subnet}" '. + [$subnet]')"
```

```
                else
                    echo " • IPv4 CIDR subnet mask: ${subnet_mask}"
                    echo -e "  ${RED} • The CidrBlock subnet mask 
should be less than or equal to 28${NC}"
                    echo -e "Review the following section in the 
Administrator's Guide: The VPC for the Data Plane${NC}"
                fi
            else
                echo "The private subnet ${subnet_id} is located in an 
unsupported Availability Zone: ${availability_zone_id}"
                echo -e "  ${RED} • Unsupported Availability Zones: 
(use1-az3, usw1-az2, or cac1-az3)${NC}"
                echo "    • Hint: Review the AWS documentation for EKS 
network configuration https://docs.aws.amazon.com/eks/latest/userguide/
network_reqs.html#network-requirements-subnets"
            fi
        fi
    done <<<"$(echo "${private_subnets}" | jq -c '.[]')"

    # Validate public subnets
    echo
    while read -r subnet; do
        cidr_block=$(echo "${subnet}" | jq -r '.CidrBlock')
        subnet_mask="${cidr_block##*/}"
        tags=$(echo "${subnet}" | jq -r '.Tags')
        subnet_id=$(echo "${subnet}" | jq -r '.SubnetId')
        elb_tag_exists="$(echo "${tags}" | jq '.[] | select(.Key == 
"kubernetes.io/role/elb" and .Value == "1") | length > 0')"

        # Public subnets are for the SAS Viya Workbench Public Subnets
        # Validate subnet for the SAS Viya Workbench Public Subnets, 
which requires that CidrBlock is at least /28 and the tag 
kubernetes.io/role/elb:1
        echo "Validate Public Subnet: ${subnet_id}"
        if [[ "${subnet_mask}" -le 28 ]]; then
            if [[ -n "${elb_tag_exists}" ]]; then
                echo -e "  ${GREEN} • IPv4 CIDR subnet mask: $
{subnet_mask}${NC}"
                echo -e "  ${GREEN} • Expected Tag: kubernetes.io/role/
elb: 1${NC}"

                valid_sas_viya_workbench_public_subnets="$(echo "$
{valid_sas_viya_workbench_public_subnets}" | jq --argjson "subnet" "$
{subnet}" '. + [$subnet]')"
            elif [[ -z "${elb_tag_exists}" ]]; then
                echo -e "  ${RED} • Missing tag: kubernetes.io/role/
elb: 1${NC}"
                echo -e "            Review the following section in 
the Administrator's Guide: The VPC for the Data Plane${NC}"
            fi
        else
            echo " • IPv4 CIDR subnet mask: ${subnet_mask}"
            echo -e "  ${RED} • The CidrBlock subnet mask should be 
less than or equal to 28"
            echo -e "Review the following section in the 
Administrator's Guide: The VPC for the Data Plane${NC}"
```

```
        fi
    done <<<"$(echo "${public_subnets}" | jq -c '.[]')"

    valid_eks_control_plane_private_subnets_count="$(echo $
{valid_eks_control_plane_private_subnets} | jq -r 'length')"
    valid_sas_viya_workbench_private_subnets_count="$(echo $
{valid_sas_viya_workbench_private_subnets} | jq -r 'length')"
    valid_sas_viya_workbench_public_subnets_count="$(echo $
{valid_sas_viya_workbench_public_subnets} | jq -r 'length')"

    echo
    # We need at least 2 EKS Control Plane Private Subnets
    echo "Number of valid EKS Control Plane Private Subnets: $
{valid_eks_control_plane_private_subnets_count}"
    if [[ "${valid_eks_control_plane_private_subnets_count}" -lt 2 ]];
then
        echo
        echo -e "  ${RED} • Expected 2 valid EKS Control Plane Private
Subnets"
        echo -e "Review the following section in the Administrator's
Guide: The VPC for the Data Plane${NC}"
        exit 1
    fi

    # We need at least 1 SAS Viya Workbench Private Subnet
    echo "Number of valid SAS Viya Workbench Private Subnets: $
{valid_sas_viya_workbench_private_subnets_count}"
    if [[ "${valid_sas_viya_workbench_private_subnets_count}" -lt
1 ]]; then
        echo
        echo -e "  ${RED} • Expected at least 1 valid SAS Viya
Workbench Private Subnet${NC}"
        echo "    • Hint: Check if you have set tag: kubernetes.io/
role/internal-elb: 1"
        echo "      Review the following section in the
Administrator's Guide: The VPC for the Data Plane"
        exit 1
    fi

    # We need at least 1 SAS Viya Workbench Public Subnets
    echo "Number of valid SAS Viya Workbench Public Subnets: $
{valid_sas_viya_workbench_public_subnets_count}"
    if [[ "${valid_sas_viya_workbench_public_subnets_count}" -lt 1 ]];
then
        echo
        echo -e "  ${RED} • Expected at least 1 valid SAS Viya
Workbench Public Subnets"
        echo -e "           Review the following section in the
Administrator's Guide: The VPC for the Data Plane${NC}"
        exit 1
    fi
}

function validate_endpoint() {
    vpc_id="${1}"
```

```
    # VPC Endpoint must have the name sas-private-endpoint-workbench
and tag workbench.sas.com/access-enabled:1
    vpc_endpoint="$(aws ec2 describe-vpc-endpoints --filters Name=vpc-
id,Values="${vpc_id}" Name=tag:Name,Values="sas-private-endpoint-
workbench" Name=tag:workbench.sas.com/access-enabled,Values="1" --
query "VpcEndpoints[0]")"

    if [[ "${vpc_endpoint}" != "null" ]]; then
        vpc_endpoint_id="$(echo "${vpc_endpoint}" | jq
'.VpcEndpointId')"
        echo
        echo "VPC Endpoint: ${vpc_endpoint_id}"
        echo -e "  ${GREEN} • Name: sas-private-endpoint-workbench$
{NC}"
        echo -e "  ${GREEN} • Expected Tag: workbench.sas.com/access-
enabled: 1${NC}"

        private_dns_enabled_valid="$(echo "${vpc_endpoint}" | jq
'.PrivateDnsEnabled')"
        if [[ "${private_dns_enabled_valid}" == "true" ]]; then
            echo -e "  ${GREEN} • Private DNS Enabled: $
{enable_dns_support}${NC}"
        else
            echo " • Private DNS Enabled: ${enable_dns_support}"
            echo -e "  ${RED} • PrivateDnsEnabled value should be true"
            echo -e "Review the following section in the
Administrator's Guide: The VPC for the Data Plane${NC}"
        fi
        endpoint_type="$(echo "${vpc_endpoint}" | jq -r
'.VpcEndpointType')"
        if [[ "${endpoint_type}" == "Interface" ]]; then
            echo -e "  ${GREEN} • VpcEndpointType: ${endpoint_type}$
{NC}"
        else
            echo " • VpcEndpointType: ${endpoint_type}"
            echo -e "  ${RED} • VpcEndpointType value sould be
Interface${NC}"
            echo -e "Review the following section in the
Administrator's Guide: The VPC for the Data Plane${NC}"
        fi

    else
        echo -e "  ${RED} No VPC endpoint exists in VPC(${vpc_id})
that meets the requirements to connect to the SAS Viya Workbench
control plane"
        echo -e "A VPC endpoint is required that has the name sas-
private-endpoint-workbench and tag workbench.sas.com/access-enabled: 1"
        echo -e "Review the following section in the Administrator's
Guide: The VPC Endpoints${NC}"
        exit 1
    fi
}
function validate_nat_gateway() {
    vpc_id="${1}"
    echo
```

```
    nat_gateway="$(aws ec2 describe-nat-gateways --filter Name=vpc-
id,Values="${vpc_id}" --query "NatGateways[0]")"
    if [[ "${nat_gateway}" != "null" ]]; then
        nat_gateway_id="$(echo "${nat_gateway}" | jq '.NatGatewayId')"
        echo "Validate NatGateway: ${nat_gateway_id}"

        state="$(echo "${nat_gateway}" | jq -r '.State')"
        if [[ "${state}" == "available" ]]; then
            echo -e "  ${GREEN} • State: ${state}${NC}"
        else
            echo " • State: ${state}"
            echo -e "  ${RED} • State value should be available${NC}"
        fi

        elastic_ip_id="$(echo "${nat_gateway}" | jq -r
'.NatGatewayAddresses[0].AllocationId')"
        elastic_ip="$(aws ec2 describe-addresses --filters
Name=allocation-id,Values="${elastic_ip_id}" --query "Addresses[0]")"
        if [[ "${elastic_ip}" != "null" ]]; then
            echo -e "  ${GREEN} • Associated Elastic IP: $
{elastic_ip_id}${NC}"
        else
            echo -e "  ${RED} • There is no Associated Elastic IP with
NatGateway${NC}"
        fi
    else
        echo -e "  ${RED} No NAT Gateway exists in VPC ${vpc_id}${NC}"
        exit 1
    fi
}
function validate_route_table() {
    vpc_id="${1}"
    echo
    route_tables="$(aws ec2 describe-route-tables --filters Name=vpc-
id,Values="${vpc_id}" --query "RouteTables")"
    while read -r route_table; do
        route_table_id=$(echo "${route_table}" | jq -r '.RouteTableId')
        igw_routes=$(echo "${route_table}" | jq '.Routes[] |
select(.GatewayId | strings | startswith("igw"))')
        nat_routes=$(echo "${route_table}" | jq '.Routes[] |
select(.NatGatewayId | strings | startswith("nat"))')

        if [[ "${route_table_id}" == "null" ]]; then
            echo -e "  ${RED} RouteTable does not exist${NC}"
            exit 1
        fi

        echo "Validate RouteTable: ${route_table_id}"
        if [[ "${igw_routes}" != "" ]]; then
            igw_id=$(echo "${igw_routes}" | jq -r '.GatewayId')
            echo "Associated Internet Gateway: ${igw_id}"
            while read -r association; do
                subnet_id=$(echo "${association}" | jq -r '.SubnetId')
                associated_public_subnet=$(echo "$
{valid_sas_viya_workbench_public_subnets}" | jq -r --arg "subnet_id" "$
{subnet_id}" '.[] | select(.SubnetId == $subnet_id)')
```

```
                if [[ "${associated_public_subnet}" != "" ]]; then
                    echo -e "  ${GREEN} • Internet Gateway is 
associated to public subnet ${subnet_id}${NC}"
                else
                    echo -e "  ${RED} • Internet Gateway (${igw_id}) 
is not associated to any public subnet${NC}"
                fi
            done <<<"$(echo "${route_table}" | jq -c 
'.Associations[]')"

        elif [[ "${nat_routes}" != "" ]]; then
            nat_id=$(echo "${nat_routes}" | jq -r '.NatGatewayId')
            echo "Associated NAT Gateway: ${nat_id}"
            while read -r association; do
                subnet_id=$(echo "${association}" | jq -r '.SubnetId')
                if [[ -z "${subnet_id}" ]] || [[ "${subnet_id}" == 
"null" ]]; then
                    continue
                fi
                associated_eks_subnet=$(echo "$
{valid_eks_control_plane_private_subnets}" | jq -r --arg "subnet_id" "$
{subnet_id}" '.[] | select(.SubnetId == $subnet_id)')
                associated_private_subnet=$(echo "$
{valid_sas_viya_workbench_private_subnets}" | jq -r --arg "subnet_id" 
"${subnet_id}" '.[] | select(.SubnetId == $subnet_id)')
                if [[ "${associated_eks_subnet}" != "" ]]; then
                    echo -e "  ${GREEN} • NAT Gateway is associated to 
private EKS subnet: ${subnet_id}${NC}"
                elif [[ "${associated_private_subnet}" != "" ]]; then
                    echo -e "  ${GREEN} • NAT Gateway is associated to 
private worker node subnet: ${subnet_id}${NC}"
                else
                    echo -e "  ${RED} • NAT Gateway (${nat_id}) does 
not have route associations with to any private subnet${NC}"
                fi
            done <<<"$(echo "${route_table}" | jq -c 
'.Associations[]')"
        else
            echo -e " • No Associated NatGateway or InternetGateway to 
RouteTable ${route_table_id}"
        fi
    done <<<"$(echo "${route_tables}" | jq -c '.[]')"
}

# Entrypoint
if [ -z "$1" ]; then
    echo "Please provide the VPC ID of the VPC you want to validate."
    echo "Usage: validate-vpc <VPC ID>"
else
    vpc_id="${1}"

    validate_vpc "${vpc_id}"
    validate_subnets "${vpc_id}"
    validate_endpoint "${vpc_id}"
    validate_nat_gateway "${vpc_id}"
    validate_route_table "${vpc_id}"
```

```
fi
```

Run the VPC Validation Script

Enter this command to run the validation script.

```sh
export VPC_ID="vpc-0000000000000000"
./validate-vpc.sh "${VPC_ID}"
```