Technical Document

Niagara NDS API Guide

February 7, 2025

niagara4

Legal Notice

Tridium, Incorporated

3951 Western Parkway, Suite 350
Richmond, Virginia 23233
U.S.A.

Confidentiality

The information contained in this document is confidential information of Tridium, Inc., a Delaware corporation (Tridium). Such information and the software described herein, is furnished under a license agreement and may be used only in accordance with that agreement.

The information contained in this document is provided solely for use by Tridium employees, licensees, and system owners; and, except as permitted under the below copyright notice, is not to be released to, or reproduced for, anyone else.

While every effort has been made to assure the accuracy of this document, Tridium is not responsible for damages of any kind, including without limitation consequential damages, arising from the application of the information contained herein. Information and specifications published here are current as of the date of this publication and are subject to change without notice. The latest product specifications can be found by contacting our corporate headquarters, Richmond, Virginia.

Trademark notice

BACnet and ASHRAE are registered trademarks of American Society of Heating, Refrigerating and Air-Conditioning Engineers. Microsoft, Excel, Internet Explorer, Windows, Windows Vista, Windows Server, and SQL Server are registered trademarks of Microsoft Corporation. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Mozilla and Firefox are trademarks of the Mozilla Foundation. Echelon, LON, LonMark, LonTalk, and LonWorks are registered trademarks of Echelon Corporation. Tridium, JACE, Niagara Framework, and Sedona Framework are registered trademarks, and Workbench are trademarks of Tridium Inc. All other product names and services mentioned in this publication that are known to be trademarks, registered trademarks, or service marks are the property of their respective owners.

Copyright and patent notice

This document may be copied by parties who are authorized to distribute Tridium products in connection with distribution of those products, subject to the contracts that authorize such distribution. It may not otherwise, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior written consent from Tridium, Inc.

The product(s) described herein may be covered by one or more U.S. or foreign patents of Tridium.

For an important patent notice, please visit: http://www.honpat.com.

Contents

Chapter 1. About this guide 5
Document change log 5
Related Documentation 5
Chapter 2. Service account management 7
Creating a service account 7
API authentication 11
OAuth support configuration 11
Implementing an OAuth client_credentials flow 12
Deleting a service account 12
Updating a service account 13
Chapter 3. EMS API support 15
Inputs 15
Page request info (pageable) 15
Customer Id 16
Search criteria 16
searchItems 17
Chapter 4. Egress API support 21
Submitting an HTTP request 21
How to get a system GUID 22
How to get a cloudId 23
Post properties 24
Chapter 5. NDS Control API support 27
Read point values 27
Write point values 29
Clear point values 32
Read point inputs 34
Chapter 6. Alarms Service API support 39
API details 39
Inputs 40
Outputs 43
Chapter 7. Glossary 47

About this guide

This guide is targeted at partner users (developers) who intend to build their own tools to access the NCS Entity Model Service (EMS) for use cases beyond the NDS UI or want to use the Egress API support.

Product Documentation

This document is part of the Niagara technical documentation library. Released versions of Niagara software include a complete collection of technical information that is provided in both online help and PDF format. The information in this document is written primarily for Systems Integrators. To make the most of the information in this book, readers should have some training or previous experience with Niagara software, as well as experience working with JACE network controllers.

Document Content

This document provides information about how to extend the functionality of the Niagara Cloud Suite (NCS) by using APIs. It describes how you can access the APIs of the NCS Entity Model Service (EMS) for using with the Niagara Data Service (NDS) product. Also, it provides information about Egress APIs used to query the telemetry data in the cloud database and send it to client applications.

Document change log

Updates (changes and additions) to this document are listed below.

February 7, 2025

- Included the Operator Role requirement to the "NDS Control API support" chapter.
- Added "Clear point values" API topic to the "NDS Control API support" chapter.

December 12, 2024

- Added the updated cloudId value for the request body in the "Egress API support" chapter.
- Updated "Alarms Service API support" to include URL to Swagger UI.
- Added "Alarm search request example" to the "Search request" section.
- Added "Alarms Service API support".

July 1, 2024

- New - Added "*NDS Control API support*" to include *POST read request*, *POST write request*, and *POST readAll request*.
- Added "Nds Operator" role to "*Creating a service account*" chapter.

September 11, 2023

- Updated *Creating a Service Account* to include new features for managing access.

April 14, 2023

- Expand document title to read: *Niagara Data Service (NDS) API Guide*.

January 19, 2023

- Reorganized the first chapter.
- Added service account topics.
- Updated the topic "Submitting an HTTP request" to correct the URI.

August 18, 2022

- Initial release.

Related Documentation

Additional information is available in the following documents.

- *Niagara Cloud Suite (NCS) Partner Guide*

Chapter 1. Service account management

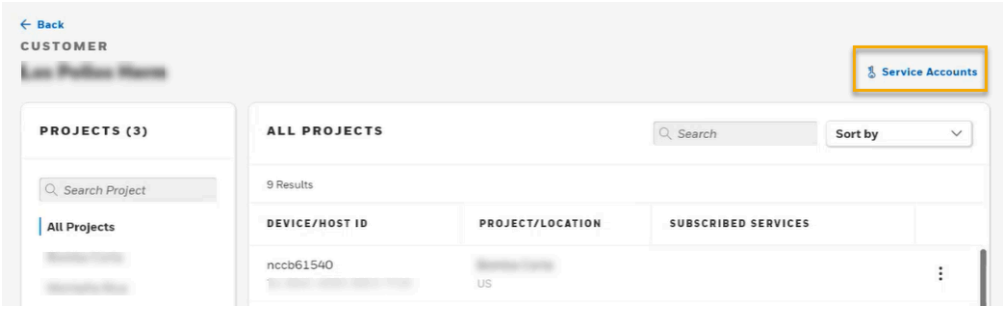
To access and use the APIs, you need credentials. Creating and managing a service account provides these needed credentials (client Id and secret) with which to access the APIs.

Creating a service account

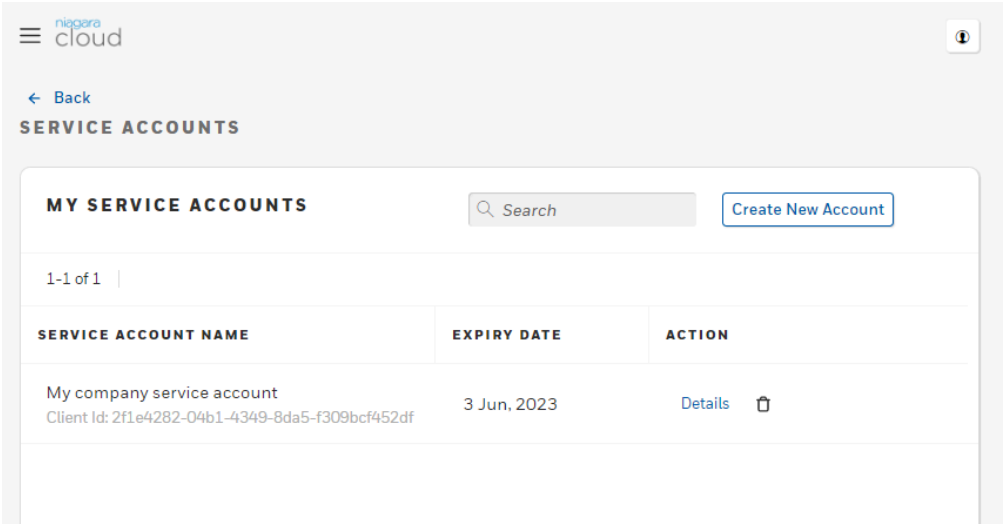
A service account authorizes you to use the provided APIs in user applications.

Prerequisites:
You are a Partner Admin and are logged in to https://www.niagara-cloud.com.

Step 1. Navigate to the home page and click the customer.
The **CUSTOMER** page opens.

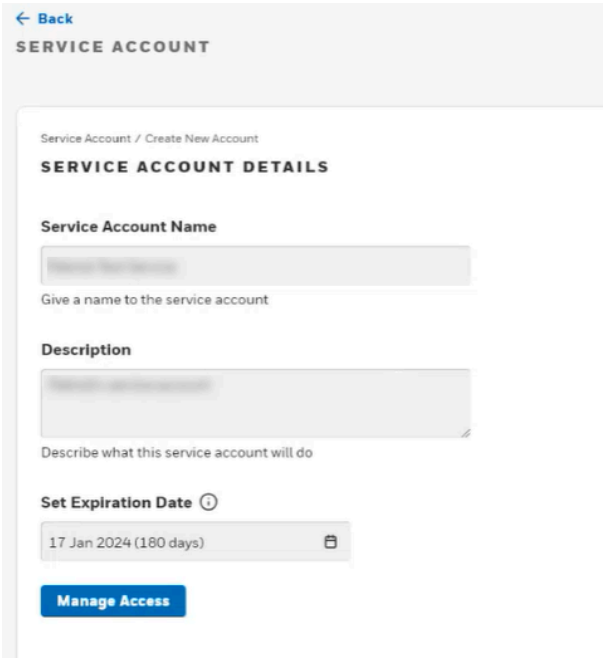


Step 2. Click the **Service Accounts** link in the upper right corner of the page.
The **SERVICE ACCOUNTS** page opens.



Step 3. Click **Create New Account**.

The **SERVICE ACCOUNT DETAILS** page opens.



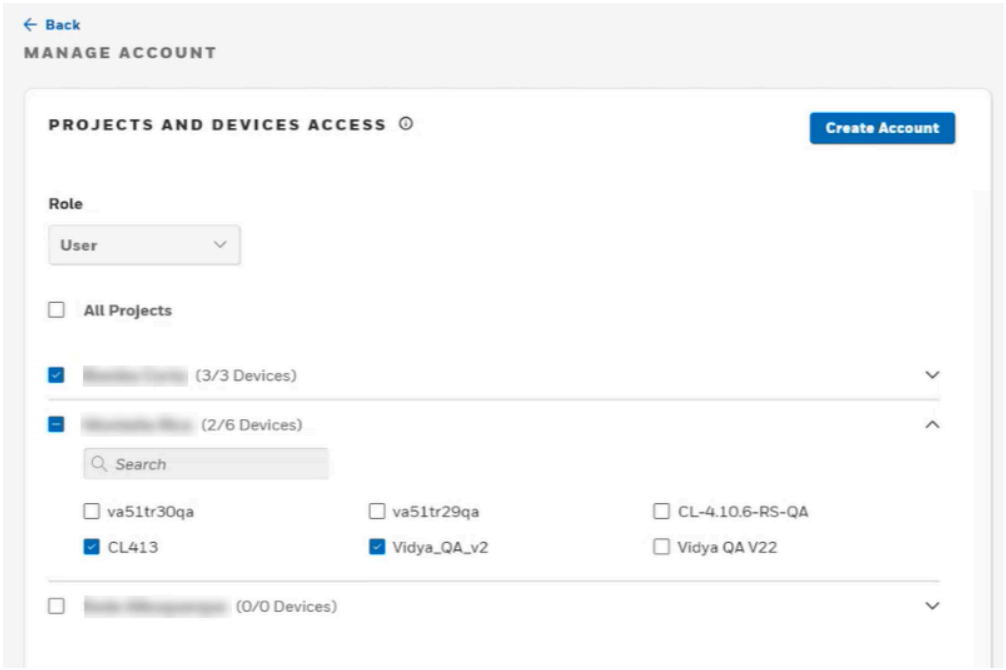
The software activates the **Manage Access** button after you enter an account name.

Step 4. Provide a name, description and expiration date for the service account and click **Manage Access**.
The **`Set Expiration`** date defaults to 180 days, which is the maximum number of days.
The **Projects and Devices Access** view opens.

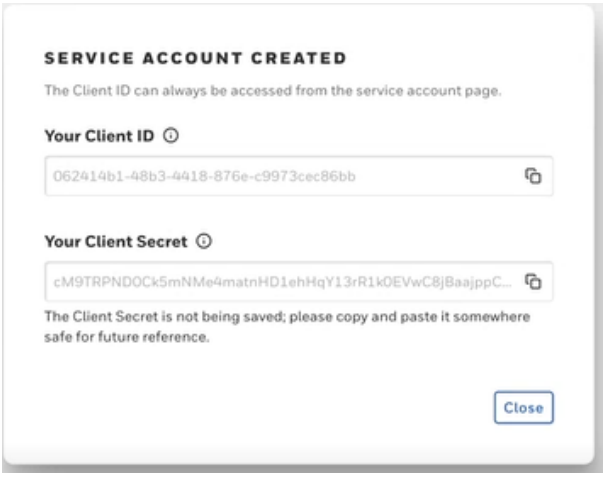
Step 5. Choose from the following options.

- To grant access to all projects within the customer account, select the **All Projects** check box.
- To grant access to only individual projects and devices, select the check boxes of the targeted projects and devices, and click **Create Account**.

  NOTE: Only projects assigned to this particular SI will be displayed under **All Projects**. Projects assigned to other SIs will not be shown.



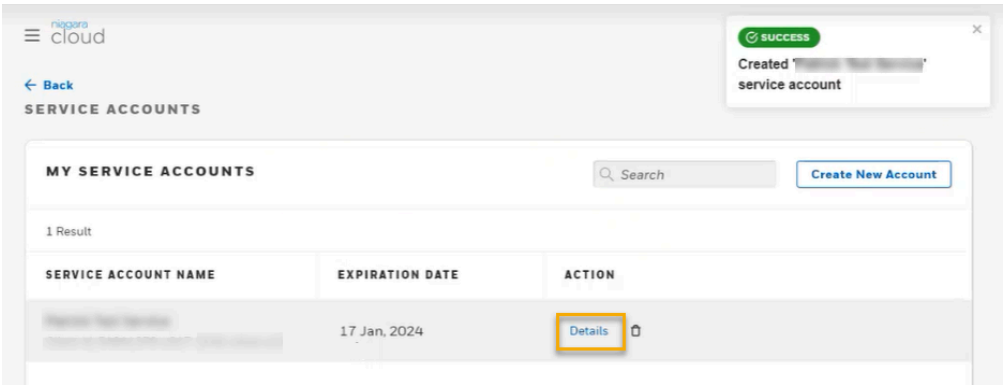
The system creates the account and opens the **SERVICE ACCOUNT CREATED** window.



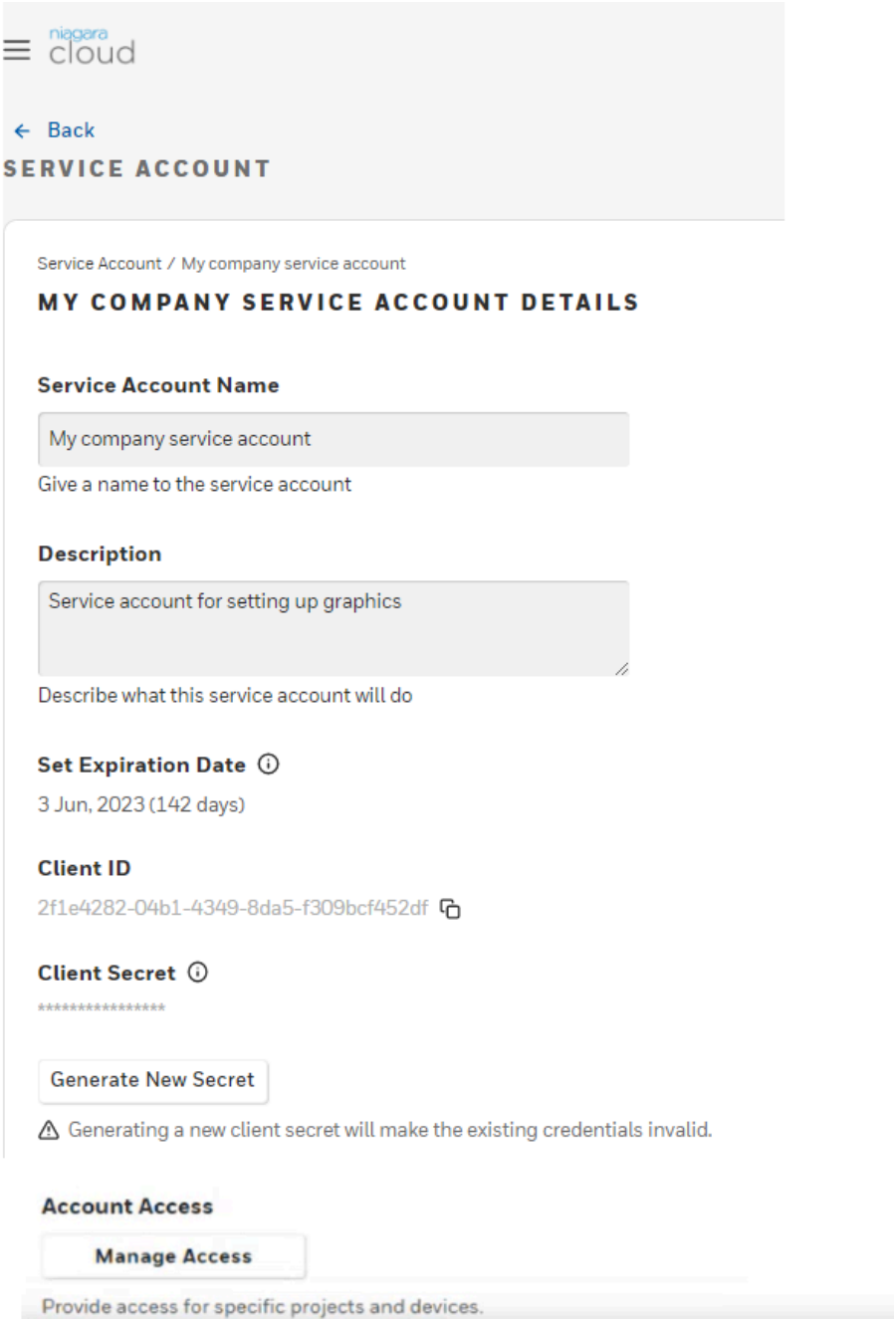
This window provides the client ID and secret.

Step 6. Before you continue, copy and paste the client secret to a safe location. To copy the client secret, click the copy icon ( ) and click **Close**.
The **SERVICE ACCOUNTS** page opens. A popup window in the upper right corner notifies you that the creation of the service account was successful. This page includes the account you just created.

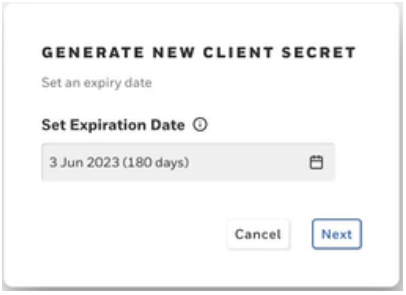
Step 7. To view and edit the access rights for the newly created service account, click on the **Details** link in the **Action** column for the new account.



The system shows the account details without displaying the **`Client Secret`**.



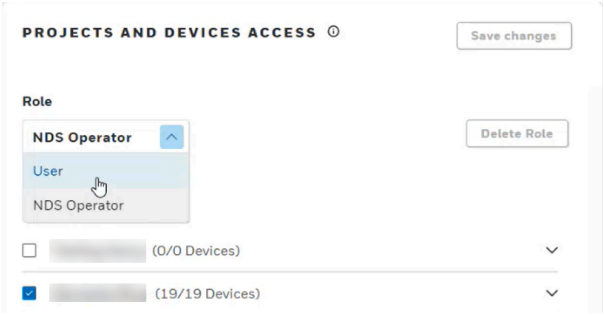
Step 8. If you have lost your client secret, click **Generate New Secret**.
The **GENERATE NEW CLIENT SECRET** window opens.



Step 9. Set the expiration date and click **Next**.
The system generates the new client secret.

CAUTION: When you generate a new secret, the old secret becomes invalid. Any application that uses the old secret must be updated.

Step10. Under **Account Access**, click **Manage Access** to provide access for specific projects and devices and select the appropriate role.



- The **User** role only allows limited access to projects and devices.
- The **NDS Operator** role allows the writing of value points and provides basic Niagara Data Service permissions. For more information, see "*NCS role-based access control*" in the NCS Partner Guide.

Step11. To continue click **Copy & Close** and save the secret in a secure location.

API authentication

A service account for each customer provides this authentication using a client Id and secret.

Service accounts for Niagara Cloud Suite use the OAuth client credentials flow. It is simple to use a client Id and client secret to obtain an access token that can then be used to authenticate to APIs.

There are two ways to authenticate the service account to APIs using the client Id and client secret:

- You can configure the settings of the HTTP libraries and utilities that have integrated OAuth support.
- You can implement the OAuth 2.0 client_credentials flow independently.

OAuth support configuration

The NCS APIs allow authentication with the OAuth 2.0 client_credentials flow. Many HTTP libraries and utilities have built-in OAuth support that needs configuration.

- **Grant Type**: client_credentials
- **Client Id**: Your client Id
- **Client Secret**: Your client secret
- **Scope**: ncp:read

- **Authentication Method**: client secret basic

Implementing an OAuth client_credentials flow

The following describes how to implement the OAuth 2.0 client_credentials flow by following these steps.

Prerequisites:
You created a service account and have a client Id and secret.

Step 1. Request authentication to the token endpoint at:
https://auth.pingone.com/2bb83e4b-dd73-47d3-984b-6e601302d766/as/token

This is a POST request with the following requirements:

- "Content-Type" header is set to "application/x-www-form-urlencoded"
- "Authorization" header is set to Base64 encoding of <clientId>:<clientSecret>

  Example: For client Id "1234" and client secret "abcd", you concatenate to "1234:abcd", and Base64 encode the result for a value of "MTIzNDphYmNk".
- The body has the following values as URL encoded form attributes:
  a. grant_type=client_credentials
  b. scope=ncp:read

Step 2. After the authentication succeeded, the response has a 200-status code with a body in the following format:

```
{
 "access_token" : &lt;access-token&gt;,
 "token_type" : "Bearer",
 "expires_in" : 3600,
 "scope" : "ncp:read"
}
```

Step 3. To authenticate to APIs by adding an "Authorization" header with the format "Bearer <access-token>", use the value in the "access_token" field.

Step 4. Use the token until it expires as indicated by the "expires_in" field (1 hour). When the token expires, obtain a new one by making another token request as described in Step 1. You can also request a new token before the old one expires.

Result
For testing, you can retrieve an access token manually using tools, such as cURL or Postman.

Below is an example curl command to retrieve an access token. Be sure to insert your base64 encoded client Id and client secret in the placeholder for the "Authorization" header.

curl -X POST https://auth.pingone.com/2bb83e4b-dd73-47d3-984b-6e601302d766/as/token -H "Content-Type: application/x-www-form-urlencoded" -H "Authorization: Basic <base64 endcoded clientId:clientSecret>" -d "grant_type=client_credentials&scope=ncp:read"

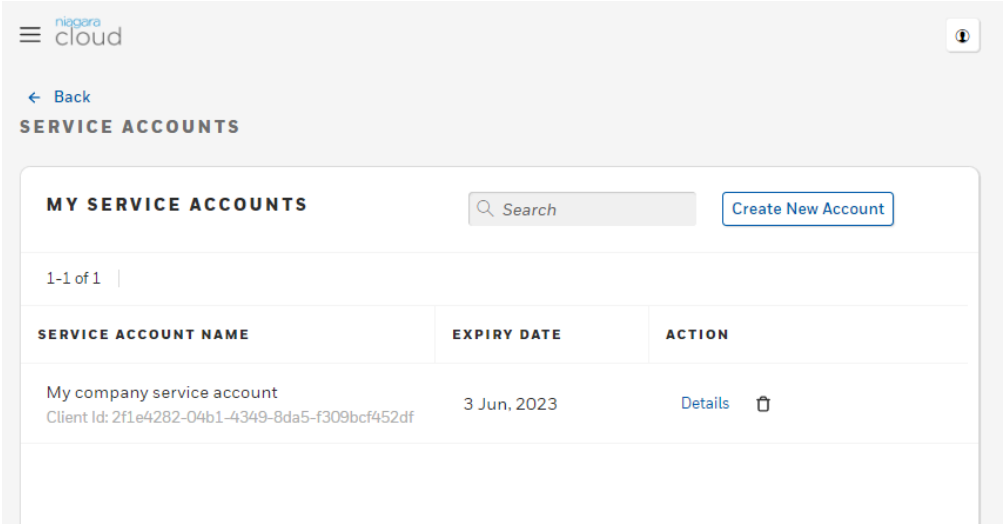
Deleting a service account

The **SERVICE ACCOUNTS** page provides service account management.

Step 1. Navigate to the home page.

Step 2. Click the **Service Accounts** link in the upper right of the page.

The **SERVICE ACCOUNTS** page opens.



Step 3. To delete an account, click the delete icon ( ) next to the **Details** link under the ACTION column.
A confirmation message opens.

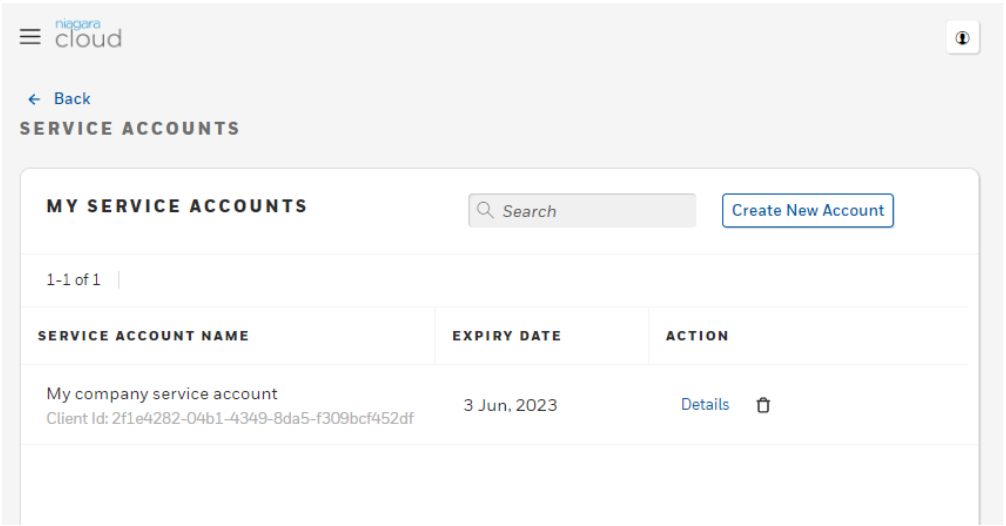
Step 4. To complete the deletion, click **Confirm**.
After deleting the service account, the client Id and secret no longer work, which means that applications no longer have access to the API.

Updating a service account

You may change service account details.

Step 1. Navigate to the home page.

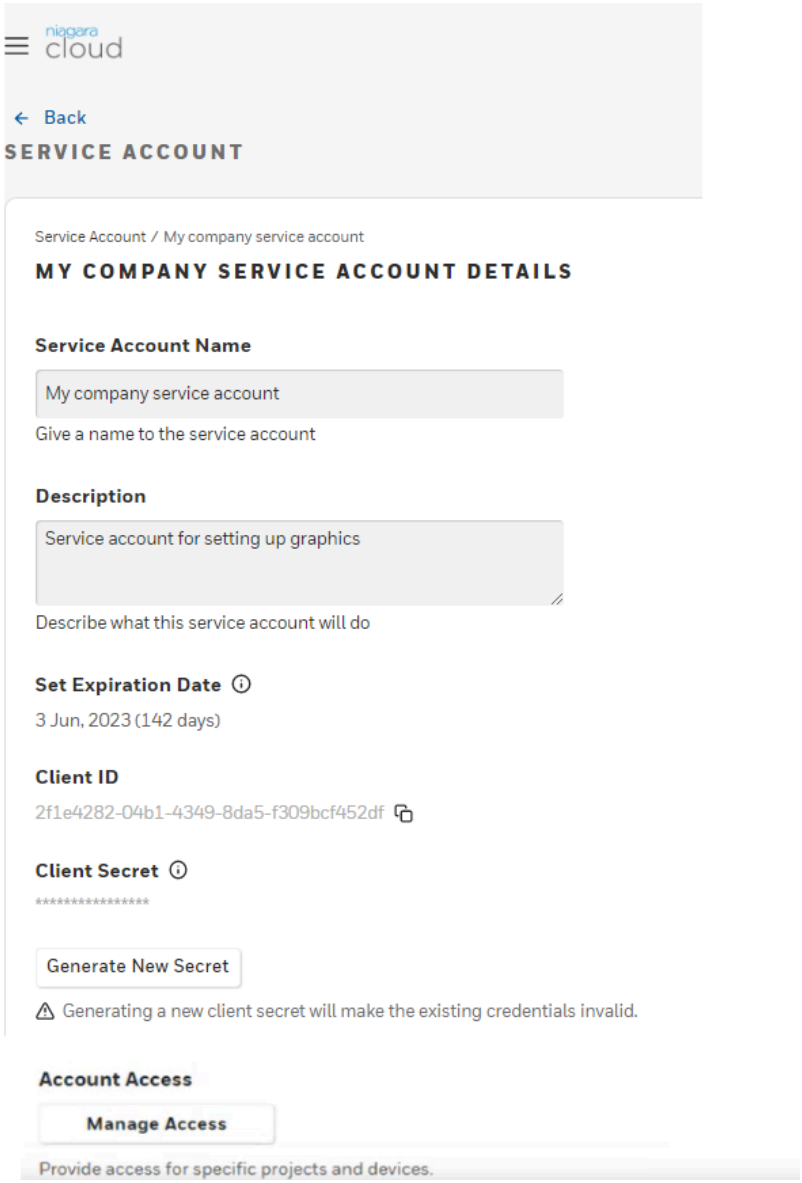
Step 2. Click the **Service Accounts** link in the upper right of the page.
The **SERVICE ACCOUNTS** page opens.



Step 3. To locate the service account to update, use the **Search** box, scroll down or click a page number at the bottom of the page.

Step 4. To edit account details click the **Details** link for the account under the ACTION column.

The **ACCOUNT DETAILS** page opens.



Step 5. Edit the details and click outside the edited property or click **Back** to save your changes.

Chapter 2. EMS API support

The Entity Model Service (EMS) provides a way to query the model database for semantic model information about point values.

The EMS API is available at:

https://www.niagara-cloud.com/api/v1/entitymodel/swagger-ui/index.html

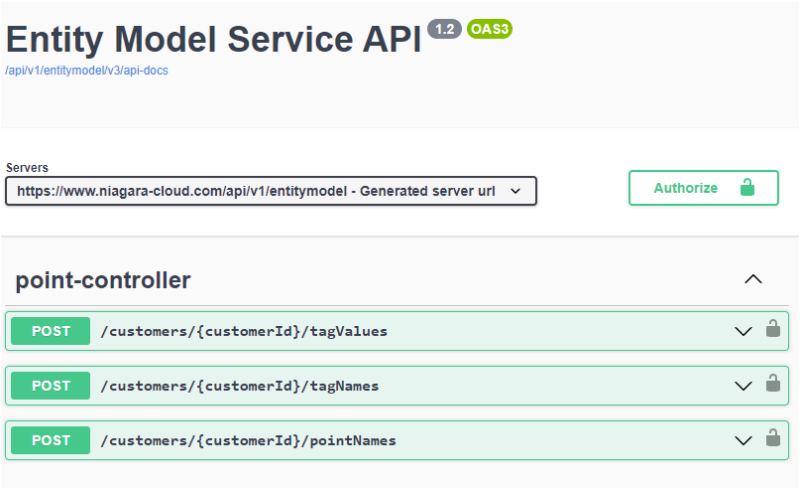
You can query the model database in three different ways:

- by point name
- by tag name
- by tag value

The results returned are items in the model database that match the query and include the following information about each item:

- cloud Id (by which it is referenced in the telemetry database of time series data)
- systemGuid of the containing system
- display name
- tags

Figure 1. API Swagger



Inputs

All EMS queries are HTTP POST methods.

The query parameters are supplied in the POST body as described in the following input topics.

Page request info (pageable)

To potentially limit the scope of results returned, you need to supply page request information.
The schema for page information is as follows:

- page: integer *(required)*: this number is zero based - the first page is 0
- size: integer *(required)*: this is the page size, consequently, the minimum number is 1

The following is an example page info object. It requests the second page, using a page size of 10, that is, the eleventh through twentieth items, sorted in descending order.

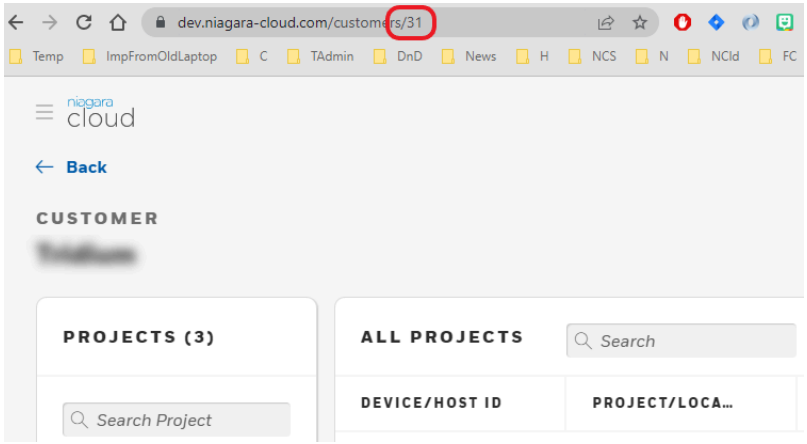
Pageable example

```
{
 "page": 1,
 "size": 10,
 "sort": [
  "desc"
 ]
}
```

Customer Id

The EMS is organized by customers who have their own separate data container. A customer cannot access another customer's container. This unique Id is supplied as a path parameter called `customerId`.

To obtain your `customerId`, navigate to the NDS home page and select a customer. The URL contains the `customerId` at the end of the URL:



Search criteria

To find items, you also need to supply search criteria. Provide the information in the following fields in the body of the search criteria:

- `systemGuid` *(required)*: string

  This is the `systemGuid` of the queried system. Only a single system's model data may be queried in one request.
- `searchType` *(required)*: string

  This is either `"pointName"`, `"tagName"`, or `"tagValue"` and must match the URI you use.
- `comparisonType` *(required)*: string

  These are the match requirements: `"any"`, `"exact"`, `"startswith"`, `"endswith"`, or `"contains"` explained in more detail below.
- `searchItems` *(required)*: string

  This is query information explained in more detail below.

searchItems

The `searchItems` is an array of strings that is used to query the EMS database. For each search type, the behaviors are as follows:

Search type pointName

The `searchItems` arguments are used to match database items by the name field. All queries (except `"any"`) are case-sensitive.

- `any`: All database items match. It can be used to retrieve every item for the supplied systemGuid. Note that if you use this comparison type, the `searchItems` field is not used.
- `exact`: Return only database items whose name field is an exact match for the argument.
- `startswith`: Return database items whose name field contains the argument at the beginning of the string.
- `endswith`: Return database items whose name field contains the argument at the end of the string.
- `contains`: Return database items whose name field contains the argument anywhere in the string.

Search type pointName: find all points

Example: Find all points for systemGuid `11223344-5566-7788-99aa-bbccddeeff00` :

```
{
 "systemGuid": "11223344-5566-7788-99aa-bbccddeeff00",
 "searchType": "pointName",
 "comparisonType": "any"
}
```

Search type pointName: find points named “SpaceTemp”

Example: Find all points for systemGuid `11223344-5566-7788-99aa-bbccddeeff00` with the name `"SpaceTemp"` :

```
{
 "systemGuid": "11223344-5566-7788-99aa-bbccddeeff00",
 "searchType": "pointName",
 "searchItems": [ "SpaceTemp" ],
 "comparisonType": "exact"
}
```

Search type pointName: find points starting with “AHU” or “VAV”

Example: Find all points for systemGuid `11223344-5566-7788-99aa-bbccddeeff00` with names beginning with `"AHU"` or `"VAV”` :

```
{
 "systemGuid": "11223344-5566-7788-99aa-bbccddeeff00",
 "searchType": "pointName",
 "comparisonType": "startswith",
 "searchItems": [ "AHU", "VAV" ]
}
```

Search type pointName: find points ending with “Cmd” or “Command”

Example: Find all points for systemGuid 11223344-5566-7788-99aa-bbccddeeff00 with names ending in "Cmd" or "Command" :

```
{
 "systemGuid": "11223344-5566-7788-99aa-bbccddeeff00",
 "searchType": "pointName",
 "comparisonType": "endswith",
 "searchItems": [ "Cmd", "Command" ]
}
```

Search type pointName: find points with “FirstFloor” in the name

Example: Find all points for systemGuid 11223344-5566-7788-99aa-bbccddeeff00 with names containing "FirstFloor" :

```
{
 "systemGuid": "11223344-5566-7788-99aa-bbccddeeff00",
 "searchType": "pointName",
 "comparisonType": "contains",
 "searchItems": [ "FirstFloor" ]
}
```

Search type tagName

The searchItems arguments are used to match database items by looking for matches in the tags list. All queries are case-sensitive. The comparisonType is ignored. To match, the tags list of an item must contain tags with a tagId (key) that matches each one of the searchItems arguments.

Example: Find all points for systemGuid 11223344-5566-7788-99aa-bbccddeeff00 with the tag "hs:temp" :

Search type tagName: find all points with the tag “hs:temp”

```
{
 "systemGuid": "11223344-5566-7788-99aa-bbccddeeff00",
 "searchType": "tagName",
 "searchItems": [ "hs:temp" ]
}
```

Search type tagName: find all points that have “n:point” AND “hs:air” tags

Example: Find all points for systemGuid 11223344-5566-7788-99aa-bbccddeeff00 with tags "n:point" and "hs:air" :

```
{
 "systemGuid": "11223344-5566-7788-99aa-bbccddeeff00",
 "searchType": "tagName",
 "searchItems": [
   "n:point",
   "hs:air"
 ]
}
```

Search type tagValue

The searchItems arguments are used to match database items by looking for matches in the tags list. All queries are case-sensitive. The comparisonType is ignored. The searchItems must be in the form "key=value",

where “key” is the tag name followed by an equals sign, and then “value” is the tag value. To match, the tags list of an item must contain a tag with tagId equal to “key” and with a tagValue equal to “value” for each one of the searchItems arguments.

Search type tagValue: find all points with the tag “hs:kind” equal to “Number” and the tag “hs:tz” equal to “New_York”

Example: Find all points for systemGuid 11223344-5566-7788-99aa-bbccddeeff00 with the "hs:kind" tag equal to "Number" and the "hs:tz" tag equal to "New_York" :

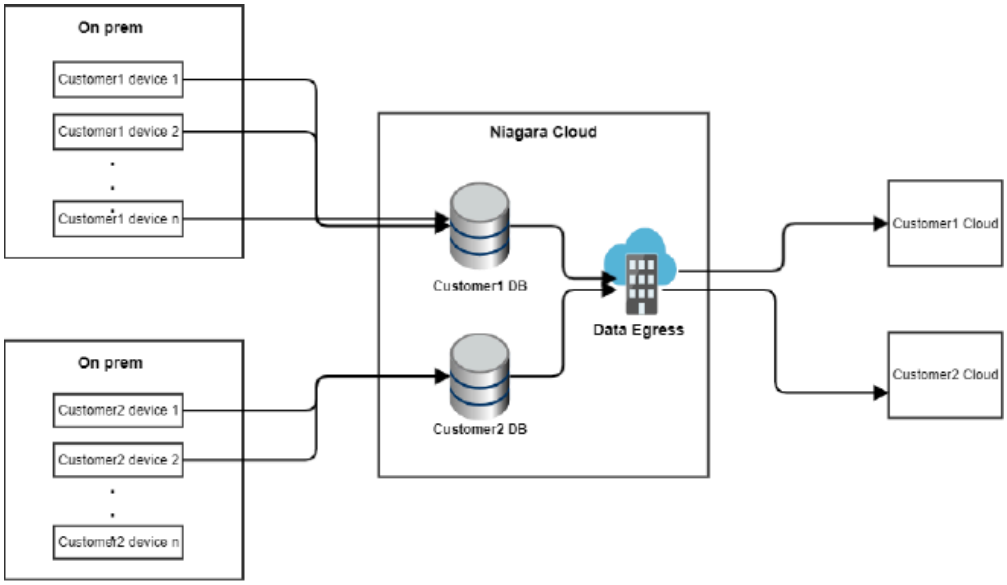
```
{
 "systemGuid": "11223344-5566-7788-99aa-bbccddeeff00",
 "searchType": "tagValue",
  "searchItems": [
   "hs:tz=New_York",
   "hs:kind=Number"
 ]
}
```

Chapter 3. Egress API support

To extend the functionality of the NCS, APIs are available.

Data Egress is the API layer used to query the telemetry data in the cloud database and send it to a client application. The Niagara Cloud supports multi-tenancy, which means it stores telemetry data coming from each customer's device in separate physical locations.

Data Egress REST APIs are used to retrieve data from each station into customer client applications.



NCS supports the following APIs. The documentation for each is available in Swagger.

- Entity Model Service (point controller), documentation: https://www.niagara-cloud.com/api/v1/entitymodel/swagger-ui/index.html
- Niagara Data Service Egress API (egress controller), documentation: https://www.niagara-cloud.com/api/v1/egress/swagger-ui/index.html

The first API queries for semantic model data, to identify data points of interest based on point name, tags, or tag values. The second API queries the stored history data for points identified by the first API.

These topics demonstrate how to post telemetry data for specified points over a given time period, obtain OAuth access tokens, and invoke the API methods using HTTP requests.

Submitting an HTTP request

This example demonstrates how to send HTTP requests to post the telemetry data from the station to the cloud.

Prerequisites:
You are using a REST client, such as Postman. You performed the step to retrieve an access token.

The URI format is: https://www.niagara-cloud.com/api/v1/egress/telemetry.

Step 1. In **HTTP Method**, select **POST**.

Step 2. For **Request URI**, enter:

https://www.niagara-cloud.com/api/v1/egress/telemetry

Step 3. Set the **`Authorization header`** to `Bearer <access-token>` where `<access token>` is retrieved by following the steps in "Getting the access token."

Step 4. In the **Body** tab, copy and paste the following JSON.

Egress API body Collapse source

```
{
  "systemGuid": "xxxxxxx",
  "cloudId": [
    "xxxxxxxxxxx",
    "yyyyyyyyyyy"
  ],
  "startTime": "2022-02-03T06:28:14.353Z",
  "endTime": "2022-02-05T08:59:49.427Z",
  "recordLimit": "5000",
  "includePreRecord": false,
  "includePostRecord": false,
  "sortAscending": false
}
```

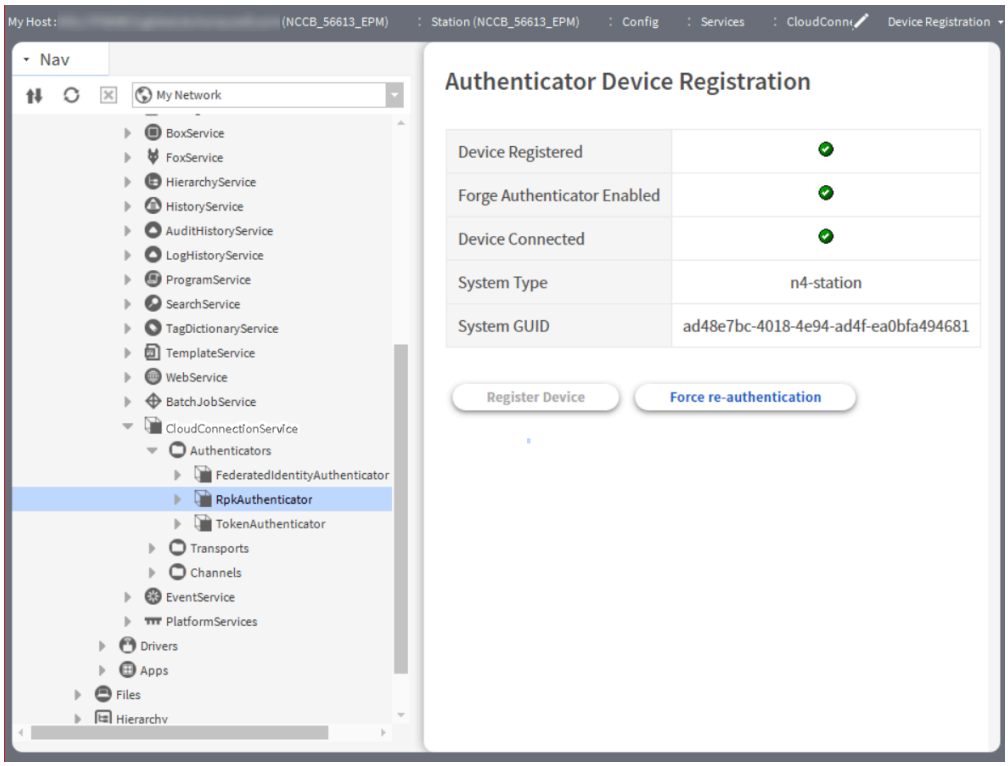
How to get a system GUID

The system GUID is the unique identifier for the station.

Prerequisites:
You are connected to the station.

Step 1. Launch Workbench and open the station.

Step 2. Navigate to **Config** > **Services** > **CloudConnectionService** and double-click **RpkAuthenticator**. The Authenticator Device Registration window opens.



Step 3. On the right, copy the system GUID.

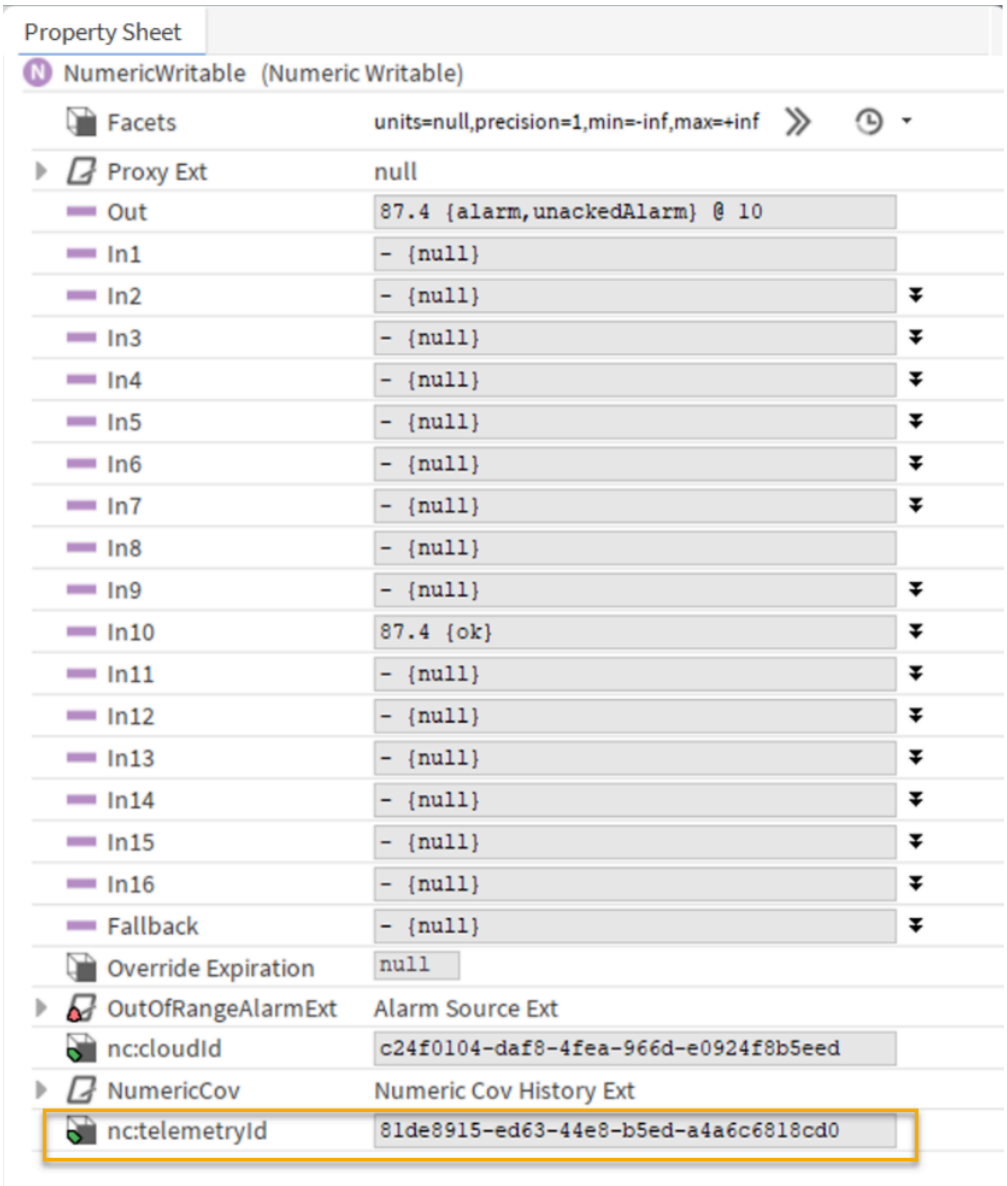
How to get a cloudId

The data Egress uses the telemetry Id as the cloudId in its body.

Prerequisites:
You are working in Workbench.

Step 1. Navigate to the point.

Step 2. To open the point **Property Sheet**, double-click on the point.
The point's property sheet opens.



The screen capture is an example of a numeric writable point.

Step 3. Scroll to the end of the **Property Sheet** and copy the **nc:telemetryId** property.
NOTE: The request body uses the **telemetry Id** in the value field of the **cloudId** attribute (key).

Post properties

In addition to the GUID and cloudId, these properties need to be configured.

Property	Value	Description
startTime	date and time	Defines the time from which to query the telemetry data.
endTime	date and time	Defines the time to end the telemetry data query.
recordLimit	integer	Defines the maximum number of records to return in the response. These are telemetry data records.
includePrerecord	Boolean true or false	Determines when to add an additional value to `startTime`. true adds the last-known value before `startTime`. false adds nothing.
includePostrecord	Boolean true or false	Determines when to add an additional value to endTime. true adds the first-known value after `endTime`. false adds nothing.

Example response


```
{
  "systemGuid": "ed5f190e-ca91-4252-8f13-c70f4e48ce13",
  "recordLimit": 100,
  "pointDetails": [
    {
      "pointId": "50c92e68-f918-4734-a783-969e12f9580b",
      "preRecord": {
        "t": "2022-02-03T12:37:54.943Z",
        "v": 16.97627830505371
      },
      "postRecord": {
        "t": "2022-02-03T2:37:54.943Z",
        "v": 17.87627830505371
      },
       "historyRecords": [
        {
          "time": "2022-02-03T12:58:00.953Z",
          "value": 36.296260833740234
        },
        {
          "time": "2022-02-03T12:57:45.932Z",
          "value": 38.795162200927734
        },
        {
          "time": "2022-02-03T12:57:30.904Z",
          "value": 41.32102966308594
        },
        {
          "time": "2022-02-03T12:57:15.881Z",
          "value": 43.87285232543945
        },
        {
          "time": "2022-02-03T12:57:00.864Z",
          "value": 46.43808364868164
        },
        {
          "time": "2022-02-03T12:56:45.838Z",
          "value": 49.0129508972168
        },
        {
          "time": "2022-02-03T12:56:30.83Z",
          "value": 51.59027099609375
        },
        {
          "time": "2022-02-03T12:56:15.816Z",
          "value": 54.163536071777344
        },
        {
          "time": "2022-02-03T12:56:00.789Z",
          "value": 56.72573471069336
        }
      ]
    }
  ]
```

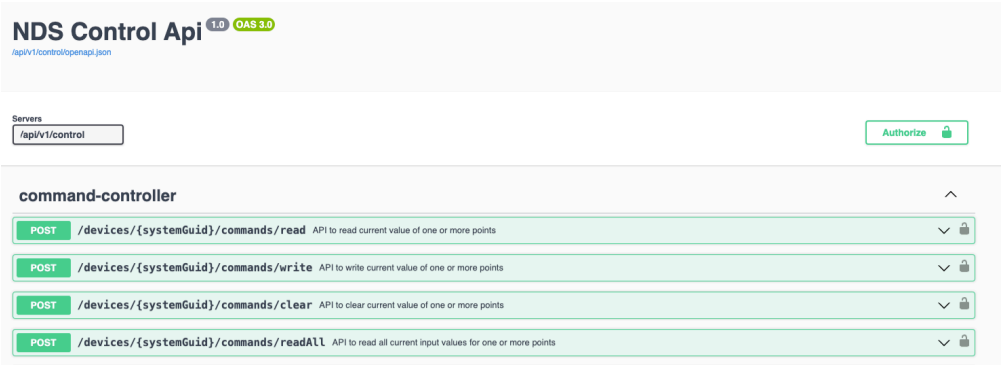
```
  }
}
```

References

Swagger link: https://www.niagara-cloud.com/api/v1/egress/swagger-ui/index.html

Chapter 4. NDS Control API support

In the context of data transport to the cloud, the Niagara Data Service (NDS) Control API allows you to communicate directly to the Niagara station to read, write, and clear live point values. It also provides scalable cloud storage for data, which can be queried via APIs without an onsite gateway for its export.



The NDS Control API is available at: https://www.niagara-cloud.com/api/v1/control/swagger-ui/index.html#/

You can read and write live data to Niagara in three different ways:

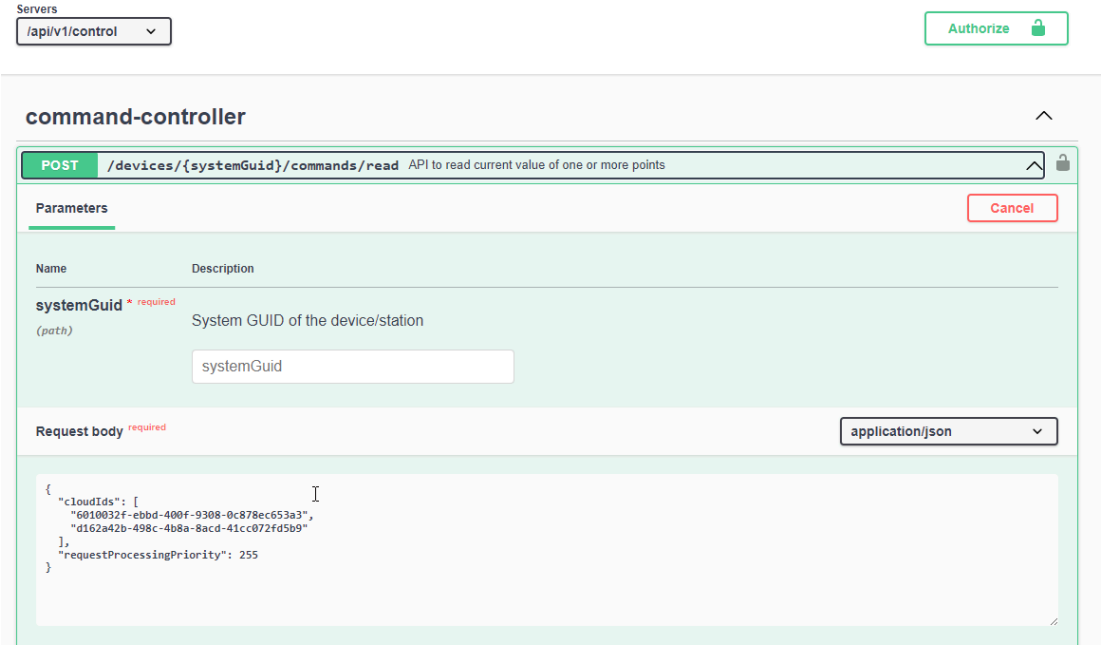
- Live read (API used to read current value of one or more points in the on-premise system)
- Live write (API used to write current value of one or more points in the on-premise system)
- Clear (API used to clear current value of one or more points)
- Live read point inputs (API used to read all current input values for one or more points)

NOTE: When commanding points from the cloud, it is important to have the Operator Role assigned to your user account. This role is necessary to ensure that you have the proper permissions and to enable seamless interaction with points through the cloud interface.

Read point values

This example shows how to send an HTTP read request to read the current value(s) of one or more points from the station to the cloud.

- URI: *https://www.niagara-cloud.com/api/v1/control/devices/{systemGuid}/commands/read*
- You performed the step to retrieve an access token (bearer authentication) to be entered for authorization. To learn more about how to obtain an access token, see "*API authentication*".
- Path parameters: *POST/devices/{systemGuid}/commands/read*



Example request

```
{
 "cloudIds": [
  "6010032f-ebbd-400f-9308-0c878ec653a3",
  "d162a42b-498c-4b8a-8acd-41cc072fd5b9"
 ],
 "requestProcessingPriority": 255
}
```

Parameters

Table 1. Path Parameter

Path Parameter	Required/ Optional	Description
systemGuid	required	Enter the system GUID of the device or station. To learn more about how to obtain the system GUID, see *"How to get a system GUID.*

Table 2. Request Body Parameter

Request Body Parameter	Required/ Optional	Description
cloudId	Required	Enter the cloudId(s) for which you want to do the point read. For more information about how to get a cloudId, see "How to get a cloudId".
requestProcessingPriority	Required	Specify the requestProcessingPriority with 1 being the highest priority and 255 being the lowest priority. You can configure the station/device to have multiple command queues. Which queue the command goes into is controlled by the priority the station/device will pull commands from the highest priority queue.

Example response


```
{
 "pointReadDetails": [
  {
   "cloudId": "6010032f-ebbd-400f-9308-0c878ec653a3",
   "value": "38.59",
   "status": "{ok} @ 10"
  },
  {
   "cloudId": "d162a42b-498c-4b8a-8acd-41cc072fd5b9",
   "value": "114.33",
   "status": "{ok} @ 5"
  }
 ]
}
```


Return values

Return value	Type	Description
cloudId	string	Repeats the cloudId(s) for which you requested the point read.
value	integer	Returns the point value on the respective cloudId at the time of reading.
status	string	Returns the status of each cloudId at the time of reading.

Status codes

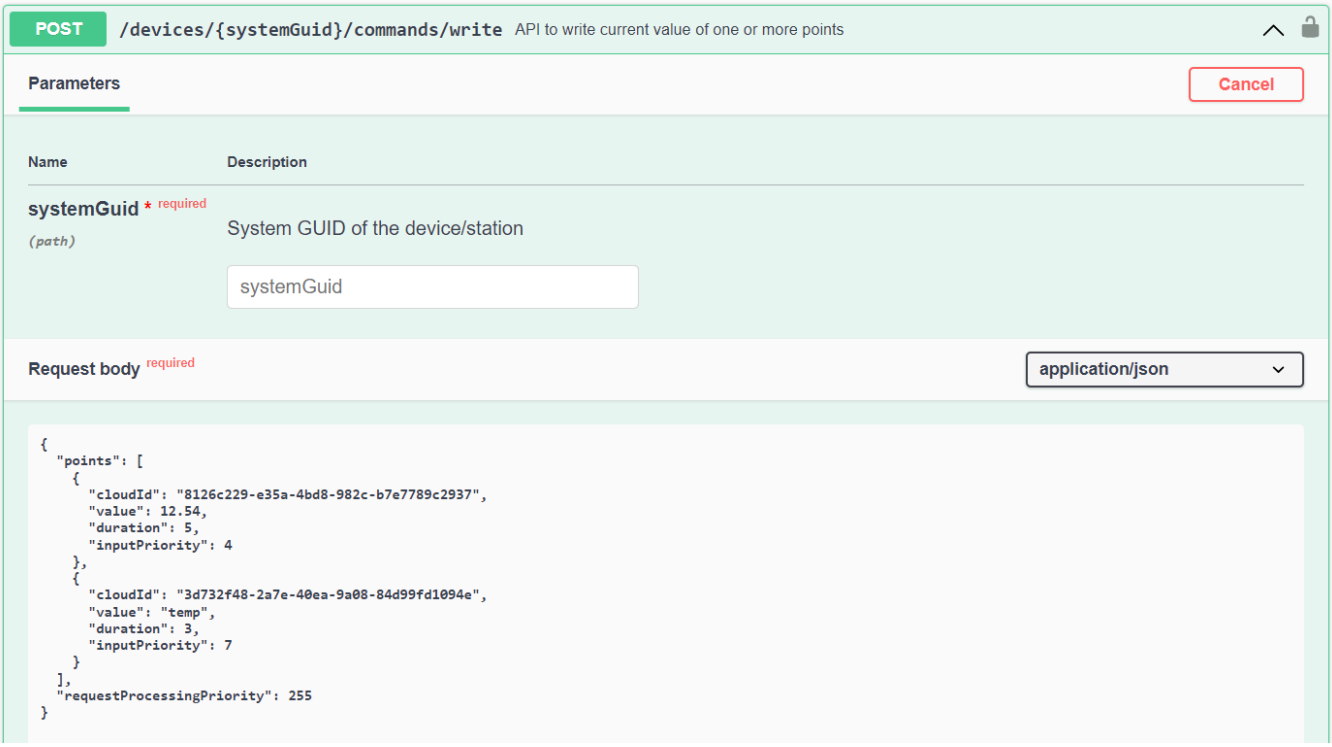
Code	Description
200	Points successfully retrieved.
400	No payload. Bad payload.
401	Unauthorized
403	Subject does not have permissions.
406	Requested media type specified in the Accept header is not supported.
415	Media type specified in the Content-Type header is not supported.
500	Internal drop of the chalupa.

Write point values

This example shows how to send an HTTP write request to write the current value of one or more points from the station to the cloud.

Prerequisite

- URI: *https://www.niagara-cloud.com/api/v1/control/devices/{systemGuid}/commands/write*
- You performed the step to retrieve an access token (bearer authentication) to be entered for authorization. To learn more about how to obtain an access token, see “API authentication”.
- Path parameters: *POST/devices/{systemGuid}/commands/write*



Example request

```
{
 "points": [
  {
   "cloudId": "8126c229-e35a-4bd8-982c-b7e7789c2937",
   "value": 12.54,
   "duration": 5,
   "inputPriority": 4
  },
  {
   "cloudId": "3d732f48-2a7e-40ea-9a08-84d99fd1094e",
   "value": "temp",
   "duration": 3,
   "inputPriority": 7
  }
 ],
 "requestProcessingPriority": 255
}
```

Parameters

Table 3. Path Parameter

Path Parameter	Required/ Optional	Description
systemGuid	required	Enter the system GUID of the device or station. To learn more about how to obtain the system GUID, see "How to get a system GUID".

Table 4. Request Body Parameter

Request Body Parameter	Required/ Optional	Type	Description

cloudId	Required	string	Enter the cloudId(s) for which you want to do the point read. For more information about how to get a cloudId, see “How to get a cloudId”.
value	Required	integer	Specify the value that you request to write of one or more points on the respective cloudId.
duration	Required	integer (defaults to 15 minutes),	Specify the time duration in minutes for which the value needs to be set.
inputPriority	Required	integer (defaults to 17/ fallback)	Specify the input priority of the desired point which ranges between 1 (minimum) and 17 (maximum).
requestProcessingPriority	Required	integer (defaults to 255)	Specify the requestProcessingPriority with 1 being the highest priority and 255 being the lowest priority. You can configure the station/device to have multiple command queues. Which queue the command goes into is controlled by the priority the station/device pulls commands from.

Example response


```
{
  "points": [
    {
      "cloudId": "8126c229-e35a-4bd8-982c-b7e7789c2937",
      "value": 12.54,
      "duration": 5,
      "inputPriority": 4
    },
    {
      "cloudId": "3d732f48-2a7e-40ea-9a08-84d99fd1094e",
      "value": "temp",
      "duration": 3,
      "inputPriority": 7
    }
  ],
  "requestProcessingPriority": 255
}
```


Return values

Return value	Type	Description
cloudId	string	Repeats the cloudId(s) for which you requested the point read.
value	integer	Returns the point value on the respective cloudId at time of reading.
status	string	Returns the point status on the respective cloudId at time of reading.

Status codes

Code	Description
200	Point write value sent successfully.
400	Bad request
401	Unauthorized, due to one of the following reasons: • Access token not supplied • Invalid access token • Access token expired
403	Access forbidden, due to on the following reasons: • Invalid input • Caller does not have the necessary role binding on the requested resource.

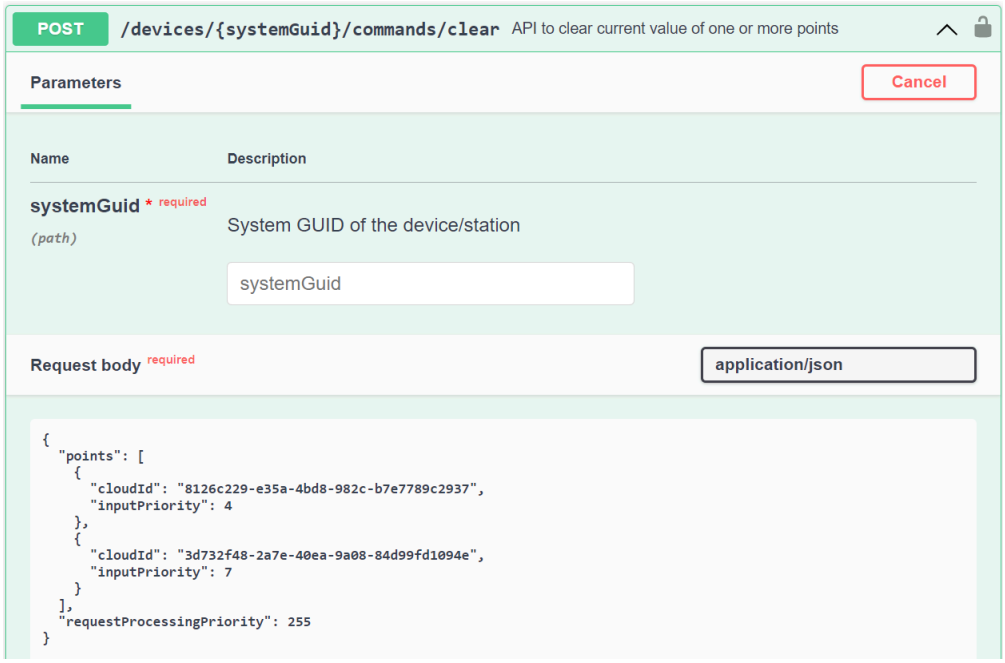
404	Page not found.
415	Unsupported media type
500	Internal server error
502	Bad gateway server error
504	Timed out

Clear point values

This example shows how to clear and release the value of a point at a particular input priority level if the point was written using a cloud write API.

Prerequisite

- URI: *https://www.niagara-cloud.com/api/v1/control/devices/{systemGuid}/commands/clear*
- You performed the step to retrieve an access token (bearer authentication) to be entered for authorization. To learn more about how to obtain an access token, see "*API authentication*".
- Path parameters: *POST/devices/{systemGuid}/commands/clear*



Example request

```
{
 "points": [
  {
   "cloudId": "8126c229-e35a-4bd8-982c-b7e7789c2937",
   "inputPriority": 4
  },
  {
   "cloudId": "3d732f48-2a7e-40ea-9a08-84d99fd1094e",
   "inputPriority": 7
  }
 ],
 "requestProcessingPriority": 255
}
```

Parameters

Table 5. Path Parameter

Path Parameter	Required/ Optional	Description
systemGuid	required	Enter the system GUID of the device or station. To learn more about how to obtain the system GUID, see "How to get a system GUID".

Table 6. Request Body Parameter

Request Body Parameter	Required/ Optional	Type	Description
cloudId	Required	string	Enter the cloudId(s) for which you want to do the point read. For more information about how to get a cloudId, see "How to get a cloudId".
inputPriority	Required	integer (defaults to 17/ fallback)	Specify the input priority of the desired point which ranges between 1 (minimum) and 17 (maximum).
requestProcessingPriority	Required	integer (defaults to 255)	Specify the requestProcessingPriority with 1 being the highest priority and 255 being the lowest priority. You can configure the station/device to have multiple command queues. Which queue the command goes into is controlled by the priority the station/device pulls commands from.

Example response


```
{
  "pointClearDetails": [
    {
      "cloudId": "8126c229-e35a-4bd8-982c-b7e7789c2937",
      "status": "success",
      "writeTime": "2024-02-26T10:44:15.535Z"
    },
    {
      "cloudId": "3d732f48-2a7e-40ea-9a08-84d99fd1094e",
      "status": "success",
      "writeTime": "2024-02-26T10:44:15.541Z"
    }
  ]
}
```


Return values

Return value	Type	Description
cloudId	string	Repeats the cloudId(s) for which you requested the point read.
status	string	Returns the point status on the respective cloudId at time of reading.
writeTime	string	Returns the point of time of when the point clear was done.

Status codes

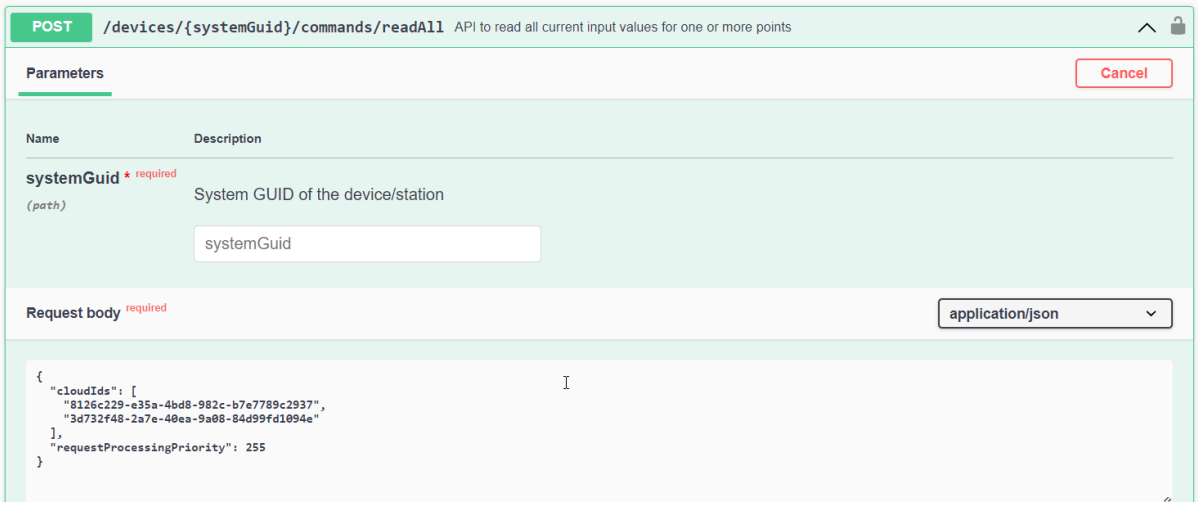
Code	Description
200	Point values cleared successfully
400	Bad request
401	Unauthorized, due to one of the following reasons: • Access token not supplied • Invalid access token • Access token expired
403	Access forbidden, due to on the following reasons: • Invalid input

	• Caller does not have the necessary role binding on the requested resource.
404	Page not found.
415	Unsupported media type
500	Internal server error
502	Bad gateway server error
504	Timed out

Read point inputs

This example shows how to send an HTTP readAll request to read all current input values of one or more points from the station to the cloud.

- URI: *https://www.niagara-cloud.com/api/v1/control/devices/{systemGuid}/commands/readAll*
- You performed the step to retrieve an access token (bearer authentication) to be entered for authorization. To learn more about how to obtain an access token, see "*API authentication*".
- Path parameters: *POST/devices/{systemGuid}/commands/readAll*



Example request

```
{
 "cloudIds": [
  "8126c229-e35a-4bd8-982c-b7e7789c2937",
  "3d732f48-2a7e-40ea-9a08-84d99fd1094e"
 ],
 "requestProcessingPriority": 255
}
```

Parameters

Table 7. Path Parameter

Path Parameter	Required/ Optional	Description
systemGuid	required	Enter the system GUID of the device or station. To learn more about how to obtain the system GUID, see "How to get a system GUID".

Table 8. Request Body Parameter

Request Body Parameter	Required/ Optional	Type	Description

cloudId	Required	string	Enter the cloudId(s) for which you want to do the point read. For more information about how to get a cloudId, see "*How to get a cloudId*".
requestProcessingPriority	Required	integer	Specify the requestProcessingPriority with 1 being the highest priority and 255 being the lowest priority. You can configure the station/device to have multiple command queues. Which queue the command goes into is controlled by the priority the station/device pulls commands from.

Example response


```
{
 "pointReadAllDetails": [
  {
   "cloudId": "8126c229-e35a-4bd8-982c-b7e7789c2937",
   "inputValues": [
    {
     "value": "0.00",
     "inputPriority": "1",
     "status": "{null}",
     "sourceId": ""
    },
    {
     "value": "0.00",
     "inputPriority": "2",
     "status": "{null}",
     "sourceId": ""
    {
     "value": "31.77",
     "inputPriority": "3",
     "status": "{ok}",
     "sourceId": "6654a3e6-d8fb-4cc1-a1a4-f652827269fa"
    },
   ]
  },
  {
   "cloudId": "3d732f48-2a7e-40ea-9a08-84d99fd1094e",
   "inputValues": [
    {
     "value": "temp",
     "inputPriority": "1",
     "status": "{ok}",
     "sourceId": ""
    },
    {
     "value": "",
     "inputPriority": "2",
     "status": "{null}",
     "sourceId": ""
    },
    {
     "value": "",
     "inputPriority": "3",
     "status": "{null}",
     "sourceId": ""
    },
    {
   ]
  }
 ]
}
```


Return values

Return value	Type	Description

cloudId	string	Repeats the cloudId(s) for which you called the point read.
inputValue	string	Returns cloudIds with arrays of input property values such as inputPriority, status, sourceId.
inputPriority	integer (defaults to 17/fallback)	Specifies the input priority of each cloudId, which ranges between 1 (minimum) and 17 (maximum).
status	string	Returns the point status on the respective cloudId at time of reading.
sourceId	string	Specifies the cloudId of the component to which the input priority slot is linked.

Status codes

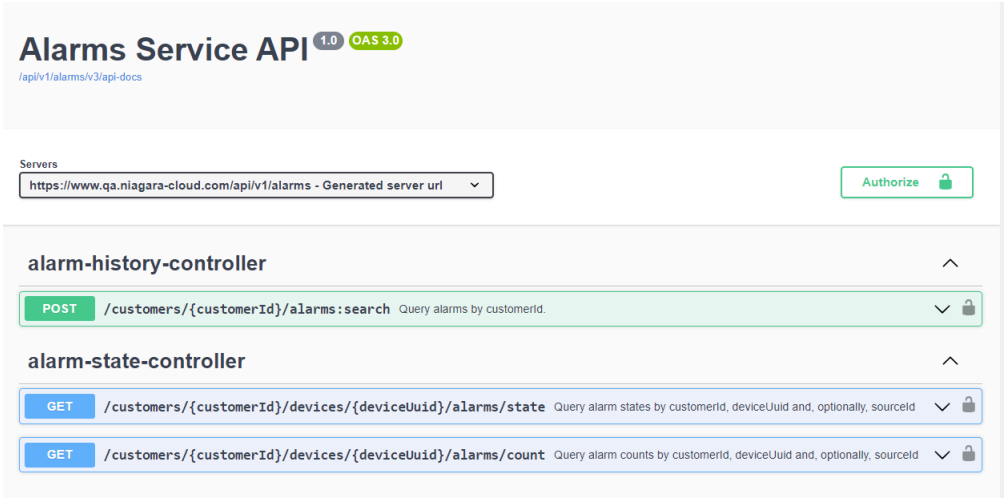
Code	Description
200	All point input values retrieved successfully
400	Bad request
401	Unauthorized, due to one of the following reasons: • Access token not supplied • Invalid access token • Access token expired
403	Access forbidden, due to on the following reasons: • Invalid input • Caller does not have the necessary role binding on the requested resource.
404	Page not found.
415	Unsupported media type
500	Internal server error
502	Bad gateway server error
504	Timed out

Chapter 5. Alarms Service API support

The Alarms Service API allows you to query the alarms database for alarm histories, alarm states, and alarm counts.
The Alarms Service API is available at: https://www.niagara-cloud.com/api/v1/alarms/swagger-ui/index.html.

- Alarm histories: Provides access to all the alarm records uploaded to the cloud, including those that may no longer exist on the originating stations due to the limited space on a device.
- Alarm states: Displays the last reported alarm record values for each individual alarm source.
- Alarm counts: Reports the number of alarm records, grouped by acknowledgment state (acked, unacked, ack pending) that the device reported last.

  NOTE: The states and counts APIs reflect the information on the device as close to real time as possible.



API Endpoints

Controller	Method	Endpoint
Alarm History Controller	POST	/customers/{customerId}/alarms:search
Alarm State Controller	GET	/customers/{customerId}/devices/{deviceUuid}/alarms/state
Alarm State Controller	GET	/customers/{customerId}/devices/{deviceUuid}/alarms/count

API details

Authentication

Perform the step to retrieve an access token (bearer authentication) to access the API. To learn more about how to obtain an access token, see "*API authentication*".

When you sign up with Tridium as a developer partner of Niagara Data Service, you will request service credentials from Tridium. A service account will be set up for you with access to a customer, and you will receive the client ID and secret through a secure channel. These can be used with the client credentials OAuth flow to obtain an access token valid for this API.

Schema

Alarm Record

The fields below are present on every (Cloud) alarm record. They are called primary fields, or primary attributes.

Field	Type	Source
deviceUuid	UUID	IoT message
alarmUuid	UUID	BAlarmRecord.uuid
iotMessageId	UUID	IoT message
cloudId	text	IoT message
generatedTime	DateTime	BAlarmRecord.timestamp
sourceState	smallint or enum	BAlarmRecord.sourceState
ackState	smallint or enum	BAlarmRecord.ackState
ackRequired	boolean	BAlarmRecord.ackRequired
source	text	BAlarmRecord.source
alarmClass	text	BAlarmRecord.alarmClass
priority	smallint	BAlarmRecord.priority
normalTime	DateTime	BAlarmRecord.normalTime
ackTime	DateTime	BAlarmRecord.ackTime
userName	text	BAlarmRecord.user
alarmTransition	smallint or enum	BAlarmRecord.alarmTransition
lastUpdateTime	DateTime	BAlarmRecord.lastUpdate
msgText	text	AlarmData

Alarm Data

Besides primary properties, custom properties can be supplied by a station using properties specific to each individual alarm source.

Examples of custom properties: **`notes`**, **`alarmValue`**, **`presentValue`**, **`offNormalValue`**, **`deadband`**, **`timeDelay`**, **`timeDelayToNormal`**, **`highLimit`**, **`lowLimit`**, **`fromState`**, **`toState`**, **`escalated`**, **`ordList`**, **`Count`**, **`TimeZone`**.

These fields are referenced with the `alarmData.` qualifier (prefix), for example `alarmData.notes`. They are stored separately.

Alarm Count

A separate table stores the alarm counts for devices and sources. There is one row per device and one row per source, which are updated when alarm count events are received.

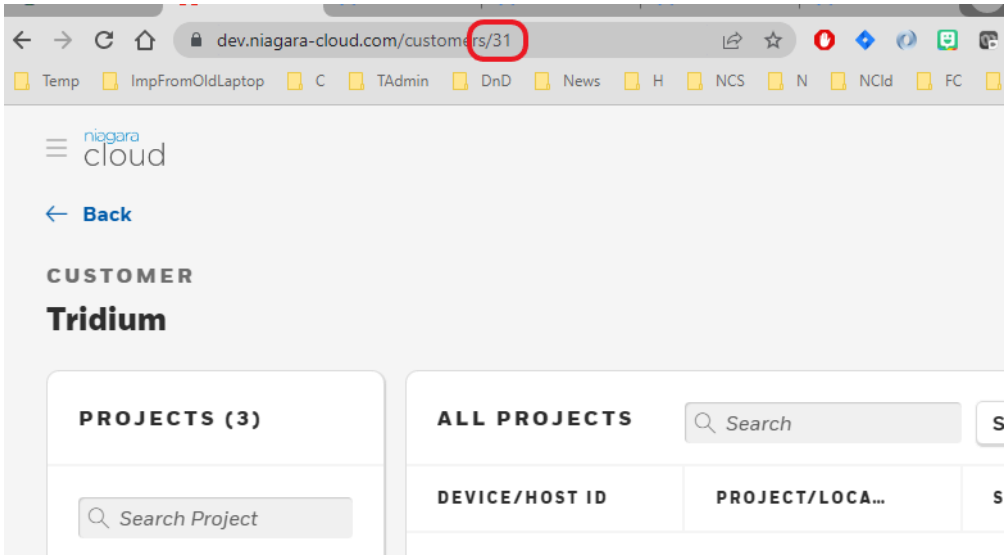
Field	Type	Notes
deviceUuid	UUID	
cloudId	text	Equal to the deviceUuid, when the count is aggregated for the device.
iotMessageId	UUID	For auditing, tracing, consistency remediation
unacked	numeric	
acked	numeric	
ackPending	numeric	
total	numeric	
lastUpdateTime	DateTime	The local system time when the alarm count was generated by the station.

Inputs

Customer ID

This ID must be supplied as a path parameter called customerId .

You can obtain your customerId by navigating to the Niagara Cloud Management Portal at *https://www.niagara-cloud.com* and selecting a customer. At the end of the URL you can see the customerId:



Device UUID

The device UUID is the same as the system ID contained in the **FederatedIdentityAuthenticator** component. It is a unique ID assigned to a device at the time of the federated identity registration with NCP.

Search request

An Alarm History API request contains 3 sections:

- Search query: a general (nested) logical expression (required)
- Pagination parameters: page size and page number (optional; default values are page number 0 of size 100)
- Sort orders (optional)

Both requests and responses are in the JSON format.

Alarm search request example

```
{
  "query": {
    "op": "and",
    "expr": [
      {
        "field": "deviceUuid",
        "op": "eq",
        "value": "f36f1268-0d4d-43d2-97fa-6c82b292ee72"
      },
      {
        "field": "generatedTime",
        "op": "gt",
        "value": "2024-07-24T00:00:00Z"
      },
      {
        "field": "alarmData.toState",
        "op": "in",
        "values": ["Offnormal", "Fault"]
      }
```

```
  ]
 },
 "page": {
  "size": 10,
  "number": 0
 },
 "sort": {
  "order": [
   {
    "by": "lastUpdateTime",
    "direction": "desc"
   }
  ]
 }
}
```

Search query

Expressions:

- Logical: {"op": "logical_op", "expr": [expr1, expr2, ...]}
- Relational:
  - {"field": "name", "op": "rel_op", "value": "v"}
  - {"field": "name", "op": "in", "values": [v1, v2, ...]}

Operators:

- Logical: "and", "or", "not"
- Relational: "eq", "neq", "lt", "gt", "le", "ge"
- String comparators: "contains", "startswith", "endswith", "like"
- Set membership: "in"

Value types:

- numeric: integer, long, double
- string: text, UUID
- boolean
- enum
- timestamp

Page request

Default page size: 100

Maximum page size: 10,000

Sort orders

A sort order is defined by the following pair: sorted field and the sort direction.

- "by": any primary alarm field name (the custom attributes, those prefixed with `alarmData.` are not sortable)
- "direction": asc, desc

Multiple sort orders are processed in the order they are specified.

The smallest alarm history request

```
{
 "query": {
```

```
  "op": "and",
  "expr": [ ]
 }
}
```

This returns one standard page (default size) of unfiltered and unsorted alarms for all devices that the caller identified by the JWT is authorized to access.

NOTE: In practice, it is not recommended to run this query because it may yield more results than you intended. For most purposes, you would limit the query using a combination of operators as shown in the examples.

Source ID

For the Alarm State and Alarm Count APIs, an additional URL parameter (sourceId) can be specified. URL parameters are appended to the URL using the syntax `?sourceId=value`.

The source ID is the cloud ID of an alarm source if this is available, or otherwise another identifying string, for example the station ord.

When a source ID is specified, the output contains the single record corresponding to that alarm source.

Outputs

Alarm history

Example Alarm History API response


```
{
 "_embedded": {
  "alarms": [
   {
    "deviceUuid": "19d7c8a8-2b96-4499-8fc6-d5ef5dfde7ea",
    "alarmUuid": "ebc8ef43-cbfd-426a-9321-6733056fc654",
    "iotMessageId": "fbafdcb8-18b9-4e96-9032-fc84fc20caff",
    "cloudId": "a4242ce0-eefe-4a0b-8b5a-67a115e6fce2",
    "generatedTime": "2024-07-24T18:06:50.043Z",
    "sourceState": "highlimit",
    "ackState": "unacked",
    "ackRequired": true,
    "source": "NumericWritable9",
    "alarmClass": "defaultAlarmClass",
    "priority": 255,
    "normalTime": "2024-07-24T18:07:17.611Z",
    "ackTime": "1970-01-01T00:00:00Z",
    "userName": "Unknown User",
    "msgText": "To Normal",
    "alarmTransition": "offnormal",
    "lastUpdateTime": "2024-07-24T18:07:17.612Z",
    "alarmData": {
     "fromState": "highLimit",
     "lowLimit": "2.0 cm²",
     "SystemGuid": "f9f9e605-04c8-4a83-bf02-94d562d81e43",
     "AlarmState": "Normal",
     "escalated": "",
     "SystemType": "pgsdemoniagara-cloud-customer-project-qa",
     "Count": 130106,
     "presentValue": "2.5 cm²",
     "TimeZone": "Asia/Calcutta (+5:30)",
```

```
        "toState": "normal",
        "alarmValue": 14,
        "offnormalValue": "13.7 cm²",
        "deadband": "1.0 cm²",
        "timeDelay": "0 ms",
        "highLimit": "10.0 cm²",
        "ordList": "local:|station:|slot:/Playground/NumericWritable9/OutOfRangeAlarmExt",
        "timeDelayToNormal": "0 ms",
        "status": "{ok} @ 10"
      }
    },
    ...
    {
      "deviceUuid": "19d7c8a8-2b96-4499-8fc6-d5ef5dfde7ea",
      "alarmUuid": "0790ca81-4a07-4010-9922-ebf1fe40b215",
      "iotMessageId": "1d0b4445-fa72-4145-b8a7-d1d20cdeebfb",
      "cloudId": "94893bd5-361f-4f55-880f-95b57bcad85c",
      "generatedTime": "2024-07-24T18:06:20.554Z",
      "sourceState": "highlimit",
      "ackState": "unacked",
      "ackRequired": true,
      "source": "test1",
      "alarmClass": "defaultAlarmClass",
      "priority": 255,
      "normalTime": "2024-07-24T18:06:46.978Z",
      "ackTime": "1970-01-01T00:00:00Z",
      "userName": "Unknown User",
      "msgText": "To Normal",
      "alarmTransition": "offnormal",
      "lastUpdateTime": "2024-07-24T18:06:46.979Z",
      "alarmData": {
        "fromState": "highLimit",
        "lowLimit": 2,
        "SystemGuid": "f9f9e605-04c8-4a83-bf02-94d562d81e43",
        "AlarmState": "Normal",
        "escalated": "",
        "SystemType": "pgsdemoniagara-cloud-customer-project-qa",
        "Count": 285212,
        "presentValue": 7,
        "TimeZone": "Asia/Calcutta (+5:30)",
        "toState": "normal",
        "alarmValue": 17,
        "offnormalValue": 17,
        "deadband": 1,
        "timeDelay": "0 ms",
        "highLimit": 10,
        "ordList": "local:|station:|slot:/Playground/test$3d$3d$3d$3d1/OutOfRangeAlarmExt",
        "timeDelayToNormal": "0 ms",
        "status": "{ok} @ 10"
      }
    }
  ]
},
"_links": {
  "first": {
    "href": "https://www.qa.niagara-cloud.com/api/v1/alarms/customers/5041237/alarms:search?page=0&size=10&sort=la
  },
```

```
  "self": {
    "href": "https://www.qa.niagara-cloud.com/api/v1/alarms/customers/5041237/alarms:search?page=0&size=10&sort=la
  },
  "next": {
    "href": "https://www.qa.niagara-cloud.com/api/v1/alarms/customers/5041237/alarms:search?page=1&size=10&sort=la
  },
  "last": {
    "href": "https://www.qa.niagara-cloud.com/api/v1/alarms/customers/5041237/alarms:search?page=95&size=10&sort=l
  }
 },
 "page": {
  "size": 10,
  "totalElements": 956,
  "totalPages": 96,
  "number": 0
 }
}
```


Alarm state

Similar to the alarm history output, the alarm state response is a list of alarm records, but not paginated and without the additional custom attributes. Each entry represents the last reported state of an alarm source identified by its unique cloud ID.

Example Alarm State API response


```
[
 {
  "deviceUuid": "bcd69c88-da80-4630-847d-4eb49a60646e",
  "alarmUuid": "b8b00f7a-a81e-44fd-acfd-8e7a36eec667",
  "iotMessageId": "638df5d8-f888-4c1d-bef8-932c9319874b",
  "cloudId": "local:|station:|slot:/Playground/point1/BooleanChangeOfStateAlarmExt",
  "generatedTime": "2024-06-23T15:59:48.258Z",
  "sourceState": "offnormal",
  "ackState": "unacked",
  "ackRequired": true,
  "source": "BooleanWritable1",
  "alarmClass": "defaultAlarmClass",
  "priority": 255,
  "normalTime": "2024-06-23T16:02:18.767Z",
  "ackTime": "1970-01-01T00:00:00Z",
  "userName": "you, sir",
  "msgText": null,
  "alarmTransition": "offnormal",
  "lastUpdateTime": "2024-06-23T16:02:18.774Z"
 },
 ...
 {
  "deviceUuid": "bcd69c88-da80-4630-847d-4eb49a60646e",
  "alarmUuid": "d25adc0e-67cb-4b17-aacc-e1beb443dc1c",
  "iotMessageId": "e0322a33-aa50-4c25-9f70-0cd17ff09978",
  "cloudId": "9e8d018c-b478-4ef7-a26d-3fbd72068bd5",
  "generatedTime": "2024-07-24T18:40:11.106Z",
  "sourceState": "offnormal",
  "ackState": "unacked",
  "ackRequired": true,
  "source": "EnumWritable",
```

```
  "alarmClass": "defaultAlarmClass",
  "priority": 255,
  "normalTime": "2024-07-24T18:41:11.123Z",
  "ackTime": "1970-01-01T00:00:00Z",
  "userName": "Unknown User",
  "msgText": "normal time",
  "alarmTransition": "offnormal",
  "lastUpdateTime": "2024-07-24T18:41:11.128Z"
 }
]
```


Alarm count

Example Alarm Count API response


```
[
 {
  "deviceUuid": "bcd69c88-da80-4630-847d-4eb49a60646e",
  "cloudId": "bfec9c82-7098-4a09-9843-939fe691ea34",
  "iotMessageId": "53b076af-c178-4c7d-9d59-e8bbc2abfcab",
  "acked": 0,
  "unacked": 157,
  "ackPending": 0,
  "total": 157,
  "lastUpdateTime": "2024-07-16T12:33:22.677Z"
 },
 {
  "deviceUuid": "bcd69c88-da80-4630-847d-4eb49a60646e",
  "cloudId": "bcd69c88-da80-4630-847d-4eb49a60646e",
  "iotMessageId": "53b076af-c178-4c7d-9d59-e8bbc2abfcab",
  "acked": 0,
  "unacked": 500,
  "ackPending": 0,
  "total": 500,
  "lastUpdateTime": "2024-07-16T12:33:22.677Z"
 }
]
```

Chapter 6. Glossary

The following glossary entries relate specifically to the topics that are included as part of this document.

To find more glossary terms and definitions refer to glossaries in other individual documents.

Alphabetical listing

API

Application Programming Interfaces open a company's applications' data and functionality to other in-house developers, external, third-party developers, and business partners. APIs allow services and products to communicate, leveraging the functions provided by each. (IBM)

Data container (EMS API)

A container that contains all customer's systems and their points. Each point is an item in the container. An item contains the component information (for example, slots, tags, relations) and metadata about the model upload it came from, for example, device systemGuid.

Entity Model Service (EMS API)

A service that maps the customer name to a customer Id. The customer name is unconstrained but maps to an internal customer Id generated when the customer is onboarded. Customers have their own separate data containers and cannot access data from another customer's data container.

GUID

Globally Unique Identifier also known as a UUID or Universally Unique Identifier, a 128-bit unique reference number that is highly unlikely to repeat.The device UUID is the same as the system ID contained in the **FederatedIdentityAuthenticator** component.

SystemGuid (EMS API)

An identifier for the device assigned upon device registration with a cloud platform. It is a GUID provided to the client registering the device, and also provided to the device upon successful authentication.