
AWS Ground Station

User Guide



AWS Ground Station: User Guide

Copyright © 2022 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

What Is AWS Ground Station?	1
How AWS Ground Station Works	2
Data Delivery to Amazon S3	2
Data Delivery to Amazon EC2	2
More Information	3
Service Terms	3
Core Components	3
Dataflow Endpoint Groups	4
Configs	4
Mission Profiles	9
AWS Ground Station Locations	10
Finding the AWS Region for a Ground Station	10
Example Ground Station Located Outside of an AWS Region	11
Setting Up AWS Ground Station	12
Step 1: Sign Up for AWS	12
Step 2: Add Permissions to Your AWS Account	12
Step 3: Customer Onboarding	13
Next Steps	14
Getting Started	15
Basic Concepts	15
Prerequisites	15
Step 1: Choose an AWS CloudFormation Template	15
Preconfigured Templates	15
Building your own template	17
Step 2: Configure an AWS CloudFormation Stack	17
Listing and Reserving Contacts	19
Using the Ground Station Console	19
Reserve a Contact	19
View Scheduled and Completed Contacts	21
Cancelling Contacts	22
Reserving and Managing Contacts with AWS CLI	23
View and List Contacts with AWS CLI	23
Reserve a Contact with AWS CLI	24
Describe a Contact with AWS CLI	25
Cancel a Contact with AWS CLI	25
Data Delivery to Amazon EC2	27
Step 1: Create EC2 SSH Key Pair	27
Step 2: Set Up Your VPC	28
Step 3: Choose and Customize an AWS CloudFormation Template	29
Configuring your Amazon EC2 Instance Settings	29
Manually Creating and Configuring Resources	30
Choose a Template	30
Step 4: Configure an AWS CloudFormation Stack	34
Step 5: Install and Configure FE Processor/Radio	35
(Optional) Install and Configure Data Defender Manually	36
Step 1: SSH Into Your EC2 Instance	36
Step 2: Install Data Defender Dependencies Using YUM	36
Step 3: Download the Data Defender Files	37
Step 4: Install Data Defender	37
Step 5: Configure Data Defender	38
Step 6: Configure the Data Defender Streams	39
Next Steps	44
Using Cross-Region Data Delivery	45
To use cross-region data delivery in the console	45

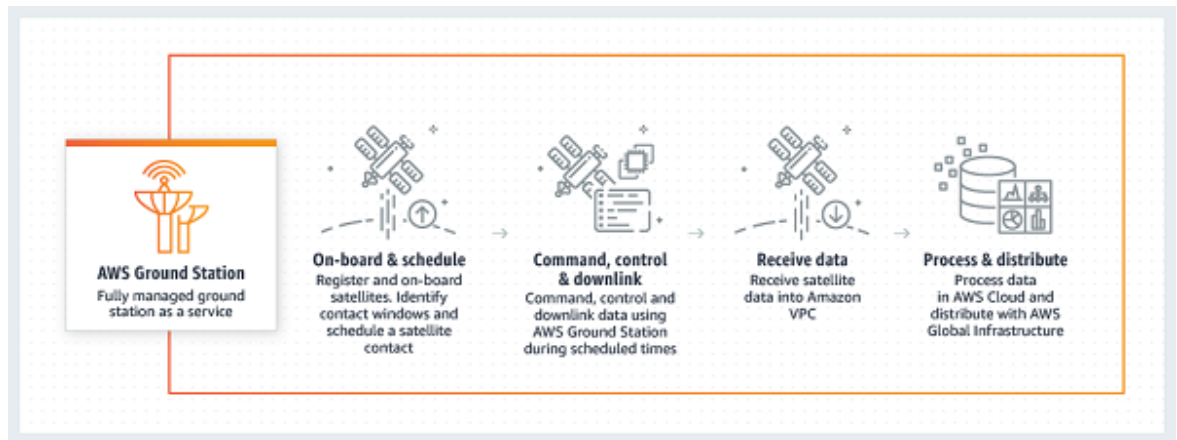
To use cross-region data delivery with AWS CLI	46
Monitoring AWS Ground Station	47
Automating with CloudWatch Events	47
Example CloudWatch Events	48
Logging API Calls with CloudTrail	49
AWS Ground Station Information in CloudTrail	50
Understanding AWS Ground Station Log File Entries	50
Metrics with Amazon CloudWatch	51
AWS Ground Station Metrics and Dimensions	51
Viewing Metrics	52
Troubleshooting	55
Troubleshooting Contacts that Deliver Data to Amazon EC2	55
Step 1: Verify that Your EC2 Instance is Running	55
Step 2: Verify that Data Defender is Running	55
Step 3: Verify that Your Data Defender Stream is Configured	57
Ground Station Contact Statuses	58
Contact Statuses	58
Security	60
Authentication and Access Control	60
Audience	60
Authentication	61
Controlling Access Using Policies	62
Learn More	63
How AWS Ground Station Works with IAM	64
Identity-Based Policy Examples	67
Troubleshooting	70
Data Encryption at rest for AWS Ground Station	73
How AWS Ground Station uses grants in AWS KMS	74
Create a customer managed key	74
To create a symmetric customer managed key	74
Key policy	74
Specifying a customer managed key for AWS Ground Station	76
AWS Ground Station encryption context	76
AWS Ground Station encryption context	76
Ephemeris Encryption Context:	76
Using encryption context for monitoring	76
Using encryption context to control access to your customer managed key	76
Monitoring your encryption keys for AWS Ground Station	77
CreateGrant (Cloudtrail)	77
DescribeKey (Cloudtrail)	78
GenerateDataKey (Cloudtrail)	79
Decrypt (Cloudtrail)	80
Satellite Ephemeris Data	82
Default Ephemeris Data	82
Which Ephemeris Is Used	82
Effect of new Ephemerides on Previously Scheduled Contacts	82
Getting the Current Ephemeris for a Satellite	83
Example GetSatellite return for a satellite using a default ephemeris	83
Example GetSatellite for a satellite using a custom ephemeris	83
Providing Custom Ephemeris Data	84
Overview	84
Creating a custom Ephemeris	84
Create a TLE Set Ephemeris via API	84
Uploading Ephemeris data from an S3 bucket	86
Troubleshooting Invalid Ephemerides	86
Reverting To Default Ephemeris Data	87
Document History	89

AWS glossary 90

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.



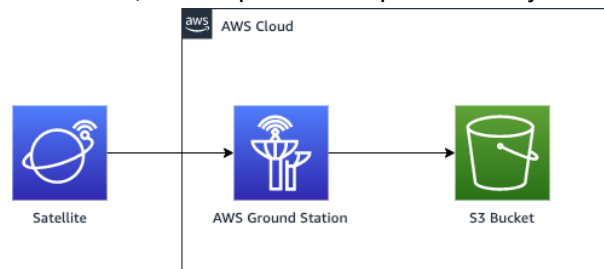
How AWS Ground Station Works

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 or delivered asynchronously to an Amazon Simple Storage Service (Amazon S3) bucket in your account.

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.

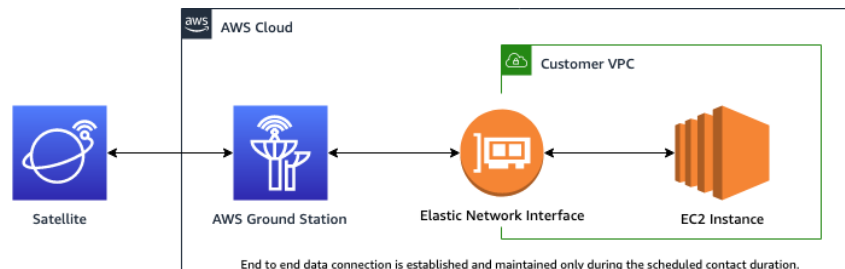
Data Delivery to Amazon S3

With data delivery to Amazon S3, your contact data is delivered asynchronously to an Amazon S3 bucket in your account. Your contact data is delivered as packet capture (pcap) files to allow replaying the contact data into a Software Defined Radio (SDR) or to extract the payload data from the pcap files for processing. The pcap files are delivered to your Amazon S3 bucket every 30 seconds as contact data is received by the antenna hardware to allow processing contact data during the contact if desired. Once received, you can process the data using your own post-processing software or use other AWS services like Amazon SageMaker or Amazon Rekognition. Data delivery to Amazon S3 is only available for downlinking data from your satellite; it is not possible to uplink data to your satellite from Amazon S3.



Data Delivery to Amazon EC2

With data delivery to Amazon EC2, your contact data is streamed to and from your Amazon EC2 instance. You can process your data in real-time on your Amazon EC2 instance or forward the data for post-processing.



More Information

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 Amazon 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 Amazon 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 \(p. 4\)](#)
- [Configs \(p. 4\)](#)
- [Mission Profiles \(p. 9\)](#)

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 concurrently, but those contacts must be executed on different dataflow endpoint groups.

See the following documentation for more information about how to perform operations on dataflow endpoint groups using AWS CloudFormation, the AWS Command Line Interface, or the AWS Ground Station API.

- [AWS::GroundStation::DataflowEndpointGroup CloudFormation resource type](#)
- [Dataflow Endpoint Group AWS CLI reference](#)
- [Dataflow Endpoint Group API reference](#)

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.

See the following documentation for more information about how to perform operations on configs using AWS CloudFormation, the AWS Command Line Interface, or the AWS Ground Station API. Links to documentation for specific config types are also provided below.

- [AWS::GroundStation::Config CloudFormation resource type](#)
- [Config AWS CLI reference](#)
- [Config API reference](#)

Dataflow Endpoint Config

Note

Dataflow endpoint configs are only used for data delivery to Amazon EC2 and are not used for data delivery to Amazon S3.

You can use dataflow endpoint configs to specify which dataflow endpoint in a [dataflow endpoint group](#) (p. 4) from which or to which you want data to flow during a contact. The two parameters of a dataflow endpoint config specify the name and region of the dataflow endpoint. When reserving a contact, AWS Ground Station analyzes the [mission profile](#) (p. 9) you specified and attempts to find a dataflow endpoint *group* that contains all of the dataflow *endpoints* specified by the dataflow endpoint *configs* contained in your mission profile.

The `dataflowEndpointName` property of a dataflow endpoint config specifies which dataflow endpoint in a dataflow endpoint group to which or from which data will flow during a contact.

The `dataflowEndpointRegion` property specifies which region the dataflow endpoint resides in. If a region is specified in your dataflow endpoint config, AWS Ground Station looks for a dataflow endpoint in the region specified. If no region is specified, AWS Ground Station will default to the contact's ground station region. A contact is considered a [cross region data delivery](#) (p. 45) contact if your dataflow endpoint's region is not the same as the contact's ground station region.

The IAM role must have a trust policy that allows the `groundstation.amazonaws.com` service principal to assume the role. See the [Example Trust Policy](#) (p. 5) section below for an example. During endpoint creation the endpoint resource id does not exist, the trust policy must use an asterisk (*) in place of *your-endpoint-id* and can be updated after creation with the endpoint resource id.

See the following documentation for more information about how to perform operations on dataflow endpoint configs using AWS CloudFormation, the AWS Command Line Interface, or the AWS Ground Station API.

- [AWS::GroundStation::Config DataflowEndpointConfig CloudFormation property](#)
- [Config AWS CLI reference](#) (see the `dataflowEndpointConfig` -> (structure) section)
- [DataflowEndpointConfig API reference](#)

Example Trust Policy

For more information on how to update a role's trust policy, see [Managing IAM roles](#) in the IAM User Guide.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "groundstation.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "your-account-id"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:groundstation:dataflow-endpoint-region:your-account-id:dataflow-endpoint-group/your-endpoint-id"
        }
      }
    }
  ]
}
```

```
}
]
}
```

S3 Recording Config

Note

S3 recording configs are only used for data delivery to Amazon S3 and are not used for data delivery to Amazon EC2.

You can use S3 recording configs to specify an Amazon S3 bucket to which you want downlinked data delivered. The two parameters of an S3 recording config specify the Amazon S3 bucket and IAM role for AWS Ground Station to assume when delivering the data to your Amazon S3 bucket. The IAM role and Amazon S3 bucket specified must meet the following criteria:

- The Amazon S3 bucket's name must begin with `aws-groundstation-`.
- The IAM role must have a trust policy that allows the `groundstation.amazonaws.com` service principal to assume the role. See the [Example Trust Policy \(p. 6\)](#) section below for an example. During config creation the config resource id does not exist, the trust policy must use an asterisk (*) in place of *your-config-id* and can be updated after creation with the config resource id.
- The IAM role must have an IAM policy that allows the role to perform the `s3:GetBucketLocation` action on the bucket and `s3:PutObject` action on the bucket's objects. If the Amazon S3 bucket has a bucket policy, then the bucket policy must also allow the IAM role to perform these actions. See the [Example Role Policy \(p. 6\)](#) section below for an example.

Example Trust Policy

For more information on how to update a role's trust policy, see [Managing IAM roles](#) in the IAM User Guide.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "groundstation.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "your-account-id"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:groundstation:config-region:your-account-id:config/s3-
recording/your-config-id"
        }
      }
    }
  ]
}
```

Example Role Policy

For more information on how to update or attach a role policy, see [Managing IAM policies](#) in the IAM User Guide.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::your-bucket-name"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::your-bucket-name/*"
      ]
    }
  ]
}
```

See the following documentation for more information about how to perform operations on S3 recording configs using AWS CloudFormation, the AWS Command Line Interface, or the AWS Ground Station API.

- [AWS::GroundStation::Config S3RecordingConfig CloudFormation property](#)
- [Config AWS CLI reference](#) (see the `s3RecordingConfig` -> (structure) section)
- [S3RecordingConfig API reference](#)

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.

See the following documentation for more information about how to perform operations on tracking configs using AWS CloudFormation, the AWS Command Line Interface, or the AWS Ground Station API.

- [AWS::GroundStation::Config TrackingConfig CloudFormation property](#)
- [Config AWS CLI reference](#) (see the `trackingConfig` -> (structure) section)
- [TrackingConfig API reference](#)

Antenna Downlink Config

You can use antenna downlink configs to configure the antenna for downlink during your contact. They consist of a spectrum config that specifies the frequency, bandwidth, and polarization that should be

used during your downlink contact. If your downlink use case requires demodulation or decoding, see the [??? \(p. 8\)](#).

See the following documentation for more information about how to perform operations on antenna downlink configs using AWS CloudFormation, the AWS Command Line Interface, or the AWS Ground Station API.

- [AWS::GroundStation::Config AntennaDownlinkConfig CloudFormation property](#)
- [Config AWS CLI reference](#) (see the `antennaDownlinkConfig` -> `(structure)` section)
- [AntennaDownlinkConfig API reference](#)

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.

See the following documentation for more information about how to perform operations on antenna downlink demod decode configs using AWS CloudFormation, the AWS Command Line Interface, or the AWS Ground Station API.

- [AWS::GroundStation::Config AntennaDownlinkDemodDecodeConfig CloudFormation property](#)
- [Config AWS CLI reference](#) (see the `antennaDownlinkDemodDecodeConfig` -> `(structure)` section)
- [AntennaDownlinkDemodDecodeConfig API reference](#)

Antenna Uplink Config

You can use antenna uplink configs to configure the antenna for uplink during your contact. They consist of a spectrum config with frequency, polarization, and target effective isotropic radiated power (EIRP). For information about how to configure a contact for uplink loopback, see [??? \(p. 8\)](#).

See the following documentation for more information about how to perform operations on antenna uplink configs using AWS CloudFormation, the AWS Command Line Interface, or the AWS Ground Station API.

- [AWS::GroundStation::Config AntennaUplinkConfig CloudFormation property](#)
- [Config AWS CLI reference](#) (see the `antennaUplinkConfig` -> `(structure)` section)
- [AntennaUplinkConfig API reference](#)

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.

See the following documentation for more information about how to perform operations on uplink echo configs using AWS CloudFormation, the AWS Command Line Interface, or the AWS Ground Station API.

- [AWS::GroundStation::Config UplinkEchoConfig CloudFormation property](#)
- [Config AWS CLI reference](#) (see the `uplinkEchoConfig` -> `(structure)` section)

- [UplinkEchoConfig API reference](#)

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 how the antenna will be configured and where data will go during your contact.

Aside from [tracking configs \(p. 7\)](#), 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.

See the following documentation for more information about how to perform operations on mission profiles using AWS CloudFormation, the AWS Command Line Interface, or the AWS Ground Station API.

- [AWS::GroundStation::MissionProfile CloudFormation resource type](#)
- [Mission Profile AWS CLI reference](#)
- [Mission Profile API reference](#)

AWS Ground Station Locations

Customers can transmit and receive data using AWS Ground Station antennas in the following locations: US (Oregon), US (Ohio), Middle East (Bahrain), Europe (Stockholm), Asia Pacific (Sydney), Europe (Ireland), Africa (Cape Town), US (Hawaii), Asia Pacific (Seoul), Asia Pacific (Singapore), and South America (Punta Arenas) [in Preview]

Customers can deliver data and configure their contacts with the AWS Ground Station console in the following regions: US West (Oregon), US East (Ohio), Middle East (Bahrain), Europe (Stockholm), Asia Pacific (Sydney), Europe (Ireland), Africa (Cape Town), US East (N. Virginia), Europe (Frankfurt), Asia Pacific (Seoul), Asia Pacific (Singapore), and South America (São Paulo) [in Preview]. More regions and antenna locations coming soon.

Note: You can only create AWS Ground Station resources in the regions that host the AWS Ground Station console mentioned in the prior paragraph.



Topics

- [Finding the AWS Region for a Ground Station \(p. 10\)](#)

Finding the AWS Region for a Ground Station

The AWS Global Network includes Ground Station locations that are not physically located in the [AWS Region](#) to which they are connected. Listing and reserving contacts at one of these Ground Station locations must be performed using the AWS Region to which the Ground Station is connected.

There are multiple methods of determining a Ground Station's AWS Region. The AWS Ground Station console page displays the Ground Station's AWS Region when displaying it in both the filters and contacts table as shown in the image below. The AWS SDK contains the Ground Station's AWS Region in the [ListGroundStation](#) response. Finally, the AWS CLI contains the Ground Station's AWS Region in the [list-ground-stations](#) response.

AWS Ground Station User Guide

Example Ground Station Located Outside of an AWS Region

Contact management (5)

Cancel contactReserve contact

Manage contacts using the table below.

Ground station

All ground stations ▲
All ground stations
Ohio 1 (us-east-2)
Oregon 1 (us-west-2)
Sydney 1 (ap-southeast-2)

Satellite catalog number

28645 ▼

Status

Available ▼

End date and time (UTC +00:00)

2020/11/23 19:55 2020/11/28 19:55

< 1 >

	Catalog number	Ground station	Start time (AOS) ▲	End time (LOS)	Maximum elevation (deg.)	Region	Status
<input type="radio"/>	28645	Ohio 1 (us-east-2)	2020-11-24T03:01:14.000Z	2020-11-24T04:59:14.000Z	29.10	us-east-2	AVAILABLE
<input type="radio"/>	28645	Ohio 1 (us-east-2)	2020-11-25T03:11:35.000Z	2020-11-25T05:09:35.000Z	30.73	us-east-2	AVAILABLE
<input type="radio"/>	28645	Ohio 1 (us-east-2)	2020-11-26T03:21:42.000Z	2020-11-26T05:19:42.000Z	32.27	us-east-2	AVAILABLE
<input type="radio"/>	28645	Ohio 1 (us-east-2)	2020-11-27T03:31:37.000Z	2020-11-27T05:29:37.000Z	33.71	us-east-2	AVAILABLE
<input type="radio"/>	28645	Ohio 1 (us-east-2)	2020-11-28T03:40:37.000Z	2020-11-28T05:38:37.000Z	35.05	us-east-2	AVAILABLE

Topics

- [Example Ground Station Located Outside of an AWS Region \(p. 11\)](#)

Example Ground Station Located Outside of an AWS Region

Hawaii 1 is an example of a Ground Station location that is not physically located in the AWS Region to which it is connected. The Hawaii 1 Ground Station is located in Hawaii, USA but is connected to the us-west-2 (Oregon) AWS Region. In order to list and reserve contacts using Hawaii 1, you must have a [mission profile \(p. 9\)](#) configured in the us-west-2 (Oregon) AWS Region and use the us-west-2 (Oregon) AWS Region in the AWS Ground Station console, AWS CLI, or AWS SDK.

- To list and [reserve contacts \(p. 19\)](#) for Hawaii 1 in the AWS Ground Station console you must use the AWS Ground Station console in the us-west-2 (Oregon) region.
- To list and reserve contacts for Hawaii 1 using the AWS CLI you must specify the region as us-west-2 using the `--region` [CLI argument](#).
- To list and reserve contacts for Hawaii 1 using the AWS SDK you must set the region of your client to us-west-2. How you set this depends on the programming language your are using. An example of how to set this using JavaScript is described in the [AWS SDK for JavaScript documentation](#). For more information, refer to the language specific [SDK documentation](#).

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. 12)
- [Step 2: Add Permissions to Your AWS Account](#) (p. 12)
- [Step 3: Customer Onboarding](#) (p. 13)
- [Next Steps](#) (p. 14)

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" \(p. 12\)](#).

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:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
    "Action": [
      "groundstation:*"
    ],
    "Resource": [
      "*"
    ]
  }
]
```

- For Read-only privileges, set **Action** to **groundstation:Get***, **groundstation:List***, and **groundstation:Describe*** as follows:

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

- For additional security through multifactor authentication, set **Action** to **groundstation:***, and **Condition/Bool** to **aws:MultiFactorAuthPresent:true** as follows:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "groundstation:*",
      "Resource": "*",
      "Condition": {
        "Bool": {
          "aws:MultiFactorAuthPresent": true
        }
      }
    }
  ]
}
```

3. In the IAM console, attach the policy you created to the desired user.

For more information about IAM users and attaching policies, see the [IAM User Guide](#).

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. 19).

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 [Create Your Resources Using a AWS CloudFormation Template \(p. 15\)](#) 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 \(p. 15\)](#) 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. AWS Ground Station delivers your contact data asynchronously to an Amazon Simple Storage Service (Amazon S3) bucket in your account or synchronously by streaming it to and from an Amazon Elastic Compute Cloud (Amazon EC2) instance in your account. The following steps describe how to configure the resources required to receive contact data asynchronously in an Amazon S3 bucket. See the [??? \(p. 27\)](#) guide for information about how to use data delivery to Amazon EC2.

Topics

- [Basic Concepts \(p. 15\)](#)
- [Prerequisites \(p. 15\)](#)
- [Step 1: Choose an AWS CloudFormation Template \(p. 15\)](#)
- [Step 2: Configure an AWS CloudFormation Stack \(p. 17\)](#)

Basic Concepts

Before you begin, you should familiarize yourself with the basic concepts in AWS Ground Station. For more information, see [??? \(p. 3\)](#).

Then, continue on to [??? \(p. 15\)](#) 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 [??? \(p. 12\)](#).

Then, continue on to [??? \(p. 15\)](#).

Step 1: Choose an AWS CloudFormation Template

After you [onboard \(p. 13\)](#) your satellite, you need to define mission profiles to define the AWS Ground Station antenna configuration to downlink data from 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?](#)

Preconfigured Templates

Today, you can configure multiple streams of data per contact to flow into an S3 bucket. 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.

AWS Ground Station provides templates for both data stream formats that demonstrate how to use the service. Use this guide to find the right template for you.

Available templates

You can use a preconfigured template to receive direct broadcast data from the Aqua, SNPP, JPSS-1/NOAA-20, and Terra satellites. These [AWS CloudFormation](#) templates contain the required AWS Ground Station and Amazon S3 resources to schedule and execute contacts and receive the data in an Amazon S3 bucket in your account. If Aqua, SNPP, JPSS-1/NOAA-20, and Terra are not onboarded to your account, see [Customer Onboarding](#) (p. 13).

- The AWS CloudFormation template named `AquaSnppJpss-1DemodDecodeS3DataDelivery.yml` contains an Amazon S3 bucket 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).

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

```
aws s3 cp s3://groundstation-cloudformation-templates-us-west-2/
AquaSnppJpss-1DemodDecodeS3DataDelivery.yml .
```

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

```
https://s3.console.aws.amazon.com/s3/object/groundstation-cloudformation-templates-us-
west-2/AquaSnppJpss-1DemodDecodeS3DataDelivery.yml
```

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

```
https://groundstation-cloudformation-templates-us-west-2.s3.us-west-2.amazonaws.com/
AquaSnppJpss-1DemodDecodeS3DataDelivery.yml
```

- The AWS CloudFormation template named `AquaSnppJpss-1TerraDigIfS3DataDelivery.yml` contains an Amazon S3 bucket and the required AWS Ground Station resources to schedule contacts and receive VITA-49 Signal/IP direct broadcast data. This template is a good starting point if you plan to process the data using a software defined radio (SDR) to demodulate and decode the data before post-processing.

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

```
aws s3 cp s3://groundstation-cloudformation-templates-us-west-2/
AquaSnppJpss-1TerraDigIfS3DataDelivery.yml .
```

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

```
https://s3.console.aws.amazon.com/s3/object/groundstation-cloudformation-templates-us-
west-2/AquaSnppJpss-1TerraDigIfS3DataDelivery.yml
```

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

```
https://groundstation-cloudformation-templates-us-west-2.s3.us-west-2.amazonaws.com/
AquaSnppJpss-1TerraDigIfS3DataDelivery.yml
```

What resources do these template define?

Both of the templates contain the same resources, with the sole difference being the antenna configs. See the **Antenna Config** description below for more information.

- **Amazon S3 Bucket** - The bucket to which the downlinked data will be delivered. The name of this bucket starts with `aws-groundstation` to meet criteria described in [S3 Recording Config \(p. 6\)](#).
- **IAM Role** - A role assumable by the `groundstation.amazonaws.com` service principal that AWS Ground Station assumes when writing the downlinked data to your Amazon S3 bucket.
- **Amazon S3 Bucket Policy** - A policy that allows the IAM Role to perform the following actions on your Amazon S3 bucket and its objects:
 - `s3:GetBucketLocation`
 - `s3:PutObject`
- **Tracking Config** - An AWS Ground Station [tracking config \(p. 7\)](#) that defines how the antenna system tracks your satellite as it moves through the sky.
- **S3 Recording Config** - An AWS Ground Station [S3 recording config \(p. 6\)](#) that references the Amazon S3 bucket and IAM role for AWS Ground Station to use when delivering your data.
- **Antenna Config** - An AWS Ground Station antenna config that specifies how to configure the AWS Ground Station antenna during a contact. The `AquaSnppJpss-1DemodDecodeS3DataDelivery.yml` template contains an [antenna downlink demod decode config \(p. 8\)](#) that configures the AWS Ground Station antenna to demodulate and decode the downlinked data before delivering it to your Amazon S3 bucket. The `AquaSnppJpss-1TerraDigIfS3DataDelivery.yml` instead contains an [antenna decode config \(p. 7\)](#) that configures the AWS Ground Station antenna to deliver the data to your Amazon S3 as VITA-49 Signal/IP packets.
- **Mission Profile** - An AWS Ground Station [mission profile \(p. 9\)](#) that groups all of the AWS Ground Station configs together to allow you to schedule and execute contacts using the configurations referenced.

Building your own template

Configuring the resources to schedule and execute contacts for your own satellites requires you configure the AWS Ground Station resources in your account to match your satellite's settings. This is difficult to do on your own. The AWS Ground Station team is available to help you configure the AWS Ground Station resources in your account to downlink from and uplink to your satellite. To configure your own satellite to use with AWS Ground Station, [contact AWS Support](#).

Step 2: 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.

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 "Step 1: Choose an AWS CloudFormation 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 "Step 1: Choose an AWS CloudFormation Template" \(p. 15\)](#). Then, choose **Next**.

4. Perform the following steps in the **Specify stack details** page:
 - 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. 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**.
6. After reviewing your stack details, select the **Capabilities** acknowledgement, and choose **Create stack**.

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 \(p. 45\)](#).

To schedule contacts, your resources must be configured. If you have not configured your resources, see [Getting Started \(p. 15\)](#).

Topics

- [Using the Ground Station Console \(p. 19\)](#)
- [Reserving and Managing Contacts with AWS CLI \(p. 23\)](#)

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 \(p. 19\)](#)
- [View Scheduled and Completed Contacts \(p. 21\)](#)
- [Cancelling Contacts \(p. 22\)](#)

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.

Manage contacts using the table below.

Ground station	Satellite catalog number	Status
All ground stations ▼	25994 ▼	Available ▼

Mission profile

TERRA ▼

Start date and time (UTC +00:00)	End date and time (UTC +00:00)
2019/05/20 18:07	2019/05/25 18:07

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

Contact management (22) Cancel contact Reserve contact

Manage contacts using the table below.

Ground station	Satellite catalog number	Status
All ground stations ▼	25994 ▼	Available ▼

Mission profile

TERRA ▼

Start date and time (UTC +00:00)	End date and time (UTC +00:00)
2019/05/20 18:19	2019/05/22 18:19

< 1 2 3 >

	Catalog number	Ground station	Start time (AOS) ▲	End time (LOS)	Maximum elevation (deg.)	Region	Status
+	25994	Oregon 1	2019-05-20T18:49:21.000Z	2019-05-20T19:01:36.000Z	77.22	us-west-2	AVAILABLE

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**.

Reserve contact ✕

You are about to reserve a contact.

Reservation information

Satellite catalog number	Ground station
25994	Ohio 1
Mission profile	Max elevation (degrees)
TERRA (us-west-2)	8.17
Start time	End time
2019-05-22T01:48:03.000Z	2019-05-22T01:51:19.000Z

Tags- optional

Add optional tags to the contact reservation.

<input type="text" value="Key"/>	<input type="text" value="Value"/>
----------------------------------	------------------------------------

Cancel 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.

Contact management (1) Cancel contact Reserve contact

Manage contacts using the table below.

Ground station: Oregon 1 Satellite catalog number: 37849 Status: Scheduled

Mission profile: 37849 SNPP And 43013 JPSS

Start date and time (UTC +00:00): 2020/03/01 14:17 End date and time (UTC +00:00): 2020/03/31 14:17

Catalog number	Ground station	Start time (AOS)	End time (LOS)	Maximum elevation (deg.)	Region	Status
37849	Oregon 1	2020-03-16T20:22:54.000Z	2020-03-16T20:35:15.000Z	64.84	us-west-2	COMPLETED

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**.

Contact management (2) Cancel contact Reserve contact

Manage contacts using the table below.

Ground station: All ground stations Satellite catalog number: 37849 Status: All

Mission profile: 37849 SNPP And 43013 JPSS

Start date and time (UTC +00:00): 2020/04/10 11:00 End date and time (UTC +00:00): 2020/04/10 14:17

Catalog number	Ground station	Start time (AOS)	End time (LOS)	Maximum elevation (deg.)	Region	Status
<input type="radio"/> 37849	Oregon 1	2020-04-10T11:09:02.000Z	2020-04-10T11:19:58.000Z	23.46	us-west-2	AVAILABLE
<input type="radio"/> 37849	Oregon 1	2020-04-10T11:09:02.000Z	2020-04-10T11:19:58.000Z	23.46	us-west-2	CANCELLED

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.

```
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 \(p. 23\)](#)
- [Reserve a Contact with AWS CLI \(p. 24\)](#)
- [Describe a Contact with AWS CLI \(p. 25\)](#)
- [Cancel a Contact with AWS CLI \(p. 25\)](#)

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 `--status-list <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.

```
aws groundstation --region us-east-2 list-contacts --ground-station 'Ohio 1'
--mission-profile-arn 'arn:aws:groundstation:us-east-2:123456789012:mission-
profile/11111111-2222-3333-4444-555555555555' --satellite-arn
'arn:aws:groundstation::123456789012:satellite/11111111-2222-3333-4444-555555555555'
--start-time '2020-04-10T00:09:22Z' --end-time '2020-04-10T00:11:22' --status-list
'AVAILABLE'
```

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

```
{
  "contactList": [
    {
      "contactStatus": "AVAILABLE",
      "endTime": "2020-04-15T03:16:35-06:00",
      "groundStation": "Oregon 1",
      "maximumElevation": {
        "unit": "DEGREE_ANGLE",
        "value": 11.22
      },
      "missionProfileArn": "arn:aws:groundstation:us-west-2:111111111111:mission-
profile/11111111-2222-3333-4444-555555555555",
      "region": "us-west-2",
      "satelliteArn":
"arn:aws:groundstation::111111111111:satellite/11111111-2222-3333-4444-555555555555",
      "startTime": "2020-04-15T03:06:08-06:00"
    }
  ]
}
```

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`

Contact reservation is an asynchronous process. The response to the `reserve-contact` command provides the contact identifier. In order to determine the outcome of the asynchronous reservation process, use `describe-contact`. For more information on this, see the section below titled [Describe a Contact with AWS CLI](#) (p. 25).

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.

```
aws groundstation reserve-contact --ground-station 'Ohio 1' --mission-  
profile-arn 'arn:aws:groundstation:us-east-2:123456789012:mission-  
profile/11111111-2222-3333-4444-555555555555' --satellite-arn  
'arn:aws:groundstation::123456789012:satellite/11111111-2222-3333-4444-555555555555' --  
start-time '2020-04-10T00:09:22Z' --end-time '2020-04-10T00:11:22Z'
```

An example of a successfully reserved contact is provided below.

```
{  
  "contactId": "11111111-2222-3333-4444-555555555555"  
}
```

Describe a Contact with AWS CLI

To see the status of a contact/reservation with AWS CLI, use the `describe-contact` CLI command. This is helpful for verifying the outcome of the asynchronous contact reservation process, monitoring the status of an in-progress contact, and determining the status of a finished contact.

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

- **Contact ID** - Specify your contact ID with `--contact-id <value>`.

You can use `list` commands to look up your resources. For more information on specifying your parameters, see [describe-contact](#).

An example command of describing a contact is provided below.

```
aws groundstation describe-contact --contact-id 11111111-2222-3333-4444-555555555555
```

An example of a successfully scheduled contact is provided below.

```
{  
  "groundStation": "Ireland 1",  
  "tags": {},  
  "missionProfileArn": "arn:aws:groundstation:us-west-2:111111111111:mission-  
profile/11111111-2222-3333-4444-555555555555",  
  "region": "us-west-2",  
  "contactId": "11111111-2222-3333-4444-555555555555",  
  "prePassStartTime": 1645850471.0,  
  "postPassEndTime": 1645851172.0,  
  "startTime": 1645850591.0,  
  "maximumElevation": {  
    "value": 12.66,  
    "unit": "DEGREE_ANGLE"  
  },  
  "satelliteArn":  
  "arn:aws:groundstation::111111111111:satellite/11111111-2222-3333-4444-555555555555",  
  "endTime": 1645851052.0,  
  "contactStatus": "SCHEDULED"  
}
```

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.

```
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.

```
{  
  "contactId": "11111111-2222-3333-4444-555555555555"  
}
```

Data Delivery to Amazon EC2

AWS Ground Station delivers your contact data asynchronously to an Amazon Simple Storage Service (Amazon S3) bucket in your account or synchronously by streaming it to and from an Amazon Elastic Compute Cloud (Amazon EC2) instance in your account. The following steps describe how to configure the resources required to stream contact data to and from an Amazon EC2 instance. See the [??? \(p. 15\)](#) guide for information about data delivery to Amazon S3.

Topics

- [Step 1: Create EC2 SSH Key Pair \(p. 27\)](#)
- [Step 2: Set Up Your VPC \(p. 28\)](#)
- [Step 3: Choose and Customize an AWS CloudFormation Template \(p. 29\)](#)
- [Step 4: Configure an AWS CloudFormation Stack \(p. 34\)](#)
- [Step 5: Install and Configure FE Processor/Radio \(p. 35\)](#)
- [\(Optional\) Install and Configure Data Defender Manually \(p. 36\)](#)
- [Next Steps \(p. 44\)](#)

Step 1: Create EC2 SSH Key Pair

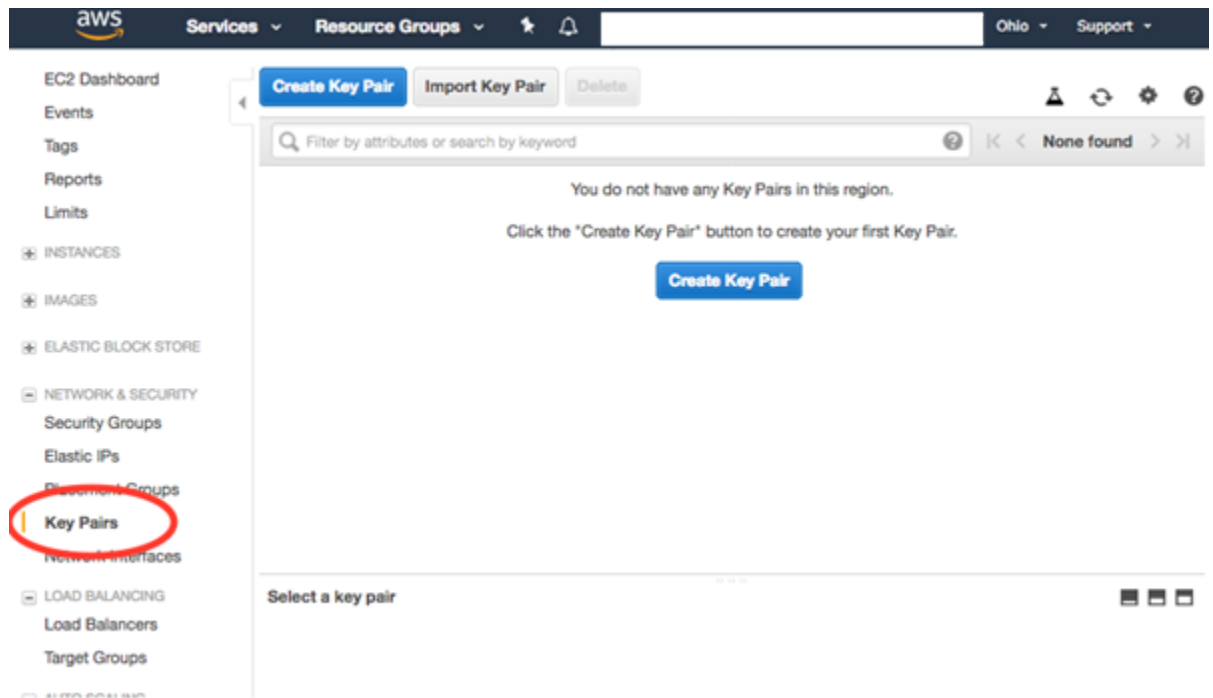
If you do not already have one, create a new key pair in the Amazon EC2 console for each AWS Region where you plan to receive data. Use the steps below.

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:
 - a. Enter a stack name for your host in the **Stack Name** box.
 - b. For **Availability Zones**, select the Availability Zones you wish to use for the subnets in the VPC. At least two Availability Zones must be selected.
 - c. For **Allowed bastion external access CIDR**, enter the CIDR block that you would like to enable SSH access from. If you are unsure, you can use the value of **0.0.0.0/0** to enable SSH access from any host that has the SSH key.
 - d. For **Key pair name**, choose the key pair name you created in [the section called "Step 1: Create EC2 SSH Key Pair" \(p. 27\)](#).
 - e. For **Bastion instance type**, choose **t2.micro**.

Important

The **t2.micro** instance type is not available for the Europe (Stockholm) Region (eu-north-1). If you are using AWS Ground Station in the Europe (Stockholm) Region (eu-north-1), choose **t3.micro**.

- f. For **TCP forwarding**, choose **true**.
 - g. (Optional) Make other edits and changes as necessary. To customize your deployment, you can change your VPC configuration, choose the number and type of bastion host instances, enable TCP or X11 forwarding, and enable a default or custom banner for your bastion hosts.
 - h. Choose **Next**.
5. In the **Configure stack options** page, make any changes or edits as necessary.
 6. Choose **Next**.
 7. Review the details of your bastion host and select the two **Capabilities** acknowledgements. Then, choose **Create stack**.

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 \(p. 13\)](#) 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](#).

In addition to configuring Amazon EC2 instance types, the AWS CloudFormation templates configure the base Amazon Machine Images (AMI) to be used for the instance. The AWS Ground Station base contains the software needed to receive data from the service in your EC2 instance. For more information about AMIs, see [Amazon Machine Images \(AMI\)](#).

Manually Creating and Configuring Resources

The sample AWS CloudFormation templates in this section configure all the resources necessary to begin executing satellite contacts. If you prefer to manually create and configure the resources required to begin executing satellite contacts, you will need to do the following:

- Create AWS Ground Station configs. For more information about manually creating AWS Ground Station configs, see [Create Config AWS CLI Command Reference](#) or [Create Config API Reference](#).
- Create an AWS Ground Station mission profile. For more information about manually creating an AWS Ground Station mission profile, see [Create Mission Profile AWS CLI Command Reference](#) or [Create Mission Profile API Reference](#).
- Create an AWS Ground Station dataflow endpoint group. For more information about manually creating an AWS Ground Station dataflow endpoint group, see [Create Dataflow Endpoint Group AWS CLI Command Reference](#) or [Create Dataflow Endpoint Group API Reference](#).
- Create an EC2 instance. For more information about manually creating an EC2 instance, see [Launch an Instance](#).
- Install and configure Data Defender on your EC2 instance to send and/or receive data during your contact. For more information on manually configuring Data Defender, see [??? \(p. 36\)](#).
- Configure your EC2 instance's security group settings to allow AWS Ground Station to send data to/from your EC2 instance. For more information about manually configuring your EC2 instance's security group settings, see [Create Security Group AWS CLI Command Reference](#) or [Create Security Group API Reference](#).

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” \(p. 31\)](#)
- [the section called “AquaSnppJpssTerraDigIF Template” \(p. 32\)](#)

Important

Satellites must be onboarded to the service in order to access AMIs with the AWS CloudFormation templates.

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-cloudformation-templates-us-west-2/
```

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 \(p. 13\)](#).

Important

The Amazon EC2 instance needs to be stopped before applying the template. Check to ensure that the instance is stopped until you are ready to use it.

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:

```
aws s3 cp s3://groundstation-cloudformation-templates-us-west-2/AquaSnppJpss.yml .
```

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

```
https://s3.console.aws.amazon.com/s3/object/groundstation-cloudformation-templates-us-west-2/AquaSnppJpss.yml
```

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

```
https://groundstation-cloudformation-templates-us-west-2.s3.us-west-2.amazonaws.com/AquaSnppJpss.yml
```

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.
- (Optional) **EC2 Verification for Contacts** - The option to use Lambda to set up a verification system of your Amazon EC2 instance(s) for contacts with SNS notification. It is important to note that this may incur charges depending on your current usage.
- **Dataflow Endpoint Group** - The AWS Ground Station [dataflow endpoint group \(p. 4\)](#) 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 \(p. 7\)](#) defines how the antenna system tracks your satellite as it moves through the sky.
- **Ground Station Amazon Machine Image Retrieval Lambda** - The option to select what software is installed in your instance and the AMI of your choice. The software options include DDX 2.6.2 Only and DDX 2.6.2 with qRadio 3.6.0. These options will continue to expand as additional software updates and features are released.

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 \(p. 13\)](#).

Important

The Amazon EC2 instance needs to be stopped before applying the template. Check to ensure that the instance is stopped until you are ready to use it.

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 <.yaml> with <.json>.

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

```
aws s3 cp s3://groundstation-cloudformation-templates-us-west-2/AquaSnppJpssTerraDigIF.yml .
```

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

```
https://s3.console.aws.amazon.com/s3/object/groundstation-cloudformation-templates-us-west-2/AquaSnppJpssTerraDigIF.yml
```

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

```
https://groundstation-cloudformation-templates-us-west-2.s3.us-west-2.amazonaws.com/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.

- (Optional) **EC2 Verification for Contacts** - The option to use Lambda to set up a verification system of your Amazon EC2 instance(s) for contacts with SNS notification. It is important to note that this may incur charges depending on your current usage.
- **Dataflow Endpoint Group** - The AWS Ground Station [dataflow endpoint group](#) (p. 4) 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. 7) 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.
- **Ground Station Amazon Machine Image Retrieval Lambda** - The option to select what software is installed in your instance and the AMI of your choice. The software options include DDX 2.6.2 Only and DDX 2.6.2 with qRadio 3.6.0. These options will continue to expand as additional software updates and features are released.

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. 30) 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. 30). 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 **CreateEC2VerificationForContacts**, choose whether or not you want to set up a verification system (utilizing Lambda) of your EC2 instance(s) for contacts with SNS notification. It is important to note that this may incur charges depending on your current usage.
- d. For **CreateReceiverInstance**, choose whether or not you want to create an Amazon EC2 receiver instance.
- e. Choose the SSH Key you created in [the section called “Step 1: Create EC2 SSH Key Pair” \(p. 27\)](#).
- 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” \(p. 28\)](#).

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” \(p. 28\)](#).
 - 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**.
 6. After reviewing your stack details, select the **Capabilities** acknowledgement, and choose **Create stack**.

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.

(Optional) Install and Configure Data Defender Manually

The installation and configuration of the Data Defender is automated in the sample AWS CloudFormation templates provided by AWS Ground Station. This step is optional and only applicable for one of the following use cases:

- You want to create your own template that does not reuse the automation built into the sample templates.
- You are building custom automation and infrastructure management processes, for example, creating an AMI for your EC2 instance(s).

To manually install and configure Data Defender instead of using the automatic configuration through the AWS CloudFormation templates, use the following steps.

Topics

- [Step 1: SSH Into Your EC2 Instance \(p. 36\)](#)
- [Step 2: Install Data Defender Dependencies Using YUM \(p. 36\)](#)
- [Step 3: Download the Data Defender Files \(p. 37\)](#)
- [Step 4: Install Data Defender \(p. 37\)](#)
- [Step 5: Configure Data Defender \(p. 38\)](#)
- [Step 6: Configure the Data Defender Streams \(p. 39\)](#)

Step 1: SSH Into Your EC2 Instance

SSH into your Amazon EC2 instance and enable port forwarding through your SSH session. You will need to access port 80 of your Amazon EC2 instance to configure the Data Defender streams using the Data Defender web UI. See [Connect to Your Linux Instance](#) for more information.

Step 2: Install Data Defender Dependencies Using YUM

1. Update the YUM packages on your Amazon EC2 instance by running the following command.

```
sudo yum update -y
```

You will see the following output when the command completes successfully.

```
Updated:
amazon-linux-extras.noarch 0:1.6.12-1.amzn2
ca-certificates.noarch 0:2019.2.32-76.amzn2.0.3
python-devel.x86_64 0:2.7.18-1.amzn2.0.1
python2-rsa.noarch 0:3.4.1-1.amzn2.0.1
amazon-linux-extras-yum-plugin.noarch 0:1.6.12-1.amzn2
python.x86_64 0:2.7.18-1.amzn2.0.1
python-libs.x86_64 0:2.7.18-1.amzn2.0.1
tzdata.noarch 0:2020a-1.amzn2

Complete!
[ec2-user@ip-10-0-40-161 ~]$
```

2. Install the Development Tools package by running the following command.

```
sudo yum groupinstall "Development Tools" -y
```

You will see the following output when the command completes successfully.

```
nettle.x86_64 0:2.7.1-8.amzn2.0.2
perl-Data-Dumper.x86_64 0:2.145-3.amzn2.0.2
perl-Git.noarch 0:2.23.3-1.amzn2.0.1
perl-Test-Harness.noarch 0:3.28-3.amzn2
perl-XML-Parser.x86_64 0:2.41-10.amzn2.0.2
subversion-libs.x86_64 0:1.7.14-11.amzn2.0.2
systemtap-devel.x86_64 0:4.2-1.amzn2.0.1
zlib-devel.x86_64 0:1.2.7-18.amzn2

Complete!
[ec2-user@ip-10-0-40-161 ~]$
```

Step 3: Download the Data Defender Files

1. Create a new directory to store the Data Defender files and download them from Amazon S3 by running the following commands.

```
mkdir "ddx_2.6.2-104"
aws --region us-west-2 s3 cp --recursive "s3://groundstation-customer-assets-us-west-2/
ddx/2.6.2-104" "ddx_2.6.2-104"
```

An example output is provided below.

```
[ec2-user@ip-10-0-40-161 ~]$ mkdir "ddx_2.6.2"
[ec2-user@ip-10-0-40-161 ~]$ aws --region us-west-2 s3 cp --recursive "s3://groundstation-customer-assets-us-west-2/ddx/2.6.2" "ddx_2.6.2"
download: s3://groundstation-customer-assets-us-west-2/ddx/2.6.2/md5-ddx-2.6.2.iso.txt to ddx_2.6.2/md5-ddx-2.6.2.iso.txt
download: s3://groundstation-customer-assets-us-west-2/ddx/2.6.2/DataDefender_2.6.2_Release_Notes.pdf to ddx_2.6.2/DataDefender_2.6.2_Release
_Notes.pdf
download: s3://groundstation-customer-assets-us-west-2/ddx/2.6.2/Kratos_DataDefender_2.6.2_User_Guide.pdf to ddx_2.6.2/Kratos_DataDefender_2.
6.2_User_Guide.pdf
download: s3://groundstation-customer-assets-us-west-2/ddx/2.6.2/ddx-2.6.2.iso to ddx_2.6.2/ddx-2.6.2.iso
[ec2-user@ip-10-0-40-161 ~]$
```

2. Verify that the hashcode from the downloaded ISO matches the expected hashcode by running the following commands.

```
cd "ddx_2.6.2-104"
md5sum "ddx-2.6.2-104.iso" > "downloaded-md5-ddx-2.6.2-104.iso.txt"
diff "md5-ddx-2.6.2-104.iso.txt" "downloaded-md5-ddx-2.6.2-104.iso.txt"
```

The diff command will only produce output to the console if there is a mismatch in the hashcode. There will be no output to the console from the diff command if the ISO's hashcode matches the expected hashcode. An example output is provided below.

```
[ec2-user@ip-10-0-40-161 ~]$ cd "ddx_2.6.2"
[ec2-user@ip-10-0-40-161 ddx_2.6.2]$ md5sum "ddx-2.6.2.iso" > "downloaded-md5-ddx-2.6.2.iso.txt"
[ec2-user@ip-10-0-40-161 ddx_2.6.2]$ diff "md5-ddx-2.6.2.iso.txt" "downloaded-md5-ddx-2.6.2.iso.txt"
[ec2-user@ip-10-0-40-161 ddx_2.6.2]$
```

Step 4: Install Data Defender

1. Mount the ISO and navigate to the mounted directory by performing the following commands.

```
mkdir "ddx_iso_mount"
sudo mount -o loop "ddx-2.6.2-104.iso" "ddx_iso_mount"
cd "ddx_iso_mount"
```

An example output is provided below.

```
[ec2-user@ip-10-0-40-161 ddx_2.6.2]$ mkdir "ddx_iso_mount"
[ec2-user@ip-10-0-40-161 ddx_2.6.2]$ sudo mount -o loop "ddx-2.6.2.iso" "ddx_iso_mount"
mount: /home/ec2-user/ddx_2.6.2/ddx_iso_mount: WARNING: device write-protected, mounted read-only.
[ec2-user@ip-10-0-40-161 ddx_2.6.2]$ cd "ddx_iso_mount"
```

2. Run the Data Defender installer by performing the following command.

```
sudo ./install.sh
```

The installer will begin installing Data Defender and will pause to ask if you'd like to use the default values. When prompted, enter "n" to provide non-default values to the installer. An example output is provided below.

```
[ec2-user@ip-10-0-40-161 ddx_iso_mount]$ sudo ./install.sh
Backing up existing configuration files for ddx
Installing: ddx-2.6.2-104.rpm
Preparing... ##### [100%]
Running pre-install script
Pre-install complete
Updating / installing...
 1:ddx-2.6.2-104 ##### [100%]
Running post-install script
Post-install complete
Installing from: /opt/rtlogic/ddx-2.6.2-104
Use default values? (Y/N): n
```

3. Enter the following values when prompted.

```
Enable SSL? (Y/N): y
Specify HTTPS port: 443
Enable forwarding from port 80 to HTTPS? (Y/N): y
Use password file? (OS is used otherwise) (Y/N): n
Single Instance Port (0 to disable): 44892
Enable LAN intelligent merge? (Y/N): n
```

The following message will be displayed when the Data Defender installation has completed

```
Installation is complete
Logs can be found in: /var/log/rtlogic/ddx
To manually start the application: systemctl start rtlogic-ddx
To manually stop the application: systemctl stop rtlogic-ddx
To change to a different version: /opt/rtlogic/ddx/setup
To view the GUI, browse to: http(s)://localhost:<port>(<secure port>)
[ec2-user@ip-10-0-40-161 ddx_iso_mount]$
```

4. Unmount and remove the ISO directory. Then, reboot the EC2 instance to complete the installation. Use the following commands.

```
cd ..
sudo umount "ddx_iso_mount"
rm -rf "ddx_iso_mount"
sudo reboot
```

Step 5: Configure Data Defender

1. Open the Data Defender config file using a text editor such as VIM. An example command is provided below.

```
sudo vim /opt/rtlogic/ddx-2.6.2-104/bin/ddx.xml
```

2. Make the following changes to the Data Defender config file and then save the file.
 - a. Add `<dtlsOverWan>true</dtlsOverWan>` to the `<dataDefender>` section.
 - b. Replace `<plugin>node_locked_licensing_plugin</plugin>` with `<plugin>not_locked_licensing_plugin</plugin>` in the `<plugins>` section.

```
</if>
<plugin>sorted_hashed_mac_system_id_plugin</plugin>
<plugin>file_based_system_id_plugin</plugin>
<plugins>not_locked_licensing_plugin</plugins>
<plugin>tcmmalloc_plugin</plugin>
<plugin>dtls_plugin</plugin>

<!-- Plugins below this point start threads so they need to be last.
      If that ever changes we can remove this plugins section -->
<if condition="agent!=true">
  <plugin>collector_manager_plugin</plugin>
</if>
<plugin>exporter_manager_plugin</plugin>
<plugin>synchronic_plugin</plugin>
</plugins>

<!-- #include(logging.xml) -->

<modules>

  <synchronicModule>
    <sharedLibrary>synchronic</sharedLibrary>
    <logLevel>information</logLevel>
    <internal>false</internal>
    <active>true</active>
  </synchronicModule>

  <dataDefender>
    <!-- <logLevel>information</logLevel> -->
    <sharedLibrary>stream_manager</sharedLibrary>

    <if condition="developerMode==true">
      <developerMode>true</developerMode>
    </if>
    <dependencies internal="true">
      <dependency>synchronicModule</dependency>
    </dependencies>
    <!-- #include(internalAttributes.xml) -->
    <!-- #include(defaultValues.xml) -->
    <!-- #include(exporterManagerConfig.xml) -->
    <!-- #include(updateStatusInterval.xml) -->
    <dtlsOverWan>true</dtlsOverWan>
  </dataDefender>
```

168, 0-1

54%

3. Open the Data Defender security file using a text editor such as VIM. An example command is provided below.

```
sudo vim /opt/rtlogic/ddx-2.6.2-104/bin/security.xml
```

4. Make the following change to the Data Defender config file. Then, save the file.
 - Change `<securityRequireSSL>true</securityRequireSSL>` to `<securityRequireSSL>false</securityRequireSSL>` in the security policy settings.

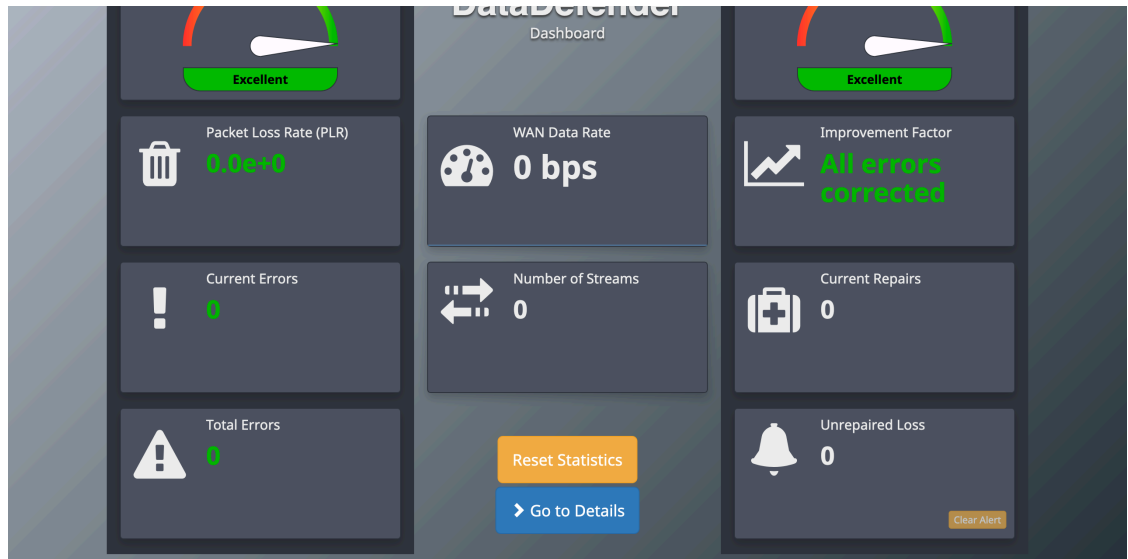
```
<!-- Security policy settings -->
<securityRequireSSL>false</securityRequireSSL>
<securityRequireAuthentication>false</securityRequireAuthentication>
```

5. Reboot the EC2 instance to apply the settings to Data Defender by using the following command.

```
sudo reboot
```

Step 6: Configure the Data Defender Streams

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**. As a reminder, you must forward port 80 of your Amazon EC2 instance through your SSH session to your localhost port 8080 to access the Data Defender web UI.
2. On the *DataDefender* dashboard, choose **Go to Details**.



3. In the **Streams** section, choose **+ Add Stream**.

Streams								
Name	Direction	WAN Protocol	WAN Rate	WAN Overhead	WAN RTT	WAN Packets Errored	WAN Packets Repaired	WAN Packets Lost
No streams configured								
<div><div>+ Add Stream</div><div><div>✓ Edit Stream</div><div>▶ Start Test Source</div><div>📊 View Analytics</div><div>🔄 Reset Statistics</div><div>🗑 Delete Stream</div></div></div>								

4. In the **WAN Transport** page of the **Stream Wizard**, enter and choose the following details. Then, choose **Next**.
- For **Stream Name**, enter **Downlink**.
 - For **Stream Direction**, choose **WAN to LAN**.
 - For **Network Interface**, choose **eth1**.
 - For **Port**, enter **55888**.

The screenshot shows the 'Stream Wizard' window with the 'WAN Transport' step selected. The window has a dark theme and a close button in the top right. At the top, there are three steps: 'WAN Transport' (selected, with a cloud-to-server icon), 'Local Endpoint' (with a server icon), and 'Finish' (with a flag icon). Below the steps, the title 'Configure DataDefender to communicate across the WAN' is displayed. The form contains the following fields: 'Stream Name' with the value 'Downlink', 'Stream Direction' with a dropdown menu set to 'WAN to LAN', and a section titled 'WAN Transport 1' containing 'Network Interface' with a dropdown menu set to 'eth1', 'Enable Multicast' with an unchecked checkbox, and 'Port' with the value '55888'. Each field has a help icon (question mark) to its right. At the bottom left is a '+ Add' button, and at the bottom right are 'Next' and 'Cancel' buttons.

Stream Wizard

WAN Transport Local Endpoint Finish

Configure DataDefender to communicate across the WAN

Stream Name Downlink ?

Stream Direction WAN to LAN ?

WAN Transport 1

Network Interface eth1 ?

Enable Multicast ☐ ?

Port 55888 ?

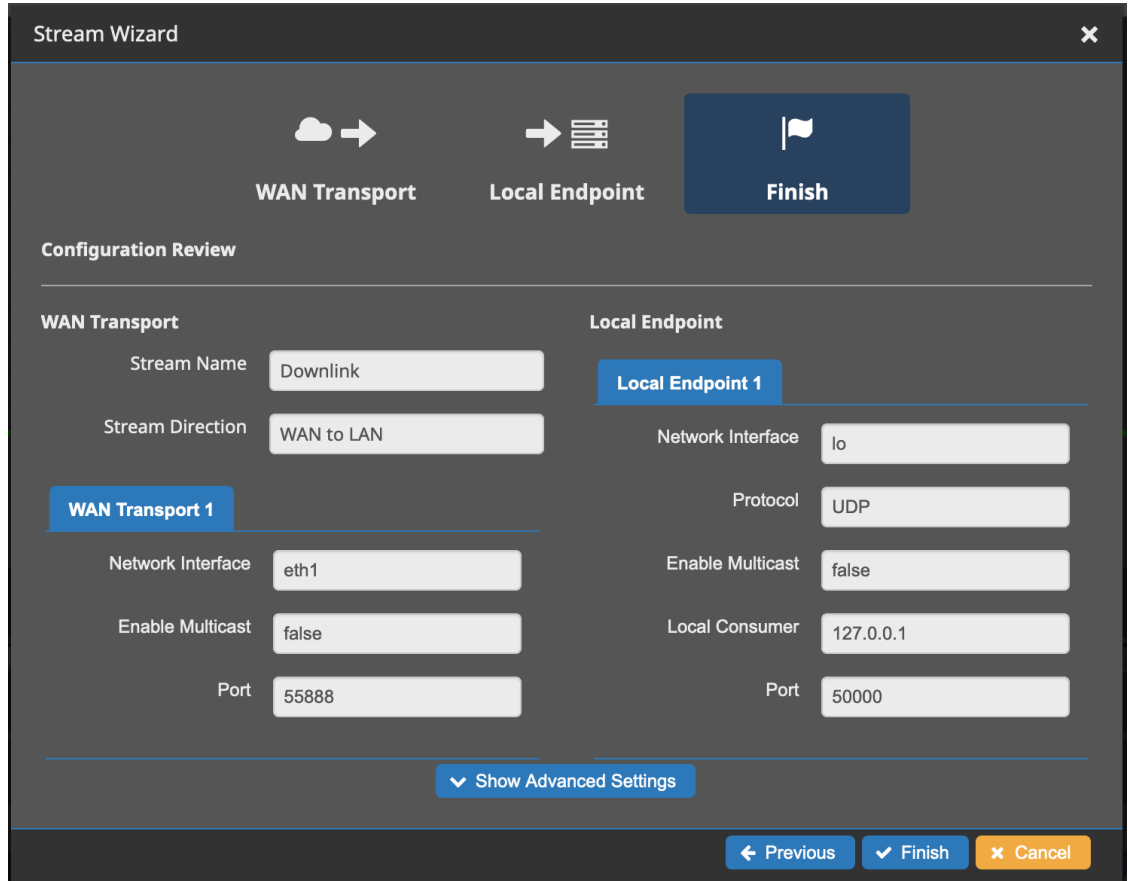
+ Add

Next Cancel

5. In the **Local Endpoint** page of the **Stream Wizard**, enter and choose the following stream details. Then, choose **Next**.
 - a. For **Network Interface**, choose **lo**.
 - b. For **Local Consumer**, enter **127.0.0.1**.
 - c. For **Port**, enter **50000**.

The screenshot shows the 'Stream Wizard' interface with three steps: 'WAN Transport', 'Local Endpoint' (selected), and 'Finish'. The 'Local Endpoint' step is titled 'Configure DataDefender to communicate with a local endpoint'. It features a tab labeled 'Local Endpoint 1' and several configuration fields: 'Network Interface' (dropdown menu showing 'lo'), 'Protocol' (dropdown menu showing 'UDP'), 'Enable Multicast' (checkbox), 'Local Consumer' (text input showing '127.0.0.1'), and 'Port' (text input showing '50000'). Each field has a help icon. At the bottom left is a '+ Add' button, and at the bottom right are 'Previous', 'Next', and 'Cancel' navigation buttons.

6. In the **Finish** page of the **Stream Wizard**, choose **Finish** to save the stream configuration.



The Stream Wizard Configuration Review screen shows the final settings for a stream. At the top, there are three icons: a cloud with an arrow (WAN Transport), a server with an arrow (Local Endpoint), and a flag (Finish). Below these are the configuration details for both WAN Transport and Local Endpoint. WAN Transport 1 is configured with Stream Name 'Downlink', Stream Direction 'WAN to LAN', Network Interface 'eth1', Enable Multicast 'false', and Port '55888'. Local Endpoint 1 is configured with Network Interface 'lo', Protocol 'UDP', Enable Multicast 'false', Local Consumer '127.0.0.1', and Port '50000'. A 'Show Advanced Settings' button is located below the configuration details. At the bottom, there are 'Previous', 'Finish', and 'Cancel' buttons.

Stream Wizard

WAN Transport **Local Endpoint** **Finish**

Configuration Review

WAN Transport

Stream Name: Downlink

Stream Direction: WAN to LAN

WAN Transport 1

Network Interface: eth1

Enable Multicast: false

Port: 55888

Local Endpoint

Local Endpoint 1

Network Interface: lo

Protocol: UDP

Enable Multicast: false

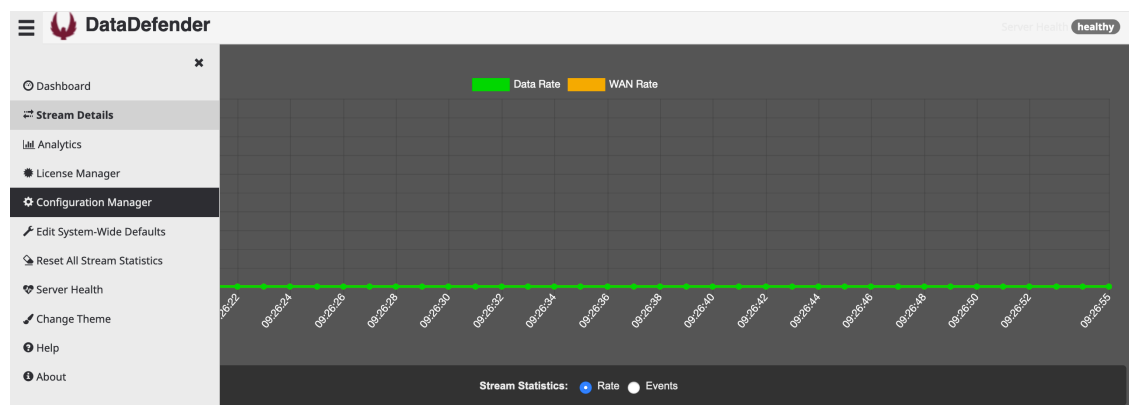
Local Consumer: 127.0.0.1

Port: 50000

[Show Advanced Settings](#)

[Previous](#) [Finish](#) [Cancel](#)

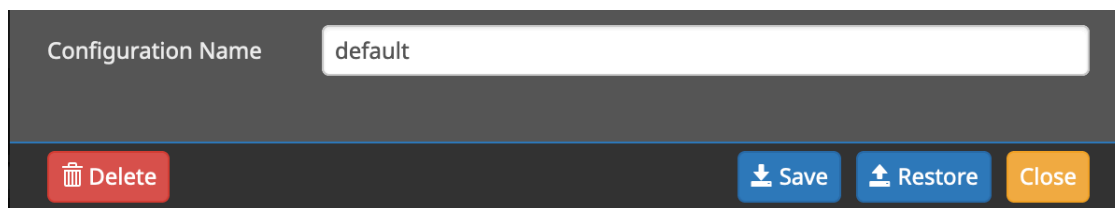
7. Open the **Data Defender** menu by choosing the menu icon in the top left corner of the **Stream Details** page. Then, choose **Configuration Manager**.



Note

The previous steps describe how to configure a single downlink stream. Depending on your use case, you may need to configure multiple downlink streams. You can repeat the previous steps in order to create additional downlink streams if needed. Creating an uplink stream is out of the scope of this guide.

8. For **Configuration Name**, enter **default**. Then, choose **Save**. The saved configuration will persist across Amazon EC2 instance restarts.



The screenshot shows a configuration interface with a dark background. At the top, there is a label "Configuration Name" followed by a text input field containing the word "default". Below this, there is a horizontal bar with four buttons: a red "Delete" button with a trash icon, a blue "Save" button with a download icon, a blue "Restore" button with an upload icon, and an orange "Close" button.

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 \(p. 19\)](#)
- [Monitoring AWS Ground Station \(p. 47\)](#)

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 is currently available in all AWS Ground Station supported regions when receiving your contact data in an Amazon S3 Bucket. It is only available in the following antenna-to-destination regions when utilizing data delivery to Amazon EC2:

- 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 [??? \(p. 29\)](#).

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

Topics

- [To use cross-region data delivery in the console \(p. 45\)](#)
- [To use cross-region data delivery with AWS CLI \(p. 46\)](#)

To use cross-region data delivery in the console

When you [reserve a contact \(p. 19\)](#) 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 \(p. 21\)](#) 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.

Contact management (1)

Cancel contact

Reserve contact

Manage contacts using the table below.

Ground station

Satellite catalog number

Status

Ohio 1

27424

Scheduled

Mission profile

37849 SNPP And 43013 JPSS

Start date and time (UTC +00:00)

End date and time (UTC +00:00)

2020/06/09

13:48

2020/06/14

13:48

<

1

>

	Catalog number	Ground station	Start time (AOS)	End time (LOS)	Maximum elevation (deg.)	Region	Status
<input type="radio"/>	27424	Ohio 1	2020-06-09T17:04:37.000Z	2020-06-09T17:08:54.000Z	11.22	us-west-2	SCHEDULED

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.

```
{
  "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
      },
      "missionProfileArn": "arn:aws:groundstation:us-west-2:123456789012:mission-profile/11111111-2222-3333-4444-555555555555",
      "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 \(p. 47\)](#)
- [Logging AWS Ground Station API Calls with AWS CloudTrail \(p. 49\)](#)
- [Metrics with Amazon CloudWatch \(p. 51\)](#)

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": [
    "arn:aws:groundstation:us-west-2:123456789012:contact/11111111-1111-1111-1111-111111111111"
  ],
  "detailType": "Ground Station Contact State Change",
  "detail": {
    "contactId": "11111111-1111-1111-1111-111111111111",
    "groundstationId": "Ground Station 1",
    "missionProfileArn": "arn:aws:groundstation:us-west-2:123456789012:mission-profile/11111111-1111-1111-1111-111111111111",
    "satelliteArn": "arn:aws:groundstation:us-west-2:123456789012:satellite/11111111-1111-1111-1111-111111111111",
    "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:us-west-2:123456789012:dataflow-endpoint-  
group/bad957a8-1d60-4c45-a92a-39febd98921d, arn:aws:groundstation:us-  
west-2:123456789012:contact/98ddd10f-f2bc-479c-bf7d-55644737fb09, arn:aws:groundstation:us-  
west-2:123456789012:mission-profile/c513c84c-eb40-4473-88a2-d482648c9234"  
  ],  
  "detailType": "Ground Station Dataflow Endpoint Group State Change",  
  "detail": {  
    "dataflowEndpointGroupId": "bad957a8-1d60-4c45-a92a-39febd98921d",  
    "groundstationId": "Ground Station 1",  
    "contactId": "98ddd10f-f2bc-479c-bf7d-55644737fb09",  
    "dataflowEndpointGroupArn": "arn:aws:groundstation:us-west-2:680367718957:dataflow-  
endpoint-group/bad957a8-1d60-4c45-a92a-39febd98921d",  
    "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.

Ground Station Ephemeris State Change

If you want to perform an action when an ephemeris changes state, you can set up a CloudWatch rule to automate this action. This allows you to perform different actions in response to an ephemeris changing state. For example, you can perform an action when an ephemeris has completed validation, and it is now ENABLED. Notification for this event will be sent to the region where the ephemeris was uploaded.

An example is provided below.

```
{  
  "id": "7bf73129-1428-4cd3-a780-95db273d1602",  
  "detail-type": "Ground Station Ephemeris State Change",  
  "source": "aws.groundstation",  
  "account": "123456789012",  
  "time": "2019-12-03T21:29:54Z",  
  "region": "us-west-2",  
  "resources": [  
    "arn:aws:groundstation::123456789012:satellite/10313191-c9d9-4ecb-a5f2-bc55cab050ec",  
    "arn:aws:groundstation::123456789012:ephemeris/111111-cccc-bbbb-a555-bcccca005000",  
  ],  
  "detail": {  
    "ephemerisStatus": "ENABLED",  
    "ephemerisId": "111111-cccc-bbbb-a555-bcccca005000",  
    "satelliteId": "10313191-c9d9-4ecb-a5f2-bc55cab050ec"  
  }  
}
```

Possible states for the `ephemerisStatus` include ENABLED, VALIDATING, INVALID, ERROR, DISABLED, EXPIRED

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](#)
- [Receiving CloudTrail Log Files from Multiple Regions](#) and [Receiving CloudTrail Log Files from Multiple Accounts](#)

All AWS Ground Station actions are logged by CloudTrail and are documented in the [AWS Ground Station API Reference](#). For example, calls to the `ReserveContact`, `CancelContact` and `ListConfigs` actions generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the [CloudTrail userIdentity Element](#).

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

```
{
  "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"
      }
    }
  },
  "eventTime": "2019-05-15T21:14:37Z",
  "eventSource": "groundstation.amazonaws.com",
  "eventName": "ReserveContact",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "127.0.0.1",
  "userAgent": "Coral/Jakarta",
  "requestParameters": {
    "satelliteArn":
      "arn:aws:groundstation::123456789012:satellite/11111111-2222-3333-4444-555555555555",
    "groundStation": "Ohio 1",
    "startTime": 1558356107,
    "missionProfileArn": "arn:aws:groundstation:us-east-2:123456789012:mission-profile/11111111-2222-3333-4444-555555555555",
    "endTime": 1558356886
  },
  "responseElements": {
    "contactId": "11111111-2222-3333-4444-555555555555"
  },
  "requestID": "11111111-2222-3333-4444-555555555555",
  "eventID": "11111111-2222-3333-4444-555555555555",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "11111111-2222-3333-4444-555555555555"
}
```

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.

Metric	Description
Es/N0	The signal to noise ratio. Units: dBm (decibels relative to milliwatts)
BitErrorRate	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
BlockErrorRate	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
ReceivedPower	The measured signal strength in the demodulator/decoder. Units: dBm (decibels relative to milliwatts)

What dimensions are used for AWS Ground Station?

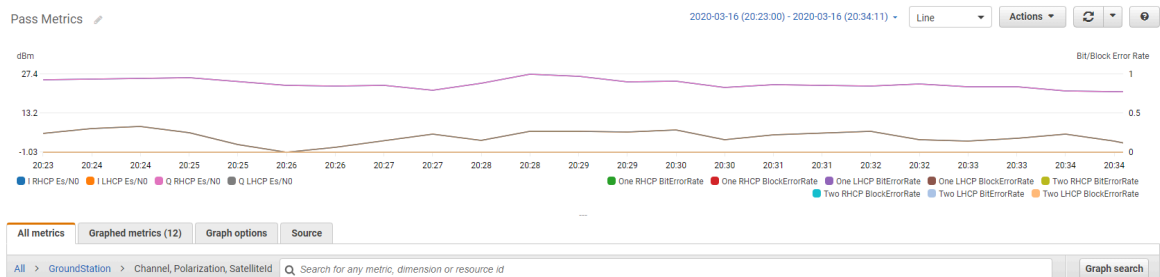
You can filter AWS Ground Station data using the following dimensions.

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.

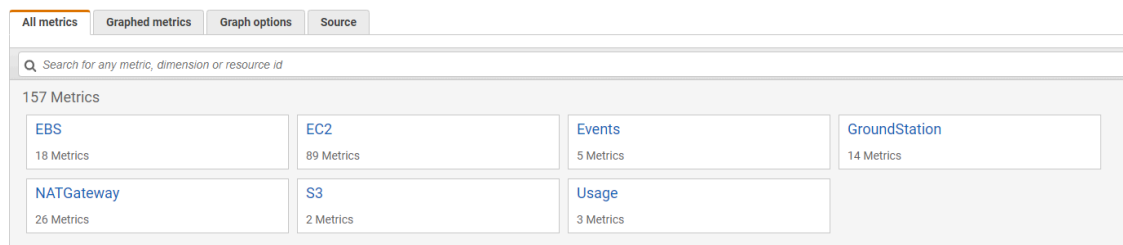
In addition, any data captured within the first 60 seconds will not contain enough information to produce meaningful metrics, and will likely not be displayed. In order to view meaningful metrics, it is recommended to view your data after 60 seconds has passed.



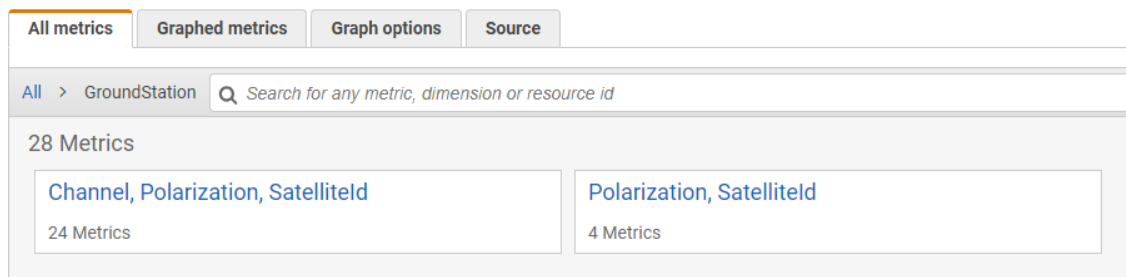
For more information about graphing AWS Ground Station metrics in CloudWatch, see [Graphing Metrics](#).

To view metrics using the console

1. Open the CloudWatch console at <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, choose **Metrics**.
3. Select the **GroundStation** namespace.



4. Select your desired metric dimensions (for example, **Channel, Polarization, Satelliteld**).



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 SatelliteID 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.

```
aws cloudwatch get-metrics-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.

```
{  
  "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 Contacts that Deliver Data to Amazon EC2 \(p. 55\)](#)
- [Ground Station Contact Statuses \(p. 58\)](#)

Troubleshooting Contacts that Deliver Data to Amazon EC2

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

```
ssh -L 8080:localhost:80 -o ProxyCommand="C:\Windows\System32\OpenSSH\ssh.exe -o  
\"ForwardAgent yes\" -W %h:%p -i \"<SSH KEY>\" ec2-user@<BASTION HOST>" -i "<SSH KEY>"  
ec2-user@<HOST>
```

For Mac

```
ssh -L 8080:localhost:80 -o ProxyCommand="ssh -A -o 'ForwardAgent yes' -W %h:%p -i <SSH  
KEY> ec2-user@<BASTION HOST>" -i <SSH KEY> ec2-user@<HOST>
```

2. Verify that Data Defender (also called DDX) is running by grepping (checking) for a running process named ddx in the output. The command for grepping (checking) for a running process and a successful example output is provided below.

```
[ec2-user@Receiver-Instance ~]$ ps -ef | grep ddx  
Rtlogic  4977      1 10 Oct16 ?        2-00:22:14 /opt/rtlogic/ddx/bin/ddx -m/  
opt/rtlogic/ddx/modules -p/opt/rtlogic/ddx/plugins -c/opt/rtlogic/ddx/bin/ddx.xml -  
umask=077 -daemon -f installed=true -f security=true -f enable HttpsForwarding=true  
Ec2-user 18787 18657  0 16:51 pts/0      00:00:00 grep -color=auto ddx
```

If Data Defender is running, skip to [the section called “Step 3: Verify that Your Data Defender Stream is Configured” \(p. 57\)](#) Otherwise, continue to the next step.

3. Start Data Defender using the command show below.

```
sudo service rtlogic-ddx start
```

If Data Defender is running after using the command, skip to [the section called “Step 3: Verify that Your Data Defender Stream is Configured” \(p. 57\)](#) Otherwise, continue to the next step.

4. Inspect the following files using the commands below to see if there were any errors while installing and configuring Data Defender.

```
cat /var/log/user-data.log  
cat /opt/aws/groundstation/.startup.out
```

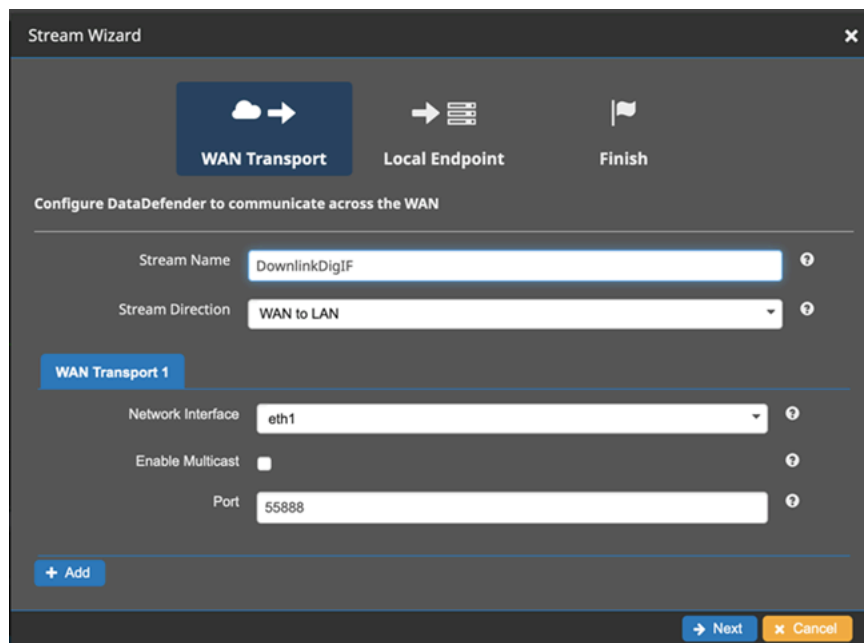
Note

A common issue discovered when inspecting these files is that the Amazon VPC that your Amazon EC2 instance is running in does not have access to Amazon S3 to download the installation files. If you discover in your logs that this is the issue, check your EC2 instance's Amazon VPC and security group settings to ensure they are not blocking access to Amazon S3.

If Data Defender is running after checking your Amazon VPC settings, continue to [the section called “Step 3: Verify that Your Data Defender Stream is Configured” \(p. 57\)](#). If the problem persists, [contact AWS Support](#) and send your log files with a description of your issue.

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**.



The screenshot shows the 'Stream Wizard' dialog box with the 'WAN Transport' pane selected. The dialog has three tabs: 'WAN Transport' (active), 'Local Endpoint', and 'Finish'. Below the tabs, the text 'Configure DataDefender to communicate across the WAN' is displayed. The 'Stream Name' field contains 'DownlinkDigIF'. The 'Stream Direction' dropdown is set to 'WAN to LAN'. Under the 'WAN Transport 1' section, the 'Network Interface' dropdown is set to 'eth1', the 'Enable Multicast' checkbox is unchecked, and the 'Port' field contains '55888'. At the bottom left is a '+ Add' button, and at the bottom right are 'Next' and 'Cancel' buttons.

- 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 on which you will receive your data after Data Defender has received it from the AWS Ground Station service. Then, choose **Next**.

The screenshot shows the 'Stream Wizard' window with the 'Local Endpoint' step selected. The title bar says 'Stream Wizard' with a close button. At the top, there are three buttons: 'WAN Transport', 'Local Endpoint' (which is highlighted with a dark blue background and a white icon of a server rack), and 'Finish'. Below these buttons, the text reads 'Configure DataDefender to communicate with a local endpoint'. Underneath, there is a section titled 'Local Endpoint 1'. It contains five configuration fields: 'Network Interface' with a dropdown menu showing 'lo', 'Protocol' with a dropdown menu showing 'UDP', 'Enable Multicast' with an unchecked checkbox, 'Local Consumer' with a text input field containing '127.0.0.1', and 'Port' with a text input field containing '50000'. Each field has a help icon (a question mark in a circle) to its right. At the bottom left of the configuration area is a '+ Add' button. At the bottom right are three navigation buttons: 'Previous' (with a left arrow), 'Next' (with a right arrow), and 'Cancel' (with an 'X' icon).

- 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](#).

Ground Station Contact Statuses

The status of an AWS Ground Station contact provides insight into what is happening to that contact at a given time.

Contact Statuses

The following is the list of statuses that a contact can have:

- **AVAILABLE** - The contact is available to be reserved.
- **SCHEDULING** - The contact is in the process of scheduling.
- **SCHEDULED** - The contact was successfully scheduled.
- **FAILED_TO_SCHEDULE** - The contact failed to schedule.
- **PREPASS** - The contact is starting soon and resources are being prepared.
- **PASS** - The contact is currently executing and the satellite is being communicated with.
- **POSTPASS** - The communication has completed and resources used are being cleaned up.
- **COMPLETED** - The contact completed successfully.
- **FAILED** - The contact failed because of an issue with the customer resource configuration.
- **AWS_FAILED** - The contact failed because of a problem in the AWS Ground Station service.
- **CANCELLING** - The contact is in the process of being cancelled.

- **AWS_CANCELLED** - The contact was cancelled by the AWS Ground Station service. Antenna or site maintenance is one example of when this could happen.
- **CANCELLED** - The contact was cancelled by the customer.

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 \(p. 60\)](#)

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 \(p. 60\)](#)
- [Authentication \(p. 61\)](#)
- [Controlling Access Using Policies \(p. 62\)](#)
- [Learn More \(p. 63\)](#)
- [How AWS Ground Station Works with IAM \(p. 64\)](#)
- [AWS Ground Station Identity-Based Policy Examples \(p. 67\)](#)
- [Troubleshooting AWS Ground Station Authentication and Access Control \(p. 70\)](#)

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 \(p. 70\)](#).

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 \(p. 64\)](#).

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 \(p. 67\)](#).

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 create an AWS account, you begin with one 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. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see [Tasks that require root user credentials](#) in the *AWS General Reference*.

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** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see [Creating a role for a third-party Identity Provider](#) in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see [Permission sets](#) in the *AWS IAM Identity Center (successor to AWS Single Sign-On) 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 on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. 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*.

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*.

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*.
- **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*.
- **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*.

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*.

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 \(p. 64\)](#)
- [Troubleshooting AWS Ground Station Authentication and Access Control \(p. 70\)](#)

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. 64)
- [AWS Ground Station Resource-Based Policies](#) (p. 66)
- [Authorization Based on AWS Ground Station Tags](#) (p. 66)
- [AWS Ground Station IAM Roles](#) (p. 67)

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:

Action	Description
<code>CancelContact</code>	Grants permission to cancel a contact
<code>DescribeContact</code>	Grants permission to describe a contact
<code>ListContacts</code>	Grants permission to return a list of contacts
<code>ReserveContact</code>	Grants permission to reserve a contact

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:

```
"Resource": "arn:aws:groundstation:us-east-2:123456789012:config/antenna-downlink-demod-decode/11111111-2222-3333-4444-555555555555"
```

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

```
"Resource": "arn:aws:groundstation:us-east-2:123456789012:config/*"
```

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": [
    "arn:aws:groundstation:us-west-2:123456789012:config/satellite/11111111-2222-3333-4444-555555555555",
    "arn:aws:groundstation:us-west-2:123456789012:config/satellite/21111111-2222-3333-4444-555555555555"
```

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

Action	Description
CreateConfig	Grants permission to create a Config
CreateDataflowEndpointGroup	Grants permission to create a dataflow endpoint group
CreateMissionProfile	Grants permission to create a mission profile

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:

Action	Description
UpdateConfig	Grants permission to update a Config

Action	Description
UpdateMissionProfile	Grants permission to update a mission profile

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 \(p. 67\)](#).

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 Configs Based on Tags \(p. 69\)](#).

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:

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

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

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

```



```
        "Resource": [
            "*"
        ]
    }
}
```

For more information about writing IAM policies, see [IAM Policies](#) in the *IAM User Guide*.

Topics

- [Policy Best Practices](#) (p. 68)
- [Using the AWS Ground Station Console](#) (p. 68)
- [Allow Users to View Their Own Permissions](#) (p. 69)
- [Viewing AWS Ground Station ConfigIds Based on Tags](#) (p. 69)

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*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
    "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.

```
{
  "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 Configs 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.

```
{
  "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 \(p. 70\)](#)
- [I am not authorized to perform `iam:PassRole` \(p. 71\)](#)
- [I want to view my access keys \(p. 71\)](#)
- [I'm an administrator and want to allow others to access AWS Ground Station \(p. 71\)](#)
- [I want to allow people outside of my AWS account to access my AWS Ground Station resources \(p. 71\)](#)

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 \(p. 64\)](#).
- 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*.

Data Encryption at rest for AWS Ground Station

AWS Ground Station provides encryption by default to protect sensitive customer data at rest using AWS owned encryption keys.

- *AWS owned keys* - AWS Ground Station uses these keys by default to automatically encrypt personal, directly identifiable data and ephemerides. You cannot view, manage, or use AWS-owned keys, or audit their use; however, it is unnecessary to take any action or change programs to protect the keys that encrypt data. For more information, see [AWS-owned keys](#) in the [AWS Key Management Service Developer Guide](#).

Encryption of data at rest by default helps by reducing the operational overhead and complexity involved in protecting sensitive data. At the same time, it enables building secure applications that meet strict encryption compliance, as well as regulatory requirements.

AWS Ground Station enforces encryption on all sensitive, at-rest, data, however, for some AWS Ground Station resource, such as ephemerides, you can choose to use a customer managed key in place of the default AWS managed keys.

- *Customer managed keys* -- AWS Ground Station supports the use of a symmetric customer managed key that you create, own, and manage to add a second layer of encryption over the existing AWS owned encryption. Because you have full control of this layer of encryption, you can perform such tasks as:
 - Establishing and maintaining key policies
 - Establishing and maintaining IAM policies and grants
 - Enabling and disabling key policies
 - Rotating key cryptographic material
 - Adding tags
 - Creating key aliases
 - Scheduling keys for deletion

For more information, see [customer managed key](#) in the [AWS Key Management Service Developer Guide](#).

The following table summarizes resources for which AWS Ground Station supports the use of Customer Managed Keys

Data type	AWS owned key encryption	Customer managed key encryption (Optional)
Ephemeris data used to compute the trajectory of a Satellite	Enabled	Enabled

Note

AWS Ground Station automatically enables encryption at rest using AWS owned keys to protect personally identifiable data at no charge. However, AWS KMS charges apply for using

a customer managed key. For more information about pricing, see the [AWS Key Management Service pricing](#).

For more information on AWS KMS, see the [AWS KMS Developer Guide](#).

How AWS Ground Station uses grants in AWS KMS

AWS Ground Station requires a [key grant](#) to use your customer-managed key.

When you upload an ephemeris encrypted with a customer managed key, AWS Ground Station creates a key grant on your behalf by sending a CreateGrant request to AWS KMS. Grants in AWS KMS are used to give AWS Ground Station access to a KMS key in a customer account.

AWS Ground Station requires the grant to use your customer managed key for the following internal operations:

- Send GenerateDataKey requests to AWS KMS to generate data keys encrypted by your customer managed key.
- Send Decrypt requests to AWS KMS to decrypt the encrypted data keys so that they can be used to encrypt your data.
- Send Encrypt requests to AWS KMS to encrypt the provided data.

You can revoke access to the grant, or remove the service's access to the customer managed key at any time. If you do, AWS Ground Station won't be able to access any of the data encrypted by the customer managed key, which affects operations that are dependent on that data. For example, if you remove a key grant from an ephemeris currently in use for a contact then AWS Ground Station will be unable to use the provided ephemeris data for pointing the antenna during the contact. This will cause the contact to end in a FAILED state.

Create a customer managed key

You can create a symmetric customer managed key by using the AWS Management Console, or the AWS KMS APIs.

To create a symmetric customer managed key

Follow the steps for Creating symmetric customer managed key in the AWS Key Management Service Developer Guide.

Key policy

Key policies control access to your customer managed key. Every customer managed key must have exactly one key policy, which contains statements that determine who can use the key and how they can use it. When you create your customer managed key, you can specify a key policy. For more information, see [Managing access to customer managed keys](#) in the AWS Key Management Service Developer Guide.

To use your customer managed key with your AWS Ground Station resources, the following API operations must be permitted in the key policy:

[kms:CreateGrant](#) - Adds a grant to a customer managed key. Grants control access to a specified KMS key, which allows access to [grant operations](#) AWS Ground Station requires. For more information about [Using Grants](#), see the AWS Key Management Service Developer Guide.

This allows Amazon AWS to do the following:

- Call `GenerateDataKey` to generate an encrypted data key and store it, because the data key isn't immediately used to encrypt.
- Call `Decrypt` to use the stored encrypted data key to access encrypted data.
- Call `Encrypt` to use the data key to encrypt data.
- Set up a retiring principal to allow the service to `RetireGrant`.

[kms:DescribeKey](#) - Provides the customer managed key details to allow AWS Ground Station to validate the key before attempting to create a grant on the provided key.

The following are IAM policy statement examples you can add for AWS Ground Station

```
"Statement" : [  
  {"Sid" : "Allow access to principals authorized to use AWS Ground Station",  
   "Effect" : "Allow",  
   "Principal" : {  
     "AWS" : "*"  
   },  
   "Action" : [  
     "kms:DescribeKey",  
     "kms:CreateGrant"  
   ],  
   "Resource" : "*",  
   "Condition" : {  
     "StringEquals" : {  
       "kms:ViaService" : "groundstation.amazonaws.com",  
       "kms:CallerAccount" : "111122223333"  
     }  
   }  
 },  
  {"Sid": "Allow access for key administrators",  
   "Effect": "Allow",  
   "Principal": {  
     "AWS": "arn:aws:iam::111122223333:root"  
   },  
   "Action" : [  
     "kms:*"  
   ],  
   "Resource": "arn:aws:kms:region:111122223333:key/key_ID"  
 },  
  {"Sid" : "Allow read-only access to key metadata to the account",  
   "Effect" : "Allow",  
   "Principal" : {  
     "AWS" : "arn:aws:iam::111122223333:root"  
   },  
   "Action" : [  
     "kms:Describe*",  
     "kms:Get*",  
     "kms:List*",  
     "kms:RevokeGrant"  
   ],  
   "Resource" : "*"   
 }  
 ]
```

For more information about [specifying permissions in a policy](#) , see the AWS Key Management Service Developer Guide.

For more information about [troubleshooting key access](#) , see the AWS Key Management Service Developer Guide.

Specifying a customer managed key for AWS Ground Station

You can specify a customer managed key to encrypt the following resources:

- Ephemeris

When you create a resource, you can specify the data key by providing a *kmsKeyArn*

- *kmsKeyArn* - A [key identifier](#) for an AWS KMS customer managed key

AWS Ground Station encryption context

An [encryption context](#) is an optional set of key-value pairs that contain additional contextual information about the data. AWS KMS uses the encryption context as additional authenticated data to support authenticated encryption. When you include an encryption context in a request to encrypt data, AWS KMS binds the encryption context to the encrypted data. To decrypt data, you include the same encryption context in the request.

AWS Ground Station encryption context

AWS Ground Station uses the different encryption context depending on the resource being encrypted and specifies a specific encryption context for each key grant created.

Ephemeris Encryption Context:

Key grant for encrypting ephemeris resources are bound to a specific satellite ARN

```
"encryptionContext": {  
  "aws:groundstation:arn":  
    "arn:aws:groundstation::111122223333:satellite/00a770b0-082d-45a4-80ed-SAMPLE"  
}
```

Note

Key grants are re-used for the same key-satellite pair.

Using encryption context for monitoring

When you use a symmetric customer managed key to encrypt your ephemerides, you can also use the encryption context in audit records and logs to identify how the customer managed key is being used. The encryption context also appears in [logs generated by AWS CloudTrail or Amazon CloudWatch Logs](#).

Using encryption context to control access to your customer managed key

You can use the encryption context in key policies and IAM policies as conditions to control access to your symmetric customer managed key. You can also use encryption context constraints in a grant.

AWS Ground Station uses an encryption context constraint in grants to control access to the customer managed key in your account or region. The grant constraint requires that the operations that the grant allows use the specified encryption context.

The following are example key policy statements to grant access to a customer managed key for a specific encryption context. The condition in this policy statement requires that the grants have an encryption context constraint that specifies the encryption context.

```
{
  "Sid": "Enable DescribeKey",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleReadOnlyRole"
  },
  "Action": "kms:DescribeKey",
  "Resource": "*"
}, {
  "Sid": "Enable CreateGrant",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleReadOnlyRole"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:EncryptionContext:aws:groundstation:arn":
        "arn:aws:groundstation::111122223333:satellite/00a770b0-082d-45a4-80ed-SAMPLE"
    }
  }
}
```

Monitoring your encryption keys for AWS Ground Station

When you use an AWS KMS customer managed key with your AWS Ground Station resources, you can use [AWS CloudTrail](#) or [Amazon CloudWatch logs](#) to track requests that AWS Ground Station sends to AWS KMS. The following examples are AWS CloudTrail events for CreateGrant, GenerateDataKey, Decrypt, Encrypt and DescribeKey to monitor KMS operations called by AWS Ground Station to access data encrypted by your customer managed key.

CreateGrant (Cloudtrail)

When you use an AWS KMS customer managed key to encrypt your ephemeral resources, AWS Ground Station sends a CreateGrant request on your behalf to access the KMS key in your AWS account. The grant that AWS Ground Station creates are specific to the resource associated with the AWS KMS customer managed key. In addition, AWS Ground Station uses the RetireGrant operation to remove a grant when you delete a resource.

The following example event records the CreateGrant operation:

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AAAAAAAAAAAAAAAAAAAA:SampleUser01",
    "arn": "arn:aws:sts::111122223333:assumed-role/Admin/SampleUser01",
    "accountId": "111122223333",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE3",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
```

```

        "principalId": "AAAAAAAAAAAAAAAAAAAA",
        "arn": "arn:aws:iam::111122223333:role/Admin",
        "accountId": "111122223333",
        "userName": "Admin"
    },
    "webIdFederationData": {},
    "attributes": {
        "creationDate": "2022-02-22T22:22:22Z",
        "mfaAuthenticated": "false"
    }
},
"invokedBy": "AWS Internal"
},
"eventTime": "2022-02-22T22:22:22Z",
"eventSource": "kms.amazonaws.com",
"eventName": "CreateGrant",
"awsRegion": "us-west-2",
"sourceIPAddress": "111.11.11.11",
"userAgent": "ExampleDesktop/1.0 (V1; OS)",
"requestParameters": {
    "operations": [
        "GenerateDataKeyWithoutPlaintext",
        "Decrypt",
        "Encrypt"
    ],
    "constraints": {
        "encryptionContextSubset": {
            "aws:groundstation:arn":
"arn:aws:groundstation::111122223333:satellite/00a770b0-082d-45a4-80ed-SAMPLE"
        }
    },
    "granteePrincipal": "groundstation.us-west-2.amazonaws.com",
    "retiringPrincipal": "groundstation.us-west-2.amazonaws.com",
    "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
},
"responseElements": {
    "grantId": "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE"
},
"requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
"eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
"readOnly": false,
"resources": [
    {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    }
],
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111122223333",
"eventCategory": "Management"
}

```

DescribeKey (Cloudtrail)

When you use an AWS KMS customer managed key to encrypt your ephemeral resources, AWS Ground Station sends a DescribeKey request on your behalf to validate that the requested key exists in your account.

The following example event records the DescribeKey operation:

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AAAAAAAAAAAAAAAAAAAA:SampleUser01",
    "arn": "arn:aws:sts::111122223333:assumed-role/User/Role",
    "accountId": "111122223333",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE3",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AAAAAAAAAAAAAAAAAAAA",
        "arn": "arn:aws:iam::111122223333:role/Role",
        "accountId": "111122223333",
        "userName": "User"
      },
      "webIdFederationData": {},
      "attributes": {
        "creationDate": "2022-02-22T22:22:22Z",
        "mfaAuthenticated": "false"
      }
    },
    "invokedBy": "AWS Internal"
  },
  "eventTime": "2022-02-22T22:22:22Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "DescribeKey",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "AWS Internal",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "keyId": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
  },
  "responseElements": null,
  "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
  "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
  "readOnly": true,
  "resources": [
    {
      "accountId": "111122223333",
      "type": "AWS::KMS::Key",
      "ARN": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}
```

GenerateDataKey (Cloudtrail)

When you use an AWS KMS customer managed key to encrypt your ephemeral resources, AWS Ground Station sends a GenerateDataKey request to KMS in order to generate a data key with which to encrypt your data.

The following example event records the GenerateDataKey operation:

```
{
  "eventVersion": "1.08",
```

```

    "userIdentity": {
      "type": "AWSService",
      "invokedBy": "AWS Internal"
    },
    "eventTime": "2022-02-22T22:22:22Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": {
      "keySpec": "AES_256",
      "encryptionContext": {
        "aws:groundstation:arn":
"arn:aws:groundstation::11112223333:satellite/00a770b0-082d-45a4-80ed-SAMPLE",
        "aws:s3:arn":
"arn:aws:s3:::customerephemerisbucket/0034abcd-12ab-34cd-56ef-123456SAMPLE"
      },
      "keyId": "arn:aws:kms:us-
west-2:11112223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
      {
        "accountId": "11112223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-
west-2:11112223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
      }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "11112223333",
    "sharedEventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventCategory": "Management"
  }

```

Decrypt (Cloudtrail)

When you use an AWS KMS customer managed key to encrypt your ephemeris resources, AWS Ground Station uses the Decrypt operation to decrypt the ephemeris provided if it is already encrypted with the same customer managed key. For example if an ephemeris is being uploaded from an S3 bucket and is encrypted in that bucket with a given key.

The following example event records the Decrypt operation:

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AWSService",
    "invokedBy": "AWS Internal"
  },
  "eventTime": "2022-02-22T22:22:22Z",
  "eventSource": "kms.amazonaws.com",
  "eventName": "Decrypt",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "AWS Internal",
  "userAgent": "AWS Internal",
  "requestParameters": {
    "encryptionContext": {

```

```
        "aws:groundstation:arn":
"arn:aws:groundstation::111122223333:satellite/00a770b0-082d-45a4-80ed-SAMPLE",
        "aws:s3:arn":
"arn:aws:s3:::customerephemerisbucket/0034abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
},
"responseElements": null,
"requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
"eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
"readOnly": true,
"resources": [
    {
        "accountId": "111122223333",
        "type": "AWS::KMS::Key",
        "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    }
],
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111122223333",
"sharedEventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
"eventCategory": "Management"
}
```

Satellite Ephemeris Data

An [ephemeris](#), plural ephemerides, is a file or data structure providing the trajectory of astronomical objects. Historically, this file only referred to tabular data but, gradually, it has come to direct to a wide variety of data files indicating a spacecraft trajectory.

AWS Ground Station uses ephemeris data to determine when contacts become available for your satellite and correctly command antennas in the AWS Ground Station Network to point at your satellite. By default no action is required to provide AWS Ground Station with ephemerides.

Topics

- [Default Ephemeris Data](#) (p. 82)
- [Which Ephemeris Is Used](#) (p. 82)
- [Getting the Current Ephemeris for a Satellite](#) (p. 83)
- [Providing Custom Ephemeris Data](#) (p. 84)
- [Troubleshooting Invalid Ephemerides](#) (p. 86)
- [Reverting To Default Ephemeris Data](#) (p. 87)

Default Ephemeris Data

By default, AWS Ground Station uses publicly available data from [Space-Track](#), and no action is required to supply AWS Ground Station with these default ephemerides. These ephemerides are [two-line element sets](#) associated with your satellite's NORAD ID. All default ephemerides have a priority of 0. As a result, they will be overridden, always, by any non-expired, custom ephemerides uploaded via the ephemeris API, which must always have a priority of 1, or greater.

Satellites without a NORAD ID, must upload custom ephemeris data to AWS Ground Station. For example, satellites that have just launched or that are intentionally omitted from the Space-Track catalog would have no NORAD ID and would need to have custom ephemerides uploaded. For more information on providing a custom ephemeris see: [Providing Custom Ephemeris Data](#) (p. 84).

Which Ephemeris Is Used

Ephemerides have a *priority*, *expiration time*, and *enabled* flag. Together, these determine which ephemeris is used for a satellite. Only one ephemeris can be active for each satellite.

The ephemeris that will be used is the **highest-priority enabled ephemeris** whose expiration time is in the future. The available contact times returned by **ListContacts** are based on this ephemeris. If multiple enabled ephemerides have the same priority, the most recently created or updated ephemeris will be used.

If no ephemeris has been created, or if no ephemerides have ENABLED status, AWS Ground Station will use a default ephemeris for the satellite (from Space Track), if available. This default ephemeris has priority 0.

Effect of new Ephemerides on Previously Scheduled Contacts

Contacts scheduled prior to uploading new custom ephemerides will retain the originally scheduled acquisition of signal (AOS) / loss of signal (LOS) times but will use the active ephemeris for tracking.

If the spacecraft's position based on the active ephemeris differs greatly from the prior ephemeris, this may result in a result in reduced signal time or a portion of your contact being unusable due to the spacecraft being below the horizon or within a transmit/receive mask. It is therefore recommended that, when you upload a new ephemeris that differs greatly from previous ephemerides, that you also cancel and reschedule your future contacts for that satellite. Note that cancelled contacts may incur costs if cancelled too close to the time of the contact. For more information on cancelled contacts see: [Ground Station FAQs](#).

Getting the Current Ephemeris for a Satellite

The current ephemeris in use by AWS Ground Station for a specific satellite can be retrieved by calling the `GetSatellite` or `ListSatellites` actions. Both of these methods will return metadata for the ephemeris currently in use. This ephemeris metadata is different for custom ephemerides uploaded to AWS Ground Station and for default ephemerides.

Default Ephemerides will only include source and epoch fields. The epoch is the [epoch](#) of the [two-line element set](#) that was pulled from Space Track AWS Ground Station, and it is currently being used for computing the trajectory of the satellite.

A custom ephemeris will have a source value of "CUSTOMER_PROVIDED" and will include a unique identifier in the `ephemerisId` field. This unique identifier can be used to query for the ephemeris via the `DescribeEphemeris` action. An optional name field will be returned if the ephemeris was assigned a name during upload to AWS Ground Station via the `CreateEphemeris` action.

It is important to note that ephemerides are updated dynamically by AWS Ground Station so the returned data is only a snapshot of the ephemeris being used at the time of the call to the API.

Example `GetSatellite` return for a satellite using a default ephemeris

```
{
  "satelliteId": "e1cfe0c7-67f9-4d98-bad2-06dbfc2d14a2",
  "satelliteArn": "arn:aws:groundstation::111122223333:satellite/e1cfe0c7-67f9-4d98-bad2-06dbfc2d14a2",
  "noradSatelliteID": 12345,
  "groundStations": [
    "Example Ground Station 1",
    "Example Ground Station 2"
  ],
  "currentEphemeris": {
    "source": "SPACE_TRACK",
    "epoch": 8888888888
  }
}
```

Example `GetSatellite` for a satellite using a custom ephemeris

```
{
  "satelliteId": "e1cfe0c7-67f9-4d98-bad2-06dbfc2d14a2",
  "satelliteArn": "arn:aws:groundstation::111122223333:satellite/e1cfe0c7-67f9-4d98-bad2-06dbfc2d14a2",
  "noradSatelliteID": 12345,
  "groundStations": [
```



```
    "Example Ground Station 1",  
    "Example Ground Station 2"  
  ],  
  "currentEphemeris": {  
    "source": "CUSTOMER_PROVIDED",  
    "ephemerisId": "e1cfe0c7-67f9-4d98-bad2-06dbfc2d14a2",  
    "name": "My Ephemeris"  
  }  
}
```

Providing Custom Ephemeris Data

Warning

The ephemeris API is currently in a Preview state

Access to the Ephemeris API is provided only on an as-needed basis. Customers requiring the ability to upload custom ephemeris data should contact aws-groundstation@amazon.com.

Overview

The Ephemeris API allows custom ephemerides to be uploaded to AWS Ground Station for use with a satellite. These ephemerides override the default ephemerides from Space Track (see: [Default Ephemeris Data](#) (p. 82)).

Uploading customer ephemerides can improve the quality of tracking, handle early operations where no Space Track ephemerides are available to AWS Ground Station, and to account for maneuvers.

Creating a custom Ephemeris

A custom ephemeris can be created using the `CreateEphemeris` action in the AWS Ground Station API. This action will upload an ephemeris using data either in the request body or from a specified S3 bucket.

It is important to note that uploading an ephemeris sets the ephemeris to `VALIDATING` and starts an asynchronous workflow that will validate and generate potential contacts from your ephemeris. Only once an ephemeris has passed this workflow and become `ENABLED` will it be used for contacts. You should poll `DescribeEphemeris` for the ephemeris status or use Cloudwatch events to track the ephemeris' status changes.

To troubleshoot an invalid ephemeris see: [Troubleshooting Invalid Ephemerides](#) (p. 86)

Create a TLE Set Ephemeris via API

The AWS Ground Station boto3 client can be used to upload a two line element (TLE) set ephemeris to AWS Ground Station via the `CreateEphemeris` call. This ephemeris will be used in place of the default ephemeris data for a satellite (see [Default Ephemeris Data](#) (p. 82)).

A TLE set is a JSON formatted object that strings one or more TLEs together to construct a continuous trajectory. The TLEs in the TLE set must form a continuous set that we can use to construct a trajectory (i.e. no gaps in time between TLEs in a TLE set). An example TLE set is shown below:

```
# example_tle_set.json  
[  
  {  
    "tleLine1": "1 25994U 99068A   20318.54719794   .00000075   00000-0   26688-4 0  
9997",  
    "tleLine2": "2 25994   98.2007  30.6589 0001234   89.2782  18.9934  
14.57114995111906",  
  }  
]
```

```
        "validTimeRange": {
            "startTime": 12345,
            "endTime": 12346
        }
    },
    {
        "tleLine1": "1 25994U 99068A   20318.54719794   .000000075   00000-0  26688-4 0
9997",
        "tleLine2": "2 25994  98.2007  30.6589 0001234  89.2782  18.9934
14.57114995111906",
        "validTimeRange": {
            "startTime": 12346,
            "endTime": 12347
        }
    }
]
```

Note

The time ranges of the TLEs in a TLE set must match up exactly to be a valid, continuous trajectory.

A TLE set can be uploaded via the AWS Ground Station boto3 client as follows:

```
tle_ephemeris_id = ground_station_boto3_client.create_ephemeris( name="Example
Ephemeris", satelliteId="2e925701-9485-4644-b031-EXAMPLE01", enabled=True,
expirationTime=datetime.now(timezone.utc) + timedelta(days=3), priority=2,
    ephemeris = {
        "tle": {
            "tleData": [
                {
                    "tleLine1": "1 25994U 99068A   20318.54719794   .000000075   00000-0  26688-4
0 9997",
                    "tleLine2": "2 25994  98.2007  30.6589 0001234  89.2782  18.9934
14.57114995111906",
                    "validTimeRange": {
                        "startTime": datetime.now(timezone.utc),
                        "endTime": datetime.now(timezone.utc) + timedelta(days=7)
                    }
                }
            ]
        }
    })
```

This call will return an ephemeris Id that can be used to reference the ephemeris in the future. For example, we can use the provided ephemeris ID from the call above to poll for the status of the ephemeris:

```
client.describe_ephemeris(ephemerisId=tle_ephemeris_id['ephemerisId'])
```

An example response from the DescribeEphemeris action is provided below

```
{
    "creationTime": 1620254718.765,
    "enabled": true,
    "name": "Example Ephemeris",
    "ephemerisId": "fde41049-14f7-413e-bd7b-EXAMPLE01",
    "priority": 2,
    "status": "VALIDATING",
    "suppliedData": {
        "tle": {
            "ephemerisData": "[{\"tleLine1\": \"1 25994U 99068A   20318.54719794   .000000075
00000-0  26688-4 0 9997\", \"tleLine2\": \"2 25994  98.2007  30.6589 0001234  89.2782
```

```
18.9934 14.57114995111906\", \"validTimeRange\": {\"startTime\": 1620254712000, \"endTime\":  
1620859512000}}}]\"  
  }  
}  
}
```

It is recommended to poll the `DescribeEphemeris` route or use Cloudwatch events to track the status of the uploaded ephemeris as it must go through an asynchronous validation workflow before it is set to `ENABLED` and becomes usable for scheduling and executing contacts.

Note that the NORAD ID in all TLEs in the TLE set, 25994 in the examples above, must match the NORAD ID your satellite has been assigned in the Space Track database.

Uploading Ephemeris data from an S3 bucket

It is also possible to upload an ephemeris file directly from an S3 bucket by pointing to the bucket and object key. AWS Ground Station will retrieve the object on your behalf. Information about the encryption of data at rest in AWS Ground Station is detailed in: [Data Encryption At Rest For AWS Ground Station \(p. 73\)](#)

Below is an example of uploading an OEM ephemeris file from an S3 bucket

```
s3_oem_ephemeris_id = customer_client.create_ephemeris( name="2022-10-26 S3  
OEM Upload", satelliteId="fde41049-14f7-413e-bd7b-EXAMPLE01", enabled=True,  
expirationTime=datetime.now(timezone.utc) + timedelta(days=5), priority=2,  
    ephemeris = {  
        "oem": {  
            "s3Object": {  
                "bucket": "ephemeris-bucket-for-testing",  
                "key": "test_data.oem",  
            }  
        }  
    })
```

Below is an example returned data from the `DescribeEphemeris` action being called for the OEM ephemeris uploaded in the previous block of example code.

```
{  
    "creationTime": 1620254718.765,  
    "enabled": true,  
    "name": "Example Ephemeris",  
    "ephemerisId": "fde41049-14f7-413e-bd7b-EXAMPLE02",  
    "priority": 2,  
    "status": "VALIDATING",  
    "suppliedData": {  
        "oem": {  
            "sourceS3Object": {  
                "bucket": "ephemeris-bucket-for-testing",  
                "key": "test_data.oem"  
            }  
        }  
    }  
}
```

Troubleshooting Invalid Ephemerides

When a custom ephemeris is uploaded to AWS Ground Station it goes through an asynchronous validation workflow before becoming `ENABLED`. This workflow ensures that the satellite identifiers, metadata, and trajectory are valid.

When an Ephemeris fails validation, `DescribeEphemeris` will return an *EphemerisInvalidReason*, which provides insight into why the ephemeris failed validation. The potential values of the *EphemerisInvalidReason* are as follows:

Value	Description	Troubleshooting Action
METADATA_INVALID	Provided spacecraft identifiers such as satellite ID are invalid	Check the NORAD ID or other identifiers provided in the ephemeris data
TIME_RANGE_INVALID	Start, end, or expiration time(s) are invalid for the provided ephemeris	Make sure the Start time is before `now` (it is recommended to set the start time a few minutes in the past), that the end time is after the start time, and that the end time is after the expiration time
TRAJECTORY_INVALID	Provided ephemeris defines an invalid spacecraft trajectory	Confirm that the provided trajectory is continuous and is for the correct satellite.
VALIDATION_ERROR	Internal service error occurred while processing ephemeris for validation	Retry Upload

An example `DescribeEphemeris` response for an `INVALID` ephemeris is provided below:

```
{
  "creationTime": 1000000000.00,
  "enabled": false,
  "ephemerisId": "d5a8a6ac-8a3a-444e-927e-EXAMPLE1",
  "name": "Example",
  "priority": 2,
  "status": "INVALID",
  "invalidReason": "METADATA_INVALID",
  "suppliedData": {
    "tle": {
      "sourceS3Object": {
        "bucket": "my-s3-bucket",
        "key": "myEphemerisKey",
        "version": "ephemerisVersion"
      }
    }
  }
},
}
```

Reverting To Default Ephemeris Data

When you upload custom ephemeris data it will override the default ephemerides AWS Ground Station uses for that particular satellite. AWS Ground Station does not use the default ephemeris again until there are no currently enabled, unexpired customer-provided ephemerides available for use. AWS Ground Station also does not list contacts past the expiration time of the current customer-provided ephemeris, even if there is a default ephemeris available past that expiration time.

To revert back to the default Space Track ephemerides, you will need to do one of the following:

- Delete (using `DeleteEphemeris`) or disable (using `UpdateEphemeris`) all enabled customer-provided ephemerides. You can list the customer-provided ephemerides for a satellite using `ListEphemerides`.
- Wait for all existing customer-provided ephemerides to expire.

You can confirm that the default ephemeris is being used by calling `GetSatellite` and verifying that the source of the current ephemeris for the satellite is `SPACE_TRACK`. See [Default Ephemeris Data \(p. 82\)](#) for more information on default ephemerides.

Document History for the AWS Ground Station User Guide

The following table describes the important changes to the documentation since the last release of AWS Ground Station.

Change	Description	Release Date
New Feature	Updated the user guide for release of CPE Preview.	November 9, 2022
New Feature	Updated the user guide to include integration with AWS CLI.	April 17, 2020
New Feature	Updated the user guide to include integration with CloudWatch Metrics.	February 24, 2020
New Template	Public Broadcast Satellites (AquaSnppJpss Template) added to the <i>AWS Ground Station User Guide</i> .	February 19, 2020
New Feature	Updated the user guide to include cross-region data delivery.	February 5, 2020
Documentation Update	Updated examples and descriptions for monitoring AWS Ground Station with CloudWatch Events.	February 4, 2020
Documentation Update	Template locations have been updated and the Getting Started and Troubleshooting sections have been revised.	December 19, 2019
New Troubleshooting Section	Troubleshooting section added to the <i>AWS Ground Station User Guide</i> .	November 7, 2019
New Getting Started Topic	Updated the Getting Started topic, which includes the most current AWS CloudFormation templates.	July 1, 2019
Kindle Version	Published Kindle version of the <i>AWS Ground Station User Guide</i> .	June 20, 2019
New service and guide	This is the initial release of AWS Ground Station and the <i>AWS Ground Station User Guide</i> .	May 23, 2019

AWS glossary

For the latest AWS terminology, see the [AWS glossary](#) in the *AWS General Reference*.