# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>What Is AWS Ground Station?</td>
<td>1</td>
</tr>
<tr>
<td>How AWS Ground Station Works</td>
<td>2</td>
</tr>
<tr>
<td>Core Components</td>
<td>3</td>
</tr>
<tr>
<td>Service Terms</td>
<td>3</td>
</tr>
<tr>
<td>Dataflow Endpoint Groups</td>
<td>3</td>
</tr>
<tr>
<td>Configs</td>
<td>4</td>
</tr>
<tr>
<td>Mission Profiles</td>
<td>7</td>
</tr>
<tr>
<td>Setting Up AWS Ground Station</td>
<td>9</td>
</tr>
<tr>
<td>Step 1: Sign Up for AWS</td>
<td>9</td>
</tr>
<tr>
<td>Step 2: Add Permissions to Your AWS Account</td>
<td>9</td>
</tr>
<tr>
<td>Step 3: Customer Onboarding</td>
<td>10</td>
</tr>
<tr>
<td>Next Steps</td>
<td>11</td>
</tr>
<tr>
<td>Getting Started</td>
<td>12</td>
</tr>
<tr>
<td>Basic Concepts</td>
<td>12</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>12</td>
</tr>
<tr>
<td>Step 1: Create EC2 SSH Key Pair</td>
<td>12</td>
</tr>
<tr>
<td>Step 2: Set Up Your VPC</td>
<td>13</td>
</tr>
<tr>
<td>Step 3: Choose and Customize an AWS CloudFormation Template</td>
<td>14</td>
</tr>
<tr>
<td>Configuring your Amazon EC2 Instance Settings</td>
<td>14</td>
</tr>
<tr>
<td>Choose a Template</td>
<td>15</td>
</tr>
<tr>
<td>Step 4: Configure an AWS CloudFormation Stack</td>
<td>19</td>
</tr>
<tr>
<td>Step 5: Install and Configure FE Processor/Radio</td>
<td>20</td>
</tr>
<tr>
<td>Next Steps</td>
<td>20</td>
</tr>
<tr>
<td>Listing and Reserving Contacts</td>
<td>21</td>
</tr>
<tr>
<td>Using the Ground Station Console</td>
<td>21</td>
</tr>
<tr>
<td>Reserve a Contact</td>
<td>22</td>
</tr>
<tr>
<td>View Scheduled and Completed Contacts</td>
<td>23</td>
</tr>
<tr>
<td>Cancelling Contacts</td>
<td>24</td>
</tr>
<tr>
<td>Reserving and Managing Contacts with AWS CLI</td>
<td>25</td>
</tr>
<tr>
<td>View and List Contacts with AWS CLI</td>
<td>26</td>
</tr>
<tr>
<td>Reserve a Contact with AWS CLI</td>
<td>27</td>
</tr>
<tr>
<td>Cancel a Contact with AWS CLI</td>
<td>27</td>
</tr>
<tr>
<td>Using Cross-Region Data Delivery</td>
<td>29</td>
</tr>
<tr>
<td>To use cross-region data delivery in the console</td>
<td>29</td>
</tr>
<tr>
<td>To use cross-region data delivery with AWS CLI</td>
<td>30</td>
</tr>
<tr>
<td>Monitoring AWS Ground Station</td>
<td>31</td>
</tr>
<tr>
<td>Automating with CloudWatch Events</td>
<td>31</td>
</tr>
<tr>
<td>Example CloudWatch Events</td>
<td>32</td>
</tr>
<tr>
<td>Logging API Calls with CloudTrail</td>
<td>33</td>
</tr>
<tr>
<td>AWS Ground Station Information in CloudTrail</td>
<td>33</td>
</tr>
<tr>
<td>Understanding AWS Ground Station Log File Entries</td>
<td>34</td>
</tr>
<tr>
<td>Metrics with Amazon CloudWatch</td>
<td>35</td>
</tr>
<tr>
<td>AWS Ground Station Metrics and Dimensions</td>
<td>35</td>
</tr>
<tr>
<td>Viewing Metrics</td>
<td>36</td>
</tr>
<tr>
<td>Troubleshooting</td>
<td>39</td>
</tr>
<tr>
<td>Troubleshooting AWS Ground Station Contact</td>
<td>39</td>
</tr>
<tr>
<td>Step 1: Verify that Your EC2 Instance is Running</td>
<td>39</td>
</tr>
<tr>
<td>Step 2: Verify that Data Defender is Running</td>
<td>39</td>
</tr>
<tr>
<td>Step 3: Verify that Your Data Defender Stream is Configured</td>
<td>40</td>
</tr>
<tr>
<td>Security</td>
<td>43</td>
</tr>
<tr>
<td>Authentication and Access Control</td>
<td>43</td>
</tr>
<tr>
<td>Audience</td>
<td>43</td>
</tr>
<tr>
<td>Authentication</td>
<td>44</td>
</tr>
</tbody>
</table>
What Is AWS Ground Station?

AWS Ground Station is a fully managed service that enables you to control satellite communications, process satellite data, and scale your satellite operations. This means that you no longer have to build or manage your own ground station infrastructure.

AWS Ground Station enables you to focus on innovating and rapidly experimenting with new applications that ingest satellite data and dynamically scale your server and storage use, rather than spend resources on operating and maintaining your own ground stations.
How AWS Ground Station Works

A satellite reservation is also known as a contact. Your satellite communicates with an AWS Ground Station antenna during contacts. You can reserve contacts through an API or through the AWS console by specifying location, time, and mission information. Your contact data can be streamed to and from an Amazon Elastic Compute Cloud (Amazon EC2) instance. From there, you can process the data or pass it into an Amazon S3 bucket, where you can use other AWS services like Amazon Machine Learning (Amazon ML) or Amazon Rekognition for post-processing. This ensures control over your data.

You can create extensible and reusable configuration resources so that you have control over how AWS Ground Station antennas are configured during your contacts. Using mission profiles, you can specify where data is coming from, what its format should be, and where to send it.

With AWS Ground Station you can access more than 125 services via satellite communications. Note the following:

- You can receive narrowband RF data in S-band (2200 to 2300 MHz) or X-band (7750 to 8400 MHz) at bandwidths up to 54 MHz.
- S-Band RF data is digitized and provided as a digital stream in VITA-49 Signal Data/IP format.
- X-Band intermediate frequency (IF) data is digitized and provided as a digital stream in VITA-49 Signal Data/IP format.
- You can receive wideband demodulated/decoded data in X-band (7750 to 8400 MHz) at bandwidths up to 500 MHz.
• X-Band intermediate frequency (IF) data is demodulated, decoded, and provided as a digital stream in VITA-49 Extension Data/IP format.

• You can transmit RF data in S-Band (2025 to 2120 MHz) at bandwidths up to 54 MHz.
  • The RF data is provided to AWS Ground Station as a digital stream in VITA-49 Signal Data/IP format.

• You must run AWS Ground Station from an AWS Region that supports AWS Ground Station. To see a list of supported regions, see the global infrastructure Region Table.

• You can deliver data to an EC2 instance running in the same region as the antenna, or you can use cross-region data delivery to send your data from an antenna to an EC2 instance in your preferred AWS Region. The following antenna-to-destination regions are currently available:
  • US East (Ohio) Region (us-east-2) to US West (Oregon) Region (us-west-2)
  • US West (Oregon) Region (us-west-2) to US East (Ohio) Region (us-east-2)

Service Terms

You may only use the Services to store, retrieve, query, serve, and execute Your Content that is owned, licensed or lawfully obtained by you. As used in these Service Terms, (a) “Your Content” includes any “Company Content” and any “Customer Content” and (b) “AWS Content” includes “Amazon Properties.” As part of the Services, you may be allowed to use certain software (including related documentation) provided by us or third-party licensors.

Important
This software is neither sold nor distributed to you and you may use it solely as part of the Services. You may not transfer it outside the Services without specific authorization to do so.

Core Components

Dataflow endpoint groups, configs, and mission profiles are core components of AWS Ground Station. These components determine how you schedule your contacts, how the antennas communicate with your satellites, and where your data is delivered. Before getting started with AWS Ground Station, we recommend that you learn about these components. Examples are provided in their respective sections.

Topics
• Dataflow Endpoint Groups (p. 3)
• Configs (p. 4)
• Mission Profiles (p. 7)

Dataflow Endpoint Groups

Dataflow endpoints define the location where you want the data to be streamed to or from during contacts. The endpoints are identified by a name of your choosing when executing contacts. These names do not need to be unique. This allows multiple contacts to be executed at the same time using the same mission profile.

The endpoint list address consists of the following:
• name - IP address of this dataflow endpoint.
• port - The port to connect to.

The security details of an endpoint consist of the following:
• **roleArn** - The Amazon Resource Name (ARN) of a role that AWS Ground Station will assume to create Elastic Network Interfaces (ENIs) in your VPC. These ENIs serve as the ingress and egress points of data streamed during a contact.

• **securityGroupIds** - The security groups to attach to the elastic network interfaces.

• **subnetIds** - A list of subnets where AWS Ground Station places elastic network interfaces to send streams to your instances.

Dataflow endpoints are always created as part of a *dataflow endpoint group*. By including multiple dataflow endpoints in a group, you are asserting that the specified endpoints can all be used together during a single contact. For example, if a contact needs to send data to three separate dataflow endpoints, you must have three endpoints in a single dataflow endpoint group that match the dataflow endpoint configs in your mission profile.

When one or more resources in a dataflow endpoint group is in use for a contact, the entire group is reserved for the duration of that contact.

You may execute multiple contacts at a time, but those contacts must be executed on different dataflow endpoint groups. An example is provided below.

```json
{
  "endpointDetails": [
    {
      "endpoint": {
        "address": {
          "name": "192.168.1.1",
          "port": 55888
        },
        "name": "DataflowEndpoint1",
        "securityDetails": {
          "roleArn": "string",
          "securityGroupIds": [ "string" ],
          "subnetIds": [ "string" ]
        }
      },
      "endpoint": {
        "address": {
          "name": "192.168.1.1",
          "port": 55889
        },
        "name": "DataflowEndpoint2",
        "securityDetails": {
          "roleArn": "string",
          "securityGroupIds": [ "string" ],
          "subnetIds": [ "string" ]
        }
      }
    }
  ]
}
```

**Configs**

**Configs** are resources that AWS Ground Station uses to define the parameters for each aspect of your contact. Add the configs you want to a mission profile, and then that mission profile will be used when executing the contact. You can define several different types of configs.
Dataflow Endpoint Config

You can use dataflow endpoint configs to specify which dataflow endpoints you want to use during contacts. The two parameters of a dataflow endpoint config specify the name and region of the dataflow endpoint. The example below tells the contact to use a dataflow endpoint with the name DataflowEndpoint1 in the US West (Oregon) Region (us-west-2) region. You can use the example code to create multiple dataflow endpoint groups across the AWS Regions and reuse the same config to execute contacts. If no dataflowEndpointRegion is specified, the dataflow endpoint will default to the region of the ground station.

```json
{
    "configData": {
        "dataflowEndpointConfig": {
            "dataflowEndpointName": "DataflowEndpoint1",
            "dataflowEndpointRegion": "us-west-2"
        }
    },
    "name": "MyDataflowEndpointConfig"
}
```

Tracking Config

You can use tracking configs in the mission profile to determine whether autotrack should be enabled during your contacts. This config has a single parameter: autotrack. The autotrack parameter can have the following values:

- REQUIRED - Autotrack is required for your contacts.
- PREFERRED - Autotrack is preferred for contacts, but contacts can still be executed without autotrack.
- REMOVED - No autotrack should be used for your contacts.

The following is an example tracking config:

```json
{
    "configData": {
        "trackingConfig": {
            "autotrack": "REMOVED"
        }
    },
    "name": "MyTrackingConfig"
}
```

Antenna Downlink Config

You can use antenna downlink configs to configure the antenna during your contact. They consist of a spectrum config that specifies the bandwidth, frequency, and polarization that should be used during your antenna downlink. If your downlink use case requires demodulation or decode, see the the section called “Antenna Downlink Demod Decode Config” (p. 7).

The following is an example antenna downlink config:

```json
{
    "antennaDownlinkConfig": {
        "spectrumConfig": {
```
Antenna Uplink Config

You can use antenna uplink configs to configure the antenna during your uplink contact. They consist of a spectrum config with frequency, polarization, and targetEirp. For information about how to configure an uplink echo, see the section called “Antenna Uplink Echo Config” (p. 6).

The following is an example antenna uplink config:

```json
{
  "configData": {
    "antennaUplinkConfig": {
      "spectrumConfig": {
        "centerFrequency": {
          "units": "MHz",
          "value": 2091
        },
        "polarization": "RIGHT_HAND"
      },
      "targetEirp": {
        "units": "dBW",
        "value": 20
      }
    },
    "name": "MyAntennaUplinkConfig"
  }
}
```

Antenna Uplink Echo Config

Uplink echo configs tell the antenna how to execute an uplink echo. This echoes the signal sent by the antenna back to your dataflow endpoint. An uplink echo config contains the ARN of an uplink config. The antenna uses the parameters from the uplink config pointed to by the ARN when executing an uplink echo.

The following is an example antenna uplink echo config:

```json
{
  "configData": {
    "uplinkEchoConfig": {
      "enabled": true
    }
  }
}
```
Antenna Downlink Demod Decode Config

Antenna downlink demod decode configs are a more complex and customizable config type that you can use to execute downlink contacts with demod or decode. If you're interested in executing these types of contacts, contact the AWS Ground Station team. We'll help you define the right config and mission profile for your use case.

Mission Profiles

Mission profiles contain configs and parameters for how contacts are executed. When you reserve a contact or search for available contacts, you supply the mission profile that you intend to use. Mission profiles bring all of your configs together and define where data will go during your contact.

Aside from tracking configs (p. 5), all configs are contained in the `dataflowEdges` field of the mission profile. A single dataflow edge is a list of two ARNs—the first is the `from` config and the second is the `to` config. By specifying a dataflow edge between two configs, you are telling AWS Ground Station from where and to where data should flow during a contact. Tracking configs are not used as part of a dataflow edge, but are specified as a separate field.

The `name` field of the mission profile helps distinguish between the mission profiles that you create.

The following is an example mission profile for the simplest downlink use case:

```json
{
    "contactPostPassDurationSeconds": 60,
    "contactPrePassDurationSeconds": 60,
    "dataflowEdges": [
        ],
    "minimumViableContactDurationSeconds": 120,
    "name": "MySimpleAntennaDownlinkMissionProfile",
    "trackingConfigArn": "arn:aws:groundstation:us-east-2:123456789012:config/tracking/11111111-2222-3333-4444-555555555555"
}
```

The following is an example mission profile for a contact containing downlink, uplink, and uplink echo:

```json
{
    "contactPostPassDurationSeconds": 60,
    "contactPrePassDurationSeconds": 60,
    "dataflowEdges": [
        ],
```
[{
},
{
}]

"minimumViableContactDurationSeconds": 120,
"name": "MyDownlinkUplinkEchoMissionProfile",
"trackingConfigArn": "arn:aws:groundstation:us-east-2:123456789012:config/tracking/11111111-2222-3333-4444-555555555555"}
Setting Up AWS Ground Station

Before you start using AWS Ground Station, you need to know what AWS Identity and Access Management (IAM) permissions you need, and what space vehicle credentials to provide. Use the following steps to set up your account.

**Topics**
- Step 1: Sign Up for AWS (p. 9)
- Step 2: Add Permissions to Your AWS Account (p. 9)
- Step 3: Customer Onboarding (p. 10)
- Next Steps (p. 11)

### Step 1: Sign Up for AWS

To use AWS Ground Station, you need an AWS account. If you already have an AWS account, skip to the section called “Step 2: Add Permissions to Your AWS Account” (p. 9).

1. Choose the following hyperlink: https://aws.amazon.com/.  
   **Note**  
   If you previously signed in to the AWS Management Console using AWS account root user credentials, choose **Sign in to a different account**. If you previously signed in to the console using IAM credentials, choose **Sign-in using root account credentials**. Then, choose **Create a new AWS account**.

2. Choose **Sign In to the Console**.

3. Choose **Create a new AWS account**.

4. Follow the instructions for creating a new AWS account. Part of the sign-up procedure involves receiving a phone call and entering a verification code using the phone keypad.

### Step 2: Add Permissions to Your AWS Account

To use AWS Ground Station, you need to create a new policy and attach it to your AWS account.

1. Sign in to the AWS Management Console and open the (IAM) console at https://console.aws.amazon.com/iam/.

2. Create a new policy. Use the following steps:
   
   a. In the navigation pane, choose **Policies** and then choose **Create Policy**.

   b. In the **JSON** tab, edit the JSON with one of the following values. Use the JSON that works best for your application.

      - For Admin privileges, set **Action** to `groundstation:*` as follows:

```json
{  
  "Version": "2012-10-17",
  "Statement": [
    {  
      "Effect": "Allow",
```
Step 3: Customer Onboarding

To complete registration for your AWS Ground Station account, see the Satellites and Resources section in the AWS Ground Station console page for onboarding details. The AWS Ground Station team will work with you to onboard your satellites to the service. Once you onboard your satellite, the satellite will be available to use when managing a contact. Instructions for managing a contact are provided later in Listing and Reserving Contacts (p. 21).
Onboarding your satellite(s) will grant you access to send and receive data to and from the satellite. In addition to onboarding your own satellites, customers may also onboard the following satellites to downlink direct broadcast data using AWS Ground Station:

- Aqua
- SNPP
- JPSS-1/NOAA-20
- Terra

Once onboarded, these satellites can be accessed for immediate use. You can use the AquaSnppJpss.yml template and the AquaSnppJpssTerraDigIF.yml template provided in the AWS Ground Station customer assets S3 bucket and customize the template to configure your own parameters. Instructions and details for accessing and using this template are provided later in the Choose and Customize an AWS CloudFormation Template (p. 14) section of the user guide.

For more information about these satellites and the kind of data they transmit, see Aqua, JPSS-1/NOAA-20 and SNPP, and Terra.

**Next Steps**

Your AWS Ground Station account is now set up and ready for configuration. Continue to Getting Started (p. 12) to configure your resources to use AWS Ground Station.
Getting Started with AWS Ground Station

AWS Ground Station enables you to command, control, and downlink data from your satellites.

With AWS Ground Station, you can schedule access to ground station antennas on a per-minute basis and pay only for the antenna time used. Incoming data is streamed to an AWS backend infrastructure where other AWS services can store or process it.

Topics

- Basic Concepts (p. 12)
- Prerequisites (p. 12)
- Step 1: Create EC2 SSH Key Pair (p. 12)
- Step 2: Set Up Your VPC (p. 13)
- Step 3: Choose and Customize an AWS CloudFormation Template (p. 14)
- Step 4: Configure an AWS CloudFormation Stack (p. 19)
- Step 5: Install and Configure FE Processor/Radio (p. 20)
- Next Steps (p. 20)

Basic Concepts

Before you begin, you should familiarize yourself with the basic concepts in AWS Ground Station. For more information, see Core Components (p. 3).

Then, continue on to Prerequisites (p. 12) to learn about prerequisites to getting started with AWS Ground Station.

Prerequisites

Before getting starting with AWS Ground Station, ensure you have an AWS account with the proper credentials. Follow the steps in Using the Ground Station Console (p. 9).

Then, continue on to Step 1: Create EC2 SSH Key Pair (p. 12).

Step 1: Create EC2 SSH Key Pair

If you do not already have one, create a new key pair in the Amazon EC2 console for each AWS Region where you plan to receive data. Use the steps below.

1. In your AWS Management Console, choose an AWS Region in which you plan to reserve contacts. You need to create a key pair for every AWS Region you choose.
Note
AWS Ground Station is not yet available for all regions. Ensure that AWS Ground Station is supported by your desired AWS Region. For more information about AWS Ground Station antenna locations, see AWS Ground Station FAQs.

2. Choose Services > EC2 > Network & Security > Key Pairs, and then choose Create Key Pair.
3. Enter a friendly name like groundstation-ec2-access-key-<region> (for example, groundstation-ec2-access-key-us-east-2).
4. Save the private key, make it accessible to your ssh utility of choice, and set the ownership/permissions as needed (for example, chmod 400 <key name>.pem).
5. Repeat for other AWS Regions if needed.

Step 2: Set Up Your VPC

The full setup of a VPC is beyond the scope of this guide. If you don't have an existing VPC that is already customized, you can use the default VPC that is created in your AWS account. We recommend adding a Linux bastion to your VPC so that you can SSH into your Amazon EC2 instances without attaching a public IP address. For more information about configuring a Linux bastion in your VPC, see Linux Bastion Hosts on AWS.

For your convenience, instructions to quickly add a bastion host to your Linux environment in AWS are below. While this is not required, it is recommended best practice.

1. Login to your AWS account.
2. In the Linux Bastion Hosts on the AWS Cloud: Quick Start Reference Deployment page, choose Launch Quick Start (for new VPC).
3. In the Create Stack page, choose Next. The template is pre-populated.
4. In the Specify stack details page, make edits and changes in the following boxes:
Step 3: Choose and Customize an AWS CloudFormation Template

Today, you can configure multiple streams of data per contact to flow into your VPC. These data streams are available in two different formats. Data streams containing VITA-49 Signal/IP data can be configured for S-Band and X-Band signals up to 54 MHz in bandwidth. VITA-49 Extension data/IPs can be configured for demodulated and/or decoded X-Band signals up to 500 MHz in bandwidth.

After you onboard (p. 10) your satellite, you need to define mission profiles and create instances to process or push data streams from or to your satellite. To assist you with this process, we provide preconfigured AWS CloudFormation templates that use public broadcast satellites. These templates make it easy for you to start using AWS Ground Station. For more information about AWS CloudFormation, see What is AWS CloudFormation?

It is important to note that you need to have data processing software or data storage software listening to the localhost side of Data Defender of the Amazon EC2 instance. This software is what you will use to store and/or process the data delivered to the Amazon EC2 instance during a contact.

Configuring your Amazon EC2 Instance Settings

The AWS CloudFormation templates provided in this section are configured to use Amazon EC2 m5.4xlarge instance types by default. However, we encourage you to customize and choose the right Amazon EC2 instance settings for your use case. Requirements such as storage I/O and CPU performance should be considered when choosing your instance settings. For example, running a software modem on a receiver instance may require compute-optimized instances with more cores and a higher clock speed. The best way to determine the right instance settings for your use case is to test your instance
settings with your workload, and Amazon EC2 makes it easy to switch between instance settings. Use the templates and customize the instance settings for your needs.

As a general recommendation, AWS Ground Station encourages the use of instances that support enhanced networking for your uplinks and downlinks, such as AWS Nitro System. For more information about enhanced networking, see Enabling enhanced networking with the Elastic Network Adapter (ENA) on Linux instances.

Choose a Template

AWS Ground Station provides templates that demonstrate how to use the service and can be accessed in different ways. Use this guide to find the right template for you.

Using a preconfigured template

You can use a preconfigured template to receive direct broadcast data from the Aqua, SNPP, JPSS-1/NOAA-20, and Terra satellites. These templates contain the required AWS CloudFormation resources to schedule and execute contacts. The AquaSnppJpss template comprises the necessary AWS CloudFormation resources to receive demodulated and decoded direct broadcast data. Use this template as a starting point if you plan to process the data using NASA Direct Readout Labs software (RT-STPS and IPOPP). The AquaSnppJpssTerraDigIF template comprises the necessary AWS CloudFormation resources to receive raw digitized intermediate frequency (DigIF) direct broadcast data. Use this template as a starting point for processing the data using a software defined radio (SDR).

- the section called “AquaSnppJpss Template” (p. 15)
- the section called “AquaSnppJpssTerraDigIF Template” (p. 17)

Using your own satellites

Configuring your own satellites requires a different set of parameters and resources. This is difficult to do on your own. The AWS Ground Station team is available to help you configure your own satellites for use and can help you configure resources for downlink, uplink, and uplink echo streams. To configure your own satellite to use with AWS Ground Station, contact AWS Support.

Accessing Templates

You can access the templates in the regional Amazon S3 bucket below. Note that the following link uses a regional S3 endpoint. Change <us-west-2> to the region in which you are creating the AWS CloudFormation stack.

s3://groundstation-customer-assets-us-west-2/cloudformation_templates

You can also download the templates using the AWS CLI. For information on configuring the AWS CLI, see Configuring the AWS CLI.

AquaSnppJpss Template

The AWS CloudFormation template named AquaSnppJpss.yml is designed to give you quick access to start receiving data for the Aqua, SNPP, and JPSS-1/NOAA-20 satellites. It contains an Amazon EC2 instance and the required AWS Ground Station resources to schedule contacts and receive demodulated and decoded direct broadcast data. This template is a good starting point if you plan to process the data using NASA Direct Readout Labs software (RT-STPS and IPOPP).

If Aqua, SNPP, and JPSS-1/NOAA-20 are not onboarded to your account, see Customer Onboarding (p. 10).
Important
When creating an AWS CloudFormation stack from the provided template, the Amazon EC2 instance will install Data Defender upon creation and needs to complete the installation (approximately 5 minutes) before you manually stop the Amazon EC2 instance. Once Data Defender installation is complete and the instance is stopped, you will need to start the instance before a reserved contact. The template provides the option to automatically start and/or stop the instance for you or you can choose to do it manually.

You can access the template by accessing the customer onboarding S3 bucket. Note that the links below use a regional S3 bucket. Change `<us-west-2>` to the region in which you are creating the AWS CloudFormation stack.

Note
The following instructions use YAML. However, the templates are available in both YAML and JSON format. To use JSON, replace `<.yml>` with `<.json>`.

To download the template using AWS CLI, use the following command:

```

You can view and download the template in the console by navigating to the following URL in your browser:


You can specify the template directly in AWS CloudFormation using the following link:


What resources does the template define?
The AquaSnppJpss template includes the following resources:

- **Data Delivery Service Role** - AWS Ground Station assumes this role to create/delete ENIs in your account in order to stream data.
- (Optional) **Receiver Instance** - The Amazon EC2 instance that will send/receive data to/from your satellite using AWS Ground Station.
- **Instance Security Group** - The security group for your Amazon EC2 instance.
- **Instance Role** - The role for your Amazon EC2 instance.
- **Instance Profile** - The instance profile for your Amazon EC2 instance.
- **Cluster Placement Group** - The placement group in which your Amazon EC2 instance is launched.
- **Dataflow Endpoint Security Group** - The security group that the elastic network interface created by AWS Ground Station belongs to. By default, this security group allows AWS Ground Station to stream traffic to any IP address in your VPC. You can modify this in a way that limits traffic to a specific set of IP addresses.
- **Receiver Instance Network Interface** - An elastic network interface that provides a fixed IP address for AWS Ground Station to connect to. This attaches to the receiver instance on eth1.
- **Receiver Instance Interface Attachment** - An elastic network interface that attaches to your Amazon EC2 instance.
- (Optional) **CloudWatch Event Triggers** - AWS Lambda Function that is triggered using CloudWatch Events sent by AWS Ground Station before and after a contact. The AWS Lambda Function will start and optionally stop your Receiver Instance.
**Dataflow Endpoint Group** - The AWS Ground Station dataflow endpoint group (p. 3) that defines the endpoints used to send/receive data to/from your satellite. As part of the dataflow endpoint group creation, AWS Ground Station creates an elastic network interface in your account to stream data.

**Tracking Config** - The AWS Ground Station tracking config (p. 5) defines how the antenna system tracks your satellite as it moves through the sky.

In addition, the template provides the following resources for the Aqua, SNPP, JPSS-1/NOAA-20 satellites:

- A downlink demod/decode config for JPSS-1/NOAA-20 and SNPP, and a downlink demod/decode config for Aqua.
- A mission profile for JPSS-1/NOAA-20 and SNPP, and a mission profile for Aqua.

The values and parameters for the satellites in this template are already populated. These parameters make it easy for you to use AWS Ground Station immediately with these satellites. You do not need to configure your own values in order to use AWS Ground Station when using this template. However, you can customize the values to make the template work for your use case.

**Where do I receive my data?**

The dataflow endpoint group is set up to use the receiver instance network interface that part of the template creates. The receiver instance uses Data Defender to receive the data stream from AWS Ground Station on the port defined by the dataflow endpoint. Once received, the data is available for consumption via UDP port 50000 on the loopback adapter of the receiver instance. For more information about setting up a dataflow endpoint group, see AWS::GroundStation::DataflowEndpointGroup.

**AquaSnppJpssTerraDigIF Template**

The AWS CloudFormation template named AquaSnppJpssTerraDigIF.yml is designed to give you quick access to start receiving digitized intermediate frequency (DigIF) data for the Aqua, SNPP, JPSS-1/NOAA-20, and Terra satellites. It contains an Amazon EC2 instance and the required AWS CloudFormation resources to receive raw DigIF direct broadcast data. This template is a good starting point for processing the data using a software defined radio (SDR).

If Aqua, SNPP, JPSS-1/NOAA-20, and Terra are not onboarded to your account, see Customer Onboarding (p. 10).

**Important**

When creating an AWS CloudFormation stack from the provided template, the Amazon EC2 instance will install Data Defender upon creation and needs to complete the installation (approximately 5 minutes) before you manually stop the Amazon EC2 instance. Once Data Defender installation is complete and the instance is stopped, you will need to start the instance before a reserved contact. The template provides the option to automatically start and/or stop the instance for you or you can choose to do it manually.

You can access the template by accessing the customer onboarding S3 bucket. Note that the links below use a regional S3 bucket. Change `<us-west-2>` to the region in which you are creating the AWS CloudFormation stack.

**Note**

The following instructions use YAML. However, the templates are available in both YAML and JSON format. To use JSON, replace `<.yml>` with `<.json>`.

To download the template using AWS CLI, use the following command:

```bash
aws s3 cp s3://groundstation-customer-assets-us-west-2/cloudformation_templates/AquaSnppJpssTerraDigIF.yml .
```
Choose a Template

You can view and download the template in the console by navigating to the following URL in your browser:


You can specify the template directly in AWS CloudFormation using the following link:

https://groundstation-customer-assets-us-west-2.s3-us-west-2.amazonaws.com/cloudformation_templates/AquaSnppJpssTerraDigIF.yml

What resources does the template define?

The AquaSnppJpssTerraDigIF template includes the following resources:

- **Data Delivery Service Role** - AWS Ground Station assumes this role to create/delete ENIs in your account in order to stream data.
- **(Optional) Receiver Instance** - The Amazon EC2 instance that will send/receive data to/from your satellite using AWS Ground Station.
  - **Instance Security Group** - The security group for your Amazon EC2 instance.
  - **Instance Role** - The role for your Amazon EC2 instance.
  - **Instance Profile** - The instance profile for your Amazon EC2 instance.
  - **Cluster Placement Group** - The placement group in which your Amazon EC2 instance is launched.
- **Dataflow Endpoint Security Group** - The security group that the elastic network interface created by AWS Ground Station belongs to. By default, this security group allows AWS Ground Station to stream traffic to any IP address in your VPC. You can modify this in a way that limits traffic to a specific set of IP addresses.
- **Receiver Instance Network Interface** - An elastic network interface that provides a fixed IP address for AWS Ground Station to connect to. This attaches to the receiver instance on eth1.
- **Receiver Instance Interface Attachment** - An elastic network interface that attaches to your Amazon EC2 instance.
- **(Optional) CloudWatch Event Triggers** - AWS Lambda Function that is triggered using CloudWatch Events sent by AWS Ground Station before and after a contact. The AWS Lambda Function will start and optionally stop your Receiver Instance.
- **Dataflow Endpoint Group** - The AWS Ground Station dataflow endpoint group (p. 3) that defines the endpoints used to send/receive data to/from your satellite. As part of the dataflow endpoint group creation, AWS Ground Station creates an elastic network interface in your account to stream data.
- **Tracking Config** - The AWS Ground Station tracking config (p. 5) defines how the antenna system tracks your satellite as it moves through the sky.
- **Downlink Dig IF Endpoint Config** - A defined endpoint used to downlink data from your satellite.

In addition, the template provides the following resources for the Aqua, SNPP, JPSS-1/NOAA-20, and Terra satellites:

- A downlink DigIF antenna config for Aqua, SNPP, JPSS-1/NOAA-20, and Terra.
- A mission profile for JPSS-1/NOAA-20 and SNPP, a mission profile for Aqua, and a mission profile for Terra.

The values and parameters for the satellites in this template are already populated. These parameters make it easy for you to use AWS Ground Station immediately with these satellites. You do not need to configure your own values in order to use AWS Ground Station when using this template. However, you can customize the values to make the template work for your use case.
Where do I receive my data?

The dataflow endpoint group is set up to use the receiver instance network interface that part of the template creates. The receiver instance uses Data Defender to receive the data stream from AWS Ground Station on the port defined by the dataflow endpoint. Once received, the data is available for consumption via UDP port 50000 on the loopback adapter of the receiver instance. For more information about setting up a dataflow endpoint group, see AWS::GroundStation::DataflowEndpointGroup.

Step 4: Configure an AWS CloudFormation Stack

After choosing the template that best applies to your use case, configure an AWS CloudFormation stack. The resources that are created in this procedure are configured to the region that you are in when you create them. This includes the mission profile and its properties that determine to which region your data is delivered.

1. In the AWS Management Console, choose Services > CloudFormation.
2. In the navigation pane, choose Stacks. Then, choose Create stack > With new resources (standard).
3. In the Create Stack page, specify the template that you selected in the section called “Choose a Template” (p. 15) by doing one of the following.
   a. Select Amazon S3 URL as your template source, and copy and paste the URL of the template you want to use in Amazon S3 URL. Then, choose Next.
   b. Select Upload a template file as your template source and choose Choose File. Upload the template you downloaded in the section called “Choose a Template” (p. 15). Then, choose Next.
4. In the Specify stack details page, make the following changes:
   a. Enter a name in the Stack Name box. We recommend using a simple name to reduce the possibility of errors in the future.
   b. For CloudWatchEventActions, choose which actions to perform for the CloudWatch event triggers before and after a contact.
   c. For CreateReceiverInstance, choose whether or not you want to create an Amazon EC2 receiver instance.
   d. For Ec2InstanceAmi, enter the AMI to use when creating the Amazon EC2 instance. Use the default value unless you need a custom AMI.
   e. Choose the SSH Key you created in the section called “Step 1: Create EC2 SSH Key Pair” (p. 12).
   f. Choose the Subnetid in which you wish to create your Amazon EC2 instance. We recommend placing your Amazon EC2 instance on a private subnet as a best practice, though it is not required. You can use the Linux Bastion Hosts on the AWS Cloud: Quick Start Reference Deployment to automatically create a private subnet if you have not already configured your account with one in the section called “Step 2: Set Up Your VPC” (p. 13).
      Note
      Your organization may have another subnet dedicated for your Amazon EC2 instance.
   g. Choose the VPC Stack you created in the section called “Step 2: Set Up Your VPC” (p. 13).
   h. Choose Next.
5. Configure stack options and advanced options for your Amazon EC2 instance.
   a. Add any tags and permissions in the Tags and Permissions sections.
   b. Make any changes for your Stack policy, Rollback configuration, Notification options, and Stack creation options.
   c. Choose Next.
Step 5: Install and Configure FE Processor/Radio

The Amazon EC2 instance defined in the AWS CloudFormation template does not have a Front End (FE) processor or software defined radio (SDR) installed by default. You need to install an FE processor or SDR in order to process the VITA-49 packets streamed to/from the AWS Ground Station antenna system.

How you install and configure your FE processor or SDR depends on which FE processor or SDR you are using. Installation of an FE processor or SDR is beyond the scope of this user guide.

To install and configure FE processor/radio, contact AWS Support.

Important
It’s a best practice to run your FE processor or SDR on the instances created by the AWS CloudFormation template to ensure the benefits of the DTLS data streams to/from Data Defender.

Next Steps

Your AWS Ground Station account and resources are now configured and ready for use. These resources are available to use in the AWS Ground Station console where you can enter satellite data, identify antenna locations, communicate, and schedule antenna time for selected satellites. You can also begin using different tools to monitor activity and configure alarms.

Use the following topics for more information:

- Listing and Reserving Contacts (p. 21)
- Monitoring AWS Ground Station (p. 31)
Listing and Reserving Contacts

You can enter satellite data, identify antenna locations, communicate, and schedule antenna time for selected satellites by using the AWS Ground Station console or AWS CLI. You can review, cancel, and reschedule contact reservations up to eight days prior to scheduled time. In addition, you can view the details of your reserved minutes pricing plan if you are using the AWS Ground Station reserved minutes pricing model.

AWS Ground Station supports cross-region data delivery. The dataflow endpoint configs that are part of the mission profile you select determine to which region(s) the data is delivered. For more information about using cross-region data delivery, see Using Cross Region Data Delivery Service (p. 29).

To schedule contacts, your resources must be configured. If you have not configured your resources, see Getting Started (p. 12).

Topics
- Using the Ground Station Console (p. 21)
- Reserving and Managing Contacts with AWS CLI (p. 25)

Using the Ground Station Console

You can use the AWS Ground Station console to reserve, view, and cancel contact reservations. To use the AWS Ground Station console, open the AWS Ground Station console and choose Reserve contacts now.

Use the following topics to use the AWS Ground Station console to reserve, view, and cancel contacts.

Topics
- Reserve a Contact (p. 22)
- View Scheduled and Completed Contacts (p. 23)
- Cancelling Contacts (p. 24)
Reserve a Contact

After accessing the AWS Ground Station console, use your configured resources to reserve contacts in the Contact management table.

1. In the Contact management table, choose the parameters you want to use to search for available contacts. Ensure that you are viewing Available contacts by using the Status filter.

2. Choose a contact that meets your requirements and then choose Reserve contact.

3. In the Reserve Contact dialog box, review your contact reservation information.
   a. (optional) Under Tags, enter a key and value for each tag you want to add.
   b. Choose Reserve.
AWS Ground Station will use the configuration data from your mission profile to execute a contact at the specified ground station.

**View Scheduled and Completed Contacts**

Once you schedule contacts, you can use the AWS Ground Station console to view the details of scheduled and completed contacts.

In the **Contact management** table, choose the parameters you want to use to search for scheduled and completed contacts. Ensure that you are viewing **Scheduled** or **Completed** contacts by using the **Status** filter.
Your scheduled or completed contact(s) will be listed if the contact(s) matches the parameters.

### Cancelling Contacts

You can use the AWS Ground Station console to cancel scheduled contacts

1. In the **Contact management** table, choose the parameters you want to use to search for scheduled and completed contacts. Ensure that you are viewing **Scheduled** contacts by using the **Status** filter.
2. Choose the contact you want to cancel in the list of scheduled contacts. Then, choose **Cancel Contact**.
3. In the **Cancel contact** dialog box, choose **Ok**.
The contact's status will be `CANCELLED`.

## Reserving and Managing Contacts with AWS CLI

You can use AWS CLI to reserve and manage your contacts in AWS Ground Station. Before using AWS CLI to reserve and manage contacts, the following AWS CLI prerequisites must be fulfilled:

- Ensure that AWS CLI is installed. For information about installing AWS CLI, see [Installing the AWS CLI version 2](#).
- Ensure that AWS CLI is configured. For information about configuring AWS CLI, see [Configuring the AWS CLI version 2](#).
- Save your frequently used configuration settings and credentials in files that are maintained by the AWS CLI. You need these settings and credentials to reserve and manage your AWS Ground Station contacts with AWS CLI. For more information about saving your configuration and credential settings, see [Configuration and Credential File Settings](#).

Once AWS CLI is configured and ready to use, review the [AWS Ground Station CLI Command Reference page](#) to familiarize yourself with available commands. Follow the AWS CLI command structure when using this service and prefix your commands with `groundstation` to specify AWS Ground Station as
the service you want to use. For more information on the AWS CLI command structure, see Command Structure in the AWS CLI page. An example command structure is provided below.

```bash
aws groundstation <command> <subcommand> [options and parameters]
```

Use the following topics to reserve, view, and cancel contacts with AWS CLI.

**Topics**
- View and List Contacts with AWS CLI (p. 26)
- Reserve a Contact with AWS CLI (p. 27)
- Cancel a Contact with AWS CLI (p. 27)

## View and List Contacts with AWS CLI

To list and view CANCELLED, COMPLETED, or SCHEDULED contacts with AWS CLI, run `aws groundstation list-contacts` with the following parameters.

- **Start Time** - Specify the start time of your contact with `--start-time <value>`. The following is an acceptable time value format: `YYYY-MM-DDTHH:MM:SSZ`
- **End Time** - Specify the end time of your contact with `--end-time <value>`. The following is an acceptable time value format: `YYYY-MM-DDTHH:MM:SSZ`
- **Status List** - Specify the status of your contact with `--end-time <value>`. Acceptable values include AVAILABLE, CANCELLED, COMPLETED, or SCHEDULED. To see a full list of valid values, see `list-contacts`.

To list and view AVAILABLE contacts with AWS CLI the following parameters are required in addition to the ones listed above.

- **Ground Station ID** - Specify your ground station's ID with `--ground-station <value>`.
- **Mission Profile ARN** - Specify your mission profile's ARN with `--mission-profile-arn <value>`.
- **Satellite ARN** - Specify your satellite ARN with `--satellite-arn <value>`.

You can use `list` commands to look up your resources. For more information on specifying your parameters, see `list-contacts`.

An example command to list available contacts is provided below.

```bash
```

An example of a list of available contacts is provided below.

```json
{
    "contactList": [
        {
            "contactStatus": "AVAILABLE",
            "endTime": "2020-04-10T00:09:22Z",
            "groundStation": "Oregon 1",
            "maximumElevation": {
                "unit": "DEGREE_ANGLE",
                "elevation": 0.0
            }
        }
    ]
}
```
Reserve a Contact with AWS CLI

AWS CLI gives you the option to reserve contacts by the minute. This feature is unique to the AWS CLI and cannot be done in the AWS Ground Station console.

To reserve contacts with AWS CLI, run `aws groundstation reserve-contact` with the following parameters.

- **Ground Station ID** - Specify your ground station's ID with `--ground-station <value>`.
- **Mission Profile ARN** - Specify your mission profile's ARN with `--mission-profile-arn <value>`.
- **Satellite ARN** - Specify your satellite ARN with `--satellite-arn <value>`.
- **Start Time** - Specify the start time of your contact with `--start-time <value>`. The following is an acceptable time value format: `YYYY-MM-DDTHH:MM:SSZ`
- **End Time** - Specify the end time of your contact with `--end-time <value>`. The following is an acceptable time value format: `YYYY-MM-DDTHH:MM:SSZ`

You can use list commands to look up your resources. For more information on specifying your parameters, see `reserve-contact`.

An example command of reserving a contact is provided below.

```bash
```

An example of a successfully reserved contact is provided below.

```json
{
    "contactId": "1111111-2222-3333-4444-555555555555"
}
```

Cancel a Contact with AWS CLI

To cancel a contact with AWS CLI, run `aws groundstation cancel-contact` with the following parameters.

- **Region** - Specify your ground station's region with `--region <value>`.
- **Contact ID** - Specify the contact ID with `--contact-id <value>`.

You can use list commands to look up your resources. For more information on specifying your parameters, see `cancel-contacts`
An example command of reserving a contact is provided below.

```bash
aws groundstation --region us-east-2 cancel-contact --contact-id '11111111-2222-3333-4444-555555555555'
```

An example of a successfully cancelled contact is provided below.

```json
{
  "contactId": "11111111-2222-3333-4444-555555555555"
}
```
Using Cross-Region Data Delivery Service

The AWS Ground Station cross-region data delivery feature gives you the flexibility to send your data from an antenna to an Amazon EC2 instance in your AWS Region. Cross-region data delivery service is currently available in the following antenna-to-destination regions:

- US East (Ohio) Region (us-east-2) to US West (Oregon) Region (us-west-2)
- US West (Oregon) Region (us-west-2) to US East (Ohio) Region (us-east-2)

To use cross-region data delivery, you should have an AWS CloudFormation template configured. For more information about choosing and customizing AWS CloudFormation templates, see Step 3: Choose and Customize an AWS CloudFormation Template (p. 14).

Use the following topics to use cross-region data delivery in AWS Ground Station.

Topics
- To use cross-region data delivery in the console (p. 29)
- To use cross-region data delivery with AWS CLI (p. 30)

To use cross-region data delivery in the console

When you reserve a contact (p. 22) in the AWS Ground Station console, choose the mission profile that is configured to deliver the contact data to your desired region. Ensure that all of your parameters are correct and choose Reserve contact. If you do not see the desired mission profile in the console, check to make sure you created the mission profile in the region where you are viewing the console.

After reserving your contact, you can view scheduled contacts (p. 23) to verify that you have scheduled cross-region data delivery by viewing the location of the ground station antenna and the destination region. The following image shows a contact that is scheduled for cross-region data delivery. The contact is configured to use the Ohio ground station antennas and deliver data to Oregon.
To use cross-region data delivery with AWS CLI

When you reserve a contact in AWS CLI, choose the mission profile that is configured to deliver the contact data to your desired region. Specify the desired mission profile’s ARN with `--mission-profile-arn <value>`. Ensure that all of your parameters are correct and run the command. If you do not see the desired mission profile ARN when viewing and listing contacts, check to make sure you created the mission profile in the region where you are running AWS CLI.

After reserving your contact, you can view scheduled contacts to verify that you have scheduled cross-region data delivery by viewing the location of the ground station antenna and the destination region. The following output shows a contact that is scheduled for cross-region data delivery. The contact is configured to use the Ohio ground station antennas and deliver the data to Oregon.

```json
{
    "contactList": [
        {
            "contactId": "11111111-2222-3333-4444-555555555555",
            "contactStatus": "SCHEDULED",
            "endTime": "2020-05-05T03:16:35-06:00",
            "groundStation": "Ohio 1",
            "maximumElevation": {
                "unit": "DEGREE_ANGLE",
                "value": 26.74
            },
            "postPassEndTime": "2020-05-05T03:17:35-06:00",
            "prePassStartTime": "2020-05-05T03:04:08-06:00",
            "region": "us-west-2",
            "satelliteArn": "arn:aws:groundstation::123456789012:satellite/11111111-2222-3333-4444-555555555555",
            "startTime": "2020-05-05T03:06:08-06:00"
        }
    ]
}
```
Monitoring AWS Ground Station

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Ground Station. AWS provides the following monitoring tools to watch AWS Ground Station, report when something is wrong, and take automatic actions when appropriate.

- **Amazon CloudWatch Events** delivers a near real-time stream of system events that describe changes in AWS resources. CloudWatch Events enables automated event-driven computing, as you can write rules that watch for certain events and trigger automated actions in other AWS services when these events happen. For more information about Amazon CloudWatch Events, see the Amazon CloudWatch Events User Guide.

- **AWS CloudTrail** captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information about AWS CloudTrail, see the AWS CloudTrail User Guide.

- **Amazon CloudWatch Metrics** captures metrics for your scheduled contacts when using AWS Ground Station. CloudWatch Metrics enables you to analyze data based on your channel, polarization, and satellite ID to identify signal strength and errors in your contacts. For more information, see Using Amazon CloudWatch Metrics.

Use the following topics to monitor AWS Ground Station.

**Topics**
- Automating AWS Ground Station with CloudWatch Events (p. 31)
- Logging AWS Ground Station API Calls with AWS CloudTrail (p. 33)
- Metrics with Amazon CloudWatch (p. 35)

Automating AWS Ground Station with CloudWatch Events

Amazon CloudWatch Events enables you to automate your AWS services and respond automatically to system events such as application availability issues or resource changes. Events from AWS services are delivered to CloudWatch Events in near real time. You can write simple rules to indicate which events are of interest to you, and what automated actions to take when an event matches a rule. The actions that can be automatically triggered include the following:

- Invoking an AWS Lambda function
- Invoking Amazon EC2 Run Command
- Relaying the event to Amazon Kinesis Data Streams
- Activating an AWS Step Functions state machine
- Notifying an Amazon SNS topic or an AWS SMS queue

Some examples of using CloudWatch Events with AWS Ground Station include:

- Invoking a Lambda function to automate the starting and stopping of Amazon EC2 instances based off the event state.
• Publishing to an Amazon SNS topic whenever a contact changes states. These topics can be set up to send out email notices at the beginning or end of contacts.

For more information, see the Amazon CloudWatch Events User Guide.

Example CloudWatch Events

Ground Station Contact State Change

If you want to perform a specific action when an upcoming contact is changing states, you can setup a CloudWatch rule to automate this action. This is helpful for when you want to receive notifications about the state changes of your contact. The events are sent to the region that the contact was scheduled from.

An example is provided below.

```
{
  "version": "0",
  "id": "01234567-0123-0123",
  "account": "123456789012",
  "time": "2019-05-30T17:40:30Z",
  "region": "us-west-2",
  "source": "aws.groundstation",
  "resources": [
  ],
  "detailType": "Ground Station Contact State Change",
  "detail": {
    "contactId": "11111111-1111-1111-1111-111111111111",
    "groundstationId": "Ground Station 1",
    "satelliteArn": "arn:aws:groundstation::123456789012:satellite/11111111-1111-1111-111111111111",
    "contactStatus": "PASS"
  },
  "account": "123456789012"
}
```

Possible states for the contactStatus include PREPASS, PASS, POSTPASS, and COMPLETED.

Ground Station Dataflow Endpoint Group State Change

If you want to perform an action when your dataflow endpoint group is being used to receive data, you can set up a CloudWatch rule to automate this action. This will allow you to perform different actions in response to the dataflow endpoint group status changing states. This event will be sent to the region of the dataflow endpoint group.

An example is provided below.

```
{
  "version": "0",
  "id": "01234567-0123-0123",
  "account": "123456789012",
  "time": "2019-05-30T17:40:30Z",
  "region": "us-west-2",
  "source": "aws.groundstation",
  "resources": [
    "arn:aws:groundstation::123456789012:contact/11111111-1111-1111-1111-111111111111"
  ],
  "detailType": "Ground Station Contact State Change",
  "detail": {
    "contactId": "11111111-1111-1111-1111-111111111111",
    "groundstationId": "Ground Station 1",
    "satelliteArn": "arn:aws:groundstation::123456789012:satellite/11111111-1111-1111-111111111111",
    "contactStatus": "PASS"
  },
  "account": "123456789012"
}
```
"resources": [
],
"detailType": "Ground Station Dataflow Endpoint Group State Change",
"detail": {
    "dataflowEndpointGroupId": "bad957a8-1d60-4c45-a92a-39feb98921d",
    "groundstationId": "Ground Station 1",
    "contactId": "98ddd10f-f2bc-479c-bf7d-55644737fb09",
    "missionProfileArn": "arn:aws:groundstation:us-west-2:123456789012:mission-profile/c513c84c-eb40-4473-88a2-d482648c9234",
    "dataflowEndpointGroupState": "PREPASS"
},
"account": "123456789012"
}

Possible states for the `dataflowEndpointGroupState` include PREPASS, PASS, POSTPASS, and COMPLETED.

Logging AWS Ground Station API Calls with AWS CloudTrail

AWS Ground Station is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Ground Station. CloudTrail captures all API calls for AWS Ground Station as events. The calls captured include calls from the AWS Ground Station console and code calls to the AWS Ground Station API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS Ground Station. If you don’t configure a trail, you can still view the most recent events in the CloudTrail console in Event history. Using the information collected by CloudTrail, you can determine the request that was made to AWS Ground Station, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

AWS Ground Station Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Ground Station, that activity is recorded in a CloudTrail event along with other AWS service events in Event history. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for AWS Ground Station, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
Understanding AWS Ground Station Log File Entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the ReserveContact action.

Example: ReserveContact

```json
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EX_PRINCIPLE_ID",
        "arn": "arn:aws:sts::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-05-15T21:11:59Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EX_PRINCIPLE_ID",
                "arn": "arn:aws:iam::123456789012:role/Alice",
                "accountId": "123456789012",
                "userName": "Alice"
            }
        }
    },
    "eventSource": "groundstation.amazonaws.com",
    "eventName": "ReserveContact",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "Coral/Jakarta",
    "requestParameters": {
```
Metrics with Amazon CloudWatch

During a contact, AWS Ground Station automatically captures and sends data to CloudWatch for analysis. Your data can be viewed on a graph or as source code in the Amazon CloudWatch console. For more information about accessing and CloudWatch Metrics, see Using Amazon CloudWatch Metrics.

AWS Ground Station Metrics and Dimensions

What metrics are available?

The following metrics are available from AWS Ground Station.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Es/N0</td>
<td>The signal to noise ratio. Units: dBm (decibels relative to milliwatts)</td>
</tr>
<tr>
<td>BitErrorRate</td>
<td>The unrecoverable error rate on bits in a given number of bit transmissions. Bit errors are caused by noise, distortion, or interference. Units: Bits errors per unit time</td>
</tr>
<tr>
<td>BlockErrorRate</td>
<td>The error rate of blocks in a given number of received blocks. Block errors are caused by interference. Units: Erroneous blocks / Total number of blocks</td>
</tr>
<tr>
<td>ReceivedPower</td>
<td>The measured signal strength in the demodulator/decoder. Units: dBm (decibels relative to milliwatts)</td>
</tr>
</tbody>
</table>

What dimensions are used for AWS Ground Station?

You can filter AWS Ground Station data using the following dimensions.
Dimension | Description
---|---
Channel | The channels for each contact include One, Two, I (in-phase), and Q (quadrature).
Polarization | The polarization for each contact include LHCP (Left Hand Circular Polarized) or RHCP (Right Hand Circular Polarized).
SatelliteId | The satellite ID contains the ARN of the satellite for your contacts.

**Viewing Metrics**

When viewing graphed metrics, it is important to note that the aggregation window determines how your metrics will be displayed. Each metric in a contact can be displayed as data per second for 3 hours after the data is received. Your data will be aggregated by CloudWatch Metrics as data per minute after that 3 hour period has elapsed. If you need to view your metrics on a data per second measurement, it is recommended to view your data within the 3 hour period after the data is received or persist it outside of CloudWatch Metrics.

In addition, any data captured within the first 60 seconds will not contain enough information to produce meaningful metrics, and will likely not be displayed. In order to view meaningful metrics, it is recommended to view your data after 60 seconds has passed.

For more information about graphing AWS Ground Station metrics in CloudWatch, see [Graphing Metrics](#).

**To view metrics using the console**

2. In the navigation pane, choose Metrics.
3. Select the **GroundStation** namespace.
4. Select your desired metric dimensions (for example, **Channel**, **Polarization**, **SatelliteId**).
5. The **All metrics** tab displays all metrics for that dimension in the namespace. You can do the following:

   a. To sort the table, use the column heading.

   b. To graph a metric, select the check box associated with the metric. To select all metrics, select the check box in the heading row of the table.

   c. To filter by resource, choose the resource ID and then choose **Add to search**.

   d. To filter by metric, choose the metric name and then choose **Add to search**.

**To view metrics using AWS CLI**

1. Ensure that AWS CLI is installed. For information about installing AWS CLI, see Installing the AWS CLI version 2.

2. Create a CloudWatch agent configuration JSON file. For instructions on creating a CloudWatch agent configuration file, see Create the CloudWatch Agent Configuration File.

3. List the available CloudWatch metrics by running `aws cloudwatch list-metrics`.

4. Modify the JSON file you created in step 2 to match the Satellite ID from your metrics.

   **Note**
   Do not reduce the **Period** field to a value under 60. AWS Ground Station publishes metrics every 60 seconds and no metrics will be returned if the value is reduced.

5. Run `aws cloudwatch get-metric-data` with time periods of your passes and your CloudWatch agent configuration JSON file. An example is provided below.

   ```bash
   aws cloudwatch get-metric-data
   -start-time 2020-02-26T19:12:00Z
   -end-time 2020-02-26T19:24:00Z
   -metric-data-queries file://metricdata.json
   ```

   Metrics will be provided with timestamps from your contact. An example output of AWS Ground Station metrics is provided below.

   ```json
   {
   "MetricDataResults": [
   {
   "Id": "myQuery",
   "Label": "Es/N0",
   "Timestamps": ["2020-02-18T19:44:00Z",
   "2020-02-18T19:43:00Z",
   "2020-02-18T19:42:00Z",
   "2020-02-18T19:41:00Z",
   "2020-02-18T19:40:00Z",
   "2020-02-18T19:39:00Z",
   "2020-02-18T19:38:00Z",
   "2020-02-18T19:37:00Z",
   ```
{ }

"Values": [
    24.58344556958329,
    24.251638725562216,
    22.919391450230158,
    22.83838908204037,
    23.303086848486842,
    22.845261784583364,
    21.34531397048953,
    19.171561698261222
],

"StatusCode": "Complete"

"Messages": []
}
Troubleshooting

The following documentation can help you troubleshoot issues that may prevent an AWS Ground Station contact from completing successfully.

Topics
• Troubleshooting AWS Ground Station Contact (p. 39)

Troubleshooting AWS Ground Station Contact

If you are unable to successfully complete an AWS Ground Station contact, you will need to verify that your Amazon EC2 instance is running, verify that Data Defender is running, and verify that your Data Defender stream is configured properly.

Prerequisite

The following procedures assume that an Amazon EC2 instance is already set up. To set up an Amazon EC2 instance in AWS Ground Station, see Getting Started.

Step 1: Verify that Your EC2 Instance is Running

1. Locate the Amazon EC2 instance that was used for the contact you are troubleshooting. Use the following steps:
   a. In your CloudFormation dashboard, select the stack that contains your Amazon EC2 instance.
   b. Choose the Resources tab and locate your Amazon EC2 instance in the Logical ID column. Verify that the instance is created in the Status column.
   c. In the Physical ID column, choose the link for your Amazon EC2 instance. This will take you to the Amazon EC2 management console.
2. In the Amazon EC2 management console, ensure that your Amazon EC2 Instance State is running.
3. If your instance is running, continue to the next step. If your instance is not running, start the instance by using the following step:
   • With your Amazon EC2 instance selected, choose Actions > Instance State > Start.

Step 2: Verify that Data Defender is Running

Verifying the status of Data Defender requires you to connect to your instance in Amazon EC2. For more details on connecting to your instance, see Connect to Your Linux Instance.

The following procedure provides troubleshooting steps using commands in an SSH client.

1. Open a terminal or command prompt and connect to your Amazon EC2 instance by using SSH. Forward port 80 of the remote host in order to view the Data Defender web UI. The following commands demonstrate how to use SSH to connect to an Amazon EC2 instance through a bastion with port forwarding enabled.

   Note
   You must replace <SSH KEY>, <BASTION HOST>, and <HOST> with your specific ssh key, bastion host name, and Amazon EC2 instance host name.

   For Windows
Step 3: Verify that Your Data Defender Stream is Configured

1. In a web browser, access your DDX Web User Interface by entering the following address in the address bar: localhost:8080. Then, press Enter.

2. On the DataDefender dashboard, choose Go to Details.

3. Select your stream from the list of streams, and choose Edit Stream.
4. In the **Stream Wizard** dialog box, do the following:

   a. In the **WAN Transport** pane, ensure **WAN to LAN** is selected for **Stream Direction**.

   b. In the **Port** box, ensure the WAN port you have chosen for your dataflow endpoint group is present. By default, this port is 55888. Then, choose **Next**.

   c. In the **Local Endpoint** pane, ensure that a valid port is present in the **Port** box. By default, this port is 50000. This is the port that you will receive your data after Data Defender has received it from the AWS Ground Station service. Then, choose **Next**.

   d. Choose **Finish** in the remaining menu if you have changed any values. Otherwise, you can cancel out of the **Stream Wizard** menu.
You have now ensured that your Amazon EC2 instance and Data Defender are both running and configured properly to receive data from AWS Ground Station. If you continue experience issues, contact AWS Support.
AWS Ground Station Security

Cloud security at AWS is the highest priority. As an AWS customer, you will benefit from a data center and network architecture built to meet the requirements of the most security-sensitive organizations. AWS provides security-specific tools and features to help you meet your security objectives. These tools and features include network security, configuration management, access control, and data security.

When using AWS Ground Station, we recommend that you follow industry best practices and implement end-to-end encryption. AWS provides APIs for you to integrate encryption and data protection. For more information about AWS security, see the Introduction to AWS Security whitepaper.

Use the following topics to learn how to secure your AWS Ground Station resources.

Topics
- Authentication and Access Control for AWS Ground Station (p. 43)

Authentication and Access Control for AWS Ground Station

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be authenticated (signed in) and authorized (have permissions) to use AWS Ground Station resources. IAM is a feature of your AWS account offered at no additional charge.

Topics
- Audience (p. 43)
- Authentication (p. 44)
- Controlling Access Using Policies (p. 45)
- Learn More (p. 46)
- How AWS Ground Station Works with IAM (p. 47)
- AWS Ground Station Identity-Based Policy Examples (p. 50)
- Troubleshooting AWS Ground Station Authentication and Access Control (p. 53)

Audience

Authentication and access control matters to you in different ways, depending on the work you do in AWS Ground Station.

Service user – If you use the AWS Ground Station service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AWS Ground Station features to do your work, you might need additional permissions. Understanding access control can help you request the right permissions from your administrator. If you cannot access a feature in AWS Ground Station, see Troubleshooting AWS Ground Station Authentication and Access Control (p. 53).

Service administrator – If you're in charge of AWS Ground Station resources at your company, you probably have full access to AWS Ground Station. It's your job to determine which AWS Ground Station
features and resources your employees should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of authentication and controlling access. To learn more about how your company can use IAM with AWS Ground Station, see How AWS Ground Station Works with IAM (p. 47).

**IAM administrator** – If you’re an IAM administrator, you already understand the concepts of authentication and access control. But you want to learn details about how you can write policies to control access to AWS Ground Station. To view example AWS Ground Station identity-based policies that you can use in IAM, see AWS Ground Station Identity-Based Policy Examples (p. 50).

**Authentication**

Authentication is how you sign in to AWS using your credentials.

You must be **authenticated** (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role. You can sign in to the AWS Management Console or access AWS programmatically. AWS provides SDK and command line tools to cryptographically sign your request using your credentials. If you don’t use AWS tools, you must sign the request yourself. Do this using **Signature Version 4**, a protocol for authenticating inbound API requests. For more information about authenticating requests, see **Signature Version 4 Signing Process** in the **AWS General Reference**. For more information about signing in using the AWS Management Console, see The IAM Console and Sign-in Page in the **IAM User Guide**.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Using Multi-Factor Authentication (MFA) in AWS in the **IAM User Guide**.

**AWS Account Root User**

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account **root user** and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

**IAM Users and Groups**

An **IAM user** is an entity within your AWS account that has specific permissions for a single person or application. An IAM user can have long-term credentials such as a user name and password or a set of access keys. To authenticate from the AWS Management Console as an IAM user, you must sign in with your user name and password. To learn more about signing in to the console, see How IAM Users Sign In to AWS in the **IAM User Guide**. To be authenticated from the AWS CLI or AWS API, you must provide your IAM user access key ID and secret key. To learn how to generate access keys, see Managing Access Keys for IAM Users in the **IAM User Guide**. When you generate access keys for an IAM user, make sure you view and save the key pair. You cannot recover the secret access key in the future. Instead, you must generate a new access key pair.

An IAM **group** is a collection of IAM users. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named AdminIAM and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see When to Create an IAM User (Instead of a Role) in the **IAM User Guide**.
IAM Roles

An IAM role is an entity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM Roles in the IAM User Guide.

IAM roles with temporary credentials are useful in the following situations:

• **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.

• **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as federated users. AWS assigns a role to a federated user when access is requested through an identity provider. For more information about federated users, see Federated Users and Roles in the IAM User Guide.

• **Cross-account access** – You can use an IAM role to allow a trusted principal in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see How IAM Roles Differ from Resource-based Policies in the IAM User Guide.

• **AWS service access** – A service role is an IAM role that a service assumes to perform actions in your account on your behalf. When you set up some AWS service environments, you must define a role for the service to assume. This service role must include all the permissions that are required for the service to access the AWS resources that it needs. Service roles vary from service to service, but many allow you to choose your permissions as long as you meet the documented requirements for that service. Service roles provide access only within your account and cannot be used to grant access to services in other accounts. You can create, modify, and delete a service role from within IAM. For example, you can create a role that allows Amazon Redshift to access an Amazon S3 bucket on your behalf and then load data from that bucket into an Amazon Redshift cluster. For more information, see Creating a Role to Delegate Permissions to an AWS Service in the IAM User Guide.

• **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances in the IAM User Guide.

To learn whether to use IAM roles, see When to Create an IAM Role (Instead of a User) in the IAM User Guide.

Controlling Access Using Policies

You control access in AWS by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an entity or resource, defines their permissions. AWS evaluates these policies when a principal, such as a user, makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON Policies in the IAM User Guide.

An IAM administrator can use policies to specify who has access to AWS resources, and what actions they can perform on those resources. Every IAM entity (user or role) starts with no permissions. In other words, by default, users can do nothing, not even change their own password. To give a user permission
to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the IAM `GetUser` action. A user with that policy can get user information from the AWS Management Console, the AWS CLI, or the AWS API.

### Identity-Based Policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, role, or group. These policies control what actions that identity can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM Policies in the [IAM User Guide](https://docs.aws.amazon.com/IAM/latest/UserGuide/idpolicies.html).

Identity-based policies can be further categorized as **inline policies** or **managed policies**. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choosing Between Managed Policies and Inline Policies in the [IAM User Guide](https://docs.aws.amazon.com/IAM/latest/UserGuide/idpolicies.html).

### Other Policy Types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity. You can set a permissions boundary for an entity. That entity can then perform only the actions that are allowed by both its identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role as the principal are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions Boundaries for IAM Entities in the [IAM User Guide](https://docs.aws.amazon.com/IAM/latest/UserGuide/idpolicies.html).

- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing the AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see About Service Control Policies in the [AWS Organizations User Guide](https://docs.aws.amazon.com/AWSOrganizations/latest/UserGuide/what-are-service-control-policies.html).

- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The permissions for a session come from identity-based policies for the IAM entity (user or role) used to create the session and the session policy. Permissions can also come from a resource-based policy. For more information, see Session Policies in the [IAM User Guide](https://docs.aws.amazon.com/IAM/latest/UserGuide/idpolicies.html).

### Multiple Policy Types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy Evaluation Logic in the [IAM User Guide](https://docs.aws.amazon.com/IAM/latest/UserGuide/idpolicies.html).

### Learn More

For more information about authentication and access control for AWS Ground Station, continue to the following pages:
How AWS Ground Station Works with IAM

Before you use IAM to control access to AWS Ground Station, you should understand what IAM features are available to use with AWS Ground Station. To get a high-level view of how AWS Ground Station and other AWS services work with IAM, see AWS Services That Work with IAM in the IAM User Guide.

Topics
- AWS Ground Station Identity-Based Policies (p. 47)
- AWS Ground Station Resource-Based Policies (p. 49)
- Authorization Based on AWS Ground Station Tags (p. 49)
- AWS Ground Station IAM Roles (p. 50)

AWS Ground Station Identity-Based Policies

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. AWS Ground Station supports specific actions, resources, and condition keys. To learn about all of the elements that you use in a JSON policy, see IAM JSON Policy Elements Reference in the IAM User Guide.

AWS Ground Station Actions

The Action element of an IAM identity-based policy describes the specific action or actions that will be allowed or denied by the policy. Policy actions in AWS Ground Station use the following prefix before the action: groundstation. For example: groundstation:Get*, groundstation:List*, groundstation:Describe* (for all AWS Ground Station actions). For a list of the actions, see the Actions Defined by AWS Ground Station.

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word Describe, include the following action:

```
"Action": "groundstation:Describe*"
```

The following table describes actions which are common for console access:

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CancelContact</td>
<td>Grants permission to cancel a contact</td>
</tr>
<tr>
<td>DescribeContact</td>
<td>Grants permission to describe a contact</td>
</tr>
<tr>
<td>ListContacts</td>
<td>Grants permission to return a list of contacts</td>
</tr>
<tr>
<td>ReserveContact</td>
<td>Grants permission to reserve a contact</td>
</tr>
</tbody>
</table>

To see a list of AWS Ground Station actions, see the AWS Ground Station API Reference.

Resources

The Resource element specifies the object or objects to which the action applies. Statements must include either a Resource or a NotResource element. You can specify a resource using an ARN or using
the wildcard (*) to indicate that the statement applies to all resources. For more information about the format of ARNs, see Amazon Resource Names (ARNs) and AWS Service Namespaces.

The AWS Ground Station Config resource has the following ARN format:

```
arn:{Partition}:groundstation:{Region}:{AccountID}:config/{configType}/{configId}
```

To specify the example `11111111-2222-3333-4444-555555555555` Config in your statement, you would use the following ARN:

```
```

To specify all Config objects that belong to a specific account, use the wildcard (*) in the following format:

```
```

Some AWS Ground Station actions, such as those for creating resources, cannot be performed on a specific resource. In those cases, you must use the wildcard (*) in the following format:

```
"Resource": "*
```

Many AWS Ground Station API actions involve multiple resources. For example, CreateConfig can create an AWS Ground Station Config across multiple satellites, so an IAM user must have permissions to use the Config and the contact. To specify multiple resources in a single statement, separate their ARNs with commas in the following format:

```
"Resource": [
]
```

The following table summarizes how to create resources with AWS Ground Station:

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateConfig</td>
<td>Grants permission to create a Config</td>
</tr>
<tr>
<td>CreateDataflowEndpointGroup</td>
<td>Grants permission to create a dataflow endpoint group</td>
</tr>
<tr>
<td>CreateMissionProfile</td>
<td>Grants permission to create a mission profile</td>
</tr>
</tbody>
</table>

A Contact is another common resource in AWS Ground Station, which has a resource ARN. See the following example:

```
"arn:aws:groundstation:us-west-2:123456789012:contact/11111111-2222-3333-4444-555555555555"
```

The following table summarizes how to update other resources:
### How AWS Ground Station Works with IAM

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UpdateConfig</td>
<td>Grants permission to update a Config</td>
</tr>
<tr>
<td>UpdateMissionProfile</td>
<td>Grants permission to update a mission profile</td>
</tr>
</tbody>
</table>

To see a list of AWS Ground Station resource types and their ARNs, see Resources Defined by AWS Ground Station in the IAM User Guide. To learn with which actions you can specify the ARN of each resource, see Actions Defined by AWS Ground Station.

### Condition Keys

The Condition element (or Condition block) lets you specify conditions in which a statement is in effect. The Condition element is optional. You can build conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request. AWS Ground Station defines its own set of condition keys and also supports using some global condition keys. To see all AWS global condition keys, see AWS Global Condition Context Keys in the IAM User Guide.

**Note**

Do not use the aws:SourceIp AWS global condition key with AWS CloudFormation. AWS CloudFormation provisions resources by using its own IP address, not the IP address of the originating request. For example, AWS CloudFormation makes requests from its IP address to launch an Amazon EC2 instance or to create an Amazon S3 bucket. It does not use the IP address from the CreateStack operation or the `aws cloudformation create-stack` command, nor does it use the IP address of the person who makes the call.

If you specify multiple Condition elements in a statement, or multiple keys in a single Condition element, AWS evaluates them using a logical AND operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical OR operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see Policy Variables in the IAM User Guide.

All AWS Ground Station actions support the `aws:RequestedRegion` and `groundstation:Region` condition keys. For more information, see Example: Restricting Access to a Specific Region.

To see a list of AWS Ground Station condition keys, see Condition Keys for AWS Ground Station in the IAM User Guide. To learn with which actions and resources you can use a condition key, see Actions Defined by AWS Ground Station.

### Examples

To view examples of AWS Ground Station identity-based policies, see AWS Ground Station Identity-Based Policy Examples (p. 50).

### AWS Ground Station Resource-Based Policies

AWS Ground Station does not support resource-based policies. To view an example of a detailed resource-based policy page, see Using Resource-based Policies for AWS Lambda.

### Authorization Based on AWS Ground Station Tags

You can attach tags to AWS Ground Station resources or pass tags in a request to AWS Ground Station. To control access based on tags, you provide tag information in the condition element of a policy using
the `groundstation:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

To view an example identity-based policy for limiting access to a resource based on the tags on that resource, see Viewing AWS Ground Station ConfigIds Based on Tags (p. 52).

**AWS Ground Station IAM Roles**

An IAM role is an entity within your AWS account that has specific permissions.

AWS Ground Station currently does not support service-linked roles.

**Using Temporary Credentials with AWS Ground Station**

You can use temporary credentials to sign in with federation, assume an IAM role, or to assume a cross-account role. You obtain temporary security credentials by calling AWS STS API operations such as `AssumeRole` or `GetFederationToken`.

AWS Ground Station supports using temporary credentials.

**AWS Ground Station Identity-Based Policy Examples**

By default, IAM users and roles don't have permission to create or modify AWS Ground Station resources. They also can't perform tasks using the AWS Ground Station console, CLI, or API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the specified resources they need. The administrator must then attach those policies to the IAM users or groups that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see Creating Policies on the JSON Tab in the IAM User Guide.

The following example policy grants Console General Access to AWS Ground Station:

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "groundstation:*"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

The following example policy grants read-only access to AWS Ground Station:

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "groundstation:Get*",
        "groundstation:List*",
        "groundstation:Describe*"
      ]
    }
  ]
}
```
Identity-Based Policy Examples

For more information about writing IAM policies, see IAM Policies in the IAM User Guide.

Topics

- Policy Best Practices (p. 51)
- Using the AWS Ground Station Console (p. 51)
- Allow Users to View Their Own Permissions (p. 52)
- Viewing AWS Ground Station ConfigIds Based on Tags (p. 52)

Policy Best Practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete AWS Ground Station resources in your account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get Started Using AWS Managed Policies** – To start using AWS Ground Station quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see Get Started Using Permissions With AWS Managed Policies in the IAM User Guide.

- **Grant Least Privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later. For more information, see Grant Least Privilege in the IAM User Guide.

- **Enable MFA for Sensitive Operations** – For extra security, require IAM users to use multi-factor authentication (MFA) to access sensitive resources or API operations. For more information, see Using Multi-Factor Authentication (MFA) in AWS in the IAM User Guide.

- **Use Policy Conditions for Extra Security** – To the extent that it's practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions to allow requests only within a specified date or time range, or to require the use of SSL or MFA. For more information, see IAM JSON Policy Elements: Condition in the IAM User Guide.

Using the AWS Ground Station Console

To access the AWS Ground Station console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the AWS Ground Station resources in your AWS account. If you create an identity-based permissions policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities with that policy.

To ensure that those entities can still use the AWS Ground Station console, also attach the following AWS managed policy to the user. For more information, see Adding Permissions to a User in the IAM User Guide.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Resource": [
        "*"
      ]
    }
  ]
}
```
"Action": [
    "groundstation:Get*",
    "groundstation:List*",
    "groundstation:Describe*"
],
"Resource": [
    "*"
]
}

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, you need only the permissions that match the API operation that you're trying to perform.

**Allow Users to View Their Own Permissions**

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsForUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
      ],
      "Resource": [
        "arn:aws:iam::*:user/${aws:username}"
      ]
    },
    {
      "Sid": "NavigateInConsole",
      "Effect": "Allow",
      "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
      ],
      "Resource": "*"
    }
  ]
}
```

**Viewing AWS Ground Station ConfigIds Based on Tags**

You can use conditions in your identity-based policy to control access to AWS Ground Station resources based on tags. This example shows how you might create a policy that allows viewing a Config.
However, permission is granted only if the configId tag Owner has the value of that user's user name. This policy also grants the permissions necessary to complete this action on the console.

```json
{
    "Sid": "VisualEditor1",
    "Effect": "Allow",
    "Action": [
        "groundstation:GetConfig"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "groundstation:ResourceTag/Owner": "${aws:username}"
        }
    }
}
```

You can attach this policy to the IAM users in your account. If a user named richard-roe attempts to view an AWS Ground Station configId, the configId must be tagged Owner=richard-roe or owner=richard-roe. Otherwise he is denied access. The condition tag key Owner matches both Owner and owner because condition key names are not case-sensitive. For more information, see IAM JSON Policy Elements: condition in the IAM User Guide.

**Troubleshooting AWS Ground Station Authentication and Access Control**

Use the following information to help you diagnose and fix common issues that you might encounter when working with AWS Ground Station and IAM.

**Topics**
- I am not authorized to perform an action in AWS Ground Station (p. 53)
- I am not authorized to perform iam:PassRole (p. 54)
- I want to view my access keys (p. 54)
- I'm an administrator and want to allow others to access AWS Ground Station (p. 54)
- I want to allow people outside of my AWS account to access my AWS Ground Station resources (p. 54)

**I am not authorized to perform an action in AWS Ground Station**

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

The following example error occurs when the mateojackson IAM user tries to use the console to view details about a Config but does not have groundstation:Get* permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
groundstation:Get* on resource: my-example-config
```

In this case, Mateo asks his administrator to update his policies to allow him to access the my-example-config resource using the groundstation:Get* action.
I am not authorized to perform `iam:PassRole`

If you receive an error that you're not authorized to perform the `iam:PassRole` action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password. Ask that person to update your policies to allow you to pass a role to AWS Ground Station.

Some AWS services allow you to pass an existing role to that service, instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in AWS Ground Station. However, the action requires the service to have permissions granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform: iam:PassRole
```

In this case, Mary asks her administrator to update her policies to allow her to perform the `iam:PassRole` action.

I want to view my access keys

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair.

Access keys consist of two parts: an access key ID (for example, `AKIAIOSFODNN7EX`) and a secret access key (for example, `wJalrXUtnFEMI/K7M/bPxRFY`). Like a user name and password, you must use both the access key ID and secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

**Important**

Do not provide your access keys to a third party, even to help find your canonical user ID. By doing this, you might give someone permanent access to your account.

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must add new access keys to your IAM user. You can have a maximum of two access keys. If you already have two, you must delete one key pair before creating a new one. To view instructions, see Managing Access Keys in the IAM User Guide.

I'm an administrator and want to allow others to access AWS Ground Station

To allow others to access AWS Ground Station, you must create an IAM entity (user or role) for the person or application that needs access. They will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in AWS Ground Station.

To get started right away, see Creating Your First IAM Delegated User and Group in the IAM User Guide.

I want to allow people outside of my AWS account to access my AWS Ground Station resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.
To learn more, consult the following:

- To learn whether AWS Ground Station supports these features, see How AWS Ground Station Works with IAM (p. 47).
- To learn how to provide access to your resources across AWS accounts that you own, see Creating Your First IAM Delegated User and Group in the IAM User Guide.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing Access to AWS Accounts Owned by Third Parties in the IAM User Guide.
- To learn how to provide access through identity federation, see Providing Access to Externally Authenticated Users (Identity Federation) in the IAM User Guide.
- To learn the difference between using roles and resource-based policies for cross-account access, see How IAM Roles Differ from Resource-based Policies in the IAM User Guide.
The following table describes the important changes to the documentation since the last release of AWS Ground Station.

| update-history-change         | update-history-description                                                                 | update-history-date      |
|------------------------------|==========================================================================================|--------------------------|
| New Feature (p. 56)          | Updated the user guide to include integration with AWS CLI.                               | April 17, 2020           |
| New Feature (p. 56)          | Updated the user guide to include integration with CloudWatch Metrics.                    | February 24, 2020        |
| New Template (p. 56)         | Public Broadcast Satellites (AquaSnppJpss Template) added to the AWS Ground Station User Guide | February 19, 2020        |
| New Feature (p. 56)          | Updated the user guide to include cross-region data delivery.                             | February 5, 2020         |
| Documentation Update (p. 56) | Updated examples and descriptions for monitoring AWS Ground Station with CloudWatch Events.| February 4, 2020         |
| Documentation Update (p. 56) | Template locations have been updated and the Getting Started and Troubleshooting sections have been revised.| December 19, 2019        |
| New Troubleshooting Section (p. 56) | Troubleshooting section added to the AWS Ground Station User Guide.          | November 7, 2019         |
| New Getting Started topic (p. 56) | Updated the Getting Started topic, which includes the most current AWS CloudFormation templates. | July 1, 2019             |
| Kindle version (p. 56)       | Published Kindle version of the AWS Ground Station User Guide.                         | June 20, 2019            |
| New service and guide (p. 56)| This is the initial release of AWS Ground Station and the AWS Ground Station User Guide. | May 23, 2019             |
For the latest AWS terminology, see the AWS glossary in the AWS General Reference.