

# AWS Data Collection Prerequisites for an IAM Role

## Overview

Densify collects resource utilization metrics (CloudWatch data) for your AWS services (e.g. EC2, RDS, ECS, etc.), analyzes the AWS data and then makes recommendations to save costs and reduce risks in your AWS environments.

Collecting data via a cross-account IAM Role simplifies the process of connecting to multiple AWS accounts from Densify since the same role and external ID can be used across your multiple AWS accounts. As accounts are added or removed, you do not need to update the Densify cloud connection.

▶ To learn more watch the following video: [Prerequisites for Collection AWS CloudWatch Data Using an IAM Role](#).

Densify also supports connecting to AWS using;

- An IAM Role and CloudFormation template—See *AWS Data Collection Using a CloudFormation Template* (Help Topic ID 410130).

**Note:** Though the cloud connection wizard provides the option to use an IAM user and an access key, Densify recommends using the IAM Role.

## Using an IAM Role

When you create a role for cross-account access, you establish trust from the customer's account that owns the role (and the resources (trusting account) to the Densify account containing the user that will collect data (trusted account). You specify the trusted account number as the Principal in the role's trust policy when you create the role. This allows the Densify user in the trusted account to assume the role and collect utilization data.

In addition to creating connections from your linked accounts, you also need to create a connection for your payer accounts. The payer account connection provides the name of each linked accounts.

Follow the process below to create and configure the IAM role for CloudWatch data collection.

## Creating the IAM Role and Attaching a Permission Policy to Collect CloudWatch Data

This role allows you to collect resource utilization data for the selected account. You need to attach a policy that allows the role to collect the required CloudWatch resource utilization metrics.

1. Log into the AWS Management Console and navigate to **Identity and Access Management (IAM)**. In the navigation tree on the left, click **Roles**.
2. Click **Create Role** in the Roles dashboard.
3. Select **AWS account** as the of trusted entity type .
4. Enter an **Account ID**. This is the Densify account that will assume the role. Enter the following Densify account ID: 036437403198.
5. Select **Require external ID** and enter your external ID. This value is similar to a password and should be unique and difficult to guess. Densify recommends using a password generator to create a random, alphanumeric string (e.g. ae73mcf4ldjpet96) for the external ID.

You will need this external ID later, when creating the cloud connection in Densify.

6. Click **Next**.

Figure: Creating a Role

The screenshot shows the AWS IAM console interface for creating a new role. The left sidebar contains navigation links: Dashboard, Groups, Users, Roles (highlighted with a red circle 1), Policies, Identity providers, Account settings, Credential report, and Encryption keys. The main content area is titled 'Select trusted entity' and includes an 'Info' link. It features four options for the trusted entity type: AWS service, AWS account (selected with a red circle 3), SAML 2.0 federation, and Custom trust policy. Below this, the 'An AWS account' section is shown, with options for 'This account' and 'Another AWS account' (selected with a red circle 4). The 'Another AWS account' section includes a search bar, a 'Role name' dropdown, and a text input field for the 'Account ID' (containing '036437403198', highlighted with a red circle 5). The 'Options' section has a checked checkbox for 'Require external ID' and a text input field for the 'External ID' (containing 'Bae91ccf4', highlighted with a red circle 5).

7. Attach the appropriate permission policy to the role. Select AWS's predefined **ReadOnlyAccess** policy. Use the filter to find the **ReadOnlyAccess** policy. Even with the filter set you will need to go to the last page to find the "ReadOnlyAccess" policy.

**Note:** The **ReadOnlyAccess** policy is provided here as an easy option for non-sensitive accounts. In most cases, you need to create a custom permission policy to grant Densify, the permissions to collect only the required CloudWatch data. Refer to [Creating an IAM Policy with Minimum Permissions for the CloudWatch Data Collection](#) on page 1 for details.

Figure: Selecting the ReadOnlyAccess Policy

**Create role**

**Attach permissions policies**

Choose one or more policies to attach to your new role.

Create policy Refresh

Filter: Policy type Q: readonlyaccess

Policy name

- ☐ AWSStorageGatewayReadOnlyAccess
- ☐ AWSWAFReadOnlyAccess
- ☐ AWSXrayReadOnlyAccess
- ☐ CloudFrontReadOnlyAccess
- ☐ CloudSearchReadOnlyAccess
- ☐ CloudWatchEventsReadOnlyAccess
- ☐ CloudWatchLogsReadOnlyAccess
- ☐ CloudWatchReadOnlyAccess
- ☐ IAMReadOnlyAccess
- ☒ **ReadOnlyAccess**
- ☐ ResourceGroupsandTagEditorReadOnlyAccess
- ☐ ServiceCatalogAdminReadOnlyAccess

\* Required

**Name, review, and create**

**Role details**

Role name  
Enter a meaningful name to identify this role.

Docs\_User 9

Maximum 64 characters. Use alphanumeric and '+,=,@,\_' characters.

Description  
Add a short explanation for this role.

Temp Role for creating videos and testing API

Maximum 1000 characters. Use alphanumeric and '+,=,@,\_' characters.

**Step 1: Select trusted entities**

```

1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": "sts:AssumeRole",
7       "Principal": {
8         "AWS": "036437403198"
9       },
10      "Condition": {
11        "StringEquals": {
12          "sts:ExternalId": "Bae91ccf4"
13        }
14      }
15    }
16  ]
17 }
```

**Step 2: Add permissions**

Permissions policy summary

Policy name	Type	Attached as
ReadOnlyAccess	AWS managed	Permissions policy 10

Cancel Previous **Create role**

- After selecting the permission policy for the role, click **Next**.
- In the Review page, specify the **Role name** and **Role description**. The role name can be any string used to identify and describe the role within the AWS account (e.g. `DensifyCrossAccountRole`).
- Click **Create role**. The new role is created.
- From the Roles page, click on the role name that you have just created, to view the role summary.

Figure: Role Summary

The screenshot displays the 'Docs\_User' role summary in the AWS IAM console. The role is a temporary role for creating videos and testing API. The ARN 'arn:aws:iam::152525148481:role/Docs\_User' is highlighted with a red circle and the number 12. The console also shows the role's permissions, including the 'ReadOnlyAccess' policy.

Creation date	ARN	Link to switch roles in console
January 18, 2023, 15:37 (UTC-05:00)	arn:aws:iam::152525148481:role/Docs_User	https://signin.aws.amazon.com/switchrole?User&account=aws-cirba

Last activity	Maximum session duration
None	1 hour

**Permissions policies (1)** Info

You can attach up to 10 managed policies.

Filter policies by property or policy name and press enter.

Policy name	Type	Description
ReadOnlyAccess	AWS managed	Provides read-only access to AWS services and resources.

12. Copy and save the **Role ARN** as you will need to paste this string into the Densify Cloud Connection wizard to create the connection.
13. You can now create an AWS connection through the Densify Public Cloud Connection wizard. See *Using the Public Cloud Connection Wizard* (Help Topic ID 380290).

You can also use the Densify API. See *Analysis: AWS* (Help Topic ID 340460).

## Optional Configuration

The following sections contain detailed instructions for optional configuration. Some of this configuration is referenced in the procedures above.

- [Creating an IAM Policy with Minimum Permissions for the CloudWatch Data Collection](#)
- [Enabling the Collection of Memory Usage Metrics](#)
- [Enabling the Collection of GPU Metrics](#)

## Creating an IAM Policy with Minimum Permissions for the CloudWatch Data Collection

To simplify setup and maintenance of either an IAM user account or an IAM role for performing the CloudWatch audit, Densify recommends attaching the AWS-managed “ReadOnlyAccess” policy to the user or role. This policy provides read-only access to your AWS services and resources and supports the requirements of the Densify CloudWatch audit. As the Densify CloudWatch audit continues to evolve and expand, you do not need to update permission policy to include newly added services and features.

Alternatively, if you must restrict the IAM user or role with the minimum permissions to perform the CloudWatch audit, you can create a custom policy with only the required permissions, as shown below.

**Note:** This custom policy must be updated periodically as Densify's standard audit requirements are updated to support additional AWS services and features.

#### Example: AWS Minimum User Permission Policy

```

1  {
2      "Version": "2012-10-17",
3      "Statement": [
4          {
5              "Sid": "Stmt1499171905000",
6              "Effect": "Allow",
7              "Action": [
8                  "autoscaling:DescribeAutoScalingGroups",
9                  "autoscaling:DescribeLaunchConfigurations",
10                 "autoscaling:DescribeScalingActivities",
11                 "ce:GetCostAndUsage",
12                 "cloudformation:DescribeStackResource",
13                 "cloudformation:DescribeStackResources",
14                 "cloudformation:ListStackInstances",
15                 "cloudformation:ListStackResources",
16                 "cloudwatch:GetMetricData",
17                 "cloudwatch:GetMetricStatistics",
18                 "cloudwatch:ListMetrics",
19                 "ec2:DescribeHosts",
20                 "ec2:DescribeImages",
21                 "ec2:DescribeInstances",
22                 "ec2:DescribeLaunchTemplateVersions",
23                 "ec2:DescribeRegions",
24                 "ec2:DescribeSnapshots",
25                 "ec2:DescribeVolumes",
26                 "ec2:DescribeSubnets",
27                 "ec2:DescribeSecurityGroupRules",
28                 "ec2:DescribeSecurityGroups",
29                 "ec2:DescribeVpcs",
30                 "ecs:DescribeCapacityProviders",
31                 "ecs:DescribeClusters",
32                 "ecs:DescribeContainerInstances",
33                 "ecs:DescribeServices",
34                 "ecs:DescribeTaskDefinition",
35                 "ecs:ListClusters",
36                 "ecs:ListContainerInstances",
37                 "ecs:ListServices",
38                 "ecs:ListTagsForResource",
39                 "ecs:ListTaskDefinitions",
40                 "eks:DescribeCluster",
41                 "eks:ListClusters",
42                 "iam:ListAccountAliases",
43                 "organizations:DescribeOrganization",
44                 "organizations:ListAccounts",
45                 "rds:DescribeDBInstances",
46                 "rds:DescribeReservedDBInstances",
47                 "rds:DescribeDBClusters",
48                 "rds:ListTagsForResource"
49             ]
50         }
51     ]
52 }
```

**Example: AWS Minimum User Permission Policy**

```
50 |                                     ],
51 |                                     "Resource": "*"
52 |                                     }
53 |                                     ]
54 |                                     }
55 | }
```

**Note:** The permissions related to CloudFormation are used for linking ASGs with ECS clusters. If these permissions are not included and the data is not available, linking the ASGs to ECS clusters may be done based on existing container instances. If the permissions are missing, Densify may not link some ASGs to their ECS clusters,

1. Log into the AWS management console and navigate to **Services > IAM**.
2. Select **Policies** and click **Create policy**.
3. Click the **JSON** tab and enter the policy from the example above.
4. Review the policy and enter a policy name (e.g. DensifyMinimumReadAccess) and a description (e.g. Minimum permissions required for Densify standard audit).

## Enabling Collection of AWS Memory Usage Metrics

Memory metrics are not collected by default and they are not required to complete the Densify analyses; however, you can manually enable collection of specific memory metrics to improve the quality of your recommendations.

**Note:** Additional metrics collected by the CloudWatch agent incur charges. See [Amazon CloudWatch Pricing](#) for details.

Table: Required Cloudwatch Metrics

Scenario	AWS Memory Metric	Collected by the CloudWatch Agent, by Default
CloudWatch agent for Linux	mem_used_percent	Yes
	mem_active	No
	mem_used	No
CloudWatch agent for Windows	Available MBytes	No
	% Committed Bytes In Use	Yes



**Note:** The CloudWatch agent must be installed and configured on each instance for which you want to obtain memory. Refer to the AWS user documentation for details. See <https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/Install-CloudWatch-Agent.html>

Once the CloudWatch agent is installed and configured, Densify requires that you use the default "CWAgent" namespace for the collected metrics collected.

## Memory Metrics for ASGs

Additionally, when working with ASGs, the ASG's EC2 members provide memory utilization data using the basic memory settings, but you need to specify "aggregation\_dimensions" to collect memory, aggregated at the ASG level.

## Linux Configuration

For Linux instances, the memory metric, "mem\_used\_percent" is collected by default, as specified in the config.json file. However, the metrics "mem\_active" and "mem\_used" need to be added for Densify's analysis.

Use the following example to customize the default CloudWatch agent's configuration, to collect the required parameters and metrics that Densify requires for memory analyses. The CloudWatch agent configuration file is located in the CWAgent namespace,

### Example: CloudWatch Agent Configuration For Linux Instances

```

1  {
2      "agent": {
3          "metrics_collection_interval": 60,
4          "run_as_user": "root"
5      },
6      "metrics": {
7          "append_dimensions": {
8              "AutoScalingGroupName": "${aws:AutoScalingGroupName}",
9              "ImageId": "${aws:ImageId}",
10             "InstanceId": "${aws:InstanceId}",
11             "InstanceType": "${aws:InstanceType}"
12         },
13         "aggregation_dimensions": [
14             ["AutoScalingGroupName"]
15         ],
16         "metrics_collected": {
17             "mem": {
18                 "measurement": [
19                     "mem_used",
20                     "mem_active",
21                     "mem_used_percent"
22                 ],

```

**Example: CloudWatch Agent Configuration For Linux Instances**

```

23 |         "metrics_collection_interval": 60
24 |     }
25 | }
26 | }
27 | }
28 |

```

## Windows Configuration

For Windows instances, the default CloudWatch agent config.json file are the same as listed above.

The memory metric, "% Committed Bytes in Use" is collected by default, as specified in the config.json file. However, the metric "Available MBytes" needs be added to the CloudWatch agent's settings, for Densify's analysis.

Use the following example to customize the default CloudWatch agent's configuration, to collect the required parameters and metrics that Densify requires for memory analyses. The CloudWatch agent configuration file is located in the CWAgent namespace,

Use the following example to customize the default CloudWatch agent configuration to collect the required parameters and metrics that Densify requires for memory analyses.

**Example: CloudWatch Agent Configuration For Windows Instances**

```

1 | {
2 |     "agent": {
3 |         "metrics_collection_interval": 60,
4 |         "logfile": "c:\\ProgramData\\Amazon\\AmazonCloudWatchAgent\\Logs\\amazon-cloudwatch-agent.log"
5 |     },
6 |     "metrics": {
7 |         "append_dimensions": {
8 |             "AutoScalingGroupName": "${aws:AutoScalingGroupName}",
9 |             "ImageId": "${aws:ImageId}",
10 |            "InstanceId": "${aws:InstanceId}",
11 |            "InstanceType": "${aws:InstanceType}"
12 |         },
13 |         "aggregation_dimensions": [
14 |             ["AutoScalingGroupName"]
15 |         ],
16 |         "metrics_collected": {
17 |             "Memory": {
18 |                 "measurement": [
19 |                     "Available MBytes",
20 |                     "% Committed Bytes In Use"
21 |                 ],
22 |                 "metrics_collection_interval": 60
23 |             }
24 |         }
25 |     }
26 | }

```

**Example: CloudWatch Agent Configuration For Windows Instances**

```

25 |     }
26 | }
27 |

```

If you are using a third-party application to collect memory metrics, the collected data can be loaded using the Receive Metrics API endpoint. See [Importing Metrics for Existing Services](#) in the API documentation.

Refer to the AWS user documentation for details on using the CloudWatch Agent to collect memory metrics.

## Enabling the Collection of GPU Metrics

To support collection of the NVIDIA® GPU data, the required metrics must be enabled through the CloudWatch agent.

Install the NVIDIA driver and CloudWatch agent on your instances. See [Collecting NVIDIA GPU Metrics](#).

- Linux Servers—You need to add the section, `nvidia_gpu` inside the `metrics_collected` section of the CloudWatch agent configuration file. For more information, see: <https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch-Agent-NVIDIA-GPU.html>
- Window Servers—Configure and use Amazon Custom CloudWatch Metrics. See <https://repost.aws/questions/QUcyzZfm3UR96Qj8JiadICAA/how-to-monitor-and-collect-gpu-metrics-for-windows-ec2-instances-using-amazon-cloudwatch>

Alternatively, you can use a third party tool such as [Telegraf](#) for metrics collection from Window Servers.

Densify requires the following GPU metrics:

Table: Required GPU Metrics

NVIDIA Metric	CloudWatch Metric Name	Unit	Description
utilization_gpu	nvidia_smi_utilization_gpu	%	The percentage of time over the past sample period during which one or more kernels on the GPU was running.
utilization_memory	nvidia_smi_utilization_memory	%	The percentage of time over the past sample period during which global (device) memory was being read or written.
memory_used	nvidia_smi_memory_used	MB	Memory used.

The following new workloads, based on the above metrics are available in the [metrics viewer](#):

- GPU\_Utilization—GPU utilization in percent
- GPU\_Mem\_Utilization\_As\_Pct—GPU memory utilization as a percent of the total memory

- GPU\_Mem\_Used—GPU memory used, in MB

## Prerequisites

1. **NVIDIA Driver**—Ensure you have the correct NVIDIA driver installed on your instances. Refer to the [NVIDIA website](#) for the driver installation guide for your specific operating system.
2. **NVIDIA Toolkit**—Install the NVIDIA Toolkit to gather GPU metrics. Refer to the [NVIDIA website](#) for details.
3. **CloudWatch Agent**—Refer to the AWS documentation for the latest agent installation instructions on your specific operating system. Configure the CloudWatch Agent as outlined below.
4. Ensure that the IAM role attached to your EC2 instances has the necessary permissions to publish metrics to CloudWatch.

## Install the NVIDIA Toolkit

The driver and toolkit may be pre-installed on your instance. Verify the elements contained on your instance before installing the toolkit or driver.

On Ubuntu:

```
sudo apt-get update sudo apt-get install -y nvidia-cuda-toolkit
```

On Amazon Linux 2:

```
sudo yum install -y nvidia cuda
```

## Create the CloudWatch Agent Configuration File

1. Modify the CloudWatch Agent configuration file (`amazon-cloudwatch-agent.json`) to add GPU metrics.

Figure: Sample Configuration File

```

AmazonCloudWatch-nvidia.txt
1  {
2      "metrics": {
3          "aggregation_dimensions": [
4              [
5                  "InstanceId"
6              ]
7          ],
8          "append_dimensions": {
9              "AutoScalingGroupName": "${aws:AutoScalingGroupName}",
10             "ImageId": "${aws:ImageId}",
11             "InstanceId": "${aws:InstanceId}",
12             "InstanceType": "${aws:InstanceType}"
13         },
14         "metrics_collected": {
15             "nvidia_gpu": {
16                 "measurement": [
17                     "utilization_gpu",
18                     "utilization_memory",
19                     "memory_used",
20                 ],
21                 "metrics_collection_interval": 60
22             },

```

2. Upload the CloudWatch Agent configuration file to your instance. For example, if using Amazon Linux 2:

```

sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl
\ -a fetch-config -m ec2 -c file:/path/to/your/amazon-cloudwatch-
agent.json -s

```

3. Start the CloudWatch Agent:

```

sudo /opt/aws/amazon-cloudwatch-agent/bin/start-amazon-cloudwatch-
agent

```

4. You can also modify the `metrics_collection_interval` to change how often metrics are collected and sent to CloudWatch.

## Verifying the Configuration

1. Verify that the CloudWatch Agent is running:

```

sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl
-m ec2 -a status

```

2. Open the CloudWatch console to review the metrics.
3. Navigate to Metrics and select the `CWAgent` namespace.
4. You should see GPU metrics under this namespace.