Service Workbench Post Deployment Guide

Table of Contents

Account Structure

Service Workbench uses *three* types of accounts. You will see these account names throughout the documentation.

- **Main**: The account from which Service Workbench is deployed. Will be billed for all AWS usage charges in this deployment.
- **Master**: Holds the AWS Organization which creates Member accounts.
- **Hosting**: User accounts created within Service Workbench for individuals.

Read the following files in the source code documentation to learn more about the different types of AWS accounts within Service Workbench:

- README.md
- main/solution/prepare-master-acc/README.md

Enable Local Users

Local users are created only within the solution. Their credentials are stored in Amazon DynamoDB. This is the easiest way to install. The alternative is to integrate with an Active Directory.

Create or Add Accounts

After logging in as **root** user for the first time, go to the '**Accounts**' page in the sidebar. Service Workbench uses AWS accounts on this page for launching research workspaces. You can add existing AWS accounts or create new ones on the '**Accounts**' tab. Accounts are responsible for the charges incurred by the resources that are deployed within the Service Workbench.

- **Create AWS Account**: Creates a new AWS account using AWS Organizations.
- **Add AWS Account**: Imports an existing AWS account, which will be responsible for its own billing.

Every user is linked to an **Account** through a **Project** and an **Index**, so at least one account must be created or added before creating the first user.

Important: *If you do not need to create new AWS accounts from within Service Workbench, then skip to the next section, 'Add AWS Account' section below.*

Create AWS Account

Prerequisites

Before creating an AWS account from Service Workbench, some prequisites must be met:

- Configure an existing AWS account to be the **Master** account for Service Workbench. When Service Workbench creates new AWS accounts, billing for those accounts will go to the **Master** account.
- Ensure the **Master** account has AWS Organizations enabled.

Configure Master Account

To configure the **Master** account:

1. Read the file: main/solution/prepare-master-acc/README.md.
2. Change directory to the **root folder** and run the command below. This command will take about 8 minutes to execute. `scripts/master-account-deploy.sh <stage>` The output of this command includes a **Master Role ARN** for the the next step.

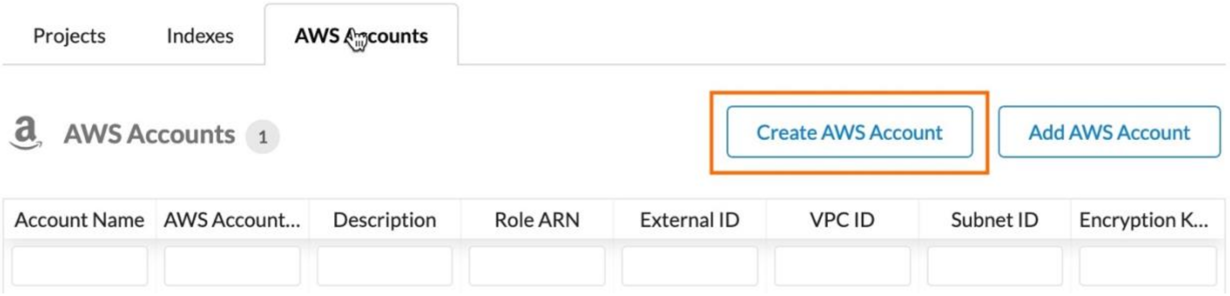
For more information on configuring an account to be the Master Account, see Prepare the Master Account in the 'Reference' section.

AWS Organizations

In the AWS Management Console, navigate to '**AWS Organizations**' to ensure that an Organization exists for the **Master** account. If it does not, then you will need to create a new one. There is no configuration to set; Service Workbench will create a new account in the AWS Organization for this deployment, named after the **Stage Name** used at deployment.

Creating a new Account

This will create a new **Member** AWS account in the Organization, whose billing will go to the **Master** account of the Organization. See **Figure 1**.



Account Name	AWS Account...	Description	Role ARN	External ID	VPC ID	Subnet ID	Encryption K...

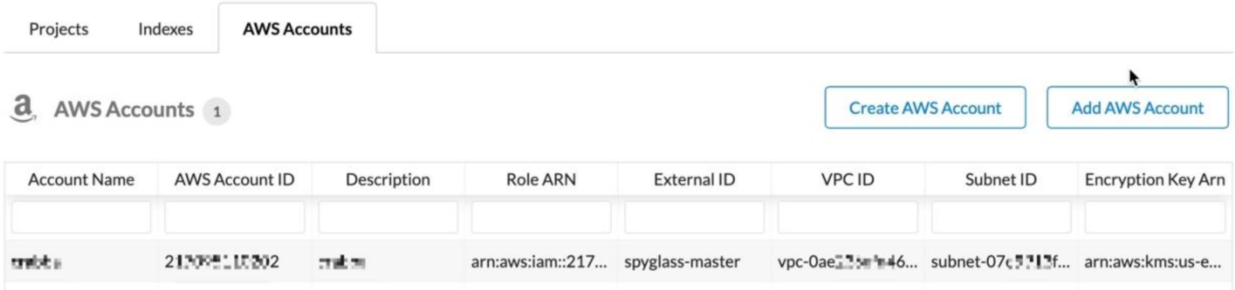
Figure 1: Create AWS Account

To create the account, perform the following actions:

1. In the Service Workbench console, navigate to '**Accounts → AWS Accounts**' and click **Create AWS Account**.
    - In **Role ARN**, fill in the **Master Role ARN** copied from the 'Configure Master Account' step described above.
    - The email address that you specify here must be unique within the Organization.
    - The **External ID** by default is the string **workbench**. See IAM for information on how to configure this to another value.
2. After a minute, the following information displays in the **AWS Accounts** tab:
    - *'Trying to create accountID: xxx'*
    - A workflow in progress in **Workflows → Provision Account** (see **Workflows** in the ***Service Workbench User Guide***)

      Note*: If instead you see an error message such as, 'Stop Internal State Account ID not found', check that there is an AWS Organization in the console of your* ***Master****account, if deploying Service Workbench in the* ***Master*** *account. If you are deploying in a* ***Member*** *account, check and ensure that you followed the steps described in [Prepare the Master Account](#).*

    - Optionally, in the AWS console, you can inspect the following resources deployed by this script:
        - In AWS CloudFormation, a stack **prep-master** will be running. It creates the **Master** role and its output is the **Master Role ARN**.
        - In the AWS Organization, in the **Master** account (see IAM), the new account will display.
        - In IAM, the new **Master** role will be created
3. Once the account is created it will be listed in **AWS Accounts**, see **Figure 2**.



Account Name	AWS Account ID	Description	Role ARN	External ID	VPC ID	Subnet ID	Encryption Key Arn
[redacted]	21[redacted]02	[redacted]	arn:aws:iam::217...	spyglass-master	vpc-0ae[redacted]46...	subnet-07c[redacted]f...	arn:aws:kms:us-e...

Figure 2: AWS Accounts with New Account

Add AWS Account

Adding an existing AWS account enables Service Workbench to launch research Workspaces into it. The existing account is reponsible for billing.

Gather Role ARNs

This step is run in the **Main** account, the account where you have deployed Service Workbench. Refer to the ***Configuration Settings*** section in the ***Service Workbench Installation Guide*** for information on how to specify the correct profile.

1. Run the following command in the `main/solution/backend` folder:

```
pnpx sls info --verbose --stage <stagename>
```

The output will contain similar lines to the following:

```
Stack Outputs
AuthenticationLayerHandlerRoleArn: arn:aws:iam::0000:role/stage-va-sw-backend-RoleAuthenticationLayerHan-
F00   EnvMgmtRoleArn: arn:aws:iam::0000:role/stage-va-sw-EnvMgmt   ApiHandlerRoleArn:
arn:aws:iam::0000:role/stage-va-sw-ApiHandler   WorkflowLoopRunnerRoleArn: arn:aws:iam::0000:role/stage-va-
sw-WorkflowLoopRunner   OpenDataScrapeHandlerRoleArn: arn:aws:iam::0000:role/stage-va-sw-backend-
RoleOpenDataScrapeHandler-F00   ServiceEndpoint: https://f00.execute-api.us-east-1.amazonaws.com/demo
ServerlessDeploymentBucketName: 0000-stage-va-sw-artifacts
```

2. Copy the values for `ApiHandlerRoleArn` and `WorkflowLoopRunnerRoleArn`.

Prepare the Existing AWS Account

This step prepares the existing AWS account that you wish to add to Service Workbench by running an onboarding template.

1. In the AWS Management Console, navigate to '**Amazon CloudFormation**'.
2. Create a new stack in CloudFormation. Select *Upload a template file* and locate the template file `addons/addon-base-raas/packages/base-raas-cfn-templates/src/templates/onboard-account.cfn.yml` from the source code.
3. On the next screen 'Specify stack details' enter the following values from **Table 3**:

Field	Value
Namespace	Short string (eg: stage name)
CentralAccountId	Service Workbench Main account ID
ExternalId	As specified (default: **workbench**)
VpcCidr	Retain default (10.0.0.0/16)
VpcPublicSubnet1Cidr	Retain default (10.0.0.0/19)
ApiHandlerArn	**ApiHandlerRoleArn** value from above
LaunchConstraintPolicyPrefix	Retain default (*)
LaunchConstraintRolePrefix	Retain default (*)
WorkflowRoleArn	**WorkflowLoopRunnerRoleArn** value from above

Table 3: Stack Details

4. Deploy the stack.
5. After the stack has deployed, view the output, which will contain values similar to the following in **Table 4**:

Key	Value
CrossAccountEnvMgmtRoleArn	arn:aws:iam::0000:role/sw-stage-xacc-env-mgmt
CrossAccountExecutionRoleArn	arn:aws:iam::0000:role/sw-stage-cross-account-role
EncryptionKeyArn	arn:aws:kms:us-east-2:0000:key/f00-f00-f00
VPC	vpc-f00f00
VpcPublicSubnet1	subnet-f00f00

Table 4: Stack Output

6. Copy the values down for the next step.

Adding the Account in Service Workbench

This step is run in the Service Workbench administrator interface and uses values from the previous step.

1. In the Service Workbench administrative interface, click the **AWS Accounts** tab. See **Figure 3**.

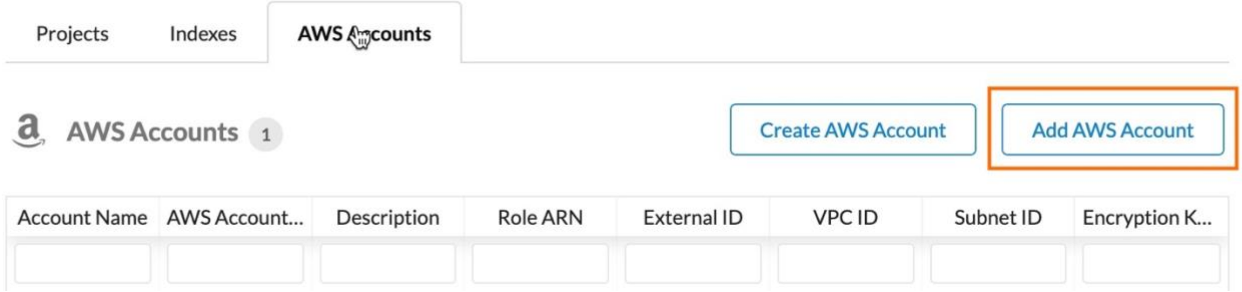


Figure 3: Add AWS Account

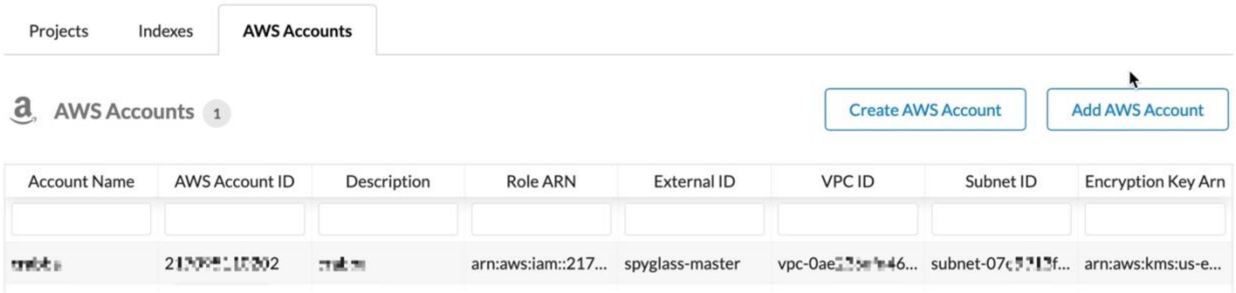
2. Click **Add AWS Account**. Enter the account information from the following **Table 5**:

Field	Value
Account Name	As desired
AWS Account ID	12-digit ID of imported account
Role ARN	**CrossAccountExecutionRoleArn** value
AWS Service Catalog Role Arn	**CrossAccountEnvMgmtRoleArn** value

Field	Value
External ID	As specified (default: **workbench**)
Description	As desired
VPC ID	**VPC** value
Subnet ID	**VpcPublicSubnet1** value
KMS Encryption Key ARN	**EncryptionKeyArn** value

Table 5: AWS Account Information

3. Once the account is added it will be listed in **AWS Accounts**, see **Figure 4**.



Account Name	AWS Account ID	Description	Role ARN	External ID	VPC ID	Subnet ID	Encryption Key Arn
[redacted]	21[redacted]302	[redacted]	arn:aws:iam::217...	spyglass-master	vpc-0ae[redacted]46...	subnet-07c[redacted]f...	arn:aws:kms:us-e...

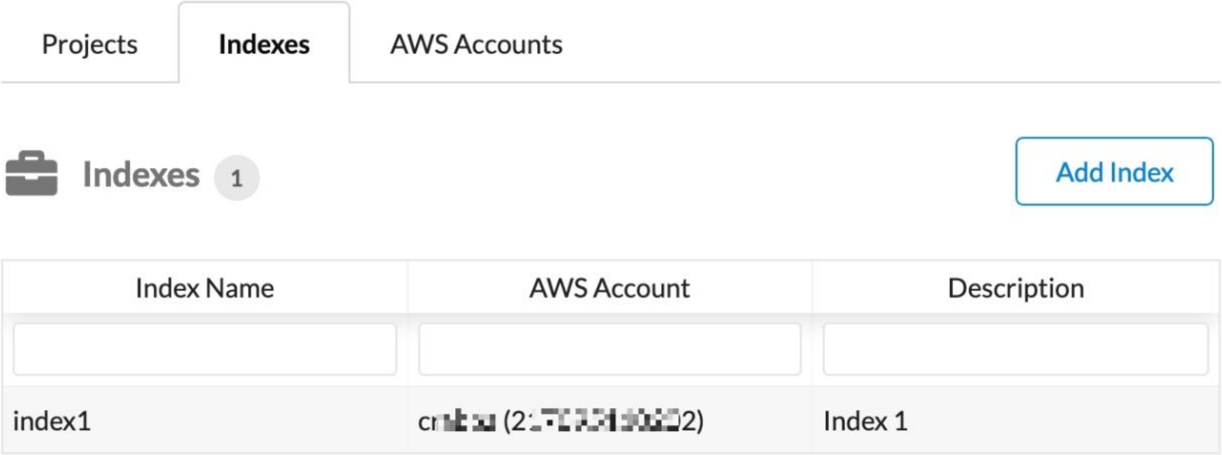
Figure 4: AWS Accounts with New Account

Create Indexes and Projects

Projects and **Indexes** form a hierarchy under '**Accounts**'. Each Account can have multiple **Indexes**, each **Index** can have multiple **Projects**. **Projects** are attached to **Users**, so you must create the **Projects** first.

After you create an account in the ‘**Accounts**’ tab of the administrative interface, create an ‘**Index**’ that links to the Account, by selecting the ‘**Account ID**’ from the drop-down list.

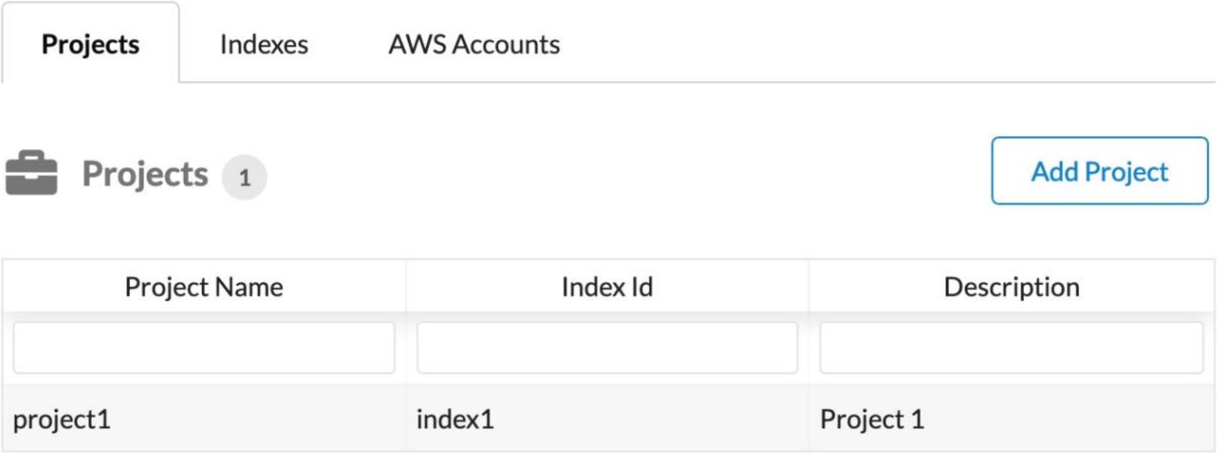
1. On the ‘**Indexes**’ tab, click ‘**Add Index**’. See **Figure 5**.



Index Name	AWS Account	Description
index1	[redacted]	Index 1

Figure 5: Create an Index

2. Create a '**Project**' that links to the new Index. In the '**Projects**' tab, click '**Add Project**'. See **Figure 6**.

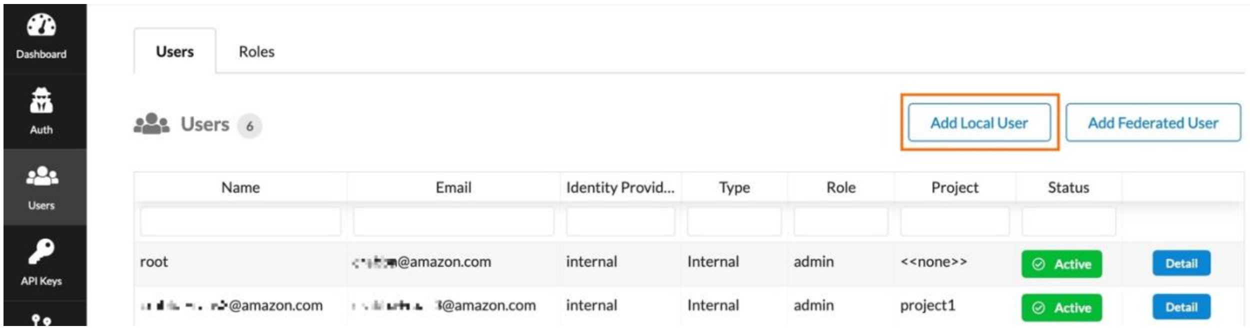


Project Name	Index Id	Description
project1	index1	Project 1

Figure 6: Create a Project

Create an Administrator User

Once you create an account and an index and project, you must create an administrator user in the '**Users**' tab. See **Figure 7**.

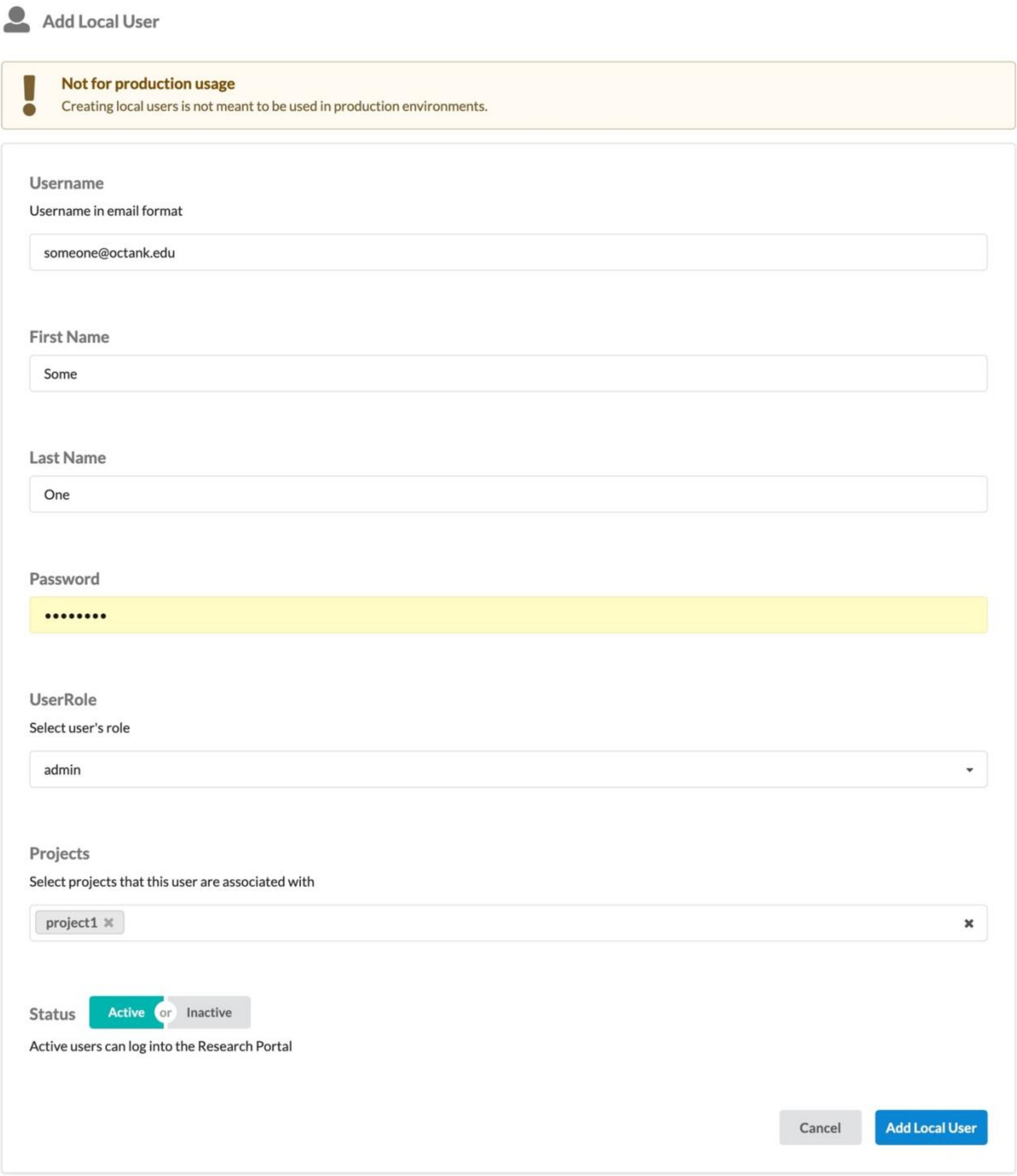


Name	Email	Identity Provid...	Type	Role	Project	Status	
root	[redacted]@amazon.com	internal	Internal	admin	<<none>>	Active	Detail
[redacted]@amazon.com	[redacted]@amazon.com	internal	Internal	admin	project1	Active	Detail

Figure 7: Create an Administrator

Note*: A root user account will already be created, however, you must not routinely use the root user account.*

For testing purposes, you can create a local user by clicking '**Add Local User**'. Assign the user the administrator's role, and associate the user with the **Project** you created, and set the status to '**Active**'. See **Figure 8**.



Add Local User

Not for production usage
Creating local users is not meant to be used in production environments.

Username
Username in email format
someone@octank.edu

First Name
Some

Last Name
One

Password
••••••••

UserRole
Select user's role
admin

Projects
Select projects that this user are associated with
project1 ✖

Status Active or Inactive
Active users can log into the Research Portal

Figure 8: Add Local User

In prod environments we highly recommend using an IDP. For more details, refer to the ***Service Workbench Configuration Guide*****.**

Import Service Catalog Products

Service Workbench uses AWS Service Catalog to manage different types of computation resources available for researchers to use through the platform.

With AWS Service Catalog integration, Service Workbench allows Admin users to create and manage catalogs of IT services that are approved for use on AWS. These IT services can include everything from virtual machine images, servers, software, and databases to complete multi-tier application architectures.

With this integration, Service Workbench helps organization to centrally manage commonly deployed IT services, and helps achieve consistent governance and meet compliance requirements, while enabling users to quickly deploy only the approved IT services they need.

When Service Workbench is deployed, an AWS Service Catalog portfolio is created by default with four commonly used products: Amazon SageMaker, Amazon EC2 for Windows, Amazon EC2 for Linux and Amazon EMR. The **administrator** needs to import and configure these products using Service Workbench user interface before they can be deployed. If you want to include additional custom products in the AWS Service Catalog portfolio, complete these steps:

1. Add the AWS CloudFormation template in the following directory:

   ```
   addons/addon-base-raas/packages/base-raas-cfn-
   templates/src/templates/service-catalog
   ```

2. Add the AWS CloudFormation template file name in the productsToCreate list in the following location:

   ```
   addons/addon-base-raas/packages/base-raas-post-
   deployment/lib/steps/create-service-catalog-portfolio.js
   ```

Import a Product

In this step, you import a pre-defined product, configure parameters to be used for product launch, and approve the configured product to be used. The following sections use Amazon EC2 Linux as an example, followed by setting different configuration required for Amazon EC2 Windows, Amazon SageMaker and Amazon EMR.

Prerequisites

Ensure the following prerequisites are met in order to import a product.

AMI

Make sure you completed the step, deploy the Machine Images SDC as part of the deployment process.

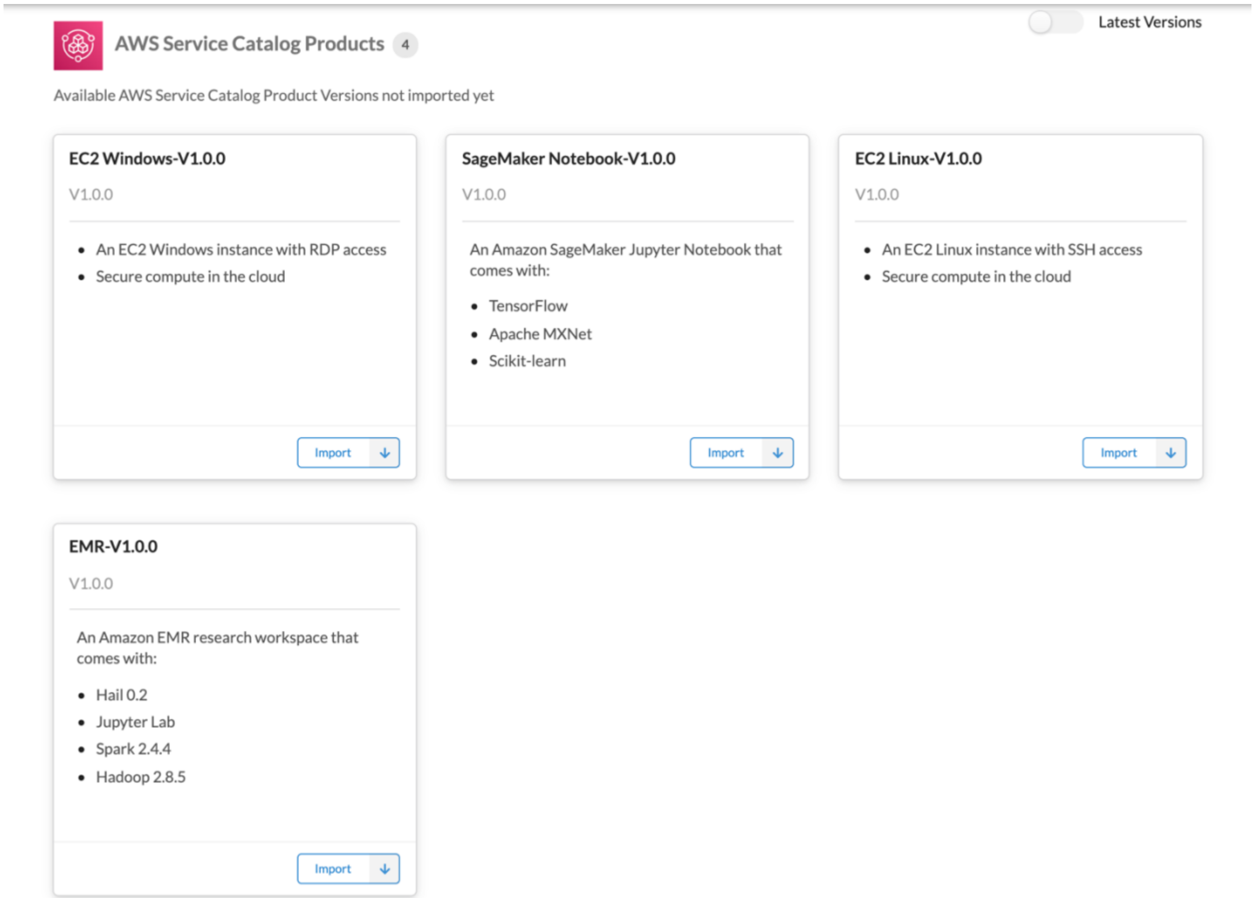
To check if AMIs were created successfully, perform the following actions:

1. Navigate to Amazon EC2.
2. Select the '**AMI**' tab.
3. Note down the 4 AMIs created for (1) Amazon EC2 Linux, (2) Amazon EC2 Windows, (3) Amazon EMR, and (4) Amazon EC2 Rstudio.
4. Copy the AMI IDs and use for workspace import and configuration. Alternatively, you can also copy these AMI IDs from the terminal when the machine-images SDC is deployed.

***Note**: If you run the machine images SDC multiple times, duplicated AMIs are created. This is okay and will not affect any Service Workbench functionalities. You can choose to remove the duplicates to avoid confusion or leave them as is.*

Service Catalog Portfolio

1. Log in to Service Workbench UI as an **administrator**.
2. Navigate to '**Workspace Types**' tab. Four AWS Service Catalog Products display as shown in **Figure 9**.



Available AWS Service Catalog Product Versions not imported yet

EC2 Windows-V1.0.0
V1.0.0
- An EC2 Windows instance with RDP access
- Secure compute in the cloud

SageMaker Notebook-V1.0.0
V1.0.0
An Amazon SageMaker Jupyter Notebook that comes with:
- TensorFlow
- Apache MXNet
- Scikit-learn

EC2 Linux-V1.0.0
V1.0.0
- An EC2 Linux instance with SSH access
- Secure compute in the cloud

EMR-V1.0.0
V1.0.0
An Amazon EMR research workspace that comes with:
- Hail 0.2
- Jupyter Lab
- Spark 2.4.4
- Hadoop 2.8.5

Figure 9: AWS Service Catalog Products

These four products come from the AWS Service Catalog portfolio created by the system during deployment. And they'll be ready for use once imported and configured.

If you wish to include other AWS computation resources in the future:

1. Add a new product to the existing Service Workbench portfolio in AWS Service Catalog
2. Update the role ServiceCatalogLaunchConstraintRole in cloudformation.yml to include permission needed to launch and terminate the product

Import

In this section, the Amazon EC2 Linux is used as an example.

1. Click the '**Import**' button under ec2-linux-instance.
2. Update **Name** and **Description** so you can easily identify the workspace.

Configure

Once you import a workspace type, perform the following actions:

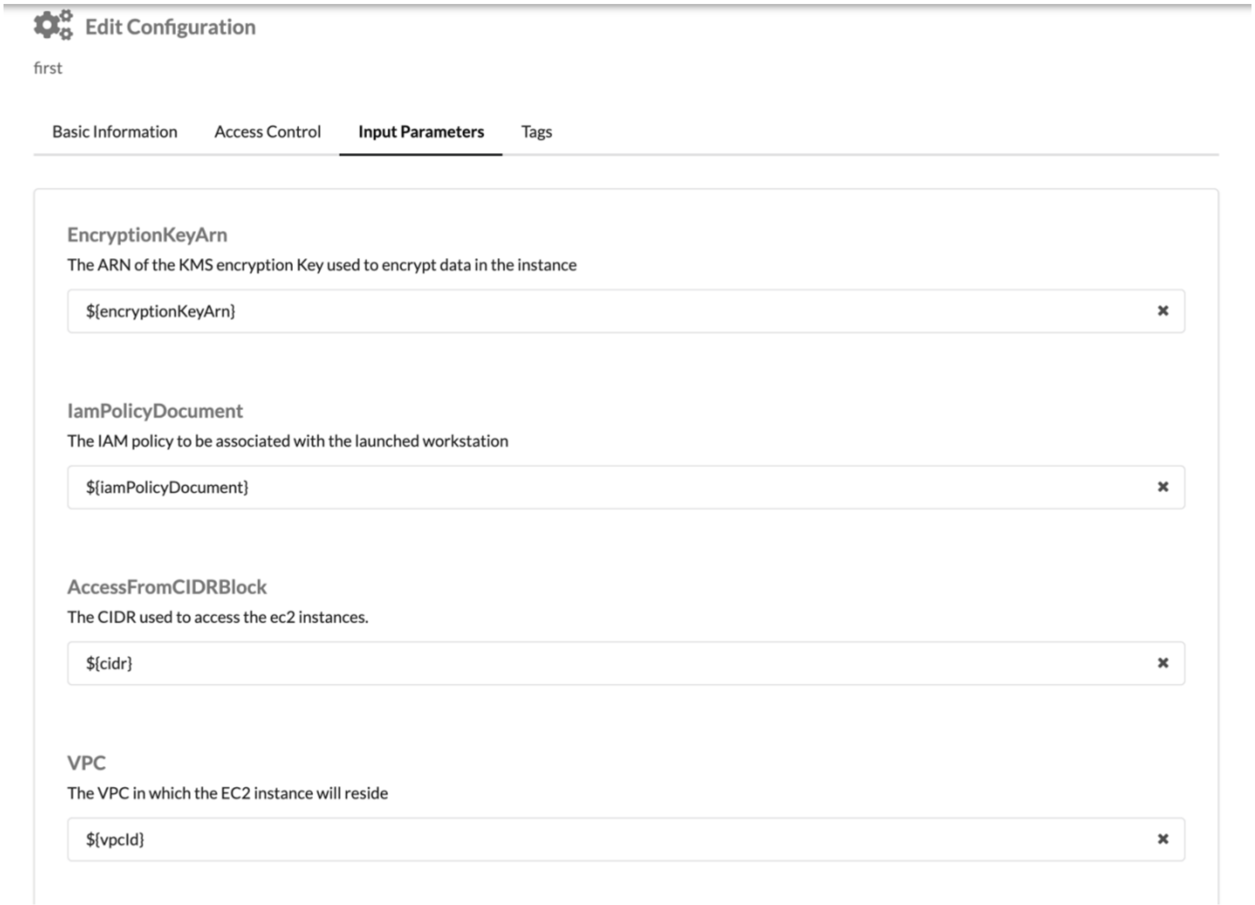
1. Click '**Add Configuration**'
2. Add **ID**, **Name**, **Description**, and **Estimated Costs** for the configuration. A common naming convention here is to attach the instance size after the product name. For example, use ec2-linux-instance-V1-small for a small Linux Amazon EC2 instance.
3. Click '**Next**'.
4. Add access control for the workspace configuration.
5. Click '**Next**'

The input parameters are parameters used for the product, AWS CloudFormation template. The number and type of parameters are different for different products. Most of the parameters used for the four system created products can be evaluated automatically at launch time. These parameters are available for selection in the drop-down when filling the input parameters page.

Configuration for EC2 Linux

For Amazon EC2 Linux, the only two fields that are not available in the drop-down are '**InstanceType**' and '**AmiId**'.

Figure 10 and **Figure 11** display screenshot images that exemplify Amazon EC2 Linux configurations.



Edit Configuration

first

EncryptionKeyArn

The ARN of the KMS encryption Key used to encrypt data in the instance

${encryptionKeyArn}

IamPolicyDocument

The IAM policy to be associated with the launched workstation

${iamPolicyDocument}

AccessFromCIDRBlock

The CIDR used to access the ec2 instances.

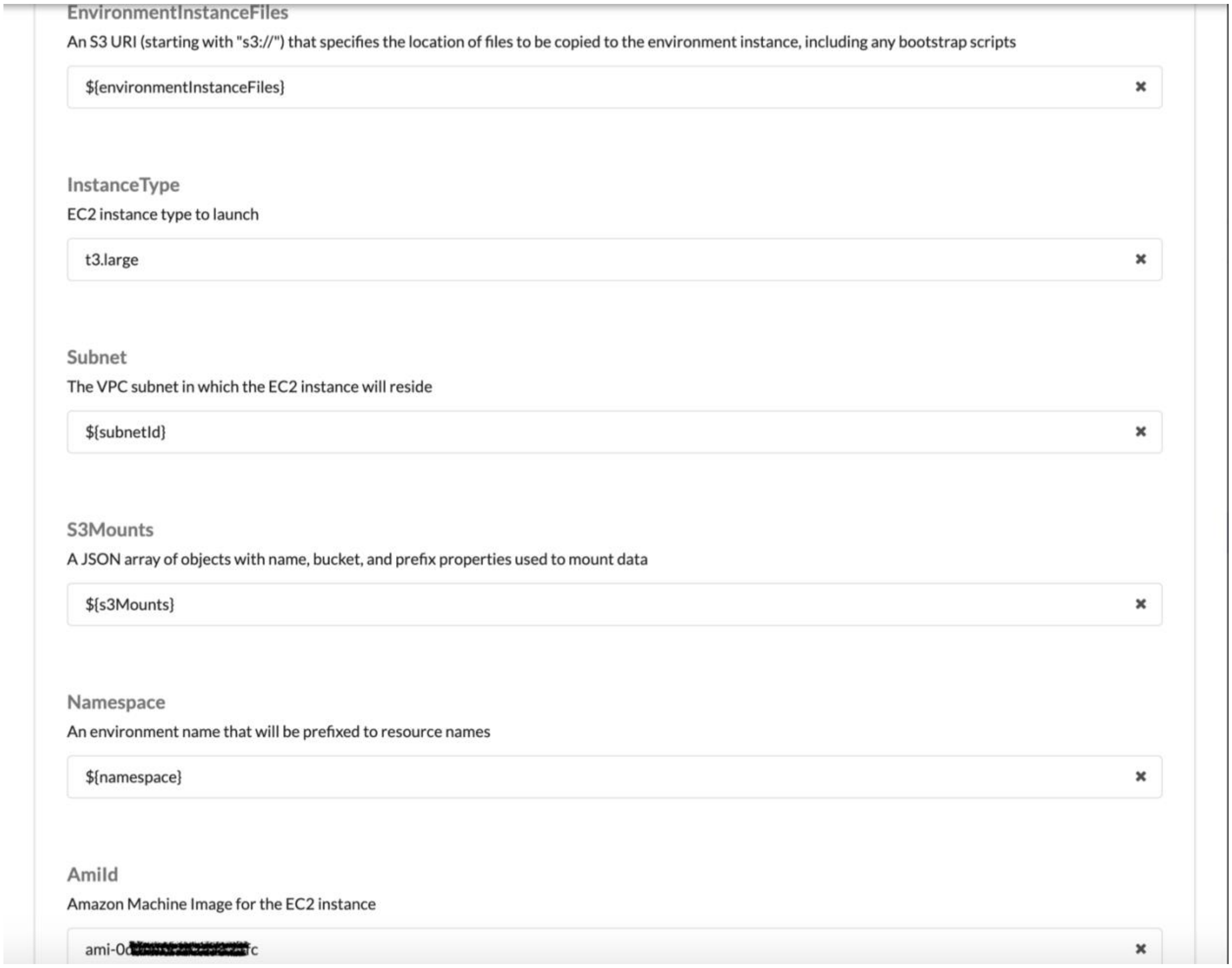
${cidr}

VPC

The VPC in which the EC2 instance will reside

${vpcId}

Figure 10: Configurations for Amazon EC2 Linux



EnvironmentInstanceFiles

An S3 URI (starting with "s3://") that specifies the location of files to be copied to the environment instance, including any bootstrap scripts

${environmentInstanceFiles} ✖

InstanceType

EC2 instance type to launch

t3.large ✖

Subnet

The VPC subnet in which the EC2 instance will reside

${subnetId} ✖

S3Mounts

A JSON array of objects with name, bucket, and prefix properties used to mount data

${s3Mounts} ✖

Namespace

An environment name that will be prefixed to resource names

${namespace} ✖

AmiId

Amazon Machine Image for the EC2 instance

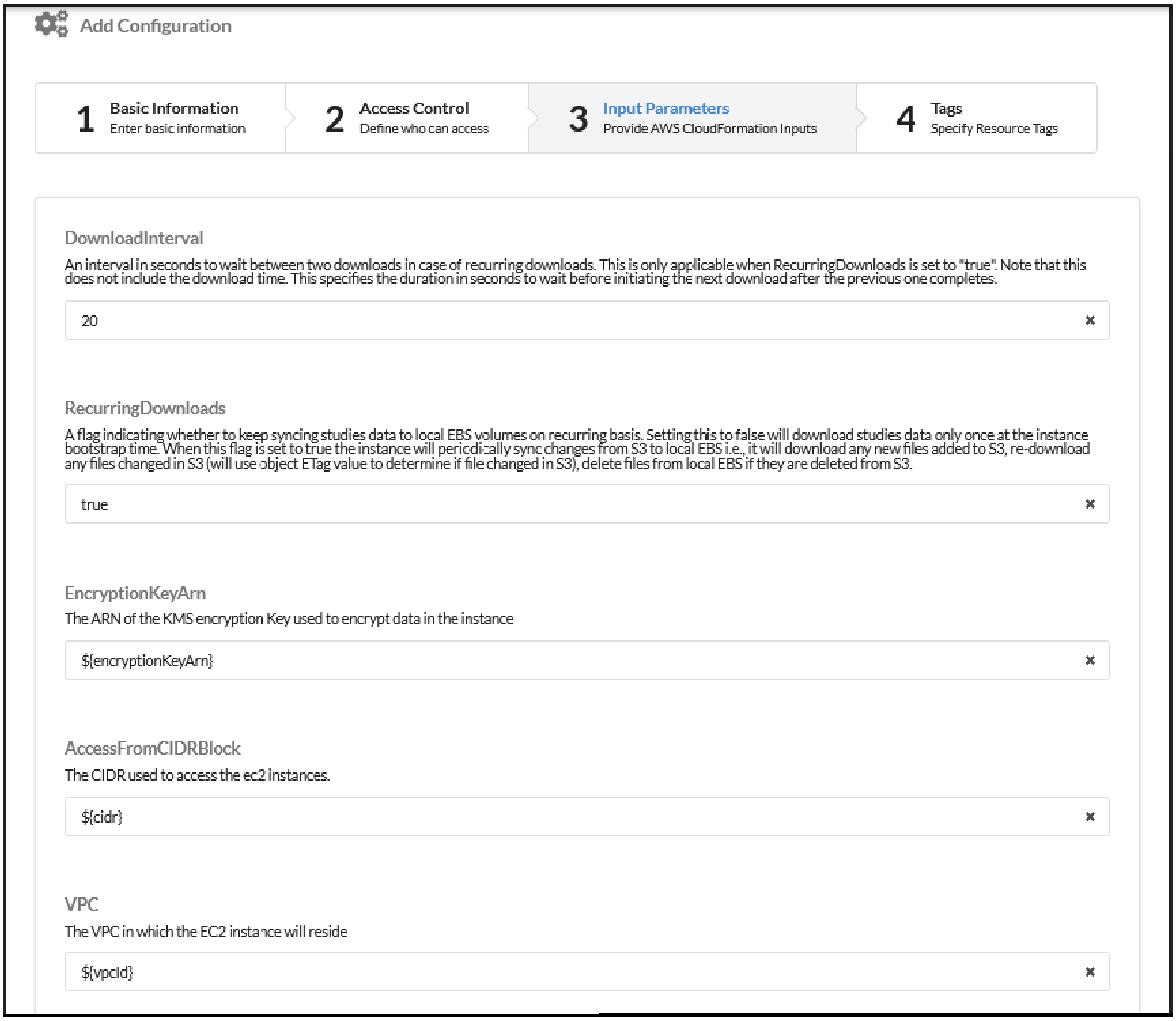
ami-0[redacted]c ✖

Figure 11: Configurations for Amazon Linux EC2

Configuration for Amazon EC2 Windows

For Amazon EC2 Windows, the only two fields that are not available in the drop-down are '**InstanceType**' and '**AmiId**'. (Use the AMI ID you copied in Prerequisites - AMI)

Figure 12, **Figure 13**, and **Figure 14** display screenshot images that exemplify Amazon EC2 Windows configurations.



Add Configuration

DownloadInterval

An interval in seconds to wait between two downloads in case of recurring downloads. This is only applicable when RecurringDownloads is set to "true". Note that this does not include the download time. This specifies the duration in seconds to wait before initiating the next download after the previous one completes.

20

RecurringDownloads

A flag indicating whether to keep syncing studies data to local EBS volumes on recurring basis. Setting this to false will download studies data only once at the instance bootstrap time. When this flag is set to true the instance will periodically sync changes from S3 to local EBS i.e., it will download any new files added to S3, re-download any files changed in S3 (will use object ETag value to determine if file changed in S3), delete files from local EBS if they are deleted from S3.

true

EncryptionKeyArn

The ARN of the KMS encryption Key used to encrypt data in the instance

${encryptionKeyArn}

AccessFromCIDRBlock

The CIDR used to access the ec2 instances.

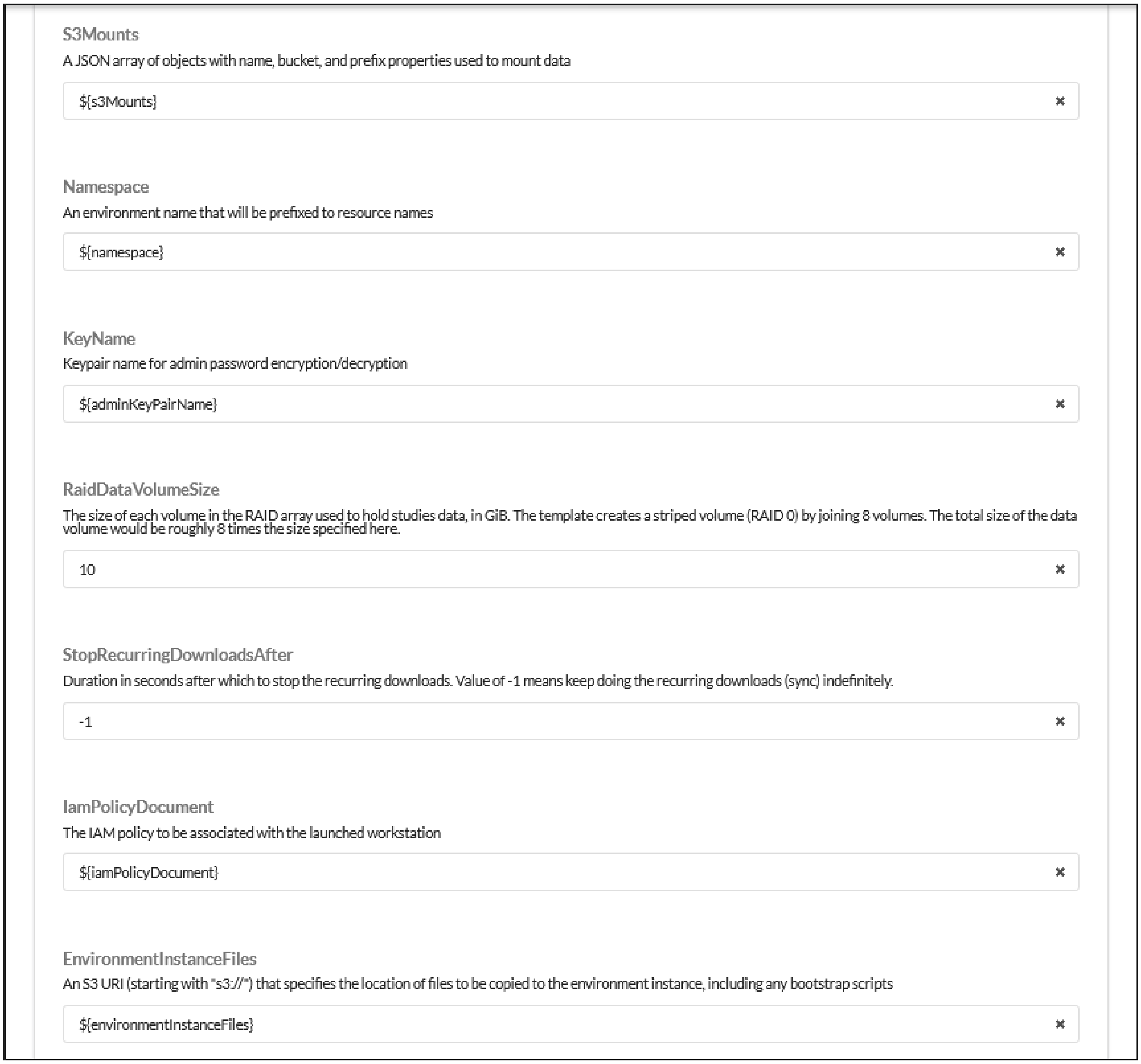
${cidr}

VPC

The VPC in which the EC2 instance will reside

${vpcId}

Figure 12: Configurations for EC2 Windows



S3Mounts

A JSON array of objects with name, bucket, and prefix properties used to mount data

${s3Mounts}

Namespace

An environment name that will be prefixed to resource names

${namespace}

KeyName

Keypair name for admin password encryption/decryption

${adminKeyPairName}

RaidDataVolumeSize

The size of each volume in the RAID array used to hold studies data, in GiB. The template creates a striped volume (RAID 0) by joining 8 volumes. The total size of the data volume would be roughly 8 times the size specified here.

10

StopRecurringDownloadsAfter

Duration in seconds after which to stop the recurring downloads. Value of -1 means keep doing the recurring downloads (sync) indefinitely.

-1

IamPolicyDocument

The IAM policy to be associated with the launched workstation

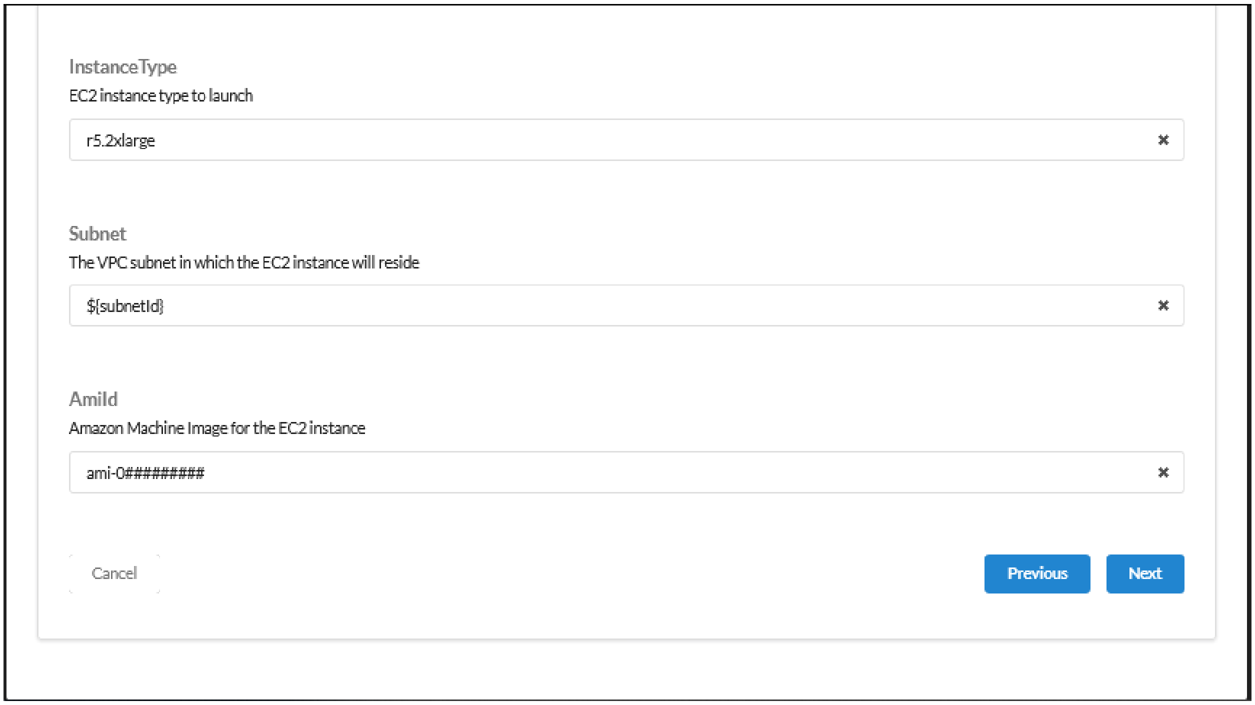
${iamPolicyDocument}

EnvironmentInstanceFiles

An S3 URI (starting with "s3://") that specifies the location of files to be copied to the environment instance, including any bootstrap scripts

${environmentInstanceFiles}

Figure 13: Configurations for EC2 Windows



InstanceType
EC2 instance type to launch

r5.2xlarge

Subnet
The VPC subnet in which the EC2 instance will reside

${subnetId}

AmiId
Amazon Machine Image for the EC2 instance

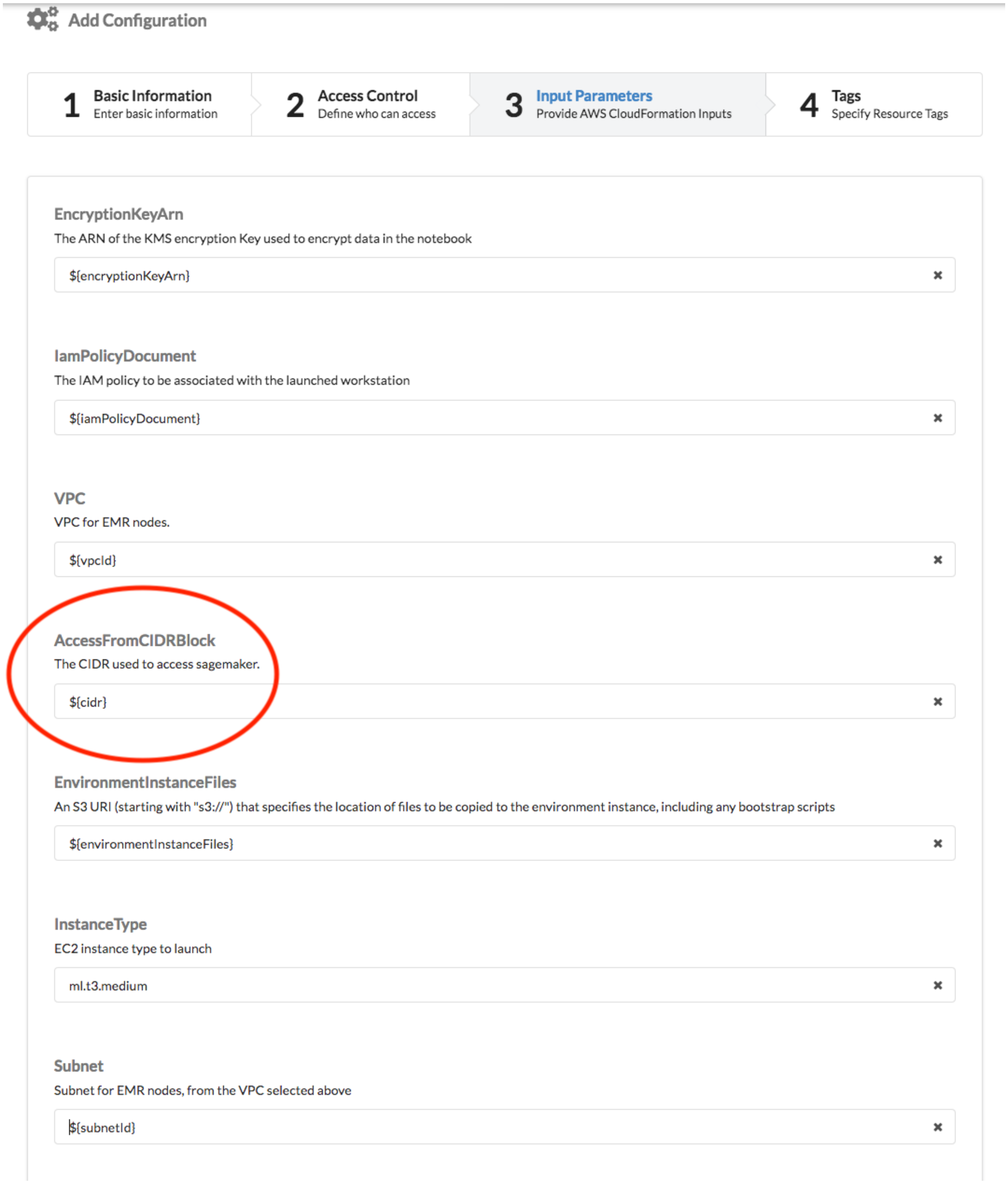
ami-0#########

Figure 14: Configurations for EC2 Windows

Configuration for Amazon SageMaker

For Amazon SageMaker, the only field that’s not available in the drop-down is ‘**InstanceType**’.

Figure 14 and **Figure 15** display screenshot images that exemplify Amazon SageMaker configurations.



Add Configuration

EncryptionKeyArn

The ARN of the KMS encryption Key used to encrypt data in the notebook

${encryptionKeyArn} ✖

IamPolicyDocument

The IAM policy to be associated with the launched workstation

${iamPolicyDocument} ✖

VPC

VPC for EMR nodes.

${vpcId} ✖

AccessFromCIDRBlock

The CIDR used to access sagemaker.

${cidr} ✖

EnvironmentInstanceFiles

An S3 URI (starting with "s3://") that specifies the location of files to be copied to the environment instance, including any bootstrap scripts

${environmentInstanceFiles} ✖

InstanceType

EC2 instance type to launch

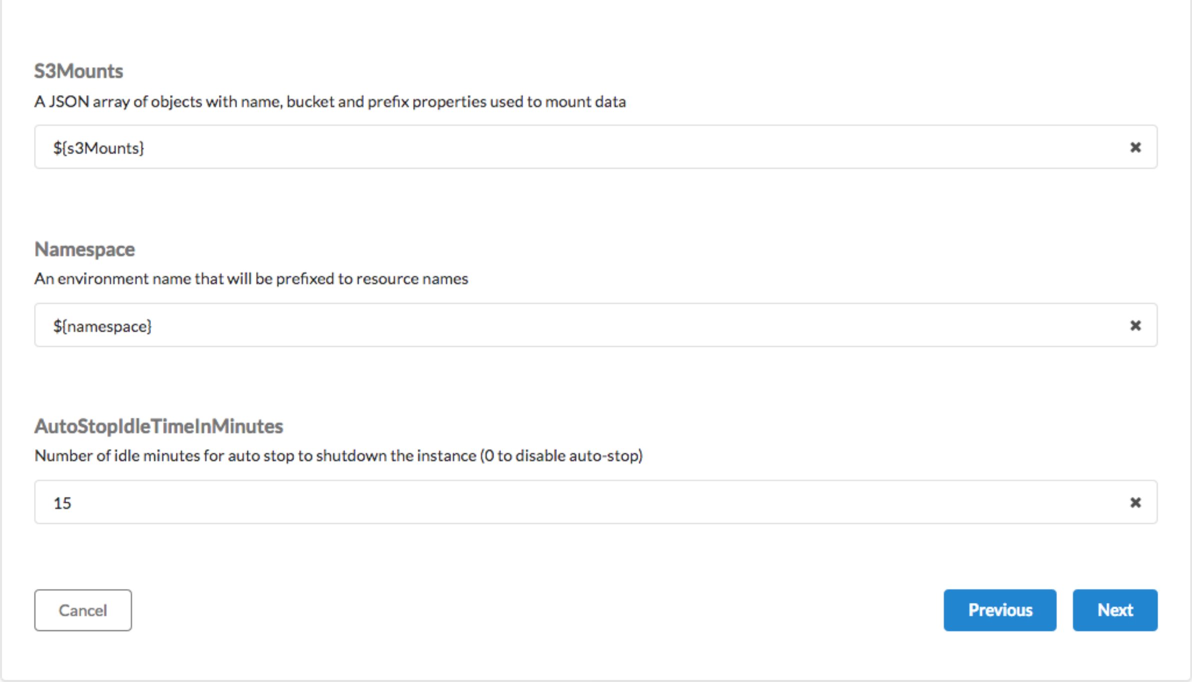
ml.t3.medium ✖

Subnet

Subnet for EMR nodes, from the VPC selected above

${subnetId} ✖

Figure 14: Configurations for Amazon SageMaker



S3Mounts

A JSON array of objects with name, bucket and prefix properties used to mount data

${s3Mounts}

Namespace

An environment name that will be prefixed to resource names

${namespace}

AutoStopIdleTimeInMinutes

Number of idle minutes for auto stop to shutdown the instance (0 to disable auto-stop)

15

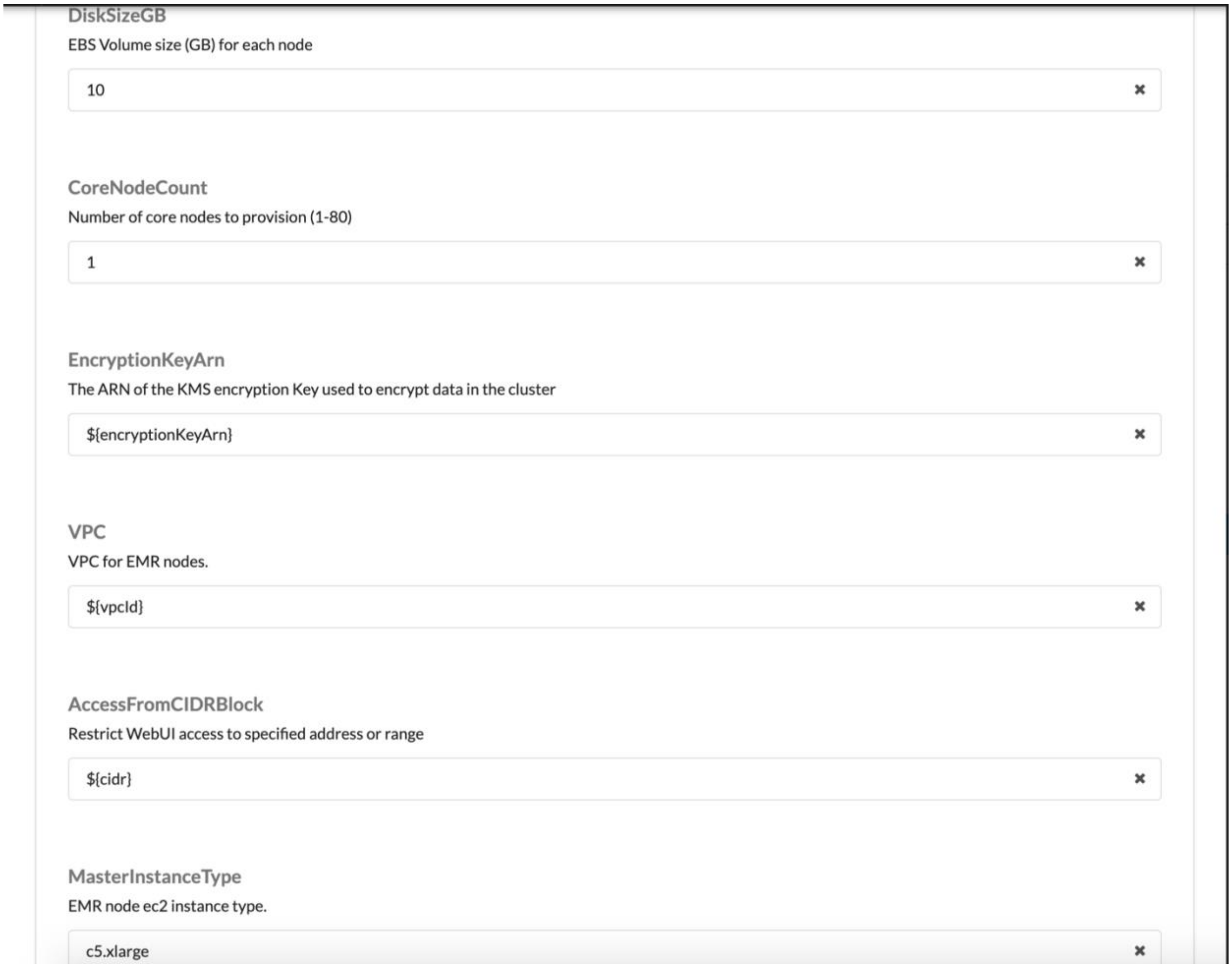
Figure 15: Configurations for Amazon SageMaker

Configuration for Amazon EMR

Amazon EMR requires a few more fields that are not available in the drop-down menu, including the following:

- DiskSizeGB (>=10)
- CoreNodeCount (1-80)
- MasterInstanceType
- Market (ON_DEMAND / SPOT)
- WorkerBidPrice (only applicable when Market = SPOT. Specify 0 for Market = ON_DEMAND)
- WorkerInstanceType
- AmiId (Use the AMI id we copied in prerequisites - AMI)

Figure 16, **Figure 17**, and **Figure 18** display screenshot images that exemplify Amazon EMR configurations.



DiskSizeGB

EBS Volume size (GB) for each node

10

CoreNodeCount

Number of core nodes to provision (1-80)

1

EncryptionKeyArn

The ARN of the KMS encryption Key used to encrypt data in the cluster

${encryptionKeyArn}

VPC

VPC for EMR nodes.

${vpcId}

AccessFromCIDRBlock

Restrict WebUI access to specified address or range

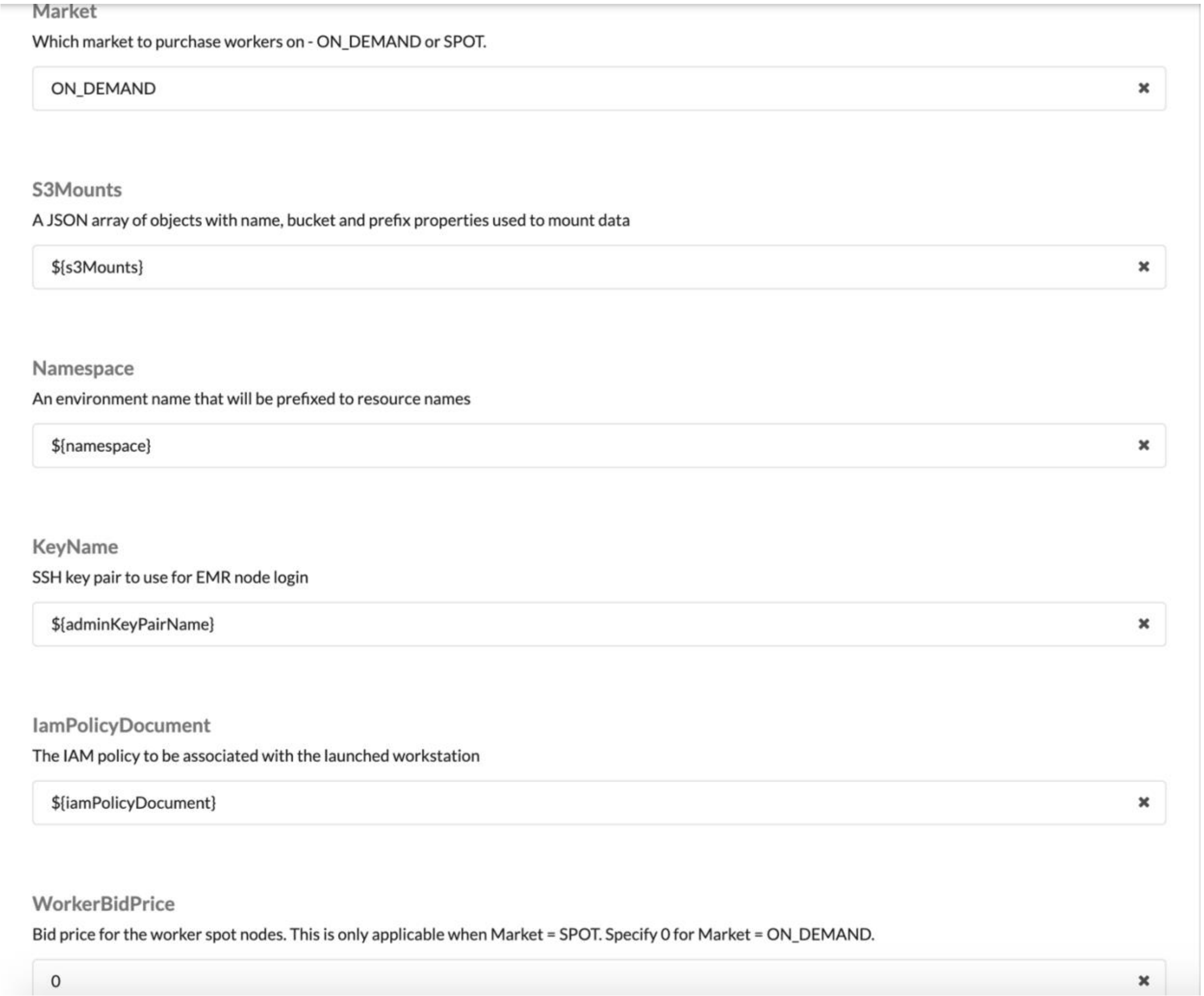
${cidr}

MasterInstanceType

EMR node ec2 instance type.

c5.xlarge

Figure 16: Configurations for Amazon EMR



Market

Which market to purchase workers on - ON_DEMAND or SPOT.

ON_DEMAND ✖

S3Mounts

A JSON array of objects with name, bucket and prefix properties used to mount data

${s3Mounts} ✖

Namespace

An environment name that will be prefixed to resource names

${namespace} ✖

KeyName

SSH key pair to use for EMR node login

${adminKeyPairName} ✖

IamPolicyDocument

The IAM policy to be associated with the launched workstation

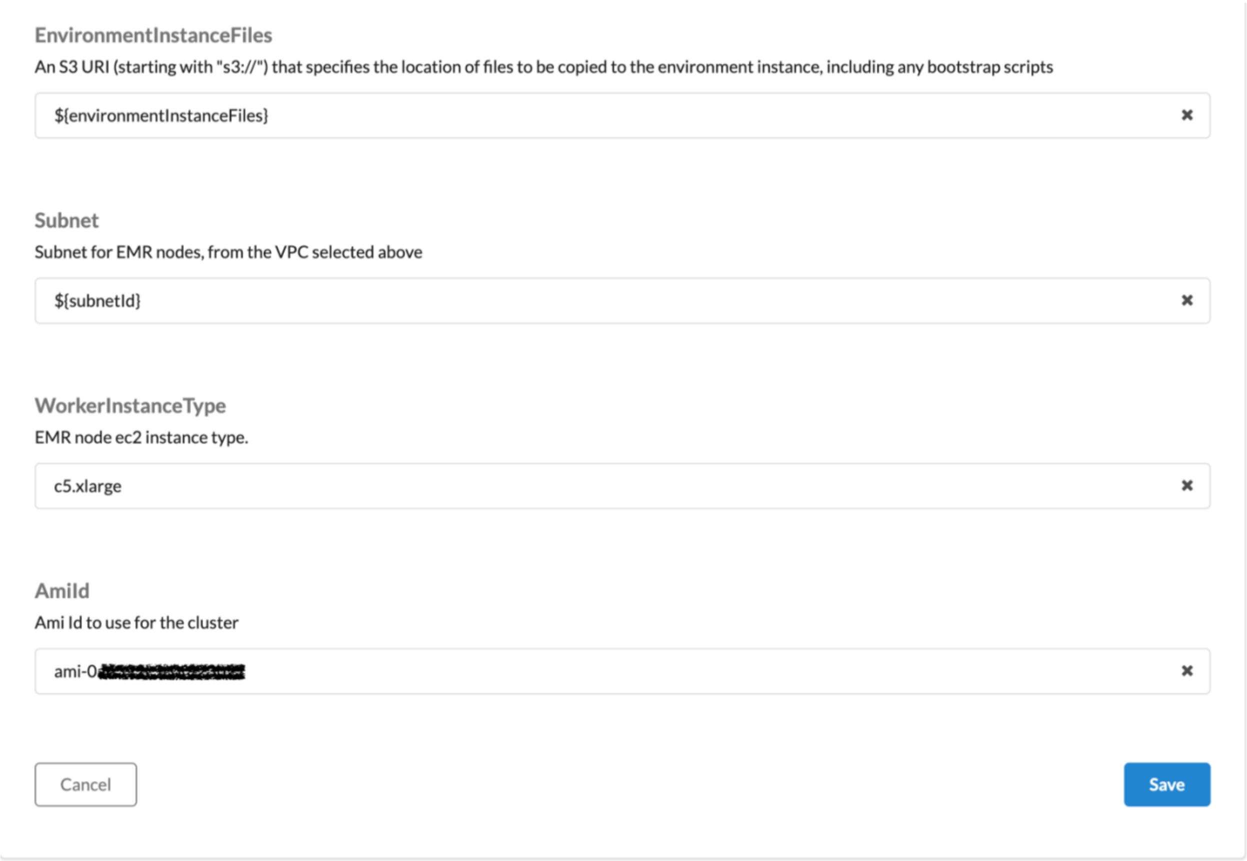
${iamPolicyDocument} ✖

WorkerBidPrice

Bid price for the worker spot nodes. This is only applicable when Market = SPOT. Specify 0 for Market = ON_DEMAND.

0 ✖

Figure 17: Configurations for Amazon EMR



EnvironmentInstanceFiles

An S3 URI (starting with "s3://") that specifies the location of files to be copied to the environment instance, including any bootstrap scripts

${environmentInstanceFiles}

Subnet

Subnet for EMR nodes, from the VPC selected above

${subnetId}

WorkerInstanceType

EMR node ec2 instance type.

c5.xlarge

AmiId

Ami Id to use for the cluster

ami-0[redacted]

Figure 18: Configurations for Amazon EMR

Approve

Once the configuration completes, click the ‘**Approve**’ button; the newly created workspace type will be available for launch in the ‘**Study and Workspace**’ tab.

Viewing logs

Viewing Service Workbench logs in CloudWatch

Service Workbench has API Gateway access logging enabled. The logs are available in CloudWatch at the /aws/api-gateway/<name of your API> log group:

Following is the format of the access logs:

```
{    "authorizer.principalId": "u-000000000000",    "error.message": "-",    "extendedRequestId":
"ZuT4rGDNoAMFxXw=",    "httpMethod": "GET",    "identity.sourceIp": "22.22.222.22",    "integration.error": "-",
"integration.integrationStatus": "200",    "integration.latency": "79",    "integration.requestId": "67394741-90ae-
4c6c-94fb-df8bf7be33ec",    "integration.status": "200",    "path": "/dev/api/user-roles",    "requestId": "468a1b4d-3015-4901-b749-37e4e0551029",    "responseLatency": "83",    "responseLength": "819",    "stage": "dev",    "status": "200"}
```

Lambda logs are also available in CloudWatch with the default log group names /aws/lambda/<lambda function name>.

Metrics

The default metrics for Lambda and API Gateway are available in CloudWatch. For the full list of available metrics, see:

- Working with AWS Lambda function metrics - AWS Lambda
- Amazon API Gateway dimensions and metrics - Amazon API Gateway

Service Workbench does not emit any custom metrics.

Deploying Updates

After a successful initial deployment, you can deploy individually to the five serverless projects that are a part of this solution.

Deploying Updates to the Infrastructure Serverless Project

```
$ cd solution/infrastructure$ pnpx sls deploy -s <stage>
```

Deploying Updates to the Backend Serverless Project

```
$ cd solution/backend$ pnpx sls deploy -s <stage>
```

Deploying Updates to the Machine-Images Serverless Project

```
$ cd solution/machine-images$ pnpx sls deploy -s <stage>
```

Deploying Updates to the Post-Deployment Serverless Project

```
$ cd solution/post-deployment$ pnpx sls invoke local -f postDeployment --env WEBPACK_ON=true -s <stage>
```

Deploying Updates to the UI Serverless Project

```
$ cd solution/ui$ pnpx sls package-ui --stage <stage> --local$ pnpx sls package-ui --stage <stage>$ pnpx sls deploy-ui --stage <stage> --invalidate-cache
```

Reference

Service Workbench on AWS interacts with multiple AWS resources, including Amazon EC2, AWS IAM, AWS Organizations, and more. You can easily add an AWS IAM role to an Amazon EC2 instance, leverage our AWS Organizations with your account structure, and create multiple Amazon S3 buckets.

Add an AWS IAM Role to an Amazon EC2 Instance

An Amazon EC2 instance can be assigned an **Instance Profile** that contains an AWS **IAM role**. The AWS **IAM role** will give the Amazon EC2 instance a set of permissions. The Amazon EC2 instance will only perform the actions defined by its AWS **IAM role**. Adding an AWS **IAM role** to the Amazon EC2 instance allows your application to make API calls securely—eliminating the need to manage security credentials.

The Service Workbench deployment application must be able to create AWS resources. The easiest way to meet this requirement is to give the Amazon EC2 instance an administrator role.

Adding an Administrator Role to a New Amazon EC2 Instance

When creating a new Amazon EC2 instance for a Service Workbench deployment, an **Instance Profile** may be assigned to the Amazon EC2 instance in '**Step 3: Configure Instance Details**'. Select '**Create a new IAM role**'—located next to the AWS IAM role drop-down. **Figure 25** displays an image of the '**Create a New IAM**' role action in the AWS Management Console.

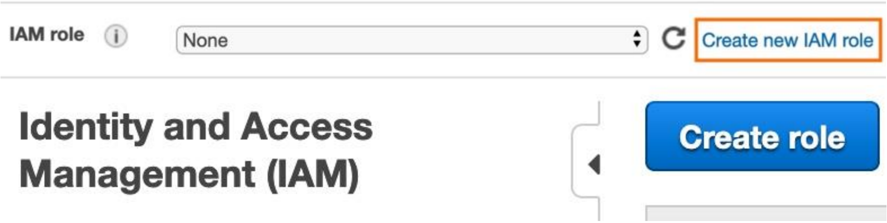
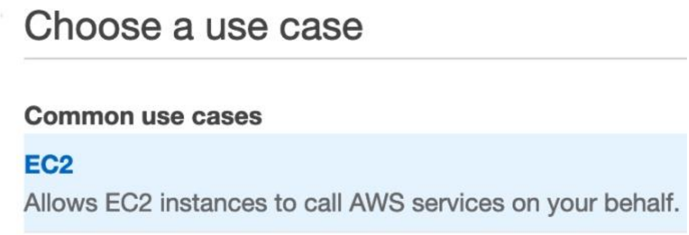


Figure 25: Create a New AWS IAM Role

To continue the process, highlight Amazon EC2 and proceed to permissions. In '**Permissions**', filter for '**AdministratorAccess**' and select it. Proceed through '**Tags**'. On the '**Review**' page, give your role a memorable name. Return to the Amazon EC2 tab, refresh the AWS IAM role drop-down, and select your administrator role to attach to the new Amazon EC2 instance. Now, proceed through the process to create an Amazon EC2 instance. **Figure 26**, **Figure 27**, **Figure 28**, and **Figure 29** display images to help you complete this process.



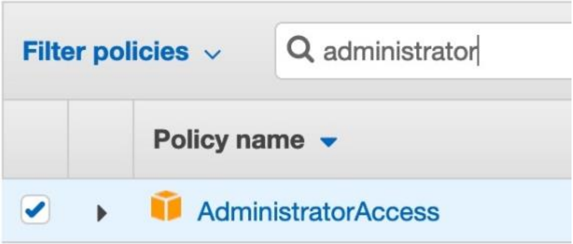
Choose a use case

Common use cases

EC2

Allows EC2 instances to call AWS services on your behalf.

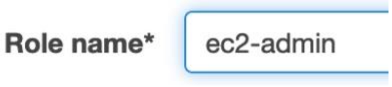
Figure 26: Permissions in Amazon EC2



administrator

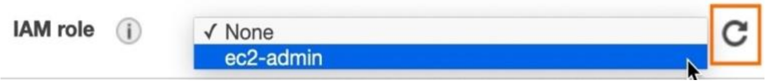
☑ AdministratorAccess

Figure 27: Filtering for AdministratorAccess



Role name* ec2-admin

Figure 28: Choosing a Role Name for an Amazon EC2 Instance



IAM role

✓ None
ec2-admin

Figure 29: Selecting the Administrator Role of the Amazon EC2 Instance

Adding a role to an existing instance

To add a role to an Amazon EC2 instance that is already running, select the Amazon EC2 instance in the EC2 Console. Open the '**Action > Instance Settings**' menu, and select '**Attach/Replace IAM Role**'. **Figure 30** shows the **Instance Settings** menu.

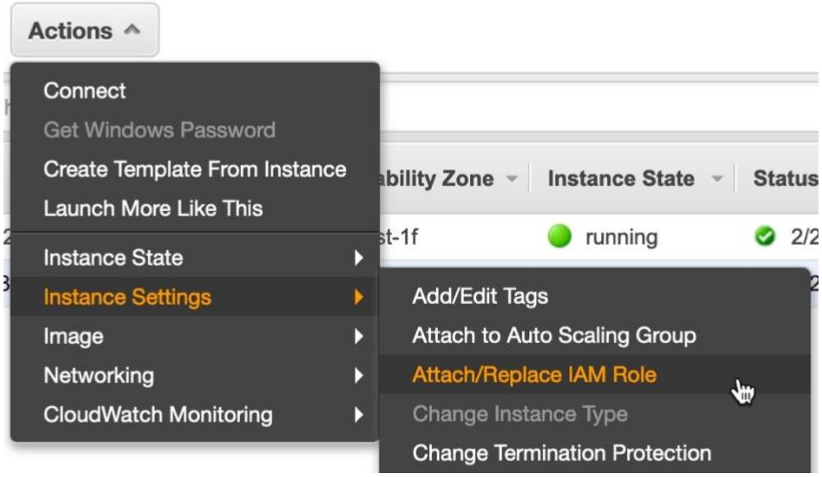
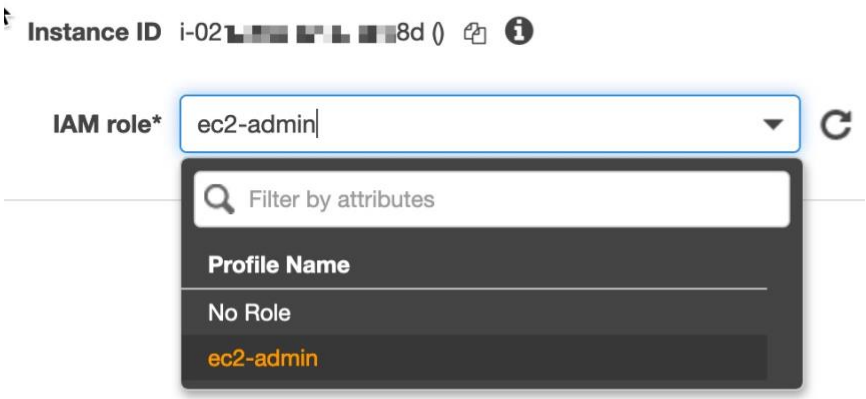


Figure 30: Attach/Replace an AWS IAM Role in the EC2 Console

In the '**Attach/Replace IAM Role**' screen, search for the role you created, select it, and click **Apply**. **Figure 31** shows the screen where you can '**Attach/Replace IAM Role**'.



Instance ID i-02[redacted]8d ()

IAM role* ec2-admin

Figure 31: AWS IAM Role Search

Usage of AWS Cloud Services

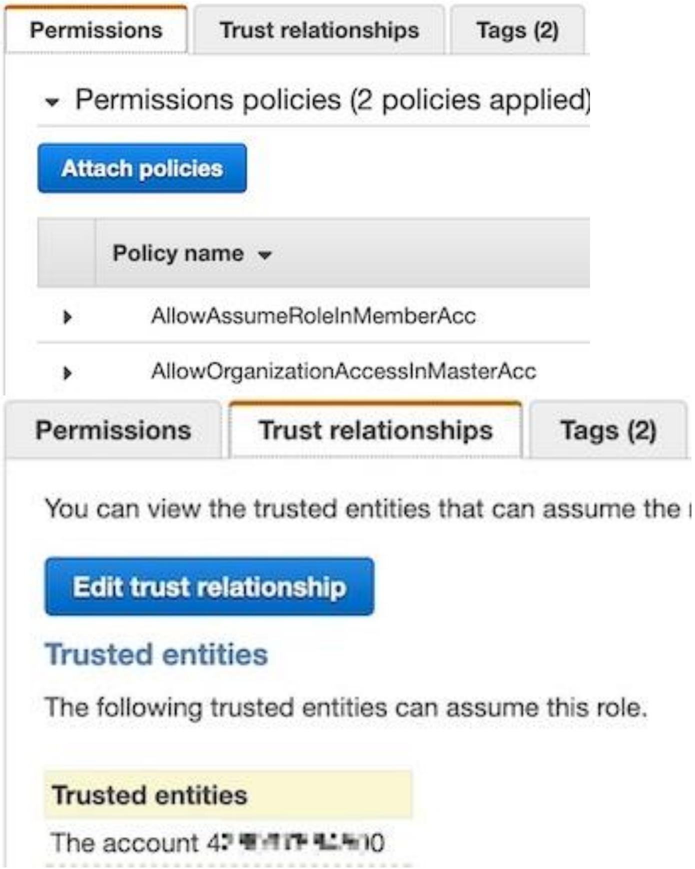
This section describes some of the AWS Cloud services used by Service Workbench. The resource names usually include the **Namespace**, including the **Stage Name** used at deployment. You can deploy multiple instances of Service Workbench from the same account if you use a different Stage Name for each deployment.

Amazon EC2

Amazon EC2 is used only as a platform from which to deploy Service Workbench. For more details see the Deployment Instance section.

AWS IAM

Service Workbench creates several roles in your account. The role `<namespace>-prep-raas-master-MasterRole-XXX` is created when you run the Post Deployment SDC. This role possesses a trust relationship with the Main account from which you deployed Service Workbench. There are two polices that allow the Main account to assume a role in this Master account. The Account Structure defines each type of account. **Figure 32** shows the AWS IAM '**Trust Relationships**' tab.



Permissions policies (2 policies applied)

Policy name
AllowAssumeRoleInMemberAcc
AllowOrganizationAccessInMasterAcc

You can view the trusted entities that can assume the

Trusted entities

The following trusted entities can assume this role.

Trusted entities
The account 4[redacted]0

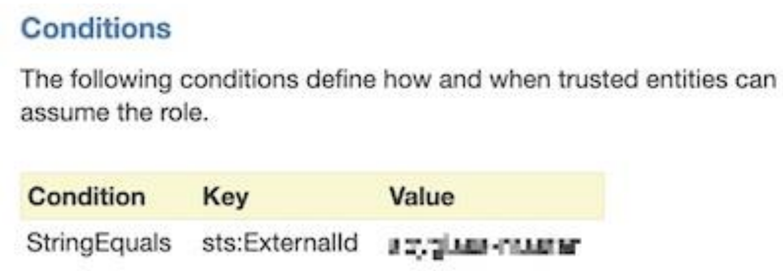
Figure 32: AWS IAM Trust Relationships Tab

An External ID is associated with the role. The External ID is an identifying string that is provided once a role is created. In order for the Trusted Entity (your Main account) to assume its role in the Master Account, it must supply this External ID. Providing the External ID of establishes a revocable relationship between the Trusted Entity and the Master account.

In the current Service Workbench deployment, the External ID is configured as a default value in the following string workbench:

```
main/solution/prepare-master-acc/config/settings/.defaults.yml
```

To change this value, create a stage-named configuration file (`mystagename.yml`) in the same directory. For more information, see the Configuration section. **Figure 33** displays a screenshot image of the conditions that define how **Trusted Entities** assume a role.



Conditions

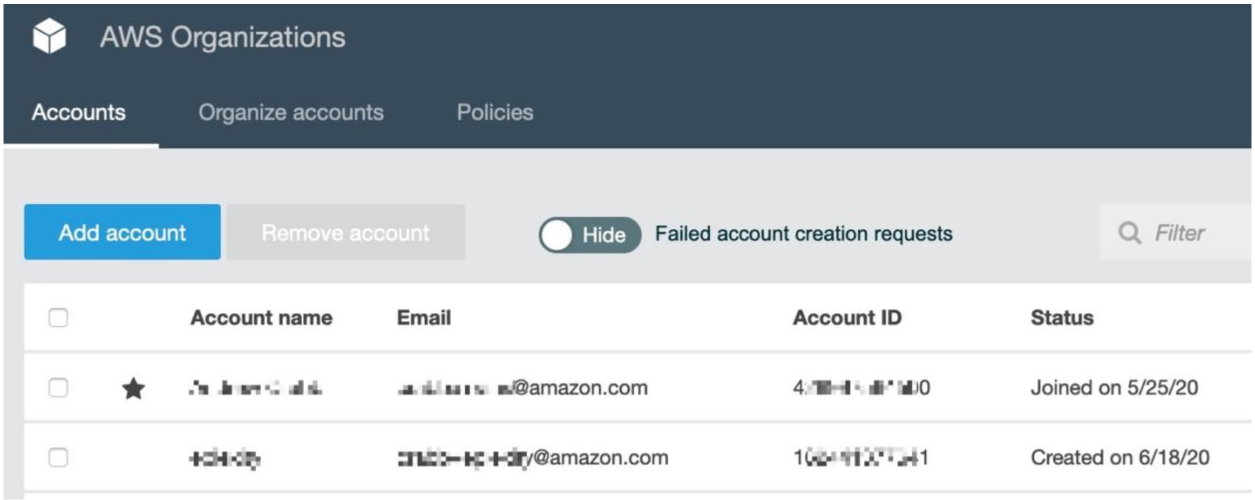
The following conditions define how and when trusted entities can assume the role.

Condition	Key	Value
StringEquals	sts:ExternalId	[redacted]

Figure 33: Defining Conditions for Trusted Entities

AWS Organizations

An AWS Organization is created in the **Master** account. The **Master** account is discussed in the Account Structure section of the Service Workbench User Guide . The AWS Organization use the **Master** account to create a separate account for each deployment. The account's name is the **Stage Name**used. **Figure 34** shows a screenshot image of the AWS Organizations '**Accounts**' tab.

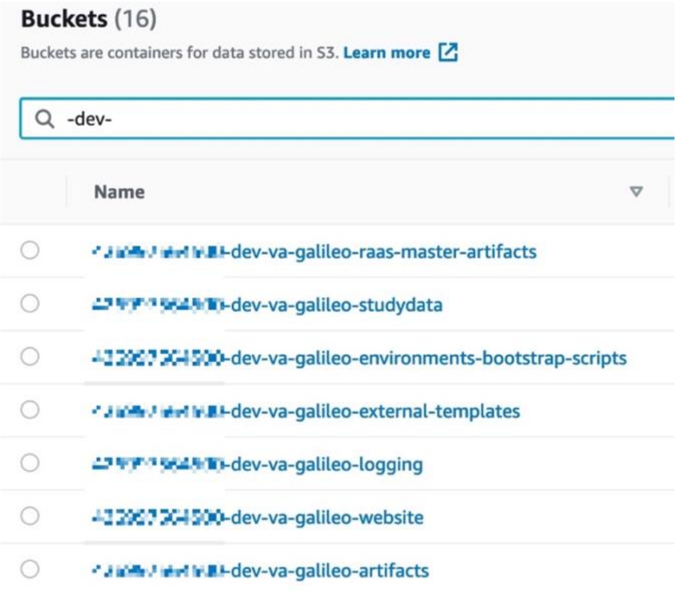


		Account name	Email	Account ID	Status
□	★	[redacted]	[redacted]@amazon.com	[redacted]	Joined on 5/25/20
□		[redacted]	[redacted]@amazon.com	[redacted]	Created on 6/18/20

Figure 34: AWS Organizations Account Page

Amazon S3

Multiple Amazon S3 buckets are created by Service Workbench. Filtering by **Stage Name** shows the Amazon S3 buckets for a deployment. **Figure 35** shows the Amazon S3 buckets for the Service Workbench deployment.



Buckets (16)

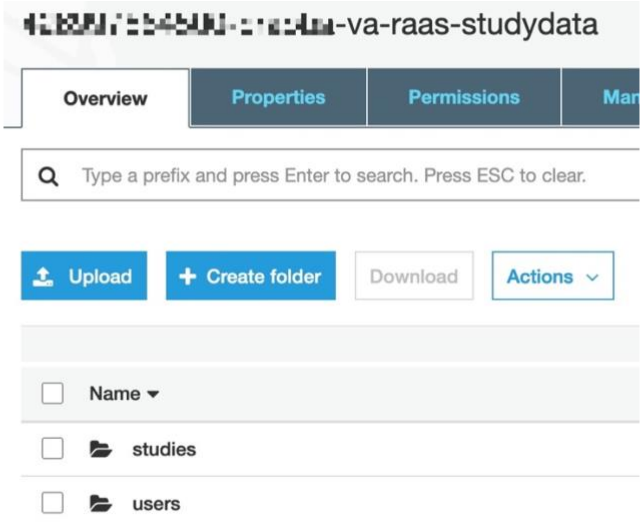
Buckets are containers for data stored in S3. **Learn more**

-dev-

	Name
○	[redacted]-dev-va-galileo-raas-master-artifacts
○	[redacted]-dev-va-galileo-studydata
○	[redacted]-dev-va-galileo-environments-bootstrap-scripts
○	[redacted]-dev-va-galileo-external-templates
○	[redacted]-dev-va-galileo-logging
○	[redacted]-dev-va-galileo-website
○	[redacted]-dev-va-galileo-artifacts

Figure 35: Amazon S3 Buckets for a Service Workbench Deployment

The '**studydata**' bucket contains all the data for the various studies in this deployment at the individual and organization level. **Figure 36** displays an image of the contents within the studydata bucket.



[redacted]-va-raas-studydata

□	Name
□	studies
□	users

Figure 36: Amazon S3 StudyData Bucket

AWS Cost Explorer

Service Workbench has the ability to show actual cost incurred by workspaces running under the Master account. This is using the AWS Cost Explorer service in the AWS Management Console. AWS Cost Explorer must be manually set up for each Master account once in order to allow requests for cost data to process. Setting this up requires background processes to complete in the Master account, which can take up to 24 hours.

Prepare the Master Account

This step is only required if Service Workbench is to be used to vend accounts in the AWS Organization, using the '**Create Account**' mechanism. If Service Workbench is to use only billing accounts imported through the '**Add Account**' mechanism, this step can be omitted.

In this step, deploy the **prepare_master_acc** SDC in the directory main/solution/prepare-master-acc. This will create in the Master account a role that allows **AssumeRole**, and has the **Main** account as its trusted entity. If you have deployed Service Workbench in a **Master** account, the **Main** account is also the **Master** account, and the trusted entity will be the same account ID as the **Master** account. If you have deployed Service Workbench in a **Member** account, the **Main** account is the **Member** account, and the trusted entity (the **Member** account) will be a different account ID than the **Master** account.

The default settings in this step are:

- **Main Account ID**: The current AWS Account ID
- **External ID**: The string **workbench**

These defaults are sufficient if you are deploying Service Workbench from the **Master**account, and the default profile has permissions for the **Master** account, since the **Main**account is also the **Master** account. If deploying Service Workbench in a **Member** account, you must create a configuration file to specify the **Main** account ID and the Profile to use. This profile must have permissions for the **Master** account.

Create a Configuration File

If deploying Service Workbench in an account other than that accessed by the current default profile, create a stage-named configuration file in the directory main/solution/prepare-master-acc/config/settings by copying example.yml to <stage>.yml. Edit the file as appropriate:

- **awsProfile**: The AWS Credentials profile with permissions for the Master account.
- **mainAccountID**: The 12 digit AWS Account ID for the Main AWS account, where the solution is deployed.
- **externalId**: As desired. The string **workbench** is often used. This string will be needed when creating an AWS account within Service Workbench.

Deploy the Prepare Master Account SDC

To deploy the prepare_master_acc SDC, perform the following:

1. Read the file: main/solution/prepare-master-acc/README.md.
2. Deploy the **Master** account SDC from the directory, main/solution/prepare-master-acc:

```
pnpx sls deploy --stage <stage>
```

3. To display the ARN of the **Master Role**, from the same directory:

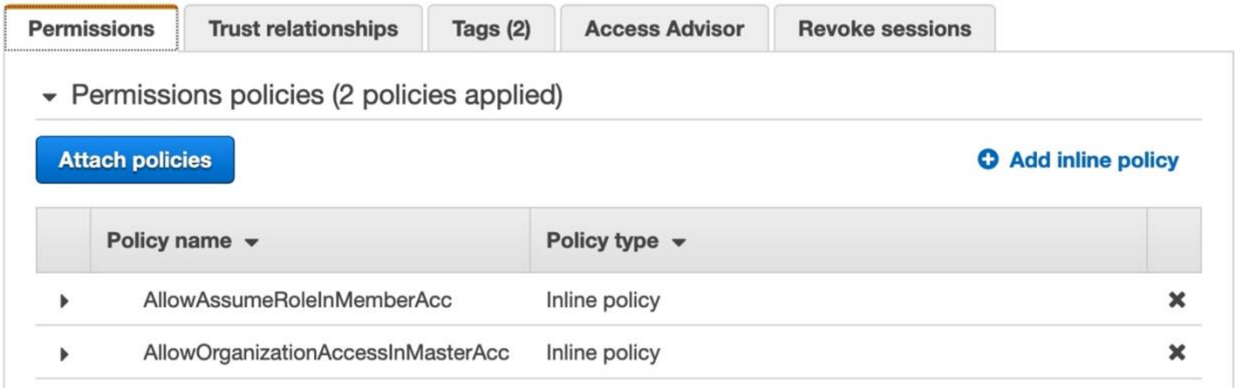
```
pnpx sls info --verbose --stage <stage>
```

The **Master Role ARN** will be needed when adding accounts within Service Workbench.

***Note**: Running the convenience script scripts/master-account-deploy.sh <stage> will perform the same steps as pnpx sls deploy, above.*

Master Role

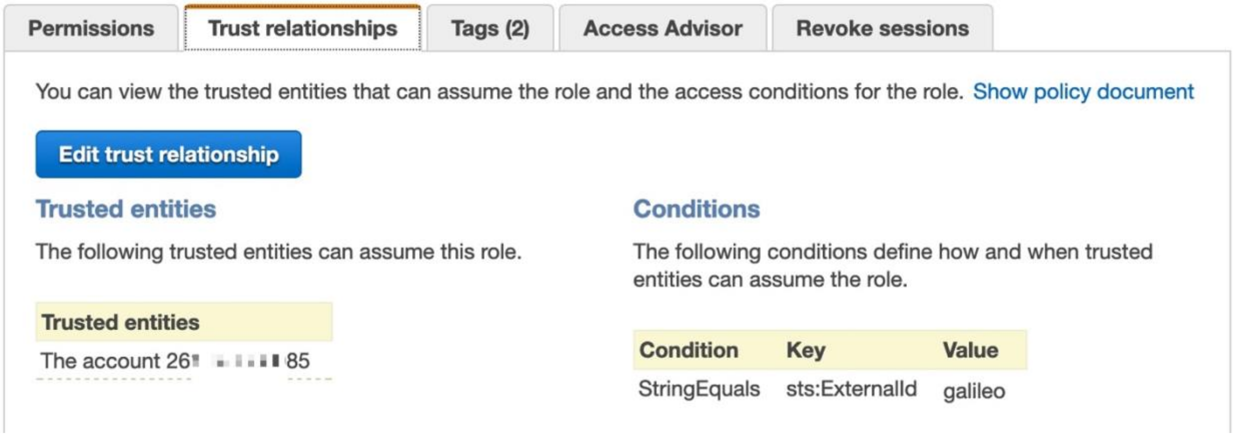
The newly-created role will contain the String **MasterRole**, will have two policies, and will trust the **Main** account (see **Figure 37** and **Figure 38**).



Permissions policies (2 policies applied)

Policy name	Policy type
AllowAssumeRoleInMemberAcc	Inline policy
AllowOrganizationAccessInMasterAcc	Inline policy

Figure 37: Permissions Policies



You can view the trusted entities that can assume the role and the access conditions for the role.

Trusted entities

The following trusted entities can assume this role.

Trusted entities
The account 26[redacted]85

Conditions

The following conditions define how and when trusted entities can assume the role.

Condition	Key	Value
StringEquals	sts:ExternalId	galileo

Figure 38: Trusted Entities