Hybrid Committed Spend 1-latest

Integrating Amazon Web Services (AWS) data into hybrid committed spend

Learn how to add and configure your AWS integrations
Integrating Amazon Web Services (AWS) data into hybrid committed spend

Learn how to add and configure your AWS integrations
Abstract

This guide describes how to add an Amazon Web Services (AWS) integration to hybrid committed spend.
# Table of Contents

CHAPTER 1. INTEGRATING AMAZON WEB SERVICES DATA INTO HYBRID COMMITTED SPEND  .......... 3  
1.1. CREATING AN AWS S3 BUCKET FOR STORING YOUR COST DATA  3  
1.2. CONFIGURE AN IAM POLICY TO ENABLE MINIMAL ACCOUNT ACCESS FOR COST AND USAGE CONSUMPTION  4  
1.3. ADDING A AMAZON WEB SERVICES ACCOUNT AND NAMING YOUR INTEGRATION  5  

CHAPTER 2. FILTERING YOUR AMAZON WEB SERVICES DATA BEFORE SENDING IT TO HYBRID COMMITTED SPEND  ................................................................................................................ 7  
2.1. CREATING AN AWS S3 BUCKET FOR STORING YOUR COST DATA  7  
2.2. CREATING AN AWS S3 BUCKET FOR FILTERED DATA REPORTING  8  
2.3. ENABLING MINIMAL ACCOUNT ACCESS FOR COST AND USAGE CONSUMPTION  8  
2.4. ADDING A AMAZON WEB SERVICES ACCOUNT AND NAMING YOUR INTEGRATION  9  
2.5. ENABLING ACCOUNT ACCESS FOR ATHENA  10  
  2.5.1. Configuring Athena for report generation  13  
  2.5.2. Creating a Lambda function for Athena  13  
  2.5.3. Creating a Lambda function to post the report files  15  

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION  ............................................... 17
Integrating Amazon Web Services (AWS) data into hybrid committed spend
CHAPTER 1. INTEGRATING AMAZON WEB SERVICES DATA INTO HYBRID COMMITTED SPEND

To add an AWS account to hybrid committed spend, you must configure your AWS account to provide metrics, then add your AWS account as an integration from the hybrid committed spend user interface.

**NOTE**

You must have a Red Hat account user with Integrations Administrator entitlements before you can add integrations to hybrid committed spend.

When you add your AWS account as an integration, this creates a read-only connection to AWS to collect cost and usage information hourly in hybrid committed spend, but does not make any changes to the AWS account.

Before you can add your AWS account to hybrid committed spend as an integration, you must configure the following services on your AWS account to allow hybrid committed spend access to metrics:

1. An S3 bucket to store cost and usage data reporting for hybrid committed spend
2. An Identity Access Management (IAM) policy and role for hybrid committed spend to process the cost and usage data
3. A Hybrid Committed Spend integration on Red Hat Hybrid Cloud Console

As you will complete some of the following steps in the AWS console, and some steps in the hybrid committed spend user interface, keep both applications open in a web browser.

Add your AWS integration to hybrid committed spend from the Integrations page.

**NOTE**

Because non-Red Hat products and documentation can change without notice, instructions for configuring the third-party integrations provided in this guide are general and correct at the time of publishing. See the AWS documentation for the most up-to-date and accurate information.

1.1. CREATING AN AWS S3 BUCKET FOR STORING YOUR COST DATA

You must create an Amazon S3 bucket with permissions configured to store billing reports.

Log in to your AWS account to begin configuring cost and usage reporting:

1. In the AWS S3 console, create a new S3 bucket or use an existing bucket. If you are configuring a new S3 bucket, accept the default settings.

2. In the AWS Billing console, create a Cost and Usage Report that will be delivered to your S3 bucket. Specify the following values (and accept the defaults for any other values):
   - Report name: `<rh_cost_report>` (note this name as you will use it later)
   - Additional report details: Include resource IDs
   - S3 bucket: `<the S3 bucket you configured previously>`
1.2. CONFIGURE AN IAM POLICY TO ENABLE MINIMAL ACCOUNT ACCESS FOR COST AND USAGE CONSUMPTION

To provide data within the web interface and API, hybrid committed spend needs to consume the Cost and Usage Reports produced by AWS. For hybrid committed spend to obtain this data with a minimal amount of access, create an IAM policy and role for hybrid committed spend to use. This configuration provides access to the stored information and nothing else.

**Procedure**

1. From the AWS Identity and Access Management (IAM) console, create a new IAM policy for the S3 bucket you configured previously.
   
   a. Select the JSON tab and paste the following content in the JSON policy text box:

   ```json
   {
   "Version": "2012-10-17",
   "Statement": [
   {
   "Sid": "VisualEditor0",
   "Effect": "Allow",
   "Action": [
   "s3:Get*",
   "s3:List*"
   ],
   "Resource": [
   "arn:aws:s3:::<your_bucket_name>",
   "arn:aws:s3:::<your_bucket_name>/*"
   ],
   "Resource": [
   "arn:aws:s3:::<your_bucket_name>/cur:DescribeReportDefinitions"
   ],
   "Resource": "*"
   }
   
   {"Sid": "VisualEditor1",
    "Effect": "Allow",
    "Action": [
    "s3:ListBucket",
    "cur:DescribeReportDefinitions"
    ],
    "Resource": "*"
   }
   ```
1. Replace `<your_bucket_name>` in both locations with the name of the Amazon s3 bucket you configured previously.

2. In the AWS IAM console, create a new IAM role:
   a. For the type of trusted entity, select **Another AWS account**
   b. Enter `589173575009` as the Account ID to provide the hybrid committed spend application with read access to the AWS account cost data.
   c. Attach the IAM policy you just configured.
   d. Enter a role name (and description if desired) and finish creating the role.

3. In the AWS IAM console under Roles, open the summary screen for the role you just created and copy the Role ARN (a string beginning with `arn:aws:`).

### 1.3. ADDING A AMAZON WEB SERVICES ACCOUNT AND NAMING YOUR INTEGRATION

You must connect your AWS account as a data integrations so the hybrid committed spend application can process the cost and usage data from your AWS account.

**Procedure**

1. From hybrid committed spend, click **>` (Settings)**.
2. On the **Settings** page, click **Integrations**.
3. In the **Cloud** tab, click **Add source**.
4. In the **Add a cloud source** wizard, select **Amazon Web Services (AWS)** as the cloud provider type and click **Next**.
5. Enter a name for your integration. Click **Next**.
6. Select **Account authorization** and provide your AWS account credentials.
7. Click **Next**.
8. In the **Select application** step, select **hybrid committed spend** and click **Next**.
9. In the Configure cost and usage reporting tab, paste the name of your S3 bucket and click **Next**.
10. In the AWS IAM console under Roles tab, open the summary screen for the role you just created and copy the Role ARN (a string beginning with `arn:aws:`).
11. Click **Next** twice as you have already created your IAM Role and Policy.
12. Enter your Role ARN and click **Next**.
13. Review the details and click **Add** to add the AWS account to hybrid committed spend.

Hybrid committed spend will begin collecting cost and usage data from your AWS account and any linked AWS accounts.
CHAPTER 2. FILTERING YOUR AMAZON WEB SERVICES DATA BEFORE SENDING IT TO HYBRID COMMITTED SPEND

You can configure a function script in AWS to copy the cost exports and object storage bucket that hybrid committed spend can access and filter your data to share a subset of your billing data with Red Hat. This option is only recommended if your organization has third party data limitations.

**NOTE**

You must have a Red Hat account user with Integrations Administrator permissions before you can add data integrations to hybrid committed spend.

To configure your AWS account to be a hybrid committed spend data integration, you must complete the following tasks:

- Create an AWS S3 bucket to store your cost data.
- Create an AWS S3 bucket to report your filtered hybrid committed spend data.
- Configure IAM roles for your cost data bucket.
- Add your AWS integrations to [Red Hat Hybrid Cloud Console](#).
- Configure IAM roles for AWS Athena.
- Enable Athena.
- Create Lambda tasks for Athena to export filtered data to your S3 bucket.

Because you will complete some of the following steps in the AWS Console, and some steps in the hybrid committed spend user interface, keep both applications open in a web browser.

### 2.1. CREATING AN AWS S3 BUCKET FOR STORING YOUR COST DATA

You must create an Amazon S3 bucket with permissions configured to store billing reports.

Log in to your AWS account to begin configuring cost and usage reporting:

1. In the AWS S3 console, create a new S3 bucket or use an existing bucket. If you are configuring a new S3 bucket, accept the default settings.

2. In the AWS Billing console, create a Cost and Usage Report that will be delivered to your S3 bucket. Specify the following values (and accept the defaults for any other values):
   - Report name: `<rh_cost_report>` (note this name as you will use it later)
   - Additional report details: Include resource IDs
   - S3 bucket: `<the S3 bucket you configured previously>`
   - Time granularity: Hourly
   - Enable report data integration for: Amazon Redshift, Amazon QuickSight (do not enable report data integration for Amazon Athena)
2.2. CREATING AN AWS S3 BUCKET FOR FILTERED DATA REPORTING

This procedure will create a bucket to store your filtered data for reporting.

1. Log into your AWS account.
2. In the AWS S3 console, create a new report.
3. Enter a report name: <any-name> (note this name as you will use it later)
4. Click include resource IDs.
5. Click Next.
6. Under Configure S3 Bucket, click Configure. Create a new bucket and apply the default policy.
7. Click Save.
8. Specify the following values (and accept the defaults for any other values):
   a. Report Path prefix: HCS-Athena
   b. Time granularity: Hourly
   c. Enable report data integration for: Amazon Athena
9. Click Next
10. Verify the information and click Create report

2.3. ENABLING MINIMAL ACCOUNT ACCESS FOR COST AND USAGE CONSUMPTION

To provide data within the web interface and API, hybrid committed spend needs to consume the Cost and Usage Reports produced by AWS. For hybrid committed spend to obtain this data with a minimal amount of access, create an IAM policy and role for hybrid committed spend to use. This configuration provides access to the stored information and nothing else.

Procedure

1. From the AWS Identity and Access Management (IAM) console, create a new IAM policy for the S3 bucket you configured previously.
   a. Select the JSON tab and paste the following content in the JSON policy text box:

```json
{

```
"Version": "2012-10-17",
"Statement": [
  {
    "Sid": "VisualEditor0",
    "Effect": "Allow",
    "Action": [
      "s3:"
    ],
    "Resource": [
      "arn:aws:s3:::<your_bucket_name>",
      "arn:aws:s3:::<your_bucket_name>/**"
    ]
  },
  {
    "Sid": "VisualEditor1",
    "Effect": "Allow",
    "Action": [
      "s3:ListBucket",
      "cur:DescribeReportDefinitions"
    ],
    "Resource": "*
  }
]

Replace <your_bucket_name> in both locations with the name of the Amazon s3 bucket you configured for storing your filtered data.

b. Provide a name for the policy and complete the creation of the policy. Keep the AWS IAM console open as you will need it for the next step.

2. In the AWS IAM console, create a new IAM role:

   a. For the type of trusted entity, select AWS account

   b. Enter 589173575009 as the Account ID to provide the cost management application with read access to the AWS account cost data.

   c. Attach the IAM policy you just configured.

   d. Enter a role name (and description if desired) and finish creating the role.

3. In the AWS IAM console under Roles, open the summary screen for the role you just created and copy the Role ARN (a string beginning with arn:aws:).

2.4. ADDING A AMAZON WEB SERVICES ACCOUNT AND NAMING YOUR INTEGRATION

You must connect your AWS account as a data integrations so the hybrid committed spend application can process the cost and usage data from your AWS account.

Procedure
1. From hybrid committed spend, click ⚙ (Settings).

2. On the Settings page, click Integrations.

3. In the Cloud tab, click Add source.

4. In the Add a cloud source wizard, select Amazon Web Services (AWS) as the cloud provider type and click Next.

5. Enter a name for your integration. Click Next.

6. Select Account authorization and click Next.

7. Click Next.

8. In the Select application step, select hybrid committed spend and click Next.

9. In the Configure cost and usage reporting tab, paste the name of your S3 bucket and click Next.

10. In the AWS IAM console under Roles tab, open the summary screen for the role you just created and copy the Role ARN (a string beginning with arn:aws:).

11. Click Next twice as you have already created your IAM Role and Policy.

12. Enter your Role ARN and click Next.

13. Review the details and click Add to add the AWS account to hybrid committed spend.

Hybrid committed spend will begin collecting cost and usage data from your AWS account and any linked AWS accounts.

2.5. ENABLING ACCOUNT ACCESS FOR ATHENA

To provide data within the web interface and API, create an IAM policy and role for hybrid committed spend to use. This configuration provides access to the stored information and nothing else.

Procedure

1. From the AWS Identity and Access Management (IAM) console, create a new IAM policy for the Athena Lambda functions you will configure.

   a. Select the JSON tab and paste the following content in the JSON policy text box:

   ```json
   {
   "Version": "2012-10-17",
   "Statement": [
   {
   "Effect": "Allow",
   "Action": [
   "athena:*"
   ],
   "Resource": [
   "*"
   ]
   }
   };
   ```
"Effect": "Allow",
"Action": [
"glue:CreateDatabase",
"glue:DeleteDatabase",
"glue:GetDatabase",
"glue:GetDatabases",
"glue:UpdateDatabase",
"glue:GetDatabase",
"glue:DeleteTable",
"glue:BatchDeleteTable",
"glue:GetTable",
"glue:GetTables",
"glue:BatchCreatePartition",
"glue:CreatePartition",
"glue:DeletePartition",
"glue:BatchDeletePartition",
"glue:GetPartition",
"glue:GetPartitions",
"glue:BatchGetPartition"
],
"Resource": ["*"]
},
{
"Effect": "Allow",
"Action": [
"s3:GetBucketLocation",
"s3:GetObject",
"s3:ListBucket",
"s3:ListBucketMultipartUploads",
"s3:ListMultipartUploadParts",
"s3:AbortMultipartUpload",
"s3:CreateBucket",
"s3:PutObject",
"s3:PutBucketPublicAccessBlock"
],
"Resource": ["arn:aws:s3:::CHANGE-ME*"]
},
{
"Effect": "Allow",
"Action": [
"s3:GetObject",
"s3:ListBucket"
],
"Resource": ["arn:aws:s3:::CHANGE-ME*"]
}

"Effect": "Allow",
"Action": [
"s3:GetObject",
"s3:ListBucket"
]
b. Provide a name for the policy and complete the creation of the policy. Keep the AWS IAM console open as you will need it for the next step.

2. In the AWS IAM console, create a new IAM role:

   a. For the type of trusted entity, select **AWS service**.
b. Select Lambda.
c. Attach the IAM policy you just configured.
d. Enter a role name (and description if desired) and finish creating the role.

### 2.5.1. Configuring Athena for report generation

You can configure Athena to provide a filtered cost and usage report for hybrid committed spend.

The following configuration provides access to additional stored information and nothing else.

**Procedure**

1. In the AWS S3 console, navigate to the filtered bucket you created and download the `crawler-cfn.yml` file.
2. From Cloudformation in the AWS console, create a new stack.
3. Select Template as Ready.
4. Upload the `crawler-cfn.yml` file you previously downloaded.
5. Click **Next**.
6. Enter a name and click **Next**.
7. Click I acknowledge that AWS Cloudformation might create IAM resources and click **Submit**.

### 2.5.2. Creating a Lambda function for Athena

You must create a Lambda function to query the cost and usage report. This lambda function queries the cost and usage report for your Red Hat related expenses and creates a report of your filtered expenses.

**Procedure**

1. Navigate to Lambda in the AWS console and click **Create function**.
2. Click **Author from scratch**.
3. Enter a name your function.
4. From the Runtime dropdown, Select python 3.7.
5. Select x86_64 as the Architecture.
6. Under Permissions select the Athena role you created.
7. Click **Create function**.
8. Paste the following code to the function:

   ```python
   import boto3
   import uuid
   import json
   ```
from datetime import datetime

now = datetime.now()
year = now.strftime("%Y")
month = now.strftime("%m")
day = now.strftime("%d")

# Vars to Change!
integration_uuid = <your_integration_uuid>  # HCS integration_uuid
bucket = <your_S3_Bucket_Name>  # Bucket created for query results
database = 'athenacurcfn_athena_cost_and_usage'  # Database to execute athena queries
output = f's3://{bucket}/{year}/{month}/{day}/{uuid.uuid4()}'  # Output location for query results

# Athena query
query = f"SELECT * FROM {database}.koku_athena WHERE ((bill_billing_entity = 'AWS Marketplace' AND line_item_legal_entity like '%Red Hat%') OR (line_item_legal_entity like '%Amazon Web Services%' AND line_item_line_item_description like '%Red Hat%') OR (line_item_legal_entity like '%Amazon Web Services%' AND line_item_line_item_description like '%RHEL%') OR (line_item_legal_entity like '%AWS%' AND line_item_line_item_description like '%Red Hat%') OR (line_item_legal_entity like '%AWS%' AND line_item_line_item_description like '%RHEL%')) AND year = '{year}' AND month = '{month}'"

def lambda_handler(event, context):
    # Initiate Boto3 athena Client
    athena_client = boto3.client('athena')

    # Trigger athena query
    response = athena_client.start_query_execution(
        QueryString=query,
        QueryExecutionContext={
            'Database': database
        },
        ResultConfiguration={
            'OutputLocation': output
        }
    )

    # Save query execution to s3 object
    s3 = boto3.client('s3')
    json_object = {
        "integration_uuid": integration_uuid,
        "bill_year": year,
        "bill_month": month,
        "query_execution_id": response.get("QueryExecutionId"),
        "result_prefix": output
    }
    s3.put_object(
        Body=json.dumps(json_object),
        Bucket=bucket,
        Key='query-data.json'
    )

    return json_object

Replace `<your_integration_uuid>` with the UUID from the source you created on console.redhat.com. Replace `<your_S3_Bucket_Name>` with the name of the S3 bucket you created to store reports.

9. Click **Deploy** to test the function.
2.5.3. Creating a Lambda function to post the report files

You must create a Lambda function to post your report files to the spend S3 bucket you created.

**Procedure**

1. Navigate to Lambda in the AWS console and click **Create function**
2. Click Author from scratch
3. Enter a name your function
4. From the Runtime dropdown, Select python 3.7.
5. Select x86_64 as the Architecture.
6. Under Permissions select the Athena role you created.
7. Click **Create function**
8. Paste the following code to the function:

```python
import boto3
import json
import requests
from botocore.exceptions import ClientError

def get_credentials(secret_name, region_name):
    session = boto3.session.Session()
    client = session.client(
        service_name='secretsmanager',
        region_name=region_name)
    try:
        get_secret_value_response = client.get_secret_value(
            SecretId=secret_name
        )
    except ClientError as e:
        raise e
    secret = get_secret_value_response['SecretString']
    return secret

secret_name = "CHANGEME"
region_name = "us-east-1"
secret = get_credentials(secret_name, region_name)
json_creds = json.loads(secret)
USER = json_creds.get("<your_username>") # console.redhat.com Username
PASS = json_creds.get("<your_password>") # console.redhat.com Password
bucket = "<your_S3_Bucket_Name>" # Bucket for athena query results

def lambda_handler(event, context):
    # Initiate Boto3 s3 and fetch query file
    s3_resource = boto3.resource('s3')
```

---

CHAPTER 2. FILTERING YOUR AMAZON WEB SERVICES DATA BEFORE SENDING IT TO HYBRID COMMITTED SPEND
json_content = json.loads(s3_resource.Object(bucket, 'query-data.json').get() ['Body']).read().decode('utf-8'))

# Initiate Boto3 athena Client and attempt to fetch athena results
athena_client = boto3.client('athena')
try:
    athena_results = 
    athena_client.get_query_execution(QueryExecutionId=json_content["query_execution_id"])
except Exception as e:
    return f"Error fetching athena query results: {e} \nConsider increasing the time between running and fetching results"

reports_list = []
prefix = json_content["result_prefix"].split(f'{bucket}/')[-1]

# Initiate Boto3 s3 client
s3_client = boto3.client('s3')
result_data = s3_client.list_objects(Bucket=bucket, Prefix=prefix)
for item in result_data.get("Contents"):
    if item.get("Key").endswith(".csv"):
        reports_list.append(item.get("Key"))

# Post results to console.redhat.com API
url = "https://console.redhat.com/api/cost-management/v1/ingress/reports/
json_data = {"source": json_content["integration_uuid"], "reports_list": reports_list,
"bill_year": json_content["bill_year"], "bill_month": json_content["bill_month"]}
resp = requests.post(url, json=json_data, auth=(USER, PASS))

return resp

Replace <your_username> with your username for console.redhat.com. Replace 
<your_password> with your password for console.redhat.com. Replace 
<your_S3_Bucket_Name> with the name of the S3 bucket you created to store reports.

9. Click Deploy to test the function.
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

If you found an error or have a suggestion on how to improve these guidelines, open an issue in the cost management Jira board and add the Documentation label.

We appreciate your feedback!