AWS Service Catalog allows organizations to create and manage catalogs of IT services that are approved for use on AWS. AWS Service Catalog allows you to centrally manage commonly deployed IT services in AWS and helps you achieve consistent governance and meet your compliance requirements while enabling users to quickly deploy only the approved IT services they need.

In this lab, you will learn how 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. In **Task 1**, you will set up your AWS Service Catalog of products by:

- Viewing how your catalog has been set up.
- Testing your Service Catalog by deploying an EC2 instance of type t2.medium.

In **Task 2**, you will perform the following operational actions:

- Set up your Service Actions library.
- Add start/stop actions to Amazon EC2 product.
- Test the service actions by launching an EC2 product and then performing actions like start/stop on it.

Service Catalog Concepts

- A product is an IT service that you want to make available for deployment on AWS. You create a product by importing a CloudFormation template.
- A provisioned product is a CloudFormation stack. When an end user launches a product, the AWS Service Catalog provisions the product in form of a CloudFormation stack.
- A portfolio is a collection of products, together with the configuration information. You can use portfolios to manage the user access to specific products.
- Constraints control the way users can deploy a product.
  - Launch Constraints - with launch constraints, you can specify a role that the AWS Service Catalog can assume to launch a product from the portfolio. This means that you don’t need to give permissions necessary to deploy the resource to your IAM user.
  - Notification Constraints - A notification constraint specifies an Amazon SNS topic to receive notifications about stack events.
  - Template Constraints - To limit the options that are available to end users when they launch a product, you apply template constraints.
- User roles:

- ○ **Service Catalog Administrator** - This user has **AWSServiceCatalogAdminFullAccess** policy associated with it. You will use this user for configuring your AWS Service Catalog.
- ○ **Service Catalog End user** - This user has **AWSServiceCatalogEndUserFullAccess** policy associated with it. You will use this user for launching products from available catalog.

Task 1: Provisioning

To enable provisioning, you will create a catalog of approved IT resources (Servers, databases, etc) for your organization. You will then grant your end users access to your catalog. Next, your end users will request cloud resources (e.g. an EC2 instance, an Amazon workspaces desktop, etc) from your catalog.

For your convenience, a sample catalog has been already set up for you. In this task, you will view the sample catalog and provision resources from the sample catalog.

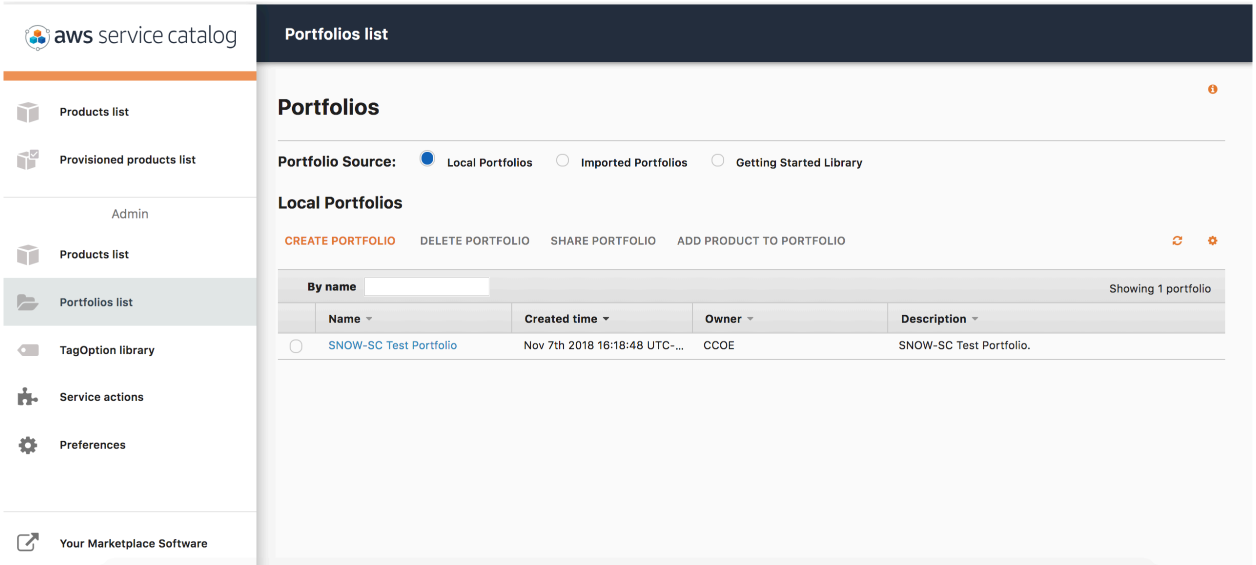
1. Make sure that you are in the correct region by looking at value of **Region** from CloudFormation output you noted earlier. This is the region code that your lab has been set up for.

You can use the AWS Regions and Endpoints page to identify the region that your region code is associated with.

2. Switch your role to **service_catalog_administrator** by:
   - Copying the value of **SwitchRoleSCAdmin** from CloudFormation output you noted earlier.
   - Pasting the value into the Management Console browser tab.
   - Clicking **Switch Role** button
3. In **AWS Management Console**, ensure you are in the same region as before.

4. On the Services menu, search and then choose **Service Catalog**.

The screen will be divided into two panels. Please see the sample screen below. Note, if you do not see a left panel as shown in the following screen, use chrome browser and reduce the resolution.



Portfolios list

Portfolios

Portfolio Source: ● Local Portfolios ○ Imported Portfolios ○ Getting Started Library

Local Portfolios

By name Showing 1 portfolio

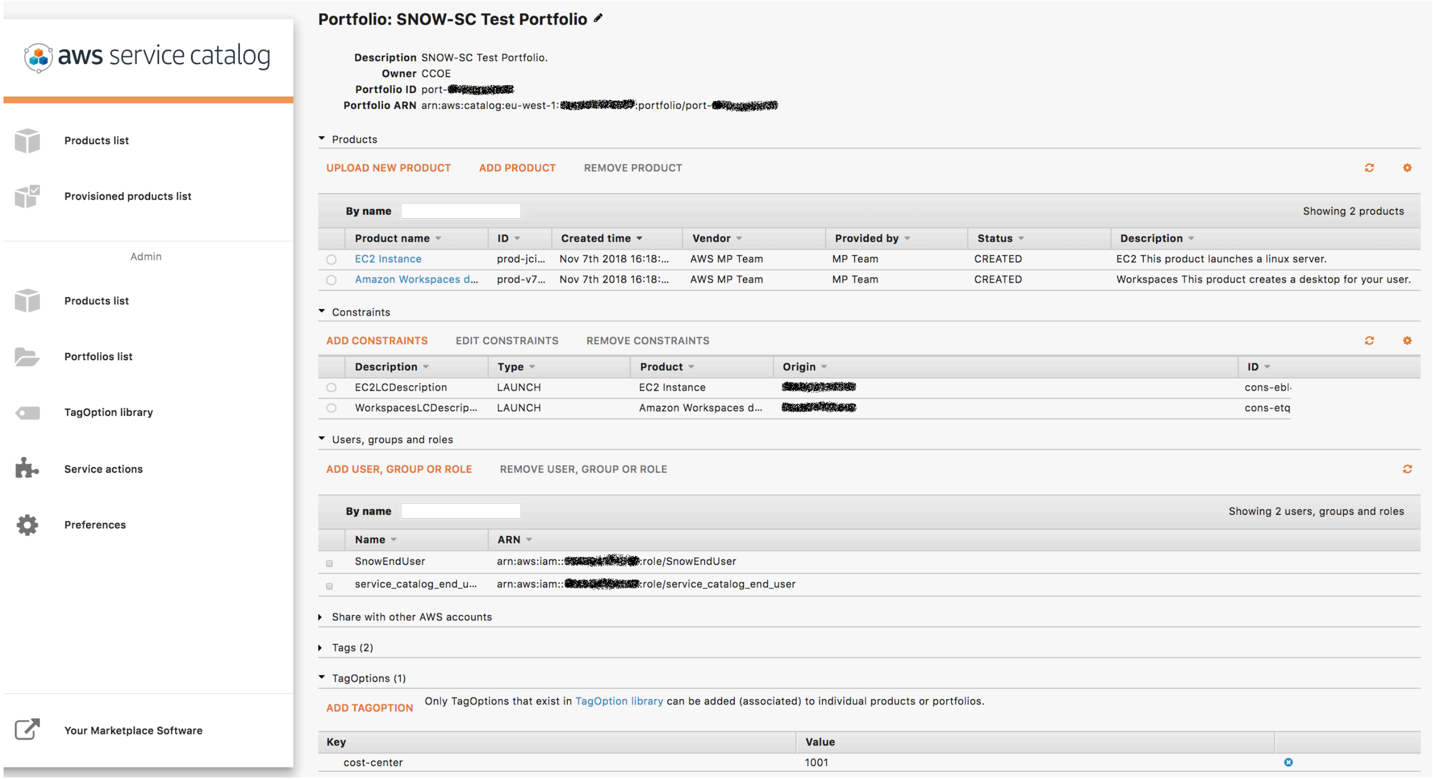
	Name	Created time	Owner	Description
○	SNOW-SC Test Portfolio	Nov 7th 2018 16:18:48 UTC-...	CCOE	SNOW-SC Test Portfolio.

5. Below the **Admin** section, click **Portfolios list**.

6. The right panel will display portfolios available.

7. Click the **SNOW-SC Test Portfolio**.

This will open the portfolio that was created for you.



Portfolio: SNOW-SC Test Portfolio

Description SNOW-SC Test Portfolio.
Owner CCOE
Portfolio ID port-[redacted]
Portfolio ARN arn:aws:catalog:eu-west-1:[redacted]:portfolio/port-[redacted]

Products

By name Showing 2 products

	Product name	ID	Created time	Vendor	Provided by	Status	Description
○	EC2 Instance	prod-jci...	Nov 7th 2018 16:18:...	AWS MP Team	MP Team	CREATED	EC2 This product launches a linux server.
○	Amazon Workspaces d...	prod-v7...	Nov 7th 2018 16:18:...	AWS MP Team	MP Team	CREATED	Workspaces This product creates a desktop for your user.

Constraints

	Description	Type	Product	Origin	ID
○	EC2LCDescription	LAUNCH	EC2 Instance	[redacted]	cons-ebl·
○	WorkspacesLCDescrip...	LAUNCH	Amazon Workspaces d...	[redacted]	cons-etq

Users, groups and roles

By name Showing 2 users, groups and roles

	Name	ARN
□	SnowEndUser	arn:aws:iam::[redacted]:role/SnowEndUser
□	service_catalog_end_u...	arn:aws:iam::[redacted]:role/service_catalog_end_user

Share with other AWS accounts

Tags (2)

TagOptions (1)

Only TagOptions that exist in TagOption library can be added (associated) to individual products or portfolios.

Key	Value	
cost-center	1001	⊗

This is the ***portfolio management screen***.

View Amazon EC2 product

In this section you will enabling self-service of EC2 instances for your end users by creating and making a standardized CloudFormation template available for your end users. For your convenience, an EC2 product has been set up for you.

8. On the **portfolio management** screen of **SNOW-SC Test Portfolio**, notice that there is a product named **EC2 instance**.

You may optionally click on the product to open it and subsequently on the version – **V1.0** to see the CloudFormation template.

Whenever your end users request an EC2 instance using EC2 instance product, the AWS Service Catalog will run the CloudFormation template you configured and create a CloudFormation Stack.

Note: If you looked at the details of your product from the same browser tab, return to the SNOW-SC Test Portfolio page by:

   a. Clicking **Portfolio list** in the left navigation pane.
   b. Clicking **SNOW-SC Test Portfolio**

View workspaces product

The amazon workspaces product can be created to allow your end users to create an Amazon Workspaces desktop in self-service manner. For your convenience, a product called “**Amazon Workspaces Desktop**” had been created for you.

On the portfolio management screen of **SNOW-SC Test Portfolio**, notice that there is a product named **Amazon Workspaces Desktop**.

You may optionally click on the **Amazon Workspaces Desktop** product and subsequently on the version to see the CloudFormation template.

Whenever your end users request a workspaces desktop using Workspaces desktop product, the AWS Service Catalog will run the CloudFormation template you configured and create a CloudFormation Stack.

Note - If you looked at the details of your product, return to the **SNOW-SC Test Portfolio**.

View a cost-center tag-option that gets associated with all your provisioned products

On the portfolio management screen of SNOW-SC Test Portfolio, expand **TagOptions**.

You will see that **cost-center=1001** tag-option has been associated with your portfolio. This means that any taggable product provisioned via users of this portfolio will have the **cost-center=1001** tag associated with it. You can click the **TagOption Library** option link to see your library of all tag-options configured across all portfolios.

View launch constraints

9. On the portfolio management screen of **SNOW-SC Test Portfolio**, expand **Constraints**.

10. Adjust the **Description** column's width to view the complete description of the constraint.

Your end user - **service_catalog_end_user** does not have IAM permissions to create an EC2 instance or request an Amazon workspaces desktop. This is because you don't want your end users to launch *any* EC2 instance. You only want them to launch EC2 instances that you have approved. To enable your end user to launch approved instances, there is a pre-created role that has permissions to create an EC2 instance with the product(s) within the portfolio. Instead of having to give access to your end user you can give a role to the AWS Service Catalog to assume. This will allow it to run the CloudFormation templates that you configured.

Similarly, you can see that another launch constraint has been set up for enabling your end users to launch Amazon workspaces desktop product without giving them any workspaces specific IAM permissions.

Verify that the service_catalog_end_user IAM role has been granted the access to your portfolio

11. On the portfolio management screen of SNOW-SC Test Portfolio, expand **Users, groups and roles**.

12. Notice that **service_catalog_end_user** has been already granted access.

The end user, **service_catalog_end_user** will be able to request an EC2 instance/Amazon workspaces desktop. Whenever they request to provision a resource via AWS service catalog, the corresponding Cloudformation template you configured will be run by AWS Service Catalog using the launch role you configured. Notice that the portfolio access has been already granted to an end user called ***SnowEndUser*** which you will use in the group **Task 3**. You can proceed to the next section.

If you need to grant other users/roles/groups access, you can add them by clicking **ADD USER, GROUP OR ROLE**. However, in this lab it is not necessary to add a user, role, or group.

Test Service Catalog by deploying an EC2 instance of type t2.medium

13. Switch your role to **service_catalog_end_user** by:
    a. Copying the value of **SwitchRoleSCEndUser** from CloudFormation output you noted earlier.
    b. Pasting the value into the Management Console browser tab.
    c. Clicking **Switch Role**
14. On the Services menu, click **Service Catalog**.

15. In **AWS Management Console**, ensure you are in the same region as before.

16. In the **Products list** page, click **EC2 Instance**.

17. Click **LAUNCH PRODUCT**

18. On the **Product Version** page, configure:

    a. **Name:** `My-EC2-instance`
    b. Select **v1.0**
19. Click **NEXT**
20. On the **Parameters** page, configure:
    - **SubnetID:** choose any subnet with the word **public** in its name
    - **InstanceType:** *t2.medium*.
    - **Security Group:** *InstanceSecurityGroup SNOW-SC-Workshop*.
    - **AMI:** Paste the value of **AMI** from CloudFormation output you noted earlier.
21. Click **NEXT**.
22. On the **TagOptions** page notice that cost-center has auto-populated. Optionally, you can add additional tags that will get associated with the taggable AWS resource created by the CloudFormation template)
23. Click **NEXT**.
24. On the **Notifications** page, click **NEXT**.

25. On the **Review** page, review the configuration information.

26. Click **LAUNCH**.

This will create a CloudFormation stack. The initial status of the product is shown as **Under change**.

27. Wait a minute, then refresh the screen.

After the product is launched, the status at the top of the page will become **Available**. This means you have successfully launched an EC2 instance from AWS Service Catalog. In the next section, you will review the EC2 instance you created. You have been given **AmazonEC2ReadOnlyAccess** so that you can view the EC2 Management Console.

28. In a new browser tab, open `https://console.aws.amazon.com/ec2/`

29. In the left navigation pane, click **instances**.

30. You will see an EC2 instance of type **t2.medium** running. Despite not having permissions to launch an EC2 instance, you were able to launch an EC2 instance.

31. Select the box next to your **t2.medium** instance.

32. Click the **Tags** tab. The tags associated with your EC2 instance will be displayed. Notice that a **cost-center** tag has been associated. There are also other tags that service catalog associates with the supported provisioned product.

33. In the **Actions** menu, click **Instance State**, **Stop**.

34. Click **Yes, Stop.**

35. Notice that you are **not** able to stop the EC2 instance. This is because you do not have permissions to stop/start this EC2 instance. In next section, you will learn how to configure service actions to stop/start only your EC2 instance via AWS Service Catalog.

36. Do not close the tab. You will come back to this tab later.

Learnings: You have just learned how to create a portfolio for your end users that will contain two products. However, you will have multiple user-groups each having requirements for launching specific AWS services. To enable these disjoint groups to launch resources, you can create multiple portfolios. Each portfolio can contain only those IT services(backed by CloudFormation templates) that the group needs.

By using AWS Service Catalog, you can allow your end users to **only** create resources using Cloudformation template that you approve of within guardrails that you define. you can additionally also associate template constraints to allow them to specify only specific values for the CloudFormation parameters. For example:

- To let them launch an EC2 instance of t2.micro/t2.medium only.
- To let them launch EC2 instance only in specific subnet
- and many more.

To learn more about template constraints go to the AWS Service Catalog Template Constraints page.

Task 2: Operational Actions

Once your catalog is ready, your end users can provision IT resources. However, there are other operational considerations that you need to think about before making these available to your end users. Things like:

- Who reboots the EC2 instance provisioned by your end users if it needs to be rebooted?
- Who stops the EC2 instance if the EC2 instance is not going to be used over the weekend?

In most cases, the answer to the above questions is the **End User**. The principle of least privilege dictates that your IAM users should have only access they need to perform actions they are supposed to. To enforce this, in past, you might have written IAM policies based on resource IDs/Tags.

Within AWS Service Catalog, you can simply give them **AWSServiceCatalogEndUserFullAccess** and not give permissions to manage the individual resources. The management of individual resources can be achieved via Service Actions. In this section, you will enable your end users to manage only their provisioned products. You will associate following operational actions with your end users:

- Stop EC2 server (using AWS provided automation document)
- Start EC2 server (using AWS provided automation document)

Here is a quick recap of the important concept:

- **Service Catalog Service actions** - AWS Service Catalog, used by enterprises to organize and govern cloud resources on AWS, enables your end users to perform self-service actions on your provisioned products, without needing to grant end users full access to AWS services. Self-service actions are based on AWS Systems Manager automation documents, and you can discover allowed public actions in the Service Catalog Self-Service Action library. A library of pre-defined public actions is available- which include stop, start, and reboot for EC2 and RDS instances. You can also define custom, private actions. Your end users can invoke these actions without needing to contact your administrator. You can define an IAM role for invocation and associate actions with Service Catalog products and product versions. After association, they become available to your users, as actions, on their provisioned products. For more information on Service actions, see the documentation page.

In this task you will:

- Set up your service actions library.
- Add start/stop actions to EC2 instance.
- Start/Stop EC2 instance you provisioned via AWS Service Catalog

Set up your service actions library

Next, you will set up your service actions library. In your service actions library, you will populate service actions for starting/stopping EC2 instance.

37. Switch your role to **service_catalog_administrator** by:
    a. Copying the value of **SwitchRoleSCAdmin** from CloudFormation output you noted earlier.
    b. Pasting the value into the Management Console browser tab.
    c. Clicking **Switch Role**

38. On the Services menu, click **Service Catalog**.

39. In **AWS Management Console**, ensure you are in the same region as before.

40. In the left navigation pane under **Admin**, click **Service actions**.

41. Click **Create new action** then configure:
    a. Select **AWS-StartEC2Instance**.
    b. Click **Next**
    c. Scroll down to the **Permissions** section and then select **Use product launch role**
    d. Click **Create action**

42. Repeat **step #41** to create an **AWS-StopEC2Instance** action.

Your Service Actions library is now set up with two actions. Notice that the Service actions library also offers an optional **AWS-RestartEC2Instance** action. However, for this workshop, you will not configure restart action.

You chose **product launch role** because you did not want to give additional permissions to the end users to manage their IT resources outside AWS Service Catalog. By choosing product launch role, you are relying on a launch role you configured earlier in the lab to have necessary permissions to perform these actions.

Associate start/stop actions to EC2 Instance product

In this section, you will add the two actions you just imported to your service actions library, with the EC2 product version. To do so:

43. In the left navigation pane, below **Admin**, click **Products list**.

44. Click **EC2 Instance**.

45. Click **v1.0**.

46. Under **Service actions**, click **ASSOCIATE AN ACTION**, then:

    a. Select **AWS-StartEC2Instance**.
    b. Click **Associate action**

47. Under **Service actions**, click **ASSOCIATE AN ACTION**, then:

    a. Select **AWS-StopEC2Instance**.
    b. Click **Associate action**

Now that you have associated these actions with your version, your end users will be able to perform these actions on their provisioned EC2 instances.

Start/Stop EC2 instance you provisioned via AWS Service Catalog

In this task, you will Start/Stop the EC2 instance that you launched earlier via service actions.

Test Service Catalog by deploying an EC2 instance of type t2.medium

48. Switch your role to **service_catalog_end_user** by:

    a. Copying the value of **SwitchRoleSCEndUser** from CloudFormation output you noted earlier.
    b. Pasting the value into the Management Console browser tab.
    c. Pressing **Enter**.
    d. Clicking **Switch Role**

49. On the Services menu, click **Service Catalog**.

50. In **AWS Management Console**, ensure you are in the same region as before.

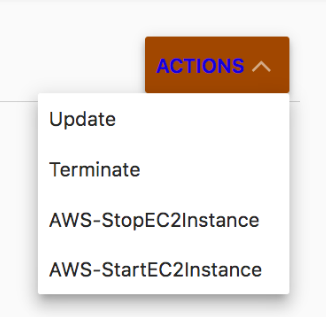
Manage your EC2 instance via Service actions

In this section, you will manage your EC2 instance via AWS Service Catalog.

51. In left navigation pane, click **Provisioned products list**.

52. Click on **My-EC2-Instance**.

53. Click **ACTIONS**. Notice that the service actions automatically became available to the EC2 instance that was launched before service actions were set up.



54. Click **AWS-StopEC2Instance**.

55. Click **RUN ACTION**.

This will take a few seconds to run. Wait a minute, then refresh the screen. The request will succeed and your EC2 instance will be stopped.

Verify Your EC2 Instance has Stopped

56. Return to your **EC2 Management Console** browser tab you had opened earlier.

57. In the **Instances** section, click the refresh icon.

Your EC2 instance of type **t2.medium** will be listed as stopped. Notice that you were able to manage only your EC2 instance from AWS Service Catalog. You are not able to manage an instance-type of **t2.micro** EC2 instance that is running in the same account. For this workshop, you don’t need to start EC2 instance using **AWS-StartEC2Instance** service action.

58. In the **Service Catalog** tab, Provisioned products list, on click on **Terminate** action.

59. Verify that the **terminate** action works. It should terminate the EC2 instance.

Learnings:

In this section, you learned how to enable your users to manage only their EC2 instance. Service actions are based on systems manager automation documents. You can create your own custom service actions by:

- Authoring an Automation document.
- Populating it in your service action library.
- Associating it with your product version.

With the support for **aws:executeAwsApi** automation action you can enable your end users to perform most API actions while ensuring that they can manage only their resources within boundaries you define. For more information on **aws:executeAwsApi**, see documentation.

Additional Resources

- Service Catalog Documentation
- Service Catalog Developer Resources
- AWS Service Catalog Partners