Cost Management Service 1-latest

Integrating Microsoft Azure data into cost management

Learn how to add your Microsoft Azure integration and RHEL metering

Last Updated: 2024-11-01

Cost Management Service 1-latest Integrating Microsoft Azure data into cost management

Learn how to add your Microsoft Azure integration and RHEL metering

Legal Notice

Abstract

Learn how to add a Microsoft Azure integration to cost management.

Table of Contents

PART I. CHOOSING A BASIC OR ADVANCED AZURE INTEGRATION

To create an Azure integration, first decide if you want to take a basic or advanced integration path.

Basic

For the basic option, go to Creating a Microsoft Azure integration: Basic.

The basic path enables cost management to directly read your billing reports from Azure at a scope that you indicate.

Advanced

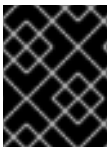
For the advanced option, go to Creating a Microsoft Azure integration: Advanced.

The advanced path enables you to customize or filter your data *before* cost management reads it. You might also use the advanced path if you want to share billing data only to certain Red Hat products. The advanced path has more complex set-up and configuration.

NOTE

You must select either basic or advanced, you cannot choose both.

CHAPTER 1. CREATING A MICROSOFT AZURE INTEGRATION: BASIC



IMPORTANT

If you want to create an Azure integration by using the advanced path, do not complete the following steps. Instead, go to Creating a Microsoft Azure integration: Advanced.

If you are using RHEL metering, after you integrate your data with cost management, go to Adding RHEL metering to a Microsoft Azure integration to finish configuring your integration for RHEL metering.

You must create a Microsoft Azure integration for cost management from the Integrations page and configure your Microsoft Azure account to allow cost management access.

To create an Azure integration, you will complete the following tasks:

1. Create a storage account and resource group
2. Choose the appropriate scope for your cost export
3. Configure a Storage Account Contributor and Reader roles for access
4. Schedule daily cost exports

Azure is a third-party product and its UI and documentation can change. The instructions for configuring third-party integrations are correct at the time of publishing. For the most up-to-date information, see the Microsoft Azure's documentation.

1.1. ADDING A MICROSOFT AZURE ACCOUNT

Add your Microsoft Azure account as an integration so cost management can process the cost and usage data.

Prerequisites

You must have a Red Hat user account with Cloud Administrator entitlements.

In cost management:

1. Click **Settings Menu**  > **Integrations**.
2. In the **Cloud** tab, click **Add integration**.
3. In the **Add a cloud integration** wizard, select **Microsoft Azure** and click **Next**.
4. Enter a name for your integration and click **Next**.
5. In the **Select application** step, select **Cost management** and click **Next**.
6. In the **Specify cost export scope** step, select **I am OK with sending the default data to Cost Management**.

- If you are registering RHEL usage billing, select **Include RHEL usage**. Otherwise, proceed to the next step.

7. Select the scope of your cost data export from the menu. You can export data at the subscription level or by other scopes in your subscription.

8. Copy the command that is generated.

In your Microsoft Azure account:

9. Click **Cloud Shell** and run the command that you copied from cost management. Copy the returned value.

In cost management:

10. In the **Specify cost export scope** step, paste the value that you copied from Microsoft Azure into **Cost export scope**.

11. Click **Next**.

You will continue using the wizard in the following sections.

1.2. CREATING A MICROSOFT AZURE RESOURCE GROUP AND STORAGE ACCOUNT

Create a storage account in Microsoft Azure to house your cost data and metrics. In the **Add a cloud integration** wizard in cost management, enter the storage account name in the corresponding fields.

Prerequisites

You must have a Red Hat user account with Cloud Administrator entitlements.

In your Microsoft Azure account:

1. Search for **storage** and click **Storage accounts**.

2. On the **Storage accounts** page, click **Create**.

3. On the **Create a storage account** page, in the **Resource Group** field, click **Create new**. Enter a name and click **OK**. In this example, use **cost-data-group**.

4. In **Instance details**, enter a name in the **Storage account name** field. In this example, use **costdata**.

5. Copy the names of the resource group and storage account so that you can add them to Red Hat Hybrid Cloud Console later.

6. Click **Review**.

7. Review the storage account and click **Create**.

In cost management:

8. In the **Add a cloud integration** wizard, paste the resource group and storage account names that you copied into **Resource group name** and **Storage account name**

9. Click **Next**.

You will continue using the wizard in the following sections.

1.3. CONFIGURING A DAILY MICROSOFT AZURE DATA EXPORT SCHEDULE

Next, set up an automatic export of your cost data to your Microsoft Azure storage account so that cost management can retrieve your data daily.

In your Microsoft Azure account:

1. In the search bar, enter "cost exports" and click the result.
2. Click **Create**.
3. Under **Select a template**, click **Cost and usage (actual)** to export your standard usage and purchase charges.
4. Follow the steps in the Azure wizard.
   - Select the correct subscription and **Storage account** that you created in the previous sections.
   - You must set **Format** to **CSV**.
   - Set **Compression type** to **None** or **Gzip**.
5. Review the information and click **Create**.

In cost management:

6. Return to the **Add a cloud integration** wizard and complete the steps in **Daily export**.
7. Click **Next**.

You will continue using the wizard in the following sections. For more help with creating exports in Azure, see Microsoft's documentation.

1.4. FINDING YOUR MICROSOFT AZURE SUBSCRIPTION ID

Find your **subscription_id** in the Microsoft Azure Cloud Shell and add it to the **Add a cloud integration** wizard in cost management.

In your Microsoft Azure account:

1. Click **Cloud Shell**.
2. Enter the following command to get your Subscription ID:

```
az account show --query "{subscription_id: id }"
```

3. Copy the value that is generated for **subscription_id**.

Example response

```
{
   "subscription_id": 00000000-0000-0000-000000000000
   }
```

In cost management:

4. In the **Subscription ID** field of the **Add a cloud integration** wizard, paste the value that you copied in the previous step.
5. Click **Next**.

You will continue using the wizard in the following sections.

1.5. CREATING MICROSOFT AZURE ROLES FOR RED HAT ACCESS

To grant Red Hat access to your data, you must configure dedicated roles in Microsoft Azure. If you have an additional resource under the same Azure subscription, you might not need to create a new service account.

In cost management:

1. In the **Roles** section of the **Add a cloud integration** wizard, copy the **az ad sp create-for-rbac** command to create a service principal with the Cost Management Storage Account Contributor role.

In your Microsoft Azure account:

2. Click **Cloud Shell**.
3. In the cloud shell prompt, paste the command that you copied.
4. Copy the values from the returned data for the client ID, secret, and tenant:

Example response

```
{
   "client_id": "00000000-0000-0000-000000000000",
   "secret": "00000000-0000-0000-000000000000",
   "tenant": "00000000-0000-0000-000000000000"
}
```

In cost management:

5. Return to the **Add a cloud integration** wizard and paste the values that you copied into their corresponding fields on the **Roles** page.
6. Copy the second **az** role assignment command that is generated from the wizard.

In your Microsoft Azure account:

7. Return to the cloud shell prompt and paste the command to create a **Cost management reader** role.

In cost management:

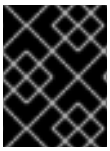
8. Return to the **Add a cloud integration**wizard and click **Next**.

9. Review the information that you provided and click **Add**.

1.6. VIEWING YOUR DATA

You have now successfully created your basic integration. To learn more about what you can do with your data, continue to Next steps for managing your costs . Do *not* follow the instructions in Creating a Microsoft Azure integration: Advanced.

CHAPTER 2. CREATING YOUR MICROSOFT AZURE INTEGRATION: ADVANCED



IMPORTANT

If you created an Azure integration by using the basic path, do not complete the following steps. Your Azure integration is already complete.

If you are using RHEL metering, after you integrate your data with cost management, go to Adding RHEL metering to a Microsoft Azure integration to finish configuring your integration for RHEL metering.

To share a subset of your billing data with Red Hat, you can configure a function script in Microsoft Azure. This script will filter your billing data and export it to object storage so that cost management can then access and read the filtered data. Add your Microsoft Azure integration to cost management from the Integrations page.

Azure is a third-party product and its processes can change. The instructions for configuring third-party integrations are correct at the time of publishing. For the most up-to-date information, see Microsoft Azure's documentation.

To create an Azure integration, you will complete the following tasks:

1. Create a storage account and resource group.
2. Configure Storage Account Contributor and Reader roles for access.
3. Create a function to filter the data that you want to send to Red Hat.

2.1. ADDING A MICROSOFT AZURE ACCOUNT

Add your Microsoft Azure account as an integration so that cost management can process the cost and usage data.

Prerequisites

- You must have a Red Hat user account with Cloud Administrator entitlements.
- You must have a service account.
- Your service account must have the correct roles assigned in Hybrid Cloud Console to enable cost management access. For more information, see the User Access Configuration Guide .

In cost management:

1. Click **Settings Menu** > **Integrations**.
2. In the **Cloud** tab, click **Add integration**.
3. In the **Add a cloud integration** wizard, select **Microsoft Azure** and click **Next**.
4. Enter a name for your integration and click **Next**.

5. In the **Select application** step, select **Cost management** and click **Next**.
6. In the **Specify cost export scope** step, select **I wish to manually customize the data set sent to Cost Management**.
    - If you are registering RHEL usage billing, select **Include RHEL usage**. Otherwise, proceed to the next step.
7. Click **Next**.

2.2. CREATING A MICROSOFT AZURE RESOURCE GROUP AND STORAGE ACCOUNT

Create a storage account in Microsoft Azure to house your billing exports and a second storage account to house your filtered data.

In your Microsoft Azure account:

1. In the search bar, enter "storage" and click **Storage accounts**.
2. On the **Storage accounts** page, click **Create**.
3. In the **Resource Group** field, click **Create new**. Enter a name and click **OK**. In this example, use **filtered-data-group**.
4. In the **Instance details** section, enter a name in the **Storage account name** field. For example, use **filtereddata**.
5. Copy the names of the resource group and storage account so you can add them to Red Hat Hybrid Cloud Console later.
6. Click **Review**.
7. Review the storage account and click **Create**.

In cost management:

8. In the **Add a cloud integration** wizard, paste the resource group and storage account names that you copied into **Resource group name** and **Storage account name**

You will continue using the wizard in the following sections.

2.3. CREATING A DAILY EXPORT IN MICROSOFT AZURE

Next, set up an automatic export of your cost data to your Microsoft Azure storage account before you filter it for cost management.

In your Microsoft Azure account:

1. In the search bar, enter "cost exports" and click the result.
2. Click **Create**.
3. In **Select a template**, click **Cost and usage (actual)** to export your standard usage and purchase charges.

4. Follow the steps in the Azure wizard:
    - You can either create a new resource group and storage account or select existing ones. In this example, we use **billingexportdata** for the storage account and **billinggroup** for the resource group.
    - You must set **Format** to **CSV**.
    - Set **Compression type** to **None** or **Gzip**.
5. Review the information and click **Create**.

In cost management:

6. Return to the **Add a cloud integration** wizard and complete the steps in **Daily export**
7. Click **Next**.

You will continue using the wizard in the following sections.

2.4. FINDING YOUR MICROSOFT AZURE SUBSCRIPTION ID

Find your **subscription_id** in the Microsoft Azure Cloud Shell and add it to the **Add a cloud integration** wizard in cost management.

In your Microsoft Azure account:

1. Click **Cloud Shell**.
2. Enter the following command to get your Subscription ID:

```
az account show --query "{subscription_id: id }"
```

3. Copy the value that is generated for **subscription_id**.

Example response

```
{
   "subscription_id": 00000000-0000-0000-000000000000
   }
```

In cost management:

4. In the **Subscription ID** field of the **Add a cloud integration** wizard, paste the value that you copied in the previous step.
5. Click **Next**.

You will continue using the wizard in the following sections.

2.5. CREATING MICROSOFT AZURE ROLES FOR RED HAT ACCESS

To grant Red Hat access to your data, you must configure dedicated roles in Microsoft Azure. If you have an additional resource under the same Azure subscription, you might not need to create a new service account.

In cost management:

1. In the **Roles** section of the **Add a cloud integration** wizard, copy the **az ad sp create-for-rbac** command to create a service principal with the Cost Management Storage Account Contributor role.

In your Microsoft Azure account:

2. Click **Cloud Shell**.
3. In the cloud shell prompt, paste the command that you copied.
4. Copy the values that are generated for the client ID, secret, and tenant:

   Example response

```
{
    "client_id": "00000000-0000-0000-000000000000",
    "secret": "00000000-0000-0000-000000000000",
    "tenant": "00000000-0000-0000-000000000000"
}
```

In cost management:

5. Return to the **Add a cloud integration** wizard and paste the values that you copied into their corresponding fields on the **Roles** page.
6. Click **Next**.
7. Review your information and click **Add** to complete your integration.
8. In the pop-up screen that appears, copy the **Source UUID** for your function script.

2.6. CREATING A FUNCTION IN MICROSOFT AZURE

Creating a function in Azure filters your data and adds it to the storage account that you created to share with Red Hat. You can use the example Python script in this section to gather and share the filtered cost data from your export.

Prerequisites

- You must have Visual Studio Code installed on your device.
- You must have the Microsoft Azure functions extension installed in Visual Studio Code. To create an Azure function, Microsoft recommends that you use their Microsoft Visual Studio Code IDE to develop and deploy code. For more information about configuring Visual Studio Code, see Quickstart: Create a function in Azure with Python using Visual Studio Code .

In your Microsoft Azure account:

1. Enter **functions** in the search bar and select **Function App**.
2. Click **Create**.
3. Select a hosting option for your function and click **Select**.
4. On the **Create Function App** page, add your resource group.
    a. In the **Instance Details** section, name your function app.
    b. In **Runtime stack**, select **Python**.
    c. In **Version**, select latest.
5. Click **Review + create**:
    a. Click **Create**.
    b. Wait for the resource to be created and then click **Go to resource** to view.

In Visual Studio Code:

6. Click the **Microsoft Azure** tab and sign in to Azure.
    a. In the **Workspaces** drop-down, click **Azure Functions** which appears as an icon with an orange lightning bolt.
    b. Click **Create Function**.
7. Follow the prompts to set a local location and select a language and version for your function. In this example, select **Python**, **Model 2** and the latest version available.
8. In **Select a template** for your function dialog, select **Timer trigger**, name the function, and then press enter.
9. Set the cron expression to control when the function runs. In this example, use **0 9 * * *** to run the function daily at 9 AM:
    a. Click **Create**.
    b. Click **Open in the current window**.

In your requirements.txt file:

10. After you create the function in your development environment, open the requirements.txt file, add the following requirements, and save the file:

```
azure-functions
pandas
requests
azure-identity
azure-storage-blob
```

In *init*.py:

11. Copy the Python script and paste it into` *init*.py`.

12. Change the values in the section marked **# Required vars to update** to the values that correspond to your environment.

    - The example script uses secrets from Azure Key Vaults to configure your service account **client_id** and **client_secret** as environment variables. You can alternatively enter your credentials directly into the script, although this is not best practice.

    - The default script has built-in options for filtering your data or RHEL subscription filtering. You must uncomment the type of filtering you want to use or write your own custom filtering. Remove the comment from one of the following not both:

        - **filtered_data = hcs_filtering(df)**

        - **filtered_data = rhel_filtering(df)**

    - If you want to write more customized filtering, you must include the following required columns:

      ```
      'additionalinfo', 'billingaccountid', 'billingaccountname', 'billingcurrencycode',
      'billingperiodenddate', 'billingperiodstartdate', 'chargetype', 'consumedservice',
      'costinbillingcurrency', 'date', 'effectiveprice', 'metercategory', 'meterid', 'metername',
      'meterregion', 'metersubcategory', 'offerid', 'productname', 'publishername',
      'publishertype', 'quantity', 'reservationid', 'reservationname', 'resourcegroup', 'resourceid',
      'resourcelocation', 'resourcename', 'servicefamily', 'serviceinfo1', 'serviceinfo2',
      'subscriptionid', 'tags', 'unitofmeasure', 'unitprice'
      ```

    - Some of the columns differ depending on the report type. The example script normalizes these columns and all filtered reports must follow this example.

      ```
      column_translation = {"billingcurrency": "billingcurrencycode", "currency":
      "billingcurrencycode", "instanceid": "resourceid", "instancename": "resourceid",
      "pretaxcost": "costinbillingcurrency", "product": "productname", "resourcegroupname":
      "resourcegroup", "subscriptionguid": "subscriptionid", "servicename": "metercategory",
      "usage_quantity": "quantity"}
      ```

    - To filter the data, you must add dataframe filtering. For example:

        - Exact matching: **df.loc[(df["publishertype"] == "Marketplace")]** Filters out all data that does not have a **publisherType** of Marketplace.

        - Contains: **df.loc[df["publishername"].astype(str).str.contains("Red Hat")]** Filters all data that does not contain Red Hat in the **publisherName**.

        - You can stack filters by using **&** (for AND) and **|** (for OR) with your **df.loc** clause.

        - More useful filters:

          subscriptionid
          : Filters specific subscriptions.

          resourcegroup
          : Filters specific resource groups.

          resourcelocation
          : Filters data in a specific region.

- You can use **servicename**, **servicetier**, **metercategory** and **metersubcategory** to filter specific service types.

13. After you build your custom query, update the custom query in the example script under **# custom filtering basic example #**.
14. Save the file.

In Visual Studio Code:

14. Right click the **Function** window and click **Deploy to Function App**.
15. Select the function app that you created in the previous steps.

2.7. SETTING UP YOUR CREDENTIALS IN AZURE

For help with steps in Azure, see Microsoft's documentation: Azure Key Vault.

In your Microsoft Azure account:

1. Navigate to **Key Vaults**.
2. Click **Create**.
3. Select the resource group that your function is in and follow the Azure wizard to create a new secret.
   a. In **Key vault name**, you can enter any name of your choosing.
   b. In **Access policies**, click **Create new policy**. Then, select **Secret Management** from the templates.
   c. In the tab **Principal**, search for and select your function as the principal.
4. After you complete the wizard and click **Create Vault**, wait until Azure brings you to the successful deployment page. Then, click **Go to resource** to open the **Key Vault** page.
5. In the **Objects** drop-down, select **Secrets**.
6. You must create *two new secrets* for your service account: **client_id** and **client_secret**. Complete the following steps two times to create both:
   a. To create a secret, click **Generate/import**.
   b. In **Create a secret**, enter any name you want for your **client_id** or **client_secret**.
   c. Copy the value in **Secret Identifier**. You will use it later. You can also retrieve this value later.
   d. Repeat the previous three steps until you have a secret for both your **client_id** and **client_secret**.

2.8. ADDING VAULT CREDENTIALS TO YOUR FUNCTION

Next, go to your function in Microsoft Azure and enter information about your secrets.

In your Microsoft Azure account:

1. Navigate to your function.
2. Select **Settings → Environment variables**.
3. Click **Add**. Use the following conventions and replace **YOUR-CLIENT-ID-URI** with the **Secret Identifier** value that you copied previously:
   - Name: **ClientIdFromVault**
   - Value: **@Microsoft.KeyVault(SecretUri=YOUR-CLIENT-ID-URI)**
4. Click **Save**.
5. Repeat the process for **ClientSecretFromVault**. Use the following conventions and replace **YOUR-CLIENT-SECRET-URI** with the **Secret Identifier** value that you copied previously:
   - Value: **@Microsoft.KeyVault(SecretUri=YOUR-CLIENT-SECRET-URI)**

2.9. CONFIGURING FUNCTION ROLES IN MICROSOFT AZURE

Configure dedicated credentials to grant your function blob access to Microsoft Azure cost data. These credentials enable your function to access, filter, and transfer the data from the original storage container to the filtered storage container.

In your Microsoft Azure account:

1. Enter **functions** in the search bar and select your function.
2. In the **Settings** menu, click **Identity**.

Complete the following set of stepstwice, one time for each of the two storage accounts that you created in the section Creating a Microsoft Azure resource group and storage account:

3. Click **Azure role assignments**.
4. Click **Add role assignment**.
5. In the **Scope** field, select **Storage**.
6. In the **Resource** field, select one of your two storage accounts. Our examples used **filtereddata** and **billingeportdata**.
7. In **Role**, select **Storage Blob Data Contributor**.
8. Click **Save**.
9. Click **Add role assignment** again.
10. In the **Scope** field, select **Storage**.
11. In the **Resource** field, select *the same storage account again* .
12. This time, in **Role**, select **Storage Queue Data Contributor**.
13. Click **Save**.
14. Repeat this entire process for the other storage account that you created.

After completing these steps, you have successfully set up your Azure integration.

2.10. VIEWING YOUR DATA

You have now successfully created your advanced integration. To learn more about what you can do with your data, continue to Next steps for managing your costs .

CHAPTER 3. NEXT STEPS FOR CONFIGURING AND VIEWING YOUR DATA

After you add your OpenShift Container Platform and Microsoft Azure integrations, cost management displays the cost and usage data that is related to running your OpenShift Container Platform clusters on Azure. You might have to wait up to 24 hours for your data to display.

On the cost management **Overview** page, your cost data is in the **OpenShift** and **Infrastructure** tabs. To toggle through different views of your cost data, select **Perspective**. You can also use the global navigation menu to view additional details about your costs by cloud provider.

Additional resources

- *Integrating OpenShift Container Platform data into cost management*
- *Integrating Amazon Web Services (AWS) data into cost management*
- *Integrating Google Cloud data into cost management*
- *Integrating Oracle Cloud data into cost management*

3.1. LIMITING ACCESS TO COST MANAGEMENT RESOURCES

After you add and configure integrations in cost management, you can limit access to cost data and resources.

You might not want users to have access to all of your cost data. Instead, you can grant users access only to data that is specific to their projects or organizations. With role-based access control, you can limit the visibility of resources in cost management reports. For example, you can restrict a user's view to only AWS integrations, rather than the entire environment.

To learn how to limit access, see the more in-depth guide *Limiting access to cost management resources* .

3.2. CONFIGURING TAGGING FOR YOUR INTEGRATIONS

The cost management application tracks cloud and infrastructure costs with tags. Tags are also known as labels in OpenShift.

You can refine tags in cost management to filter and attribute resources, organize your resources by cost, and allocate costs to different parts of your cloud infrastructure.

IMPORTANT

You can only configure tags and labels directly on an integration. You can choose the tags that you activate in cost management, however, you cannot edit tags and labels in the cost management application.

To learn more about the following topics, see *Managing cost data using tagging* :

- Planning your tagging strategy to organize your view of cost data
- Understanding how cost management associates tags
- Configuring tags and labels on your integrations

3.3. CONFIGURING COST MODELS TO ACCURATELY REPORT COSTS

Now that you configured your integrations to collect cost and usage data in cost management, you can configure cost models to associate prices to metrics and usage.

A cost model is a framework that uses raw costs and metrics to define calculations for the costs in cost management. You can record, categorize, and distribute the costs that the cost model generates to specific customers, business units, or projects.

In Cost Models, you can complete the following tasks:

- Classifying your costs as infrastructure or supplementary costs
- Capturing monthly costs for OpenShift nodes and clusters
- Applying a markup to account for additional support costs

To learn how to configure a cost model, see *Using cost models*.

3.4. VISUALIZING YOUR COSTS WITH COST EXPLORER

Use cost management Cost Explorer to create custom graphs of time-scaled cost and usage information and ultimately better visualize and interpret your costs.

To learn more about the following topics, see *Visualizing your costs using Cost Explorer*:

- Using Cost Explorer to identify abnormal events
- Understanding how your cost data changes over time
- Creating custom bar charts of your cost and usage data
- Exporting custom cost data tables

CHAPTER 4. UPDATING AN INTEGRATION

If you have added an integration to cost management and want to make changes to it, you can add or remove the applications associated with your integrations in Red Hat Hybrid Cloud Console .

Procedure

1. From Red Hat Hybrid Cloud Console , click **Settings** .
2. Click **Integrations**.
3. Click the more options menu for your integration. Click **Edit**.
4. In **Metered Product**, select **Red Hat Enterprise Linux** from the drop-down to activate metering.

4.1. ADDING RHEL METERING TO A MICROSOFT AZURE INTEGRATION

If you converted from a compatible third-party Linux distribution to Red Hat Enterprise Linux (RHEL) and purchased the RHEL for third party migration listing in Microsoft Azure, you can update an Microsoft Azure integration you created to add RHEL metering.

With RHEL metering, Red Hat processes your bill to meter my hourly RHEL usage associated with a Red Hat offering in Microsoft Azure.

Procedure

1. In Microsoft Azure, tag your instances of RHEL that you want to meter. For more information about tagging your instances of RHEL in Microsoft Azure, see Adding tags to a Microsoft Azure resource.
2. From Red Hat Hybrid Cloud Console , click **Settings** .
3. Click **Integrations**.
4. Click the more options menu for your integration. Click **Edit**.
5. In **Metered Product**, select **Red Hat Enterprise Linux** from the drop-down to activate metering.

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate and prioritize your feedback regarding our documentation. Provide as much detail as possible, so that your request can be quickly addressed.

Prerequisites

- You are logged in to the Red Hat Customer Portal.

Procedure

To provide feedback, perform the following steps:

1. Click the following link: Create Issue.
2. Describe the issue or enhancement in the **Summary** text box.
3. Provide details about the issue or requested enhancement in the **Description** text box.
4. Type your name in the **Reporter** text box.
5. Click the **Create** button.

This action creates a documentation ticket and routes it to the appropriate documentation team. Thank you for taking the time to provide feedback.