Juniper Networks | Engineering Simplicity

Juniper Mist Automation Guide

Published
2026-01-20

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Juniper Mist Automation Guide

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at https://support.juniper.net/support/eula/. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

1

CHAPTER

Overview

IN THIS CHAPTER

Introduction to Juniper Mist Automation

SUMMARY

Start getting familiar with the automation and integration features of Juniper Mist™.

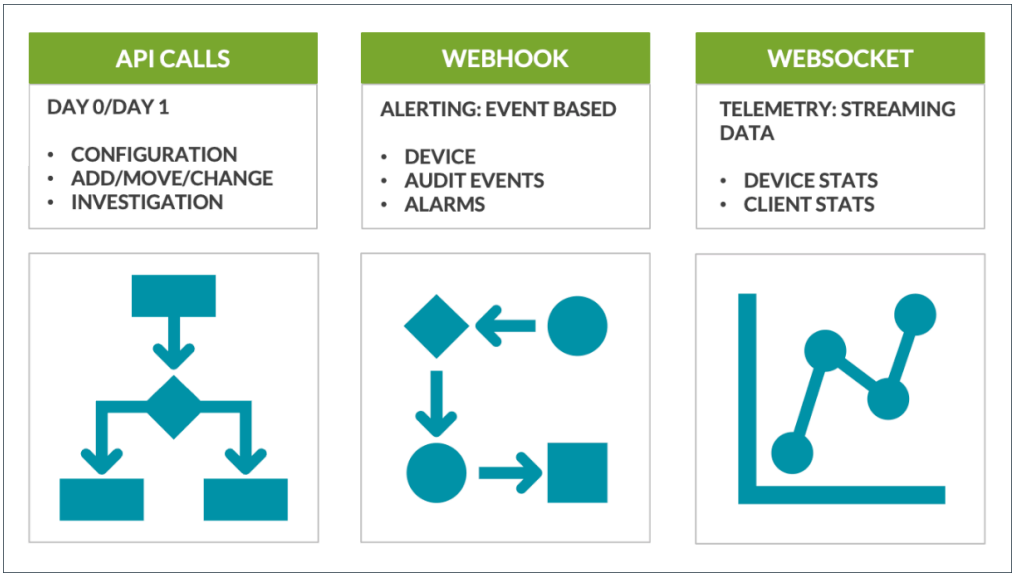
You can automate many Juniper Mist processes by using the RESTful API, webhooks, and the WebSocket API.

As a network administrator, you can use this guide to get familiar with essential concepts and available tools. You'll also walk through several use cases that illustrate the impact of automation.

Get started with these overviews:

- "RESTful API Overview" on page 4
- "Webhooks Overview" on page 174
- "WebSocket API Overview" on page 242

As shown below, you can interact with Mist using the APIs in multiple ways.



2

CHAPTER

REST API

IN THIS CHAPTER

RESTful API Overview

SUMMARY

Get familiar with Representational State Transfer (REST) and understand how you can use RESTful APIs with Juniper Mist™.

IN THIS SECTION

The 100% API architecture of Juniper Mist backs every visible feature in the Juniper Mist portal. Anything that you can do in the portal, you can automate at scale by using the API. Representational State Transfer (REST) is a stateless client/server architecture with a uniform interface. Since machines have no use for a user interface, APIs allow for a defined and faster way for machines to communicate with each other.

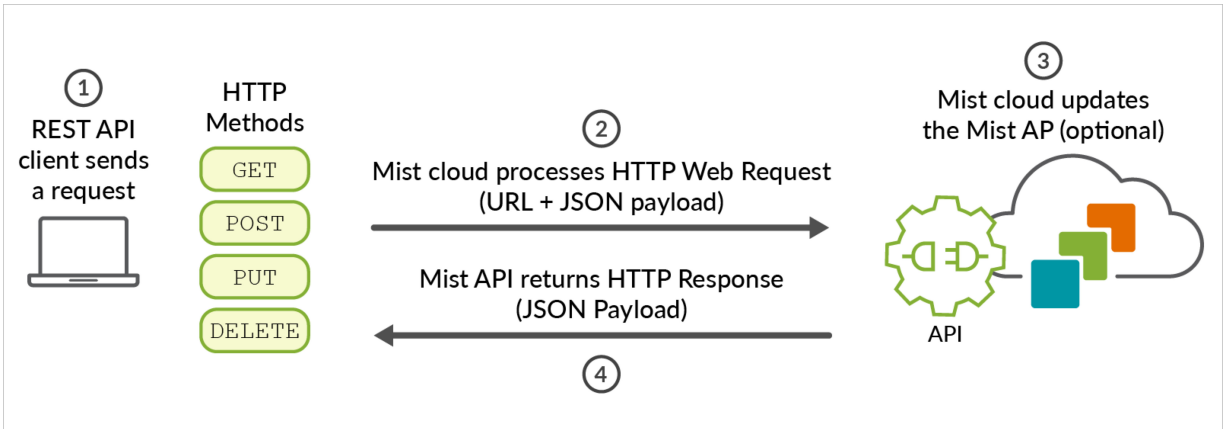
REST APIs enable you to create your own way of interacting with systems and applications. You can even create custom features. Other common use cases for REST APIs include communication and data exchange between applications, data exchange between applications and servers, communication between microservices within an application, and more. REST is stateless, which makes it ideal for cloud-based services.

The Juniper Mist API is available to any customer with a Juniper Mist account.

Also see the Mist API Reference. This contains additional documentation for developers, as well as the ability to test API calls.

Juniper Mist API Architecture

Juniper Mist uses REST APIs, which use HTTP methods (GET, POST, PUT, and DELETE) to transfer data in JavaScript Object Notation (JSON) format.



RESTful API Requests

Using RESTful APIs follows a similar practice to the CRUD (CREATE, READ, UPDATE, DELETE) methodology used in development. These are the four basic actions or functions used when working with data.

Table 1: Basic CRUD Actions

CRUD	HTTP/REST
Create	POST
Read	GET
Update	PUT
Delete	DELETE

API Endpoint URL Format

The API endpoint URL has two parts:

- API Host (or Endpoint)—The endpoint for the global region that your Juniper Mist organization is associated with. These endpoints are listed in "API Endpoints and Global Regions" on page 13.
- Function—Everything after the API endpoint represents the function that the API will call.

Example

```
https://{api-host}/api/v1/sites/{site_id}/stats/devices/{device_id}.
```

```
https://api.mist.com/api/v1/sites/13b0ee00-121a-456e-84e0-ead3008bc2f2/stats/devices/00000000-0000-0000-1000-d420b08532eb
```

NOTE: When reusing code blocks, replace placeholder values with actual values, such as your API token, organization ID, site ID, AP name, and so on.

Everything after **{api-host}** is the function. The call goes to the global cloud and requests the statistics for the specified device at the specified site.

The next section takes a deeper look at the structure that makes up an API call.

API Call Structure

The following image is an example of an API call and the different components that make it up.

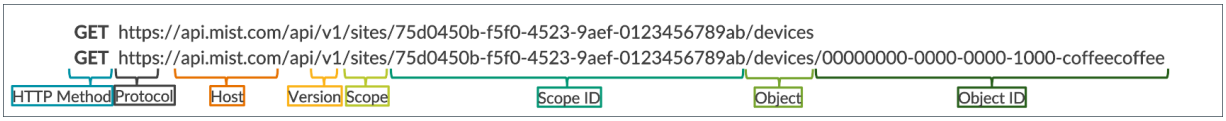


Table 2: API Call Components

API Call Component	Description

HTTP Method	• POST—Create an object.—POST overwrites any existing values with those contained in the payload. Values that are not specified in the POST payload are reverted to their original values.
	• GET—List objects.—GET returns the value of a resource or a list of resources, depending on whether an identifier is specified.
	  • `GET /api/v1/orgs/:org_id`
	  returns information about the organization based on the specified *:org_id*.
	  • `GET /api/v1/orgs/:org_id/site`
	  returns a list of sites belonging to the *:org_id*.
	  • `GET /api/v1/sites/:site_id`
	  returns information about the site specified by the *:site_id*.
	• PUT—Update an object.—PUT modifies all specified values in the payload. When updating with a PUT, simple values (strings or numbers) not specified in the payload keep their existing values. If the value contains a data structure such as an array or an object, the values included in the payload will replace that structure in its entirety. Keep this in mind to avoid unwanted changes to existing values.
	• DELETE—Remove an object.—DELETE removes a resource.

Host (or API Endpoint)	Determines the Mist Cloud to use (Global 01, EMEA 01, etc). The endpoint for the global region that your Juniper Mist organization is associated with. See "API Endpoints and Global Regions" on page 13.
Version	The API version to use (currently, all APIs use v1).

Table 2: API Call Components *(Continued)*

API Call Component	Description
Scope	Indicates the level that the request is being done at. Examples include msp, org, site, self, register, installer, const, and so on.
Scope ID	Identifies the scope to use.
Object	The type of object to use (Device, WLAN, and so on).
Object ID	Identifies the object to request.

To perform any of the above REST commands (POST, GET, PUT, DELETE) on the REST API, you need to fulfill a few requirements in each request, such as:

- Authentication:
  - You can use an API token, Juniper Mist login credentials (this authentication type to be deprecated September 2026), or an external OAuth2 provider to indicate who you are and what you have access to during the authentication process.
  - For more detail on the various authentication methods, see Authentication.

NOTE: If you are already logged in on manage.mist.com, you can simply open a new browser tab and go to https://api.mist.com/api/v1/self/apitokens and click the **POST** button. This will automatically create a new api user token.
See "Create API Tokens" on page 15 for more information about tokens.

- HTTP Header: This header specifies the content and the authorization type, as follows:
  - For Juniper Mist, the content type is always application/json.
  - The authorization can be a token or a cookie (including CSRF token and session ID).
- The endpoint for the global region that your Juniper Mist organization is associated with. See "API Endpoints and Global Regions" on page 13.
- "JSON Payload" on page 9

The following table provides examples for the different parts that make up a RESTful API request.

Table 3: RESTful API Request examples

CRUD Operation	HTTP Header Authentication	Endpoint URL	Payload (JSON)
GET	API Token	https://api.mist.com/api/v1/sites/:site_id/wlans	
DELETE	CSRF Token, Session ID	https://api.mist.com/api/v1/sites/:site_id/wlans/:wlan_id	
POST	CSRF Token, Session ID	https://api.mist.com/api/v1/orgs/:org_id/inventory	{["<claim_code>"]}
PUT	API Token	https://api.mist.com/api/v1/sites/:site_id/wlans/:wlan_id	{"ssid" : "New Name"}

JSON Payload

Different functions require different elements in the JSON payload. You can view the required details in the API documentation.

The following is a sample API call and the response (JSON payload).

API call:

```
POST
/api/v1/orgs/{org_id}/rftemplates
```

Response (JSON Payload):


```
{
  "name": "new-rf-template",
  "org_id": "a97c1b22-a4e9-411e-9bfd-d8695a0f9e61",
  "band_5": {
    "allow_rrm_disable": false,
    "ant_gain": 0,
    "channels": [],
    "disabled": false,
    "power_max": 17,
    "power_min": 8
  },
  "band_24": {
    "allow_rrm_disable": false,
    "ant_gain": 0,
    "channels": [],
    "disabled": false,
    "power_max": 17,
    "power_min": 8
  },
  "country_code": "CA"
}
```


API Rate Limiting

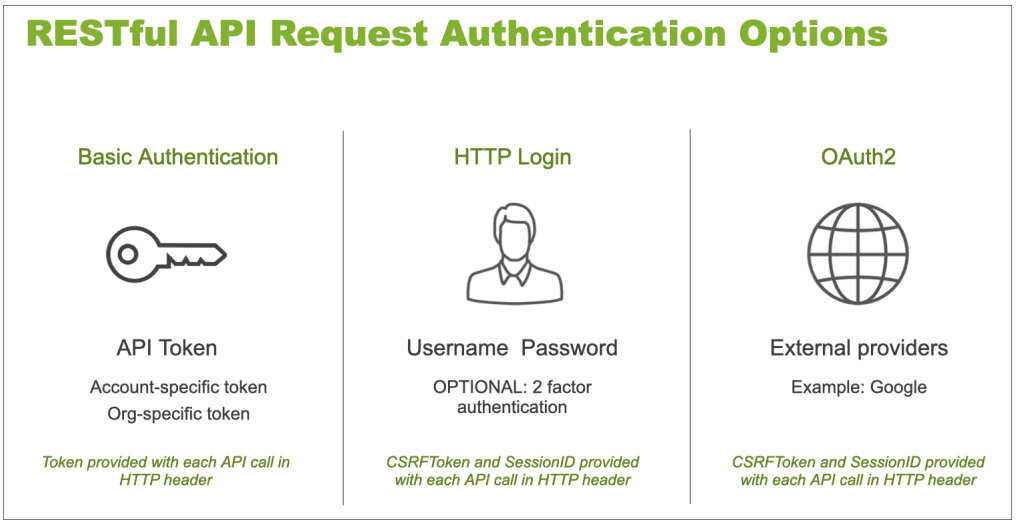
Juniper Mist limits API calls to 5,000 per hour. If you need to make more than 5,000 calls per hour, Create a Support Ticket.

For large deployments, it is recommended that you make API calls at the organization-level to avoid reaching the API call limit quickly.

NOTE: To prevent brute-force attacks, the login API (/api/v1/login) is rate-limited after three login failures.

API Authentication Options

The Juniper Mist API allows three options for requesting authentication:



- Basic Authentication—Token
  - Secure it like a password.
  - For instructions about creating an API token, see "Create API Tokens" on page 15.

  ATTENTION: To enhance security and align with industry best practices, Mist will deprecate Basic Authentication for all use cases—including admin logins and scripts—effective September 2026. Before September 2026, all integrations must transition to token-based authentication to ensure uninterrupted access and support. See "Create API Tokens" on page 15.

- HTTP Login— User name and Password
  - Is like a dashboard login.
  - Can be two-factor authentication.
- OAuth2
  - Account must be linked to an OAuth provider.
  - Requires browser access.

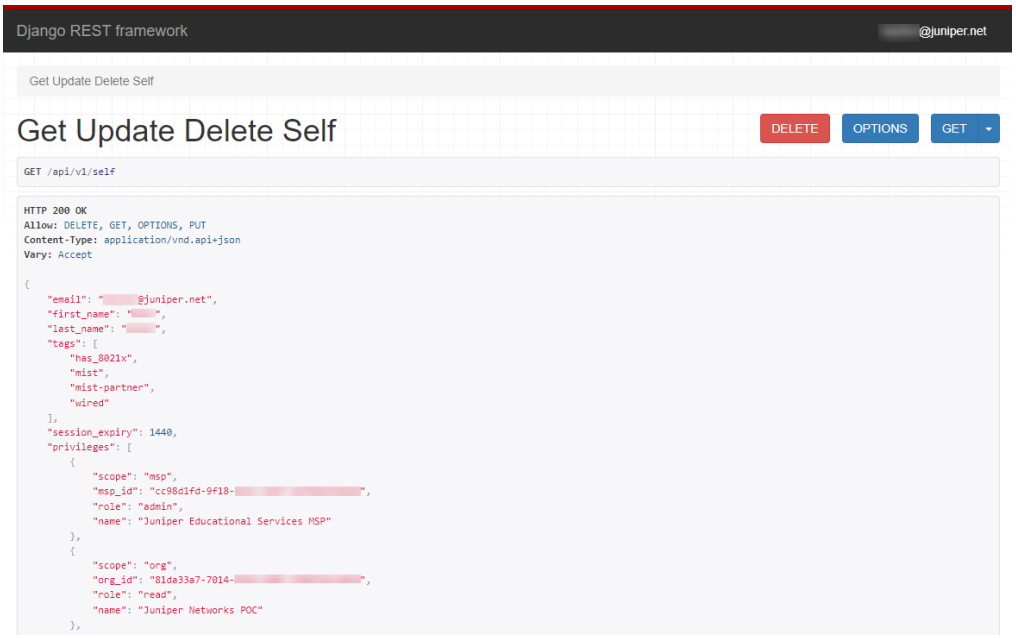
For more information about Authentication, see the Mist API Reference.

A Simple API Example

The Django API interface is a web-based interface where you can perform CRUD operations within the API. See "Use the Django Web Interface to Make API Changes " on page 312.

Using the Django API interface, you can make your first API call. After logging in to Mist, open a new window using the same browser and enter the URL **https://api.mist.com/api/v1/self**. This is the URL for the the Global 01 cloud. If you are using another cloud, this URL will be different.

This is equivalent to making this API call `GET /api/v1/self`.



The result, shown above, displays the privileges assigned to you for the organizations and sites you are associated with.

RELATED DOCUMENTATION

Additional RESTful API Documentation | 154

API Endpoints and Global Regions

SUMMARY

API endpoints vary by global region. You need to use the correct API endpoint for the region that your Juniper Mist organization is associated with.

IN THIS SECTION

Determine Your API Endpoint

You can determine the correct API endpoint URL for your organization by looking in the address bar of the Juniper Mist portal.

1. Log in to the Juniper Mist portal.
2. In the address bar, notice the first part of the URL, starting with the word *manage* and ending with *com*.

   Example: https://**manage.ac2.mist.com**/admin/?org_id=xxxxxxx-xxxx-xxx

   Your API endpoint is similar but starts with *api* instead of *manage*.

   In the above example, the resulting API endpoint URL is **api.ac2.mist.com**.

TIP: The portal URL also contains your organization ID. In the URL, the organization ID section starts with these characters: *org_id=*

List of API Endpoint URLs

Table 4: Endpoints by Global Region

Region	Admin Portal	API
Global 01	manage.mist.com/signin.html	api.mist.com

Table 4: Endpoints by Global Region *(Continued)*

Region	Admin Portal	API
Global 02	manage.gc1.mist.com	api.gc1.mist.com
Global 03	manage.ac2.mist.com	api.ac2.mist.com
Global 04	manage.gc2.mist.com	api.gc2.mist.com
Global 05	manage.gc4.mist.com	api.gc4.mist.com
EMEA 01	manage.eu.mist.com	api.eu.mist.com
EMEA 02	manage.gc3.mist.com	api.gc3.mist.com
EMEA 03	manage.ac6.mist.com	api.ac6.mist.com
EMEA 04	manage.gc6.mist.com	api.gc6.mist.com
APAC 01	manage.ac5.mist.com	api.ac5.mist.com
APAC 02	manage.gc5.mist.com	api.gc5.mist.com
APAC 03	manage.gc7.mist.com	api.gc7.mist.com

Create API Tokens

SUMMARY

Create tokens to provide authentication for your APIs.

IN THIS SECTION

API tokens contain authentication information and are bound to specific users or an entire organization. API tokens send identification information about the user or organization to the API server to indicate whether or not the user has access to the API, to ensure security.

Like many other API providers, Juniper Mist offers a way to generate API tokens for authentication (in the HTTP header). When considering tokens, Juniper Mist uses the terms *token* and *key* interchangeably.

ATTENTION: Before September 2026, all integrations must transition to token-based authentication to ensure uninterrupted access and support. Basic Authentication will be deprecated in September 2026 to enhance security and align with industry best practices.

In Juniper Mist, there are two types of API tokens:

- **Organization Token**

  Organization tokens are useful when you are building an application that automates work for your organization. Multiple users would need to access the application and thus would use the same organization token for authentication.

  - The token persists under the Mist organization.
  - The token is not bound to any specific user, meaning the access does not depend upon any user's access to the organization.
  - Supports *N* org tokens, which can have different privileges.
  - The token can only be used for that specific organization.

  - Rate limiting is done by the individual token. For example, if OrgToken1 consumes 5000 API calls and reaches the rate limit, OrgToken2 is not impacted.

- **User Token**

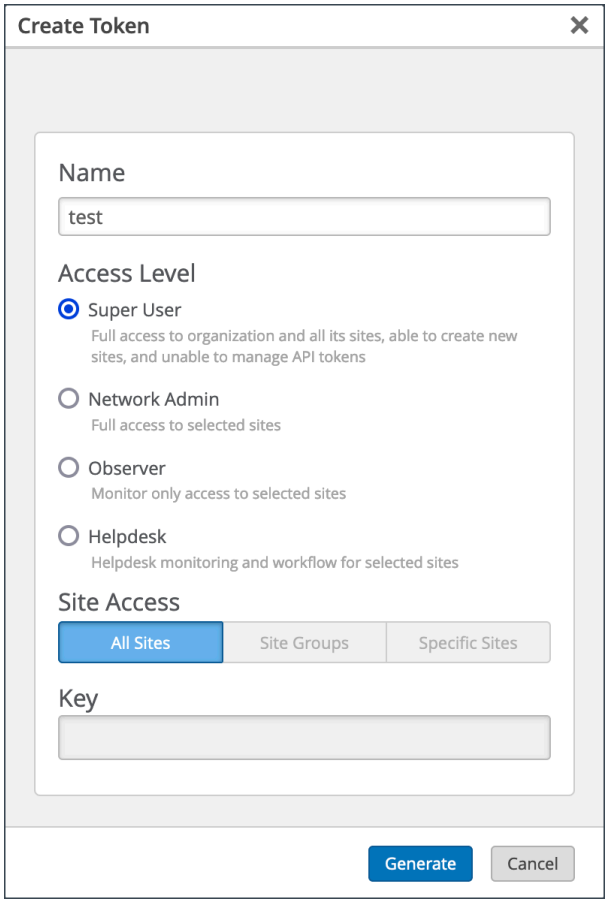
  User tokens are useful when you need to authenticate for yourself, like when running a script, for example.

  - The API token assumes the same privileges as the assigned user's account privileges.
  - The token is bound to the specific user, meaning the access directly correlates to the user's access to the organization.
  - The token can be used for any Managed Service Provider (MSP) or organization that the user has access to.
  - Supports *N* tokens, which all have the same privilege as the user account.
  - Rate limiting is done by the account that is tied to the user. For example, if UserToken1 consumes 5000 API calls and reaches the rate limit, UserToken2 AND account log in to the GUI are impacted.

You can create API tokens through the Mist Portal or REST API Explorer.

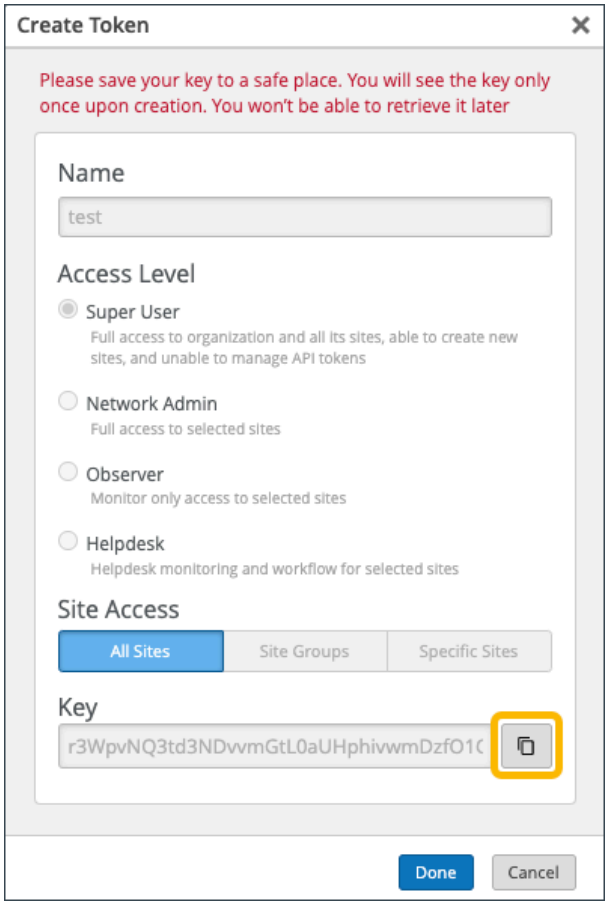
Create an Organization Token in the Mist Portal

1. From the left menu of the Juniper Mist portal, select **Organization** > **Admin** > **Settings**.
2. Scroll down to the **API Token** section and click **Create Token**.
3. Select an **Access Level** to define the permissions for the token.



4. Click **Generate**.
5. Click the copy button next to the **Key** field and store it somewhere for safekeeping.

> **NOTE**: The only time you will see the entire, untruncated key is upon creation. You will not be able to see the full key ever again. If you misplace the key, you will have to create a new key.



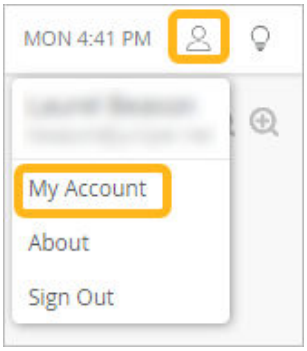
6. Click **Done** at the bottom of the window.
7. Click **Save** near the top-right corner of the page.

Create a User Token in the Mist Portal

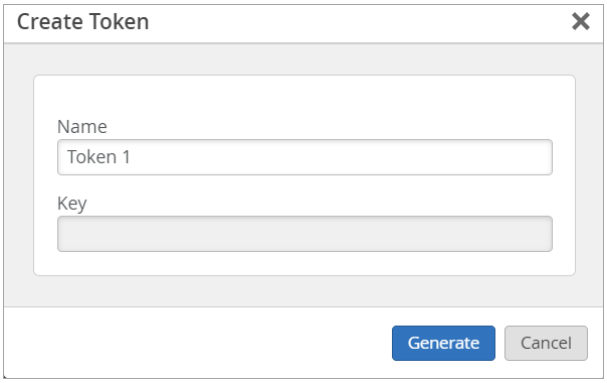
You can generate an API user token from the My Account page in the Mist portal.

To generate an API user token:

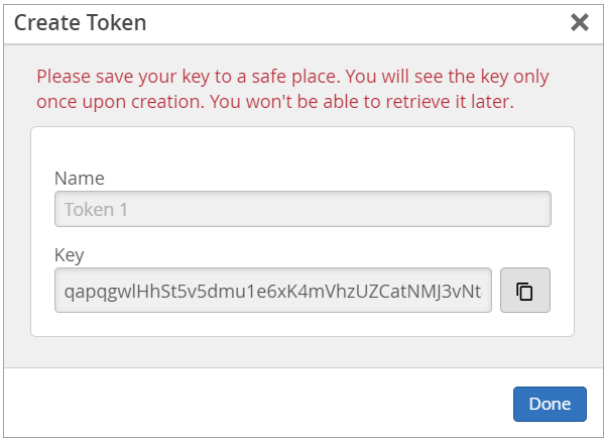
1. At the top-right corner of the Juniper Mist portal, click the Juniper Mist Account icon, and then click **My Account**.



2. In the API Token section, click **Create Token**. If you have enabled single sign-on for your organization, you will not be able to create API user tokens.
3. Enter a name for the token and click **Generate**. The generated key is the user API token.



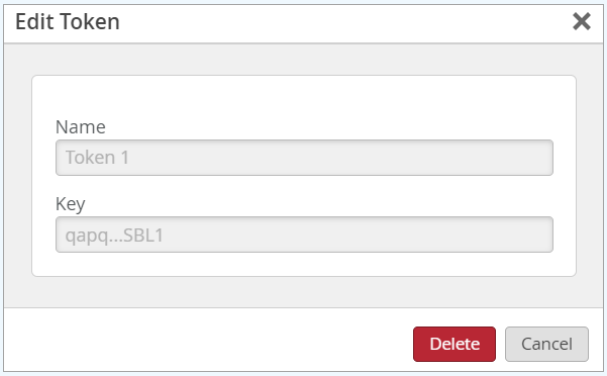
4. Click the copy button next to the **Key** field.



Ensure to store the key somewhere for safekeeping as you will not be able to see the full key again. If you misplace the key, you'll need to create a new key.

5. Click **Done**.

NOTE: If you need to delete a user token, click the token in the API Token section of the **My Profile** page, and then click **Delete** in the Edit Token page.



Create a User or Organization Token Using the REST API Explorer

1. Log in to the Juniper Mist portal.

   NOTE: You must be logged into the portal to use the REST API Explorer.

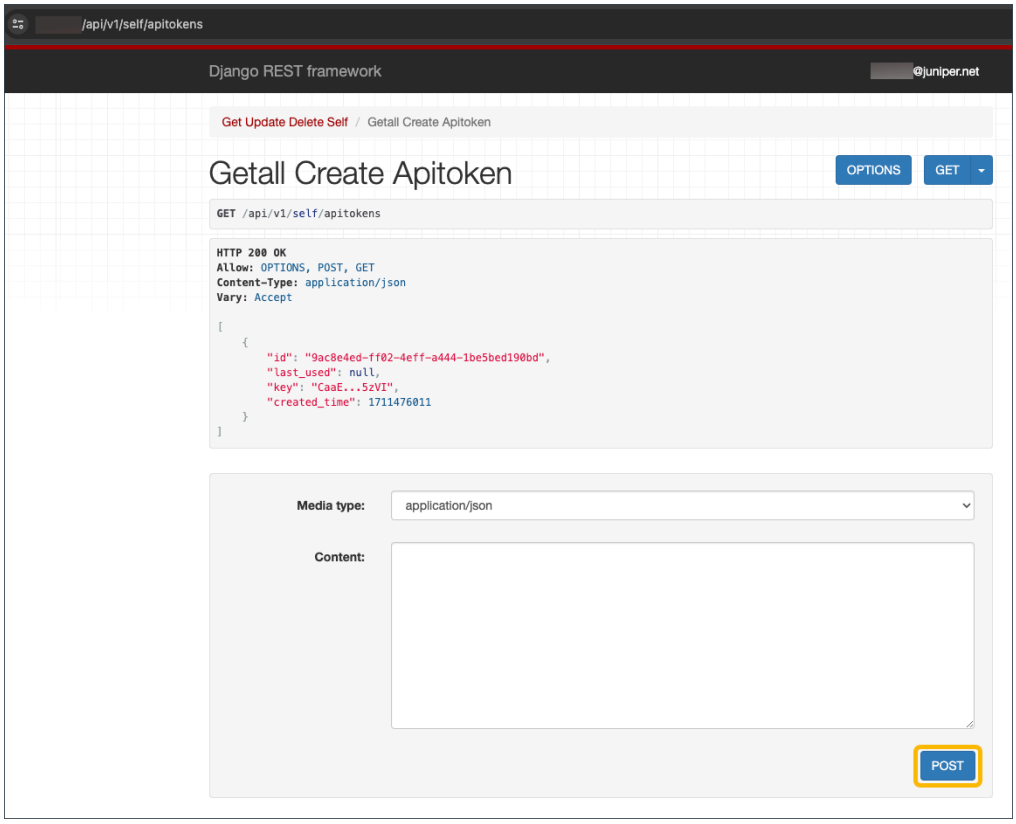
2. Open a new browser window and paste your URL: `{api-host}/api/v1/self/apitokens`.

   NOTE: In place of {api-host}, you need to use the API endpoint for your global region. See "API Endpoints and Global Regions" on page 13.

   The REST API Explorer is the API page for token control. Here you can create, read, update, and delete tokens and token information. This page initially displays the tokens that you have already created.

   This page also enables you and other users to make an API call directly from the browser. With **Media type: applications/json** already selected as the default, a GET request will be performed to show you a list of your tokens. A truncated key will display for any previously created tokens.

3. Click **POST**.



The response will be similar to this example:

```
{
  "id": "437de3bf-acd2-4bed-****-cbba973b91f8",
  "last_used": null,
  "key": "L0kN12lkn4lkn***nklnqlkewnrsFTJ",
  "created_time": 1596821422
}
```

4. Copy the key (token) and store it for safekeeping.

NOTE: The Juniper Mist API will never again display the actual token (*key*) in full, anywhere, after creating the key. After you navigate away from this page and come back, the key will appear but in a truncated version. You should treat this key as a password and store it in a safe place. If you lose this key, you will need to create a new one.

NOTE: If you plan to share Python scripts with others, be sure not to store your API token in the script itself and use an environment file (`.mist_env`) in its place. Environment files store sensitive information in the script for you, so you don't have to share your sensitive information directly in your script. For more information, see https://github.com/tmunzer/mist_library?tab=readme-ov-file#environment-file.

REST API HTTP Response Codes

SUMMARY

Refer to this table for the HTTP response codes that are used by Juniper Mist™.

Table 5: HTTP Response Codes

Status	Description
200	OK. The Mist API understood the call and answered it without errors.
400	Bad Request. The API endpoint exists but its syntax/payload is incorrect, detail may be given. Validate the data provided as part of the JSON payload and make sure it matches the API documentation.
401	Unauthorized. The authentication towards the Mist API failed. You could receive this code when your token is wrong or if your token was not sent to the API with the proper format. Validate your authentication information.
403	Permission Denied. You will receive this code if your privileges do not allow you to access some features. For instance, a user with read-only privileges will not able able to send POST, PULL or DELETE API calls.

Table 5: HTTP Response Codes *(Continued)*

Status	Description
404	Not found. The API endpoint doesn't exist, or the resource doesn't exist. Validate the full URL of your API call and make sure it matches the API documentation.
429	Too Many Requests. The API Token used for the request reached the 5,000 API Calls per hour threshold.

Gather Data Using the RESTful API

SUMMARY

Read the topics in this chapter to learn how to gather data using the RESTful API.

IN THIS SECTION

Mist offers cloud-based services that assist in identifying the root cause of poor user experience on the network. These services are known as assurances. They are automated Mist tools that provide insight into users' connection data and can proactively make configuration changes to the network. Examples of such services are WAN Assurance, Wired Assurance, Wireless Assurance, and Access Assurance.

Service Level Expectations (SLEs) and Insights are two features that are included in each of these licensed offerings. In addition to viewing SLEs and Insights from the Mist UI, the information is also available from the API.

For details on how to gather data using the RESTful API, continue to the next topics in this guide.

Use Mist SLEs and Insights with APIs

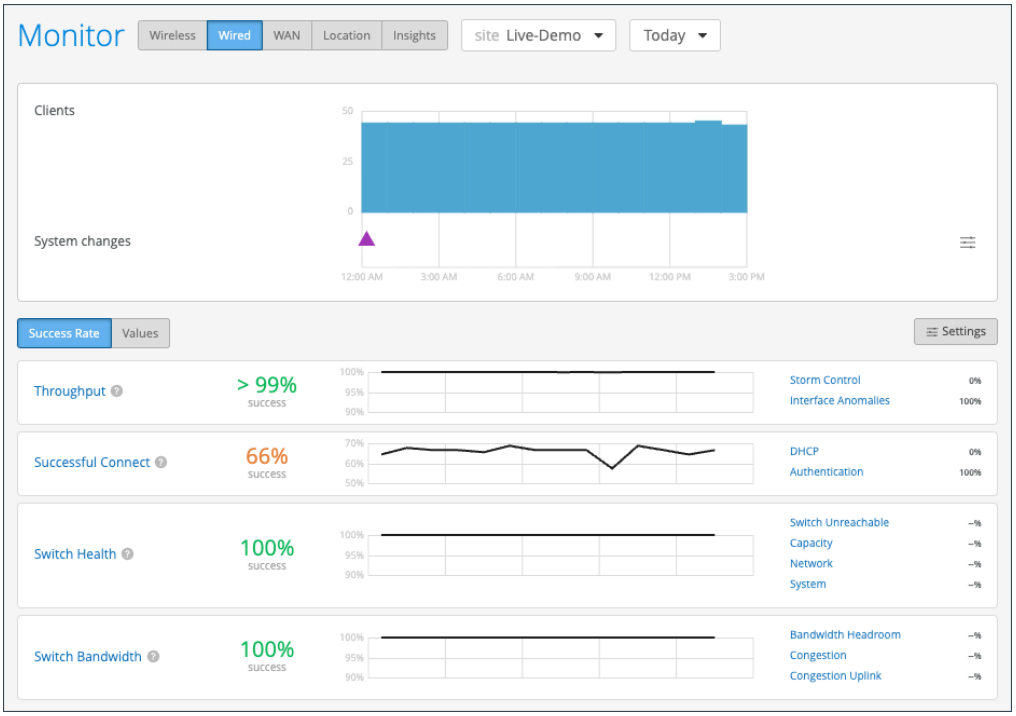
SUMMARY

Use the RESTful API to get insights about network performance.

IN THIS SECTION

When you use the Juniper Mist portal to monitor the operation of your network, you gain insight into what's happening before it becomes an issue. You can see the network from multiple viewpoints: wireless, wired, WAN, and others. Additionally, you can use the tools that Juniper Mist™ provides to troubleshoot and correct potential issues.

The primary Mist dashboard at **Monitor** > **Service Levels** presents the results of the Predictive Analytics and Correlation Engine (PACE), in the form of Service Level Expectation (SLE) metrics. SLEs leverage machine learning and the Mist PACE in the Juniper Mist cloud. Using these resources, SLEs turn streaming telemetry from the access points (APs) into visualizations representing end users' network experience in near real time. For more information about SLEs, see the Juniper Mist AI-Native Operations Guide.



As with everything seen in the Mist GUI, SLEs and Insights information are also available from the API.

SLEs

It may be useful for you to gather information on specific SLEs to use for historical reporting purposes or to trigger other automation. Like other API calls, you begin by determining which endpoint you want to gather data from. The following is an example of the API GET request for the `getOrgsSitesSle` endpoint:

```
GET
        /api/v1/orgs/{org_id}/insights/sites-sle
```

NOTE: Replace {org_id} with your organization ID.

For more information, see SLEs Overview.

Insights

Insights provide an overview of network experience across the entire Site, Access Points, or wireless Client. It's a great place to start when checking on a site.

You can find Insight information by making a GET request to one of the following Insights endpoints:

- GetSiteInsightMetrics

- GetSiteInsightMetricsForDevice
- GetSiteInsightMetricsForClient

For more information, see Insights Overview.

Metrics and Classifiers

Juniper Mist SLEs and Insights endpoints support metrics and classifiers. Metrics track whether the service level meets the configured threshold value. If a metric does not meet the threshold, then this failure may be attributed to one of the classifiers to further understand where the failure occurred.

The SLE and Insights endpoints often require the `metric` argument. This is because, in addition to raw statistics or configuration, Mist exposes calculated values resulting from internal data analysis. Rather than creating a unique API function every time a new data analytics feature is added, Mist exposes just a handful of function, but uses the `metric` argument to specify which derived data values to retrieve.

To get a list of insight metrics, you can issue the following GET call:

```
GET
        /api/v1/sites/{site_id}/insights/{metric}
```

The response would look like this:

```
{
  "bytes": {
    "description": "aggregated bytes over time",
    "example": [
      185,
      197,
      250
    ],
    "intervals": {
      "10m": {
        "interval": 600,
        "max_age": 86400
      },
      "1h": {
        "interval": 3600,
        "max_age": 1209600
      }
    },
```

```
    "report_durations": {
      "1d": {
        "duration": 86400,
        "interval": 3600
      },
      "1w": {
        "duration": 604800,
        "interval": 3600
      }
    },
    "report_scopes": [
      "site",
      "org"
    ],
    "scopes": [
      "site",
      "ap",
      "client"
    ],
    "type": "timeseries",
    "unit": "byte"
  },
  "num_clients": {
    "description": "number of client over time",
    "example": [
      18,
      null,
      15
    ],
    "intervals": {
      "10m": {
        "interval": 600,
        "max_age": 86400
      },
      "1h": {
        "interval": 3600,
        "max_age": 1209600
      }
    },
    "report_durations": {
      "1d": {
        "duration": 86400,
        "interval": 3600
```

```
      },
      "1w": {
        "duration": 604800,
        "interval": 3600
      }
    },
    "report_scopes": [
      "site",
      "org"
    ],
    "scopes": [
      "site",
      "ap",
      "device"
    ],
    "type": "timeseries",
    "unit": ""
  }
}
```

Another way to view examples of available Insight metrics is by logging into the Mist portal and opening this link in a new tab from the same browser:

https://api.mist.com/api/v1/const/insight_metrics

Using the previous GET call example **GET /api/v1/const/insight_metrics**, you would add your desired metric at the end of the call. See the example GET calls below for some of the currently supported metrics and their classifiers.

AP Uptime: ap-availability

GET call:

```
GET /api/v1/sites/{site_id}/insights/ap-availability
```

NOTE: Replace {site_id} with your site ID.

Sample Request:

```
https://api.gc2.mist.com/api/v1/sites/aee83225-1773-4e55-af64-c8b5a86b1fa6/sle/site/
aee83225-1773-4e55-af64-c8b5a86b1fa6/metric/ap-availability/summary?start&end&duration
```

Sample Response:


```
{
    "start": 1727696747,
    "end": 1727783147,
    "sle": {
        "name": "ap-availability",
        "x_label": "seconds",
        "y_label": "seconds",
        "interval": 3600,
        "samples": {
            "total": [
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null
            ],
            "degraded": [
                null,
                null,
                null,
                null,
```

```
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null
        ],
        "value": [
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
```

```
            null,
            null,
            null
        ]
    }
},
"impact": {
    "num_users": 0,
    "num_aps": 0,
    "total_users": 0,
    "total_aps": 0
},
"classifiers": [
    {
        "name": "ap-disconnected-ap-unreachable",
        "x_label": "seconds",
        "y_label": "minutes",
        "interval": 3600,
        "samples": {
            "duration": [
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
```

```
                    0
                ],
                "total": [
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0
            ]
        },
        "impact": {
            "num_users": null,
            "num_aps": 0,
            "total_users": 0,
            "total_aps": 0
        }
    },
    {
        "name": "ethernet-ethernet-errors",
        "x_label": "seconds",
        "y_label": "minutes",
        "interval": 3600,
        "samples": {
            "duration": [
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": null,
                "num_aps": 0,
                "total_users": 0,
                "total_aps": 0
            }
        },
        {
            "name": "ethernet-speed-mismatch",
            "x_label": "seconds",
            "y_label": "minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null
                ],
                "degraded": [
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": null,
                "num_aps": 0,
                "total_users": 0,
                "total_aps": 0
            }
        },
        {
            "name": "low-power",
            "x_label": "seconds",
            "y_label": "minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0
            ],
            "total": [
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
                null,
```

```
                    null,
                    null
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": null,
                "num_aps": 0,
                "total_users": 0,
                "total_aps": 0
            }
        },
        {
            "name": "ap-disconnected-switch-down",
            "x_label": "seconds",
            "y_label": "minutes",
            "interval": 3600,
            "samples": {
```

```
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
```

```
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": null,
                "num_aps": 0,
                "total_users": 0,
                "total_aps": 0
            }
        },
```

```
        {
            "name": "ap-disconnected-site-down",
            "x_label": "seconds",
            "y_label": "minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
```

```
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null,
                    null
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
```

```
                "num_users": null,
                "num_aps": 0,
                "total_users": 0,
                "total_aps": 0
            }
        },
        {
            "name": "ap-disconnected-ap-reboot",
            "x_label": "seconds",
            "y_label": "minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    null,
                    null,
                    null,
                    null,
```

```
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null,
            null
        ],
        "degraded": [
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
            0,
```

```
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": null,
                "num_aps": 0,
                "total_users": 0,
                "total_aps": 0
            }
        }
    ],
    "events": []
}
```


- **AP Reboot:** ap-reboot
- **AP Unreachable:** ap-unreachable
- **Site Down:** site-down

Capacity: capacity

GET Call:

```
GET /api/v1/sites/{site_id}/insights/capacity
```

NOTE: Replace {site_id} with your site ID.

Sample Request:

```
https://api.gc2.mist.com/api/v1/sites/aee83225-1773-4e55-af64-c8b5a86b1fa6/sle/site/
aee83225-1773-4e55-af64-c8b5a86b1fa6/metric/capacity/summary?start&end&duration
```

Sample Response:


```
{
    "start": 1727696707,
    "end": 1727783107,
    "sle": {
```

```
        "name": "capacity",
        "x_label": "seconds",
        "y_label": "%",
        "interval": 3600,
        "samples": {
            "total": [
                1204.1,
                1243.9333,
                1184.7667,
                1136.9667,
                1133.6666,
                1152.7167,
                1137.8334,
                1131.2167,
                1119.2167,
                1136.1,
                1143.9667,
                1167.2667,
                1133.8667,
                1182.5,
                1274.6333,
                1281.5333,
                1232.7333,
                1260.9833,
                1258.3167,
                1250.7167,
                1215.25,
                1236.0834,
                1247.9667,
                828.05
            ],
            "degraded": [
                228.71666,
                301.41666,
                110.13333,
                88.183334,
                82.416664,
                128.93333,
                139.23334,
                295.08334,
                170.0,
                201.7,
                151.41667,
```

```
                145.06667,
                153.91667,
                181.76666,
                256.45,
                239.83333,
                214.08333,
                215.9,
                147.68333,
                163.25,
                120.36667,
                120.2,
                135.0,
                184.68333
            ],
            "value": [
                0.58172977,
                0.5803318,
                0.5925928,
                0.6017501,
                0.61211795,
                0.611349,
                0.6141628,
                0.59855264,
                0.6093941,
                0.60869706,
                0.60907525,
                0.60685295,
                0.60197794,
                0.5995983,
                0.6004573,
                0.6096473,
                0.61228454,
                0.6112147,
                0.60956645,
                0.6057745,
                0.6139534,
                0.6133245,
                0.5969889,
                0.5707334
            ]
        }
    },
    "impact": {
```

```
        "num_users": 26,
        "num_aps": 4,
        "total_users": 30,
        "total_aps": 4
    },
    "classifiers": [
        {
            "name": "client-count",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    1204.1,
                    1243.9333,
                    1184.7667,
                    1136.9667,
```

```
        1133.6666,
        1152.7167,
        1137.8334,
        1131.2167,
        1119.2167,
        1136.1,
        1143.9667,
        1167.2667,
        1133.8667,
        1182.5,
        1274.6333,
        1281.5333,
        1232.7333,
        1260.9833,
        1258.3167,
        1250.7167,
        1215.25,
        1236.0834,
        1247.9667,
        828.05
    ],
    "degraded": [
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
        0,
```

```
                0,
                0,
                0
            ]
        },
        "impact": {
            "num_users": 0,
            "num_aps": 0,
            "total_users": 30,
            "total_aps": 4
        }
    },
    {
        "name": "wifi-interference",
        "x_label": "seconds",
        "y_label": "user-minutes",
        "interval": 3600,
        "samples": {
            "duration": [
                202.51666,
                226.98334,
                84.25,
                82.63333,
                79.28333,
                106.566666,
                105.23333,
                295.08334,
                162.65,
                190.48334,
                129.05,
                140.58333,
                146.63333,
                163.33333,
                245.2,
                237.3,
                214.08333,
                213.75,
                139.35,
                160.76666,
                115.45,
                115.0,
                115.2,
                165.38333
```

```
                ],
                "total": [
                    1204.1,
                    1243.9333,
                    1184.7667,
                    1136.9667,
                    1133.6666,
                    1152.7167,
                    1137.8334,
                    1131.2167,
                    1119.2167,
                    1136.1,
                    1143.9667,
                    1167.2667,
                    1133.8667,
                    1182.5,
                    1274.6333,
                    1281.5333,
                    1232.7333,
                    1260.9833,
                    1258.3167,
                    1250.7167,
                    1215.25,
                    1236.0834,
                    1247.9667,
                    828.05
                ],
                "degraded": [
                    202.51666,
                    226.98334,
                    84.25,
                    82.63333,
                    79.28333,
                    106.566666,
                    105.23333,
                    295.08334,
                    162.65,
                    190.48334,
                    129.05,
                    140.58333,
                    146.63333,
                    163.33333,
                    245.2,
```

```
                    237.3,
                    214.08333,
                    213.75,
                    139.35,
                    160.76666,
                    115.45,
                    115.0,
                    115.2,
                    165.38333
                ]
            },
            "impact": {
                "num_users": 26,
                "num_aps": 4,
                "total_users": 30,
                "total_aps": 4
            }
        },
        {
            "name": "client-usage",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    26.2,
                    74.433334,
                    20.983334,
                    5.55,
                    3.1333334,
                    0,
                    27.2,
                    0,
                    7.35,
                    1.7333333,
                    4.616667,
                    4.483333,
                    7.2833333,
                    8.3,
                    11.25,
                    2.5333333,
                    0,
                    2.15,
```

```
                    8.333333,
                    2.4833333,
                    4.9166665,
                    5.2,
                    2.6666667,
                    17.183332
                ],
                "total": [
                    1204.1,
                    1243.9333,
                    1184.7667,
                    1136.9667,
                    1133.6666,
                    1152.7167,
                    1137.8334,
                    1131.2167,
                    1119.2167,
                    1136.1,
                    1143.9667,
                    1167.2667,
                    1133.8667,
                    1182.5,
                    1274.6333,
                    1281.5333,
                    1232.7333,
                    1260.9833,
                    1258.3167,
                    1250.7167,
                    1215.25,
                    1236.0834,
                    1247.9667,
                    828.05
                ],
                "degraded": [
                    26.2,
                    74.433334,
                    20.983334,
                    5.55,
                    3.1333334,
                    0,
                    27.2,
                    0,
                    7.35,
```

```
                    1.7333333,
                    4.616667,
                    4.483333,
                    7.2833333,
                    8.3,
                    11.25,
                    2.5333333,
                    0,
                    2.15,
                    8.333333,
                    2.4833333,
                    4.9166665,
                    5.2,
                    2.6666667,
                    17.183332
                ]
            },
            "impact": {
                "num_users": 16,
                "num_aps": 2,
                "total_users": 30,
                "total_aps": 4
            }
        },
        {
            "name": "non-wifi-interference",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    4.9,
                    0,
                    0,
                    22.366667,
                    6.8,
                    0,
                    0,
                    9.483334,
                    17.75,
                    0,
```

```
                    0,
                    10.133333,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    17.133333,
                    22.966667
                ],
                "total": [
                    1204.1,
                    1243.9333,
                    1184.7667,
                    1136.9667,
                    1133.6666,
                    1152.7167,
                    1137.8334,
                    1131.2167,
                    1119.2167,
                    1136.1,
                    1143.9667,
                    1167.2667,
                    1133.8667,
                    1182.5,
                    1274.6333,
                    1281.5333,
                    1232.7333,
                    1260.9833,
                    1258.3167,
                    1250.7167,
                    1215.25,
                    1236.0834,
                    1247.9667,
                    828.05
                ],
                "degraded": [
                    0,
                    0,
                    4.9,
```

```
                    0,
                    0,
                    22.366667,
                    6.8,
                    0,
                    0,
                    9.483334,
                    17.75,
                    0,
                    0,
                    10.133333,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    17.133333,
                    22.966667
                ]
            },
            "impact": {
                "num_users": 11,
                "num_aps": 2,
                "total_users": 30,
                "total_aps": 4
            }
        }
    ],
    "events": []
}
```


- **AP Load:** AP-load
- **Non WiFi Interference:** non-wifi-interference
- **WiFi Interference:** wifi-interference

Coverage: coverage

GET Call:

```
GET /api/v1/sites/{site_id}/insights/coverage
```

NOTE: Replace {site_id} with your site ID.

Sample Request:

```
https://api.gc2.mist.com/api/v1/sites/aee83225-1773-4e55-af64-c8b5a86b1fa6/sle/site/
aee83225-1773-4e55-af64-c8b5a86b1fa6/metric/coverage/summary?start&end&duration
```

Sample Response:


```
{
    "start": 1727696673,
    "end": 1727783073,
    "sle": {
        "name": "coverage",
        "x_label": "seconds",
        "y_label": "dBm",
        "interval": 3600,
        "samples": {
            "total": [
                1166.2,
                1229.7,
                1180.7,
                1127.1833,
                1132.2,
                1133.5834,
                1129.7167,
                1128.1333,
                1113.1,
                1122.7667,
                1119.1333,
                1155.6333,
                1127.75,
                1147.85,
                1251.3833,
                1279.7833,
                1232.7333,
```

```
                1263.7333,
                1253.3,
                1249.0333,
                1215.25,
                1235.1,
                1242.0,
                948.4667
            ],
            "degraded": [
                14.05,
                10.433333,
                9.583333,
                4.516667,
                0.0,
                10.383333,
                0.0,
                0.0,
                2.3166666,
                4.1,
                6.5333333,
                6.3,
                5.15,
                10.15,
                41.35,
                76.03333,
                32.666668,
                42.25,
                11.35,
                3.2666667,
                1.05,
                2.0333333,
                15.433333,
                12.183333
            ],
            "value": [
                -57.629894,
                -57.33048,
                -57.447754,
                -57.30583,
                -56.43402,
                -56.627388,
                -55.95337,
                -54.501797,
```

```
                -54.907463,
                -54.84789,
                -55.376183,
                -56.49305,
                -56.228077,
                -56.321236,
                -57.631668,
                -57.97819,
                -57.058994,
                -56.97246,
                -56.41033,
                -56.125,
                -56.128677,
                -56.17078,
                -55.696415,
                -55.79176
            ]
        }
    },
    "impact": {
        "num_users": 8,
        "num_aps": 3,
        "total_users": 28,
        "total_aps": 4
    },
    "classifiers": [
        {
            "name": "asymmetry-uplink",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    9.433333,
                    7.55,
                    3.3,
                    2.8666666,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0.8833333,
                    0,
                    1.9166666,
                    23.383333,
                    31.1,
                    1.25,
                    0,
                    8.433333,
                    1.2166667,
                    0,
                    0,
                    13.25,
                    2.15
                ],
                "total": [
                    1166.2,
                    1229.7,
                    1180.7,
                    1127.1833,
                    1132.2,
                    1133.5834,
                    1129.7167,
                    1128.1333,
                    1113.1,
                    1122.7667,
                    1119.1333,
                    1155.6333,
                    1127.75,
                    1147.85,
                    1251.3833,
                    1279.7833,
                    1232.7333,
                    1263.7333,
                    1253.3,
                    1249.0333,
                    1215.25,
                    1235.1,
                    1242.0,
                    948.4667
                ],
                "degraded": [
                    9.433333,
```

```
                    7.55,
                    3.3,
                    2.8666666,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0.8833333,
                    0,
                    1.9166666,
                    23.383333,
                    31.1,
                    1.25,
                    0,
                    8.433333,
                    1.2166667,
                    0,
                    0,
                    13.25,
                    2.15
                ]
            },
            "impact": {
                "num_users": 5,
                "num_aps": 3,
                "total_users": 28,
                "total_aps": 4
            }
        },
        {
            "name": "asymmetry-downlink",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    1166.2,
                    1229.7,
                    1180.7,
                    1127.1833,
                    1132.2,
                    1133.5834,
                    1129.7167,
                    1128.1333,
                    1113.1,
                    1122.7667,
                    1119.1333,
                    1155.6333,
                    1127.75,
                    1147.85,
                    1251.3833,
                    1279.7833,
                    1232.7333,
                    1263.7333,
                    1253.3,
                    1249.0333,
                    1215.25,
```

```
                    1235.1,
                    1242.0,
                    948.4667
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 28,
                "total_aps": 4
            }
        },
        {
            "name": "weak-signal",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
```

```
            "samples": {
                "duration": [
                    4.616667,
                    2.8833334,
                    6.2833333,
                    1.65,
                    0,
                    10.383333,
                    0,
                    0,
                    2.3166666,
                    4.1,
                    6.5333333,
                    5.4166665,
                    5.15,
                    8.233334,
                    17.966667,
                    44.933334,
                    31.416666,
                    42.25,
                    2.9166667,
                    2.05,
                    1.05,
                    2.0333333,
                    2.1833334,
                    5.0333333
                ],
                "total": [
                    1166.2,
                    1229.7,
                    1180.7,
                    1127.1833,
                    1132.2,
                    1133.5834,
                    1129.7167,
                    1128.1333,
                    1113.1,
                    1122.7667,
                    1119.1333,
                    1155.6333,
                    1127.75,
                    1147.85,
                    1251.3833,
```

```
                    1279.7833,
                    1232.7333,
                    1263.7333,
                    1253.3,
                    1249.0333,
                    1215.25,
                    1235.1,
                    1242.0,
                    948.4667
                ],
                "degraded": [
                    4.616667,
                    2.8833334,
                    6.2833333,
                    1.65,
                    0,
                    10.383333,
                    0,
                    0,
                    2.3166666,
                    4.1,
                    6.5333333,
                    5.4166665,
                    5.15,
                    8.233334,
                    17.966667,
                    44.933334,
                    31.416666,
                    42.25,
                    2.9166667,
                    2.05,
                    1.05,
                    2.0333333,
                    2.1833334,
                    5.0333333
                ]
            },
            "impact": {
                "num_users": 6,
                "num_aps": 3,
                "total_users": 28,
                "total_aps": 4
            }
```

```
        }
    ],
    "events": []
}
```

- **Asymmetry Downlink:** asymmetry-downlink
- **Asymmetry Uplink:** asymmetry-uplink
- **Weak Signal:** weak-signal

Roaming: roaming

GET Call:

```
GET /api/v1/sites/{site_id}/insights/roaming
```

NOTE: Replace {site_id} with your site ID.

Sample Request:

```
https://api.gc2.mist.com/api/v1/sites/aee83225-1773-4e55-af64-c8b5a86b1fa6/sle/site/
aee83225-1773-4e55-af64-c8b5a86b1fa6/metric/roaming/summary?start&end&duration
```

Sample Response:

```
{
    "start": 1727696635,
    "end": 1727783035,
    "sle": {
        "name": "roaming",
        "x_label": "seconds",
        "y_label": "roaming-score",
        "interval": 3600,
        "samples": {
            "total": [
                45.0,
                30.0,
                3.0,
                12.0,
                1.0,
```

```
                23.0,
                2.0,
                3.0,
                13.0,
                10.0,
                46.0,
                13.0,
                29.0,
                59.0,
                37.0,
                4.0,
                null,
                null,
                1.0,
                null,
                null,
                null,
                6.0,
                33.0
            ],
            "degraded": [
                1.0,
                1.0,
                0.0,
                0.0,
                0.0,
                1.0,
                0.0,
                0.0,
                2.0,
                0.0,
                0.0,
                1.0,
                1.0,
                0.0,
                4.0,
                0.0,
                null,
                null,
                0.0,
                null,
                null,
                null,
```

```
                1.0,
                1.0
            ],
            "value": [
                1.0666667,
                1.1333333,
                1.0,
                1.0,
                1.0,
                1.173913,
                1.0,
                1.0,
                1.4615384,
                1.0,
                1.0,
                1.2307693,
                1.1034483,
                1.0,
                1.4324324,
                1.0,
                null,
                null,
                1.0,
                null,
                null,
                null,
                1.5,
                1.1515151
            ]
        }
    },
    "impact": {
        "num_users": 3,
        "num_aps": 3,
        "total_users": 9,
        "total_aps": 4
    },
    "classifiers": [
        {
            "name": "latency-slow-okc-roam",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
```

```
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    45.0,
                    30.0,
                    3.0,
                    12.0,
                    1.0,
                    23.0,
                    2.0,
                    3.0,
                    13.0,
                    10.0,
                    46.0,
                    13.0,
                    29.0,
                    59.0,
                    37.0,
```

```
                    4.0,
                    null,
                    null,
                    1.0,
                    null,
                    null,
                    null,
                    6.0,
                    33.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 9,
                "total_aps": 4
            }
```

```
        },
        {
            "name": "latency-slow-11r-roam",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    45.0,
                    30.0,
                    3.0,
                    12.0,
                    1.0,
                    23.0,
                    2.0,
                    3.0,
                    13.0,
```

```
                    10.0,
                    46.0,
                    13.0,
                    29.0,
                    59.0,
                    37.0,
                    4.0,
                    null,
                    null,
                    1.0,
                    null,
                    null,
                    null,
                    6.0,
                    33.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
```

```
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 9,
                "total_aps": 4
            }
        },
        {
            "name": "stability-failed-to-fast-roam",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    45.0,
                    30.0,
                    3.0,
```

```
                    12.0,
                    1.0,
                    23.0,
                    2.0,
                    3.0,
                    13.0,
                    10.0,
                    46.0,
                    13.0,
                    29.0,
                    59.0,
                    37.0,
                    4.0,
                    null,
                    null,
                    1.0,
                    null,
                    null,
                    null,
                    6.0,
                    33.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 9,
                "total_aps": 4
            }
        },
        {
            "name": "signal-quality-interband-roam",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    1.0,
                    0,
                    0,
                    0,
                    1.0,
                    0,
                    3.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    1.0
                ],
                "total": [
                    45.0,
                    30.0,
                    3.0,
                    12.0,
                    1.0,
                    23.0,
                    2.0,
                    3.0,
                    13.0,
                    10.0,
                    46.0,
                    13.0,
                    29.0,
                    59.0,
                    37.0,
                    4.0,
                    null,
                    null,
                    1.0,
                    null,
                    null,
                    null,
                    6.0,
                    33.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    1.0,
                    0,
                    0,
                    0,
                    1.0,
                    0,
```

```
                    3.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    1.0
                ]
            },
            "impact": {
                "num_users": 2,
                "num_aps": 3,
                "total_users": 9,
                "total_aps": 4
            }
        },
        {
            "name": "signal-quality-suboptimal-roam",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    1.0,
                    1.0,
                    0,
                    0,
                    0,
                    1.0,
                    0,
                    0,
                    1.0,
                    0,
                    0,
                    1.0,
                    0,
                    0,
                    1.0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    1.0,
                    0
                ],
                "total": [
                    45.0,
                    30.0,
                    3.0,
                    12.0,
                    1.0,
                    23.0,
                    2.0,
                    3.0,
                    13.0,
                    10.0,
                    46.0,
                    13.0,
                    29.0,
                    59.0,
                    37.0,
                    4.0,
                    null,
                    null,
                    1.0,
                    null,
                    null,
                    null,
                    6.0,
                    33.0
                ],
                "degraded": [
                    1.0,
                    1.0,
                    0,
                    0,
                    0,
                    1.0,
                    0,
                    0,
```

```
                    1.0,
                    0,
                    0,
                    1.0,
                    0,
                    0,
                    1.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    1.0,
                    0
                ]
            },
            "impact": {
                "num_users": 3,
                "num_aps": 2,
                "total_users": 9,
                "total_aps": 4
            }
        },
        {
            "name": "latency-slow-standard-roam",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    45.0,
                    30.0,
                    3.0,
                    12.0,
                    1.0,
                    23.0,
                    2.0,
                    3.0,
                    13.0,
                    10.0,
                    46.0,
                    13.0,
                    29.0,
                    59.0,
                    37.0,
                    4.0,
                    null,
                    null,
                    1.0,
                    null,
                    null,
                    null,
                    6.0,
                    33.0
                ],
                "degraded": [
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 9,
                "total_aps": 4
            }
        },
        {
            "name": "signal-quality-sticky-client",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    45.0,
                    30.0,
                    3.0,
                    12.0,
                    1.0,
                    23.0,
                    2.0,
                    3.0,
                    13.0,
                    10.0,
                    46.0,
                    13.0,
                    29.0,
                    59.0,
                    37.0,
                    4.0,
                    null,
                    null,
                    1.0,
                    null,
                    null,
                    null,
```

```
                    6.0,
                    33.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 9,
                "total_aps": 4
            }
        }
    ],
    "events": []
}
```


- **Failed to Fast Roam:** No-Fast-Roam

- **Slow 11r Roams:** Suboptimal-11r-roam
- **Slow OKC Roams:** Suboptimal-okc-roam
- **Slow Standard Roams:** Slow-Roam

Successful Connects: successful-connect

GET Call:

```
GET /api/v1/sites/{site_id}/insights/successful-connect
```

NOTE: Replace {site_id} with your site ID.

Sample Request:

```
https://api.gc2.mist.com/api/v1/sites/aee83225-1773-4e55-af64-c8b5a86b1fa6/sle/site/
aee83225-1773-4e55-af64-c8b5a86b1fa6/metric/time-to-connect/summary?start&end&duration
```

Sample Response:


```
{
    "start": 1727696454,
    "end": 1727782854,
    "sle": {
        "name": "time-to-connect",
        "x_label": "seconds",
        "y_label": "seconds",
        "interval": 3600,
        "samples": {
            "total": [
                52.0,
                32.0,
                6.0,
                15.0,
                2.0,
                24.0,
                7.0,
                3.0,
                13.0,
                13.0,
```

```
                49.0,
                15.0,
                29.0,
                62.0,
                38.0,
                4.0,
                null,
                1.0,
                1.0,
                2.0,
                null,
                null,
                9.0,
                31.0
            ],
            "degraded": [
                4.0,
                1.0,
                2.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                1.0,
                1.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                null,
                0.0,
                0.0,
                0.0,
                null,
                null,
                0.0,
                0.0
            ],
            "value": [
                1.0913653,
```

```
                0.5267187,
                6.8511662,
                0.20886666,
                1.885,
                0.038458332,
                0.059428573,
                0.039666668,
                0.044153847,
                0.27992308,
                0.09634693,
                0.0908,
                0.021689653,
                0.047080643,
                0.037736844,
                0.0205,
                null,
                0.223,
                0.027,
                0.344,
                null,
                null,
                0.15811113,
                0.04767742
            ]
        }
    },
    "impact": {
        "num_users": 4,
        "num_aps": 4,
        "total_users": 26,
        "total_aps": 4
    },
    "classifiers": [
        {
            "name": "IP-Services",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
```

```
                    null,
                    null,
                    9.0,
                    31.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 26,
                "total_aps": 4
            }
        },
        {
            "name": "authorization",
            "x_label": "seconds",
            "y_label": "attempts",
```

```
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
```

```
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
                    31.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 26,
                "total_aps": 4
```

```
            }
        },
        {
            "name": "dhcp-nack",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
```

```
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
                    31.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
```

```
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 26,
                "total_aps": 4
            }
        },
        {
            "name": "association",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    3.0,
                    1.0,
                    2.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
```

```
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
                    31.0
                ],
                "degraded": [
                    3.0,
                    1.0,
                    2.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                0,
                0,
                0,
                0,
                0
            ]
        },
        "impact": {
            "num_users": 1,
            "num_aps": 3,
            "total_users": 26,
            "total_aps": 4
        }
    },
    {
        "name": "dhcp-stuck",
        "x_label": "seconds",
        "y_label": "attempts",
        "interval": 3600,
        "samples": {
            "duration": [
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
```

```
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
                    31.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 26,
                "total_aps": 4
            }
        },
        {
            "name": "dhcp-unresponsive",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    1.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    1.0,
                    1.0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
                    31.0
                ],
                "degraded": [
                    1.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    1.0,
                    1.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 3,
                "num_aps": 3,
                "total_users": 26,
                "total_aps": 4
            }
        }
    ],
    "events": []
}
```


- **Association:** association
- **Authorization:** authorization
- **DHCP:** DHCP

Throughput: throughput

GET Call:

```
GET /api/v1/sites/{site_id}/insights/throughput
```

NOTE: Replace {site_id} with your site ID.

Sample Request:

```
https://api.gc2.mist.com/api/v1/sites/aee83225-1773-4e55-af64-c8b5a86b1fa6/sle/site/
aee83225-1773-4e55-af64-c8b5a86b1fa6/metric/throughput/summary?start&end&duration
```

Sample Response:


```
{
    "start": 1727696554,
    "end": 1727782954,
    "sle": {
        "name": "throughput",
        "x_label": "seconds",
        "y_label": "Mbps",
        "interval": 3600,
        "samples": {
            "total": [
                1167.6,
                1229.75,
                1180.7333,
                1123.25,
                1135.45,
                1134.7333,
                1125.95,
                1132.2833,
                1109.35,
                1128.3334,
                1114.8167,
                1162.75,
                1122.4333,
                1149.5,
                1249.2667,
                1276.7833,
                1238.9667,
                1254.6333,
                1257.2,
                1244.3833,
                1217.2667,
```

```
                1237.2333,
                1239.7833,
                758.88336
            ],
            "degraded": [
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0
            ],
            "value": [
                4416.122,
                4364.1084,
                4225.373,
                4254.209,
                4319.275,
                4330.7563,
                4346.148,
                4395.728,
                4414.8213,
                4376.623,
                4303.742,
                4317.217,
```

```
                4516.579,
                4510.3657,
                4462.7544,
                4514.023,
                4607.6772,
                4556.474,
                4582.5347,
                4593.3706,
                4579.6895,
                4564.7954,
                4442.6,
                4417.6904
            ]
        }
    },
    "impact": {
        "num_users": 0,
        "num_aps": 0,
        "total_users": 29,
        "total_aps": 4
    },
    "classifiers": [
        {
            "name": "capacity-excessive-client-load",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    1167.6,
                    1229.75,
                    1180.7333,
                    1123.25,
                    1135.45,
                    1134.7333,
                    1125.95,
                    1132.2833,
                    1109.35,
                    1128.3334,
                    1114.8167,
                    1162.75,
                    1122.4333,
                    1149.5,
                    1249.2667,
                    1276.7833,
                    1238.9667,
                    1254.6333,
                    1257.2,
                    1244.3833,
                    1217.2667,
                    1237.2333,
                    1239.7833,
                    758.88336
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 29,
                "total_aps": 4
            }
        },
        {
            "name": "device-capability",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    1167.6,
                    1229.75,
                    1180.7333,
                    1123.25,
                    1135.45,
                    1134.7333,
                    1125.95,
                    1132.2833,
                    1109.35,
                    1128.3334,
                    1114.8167,
                    1162.75,
                    1122.4333,
                    1149.5,
                    1249.2667,
                    1276.7833,
                    1238.9667,
                    1254.6333,
                    1257.2,
                    1244.3833,
                    1217.2667,
                    1237.2333,
                    1239.7833,
                    758.88336
                ],
```

```
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 29,
                "total_aps": 4
            }
        },
        {
            "name": "network-issues",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    1167.6,
                    1229.75,
                    1180.7333,
                    1123.25,
                    1135.45,
                    1134.7333,
                    1125.95,
                    1132.2833,
                    1109.35,
                    1128.3334,
                    1114.8167,
                    1162.75,
                    1122.4333,
                    1149.5,
                    1249.2667,
                    1276.7833,
                    1238.9667,
                    1254.6333,
                    1257.2,
```

```
                    1244.3833,
                    1217.2667,
                    1237.2333,
                    1239.7833,
                    758.88336
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 29,
                "total_aps": 4
            }
        },
        {
            "name": "coverage",
            "x_label": "seconds",
```

```
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    1167.6,
                    1229.75,
                    1180.7333,
                    1123.25,
                    1135.45,
                    1134.7333,
                    1125.95,
                    1132.2833,
                    1109.35,
                    1128.3334,
                    1114.8167,
                    1162.75,
                    1122.4333,
```

```
                    1149.5,
                    1249.2667,
                    1276.7833,
                    1238.9667,
                    1254.6333,
                    1257.2,
                    1244.3833,
                    1217.2667,
                    1237.2333,
                    1239.7833,
                    758.88336
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 29,
```

```
                "total_aps": 4
            }
        },
        {
            "name": "capacity-high-bandwidth-utilization",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    1167.6,
                    1229.75,
                    1180.7333,
                    1123.25,
                    1135.45,
                    1134.7333,
                    1125.95,
```

```
                    1132.2833,
                    1109.35,
                    1128.3334,
                    1114.8167,
                    1162.75,
                    1122.4333,
                    1149.5,
                    1249.2667,
                    1276.7833,
                    1238.9667,
                    1254.6333,
                    1257.2,
                    1244.3833,
                    1217.2667,
                    1237.2333,
                    1239.7833,
                    758.88336
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
```

```
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 29,
                "total_aps": 4
            }
        },
        {
            "name": "capacity-wifi-interference",
            "x_label": "seconds",
            "y_label": "user-minutes",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    1167.6,
```

```
                    1229.75,
                    1180.7333,
                    1123.25,
                    1135.45,
                    1134.7333,
                    1125.95,
                    1132.2833,
                    1109.35,
                    1128.3334,
                    1114.8167,
                    1162.75,
                    1122.4333,
                    1149.5,
                    1249.2667,
                    1276.7833,
                    1238.9667,
                    1254.6333,
                    1257.2,
                    1244.3833,
                    1217.2667,
                    1237.2333,
                    1239.7833,
                    758.88336
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                0,
                0,
                0,
                0,
                0,
                0
            ]
        },
        "impact": {
            "num_users": 0,
            "num_aps": 0,
            "total_users": 29,
            "total_aps": 4
        }
    },
    {
        "name": "capacity-non-wifi-interference",
        "x_label": "seconds",
        "y_label": "user-minutes",
        "interval": 3600,
        "samples": {
            "duration": [
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
                0,
```

```
                    0,
                    0,
                    0
                ],
                "total": [
                    1167.6,
                    1229.75,
                    1180.7333,
                    1123.25,
                    1135.45,
                    1134.7333,
                    1125.95,
                    1132.2833,
                    1109.35,
                    1128.3334,
                    1114.8167,
                    1162.75,
                    1122.4333,
                    1149.5,
                    1249.2667,
                    1276.7833,
                    1238.9667,
                    1254.6333,
                    1257.2,
                    1244.3833,
                    1217.2667,
                    1237.2333,
                    1239.7833,
                    758.88336
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 29,
                "total_aps": 4
            }
        }
    ],
    "events": []
}
```


- **Capacity:** capacity
- **Coverage:** coverage
- **Device Capability:** device-capability
- **Network Issues:** network-issues

Time to Connect: time-to-connect

GET Call::

```
GET /api/v1/sites/{site_id}/insights/time-to-connect
```

NOTE: Replace {site_id} with your site ID.

Sample Request:

```
https://api.gc2.mist.com/api/v1/sites/aee83225-1773-4e55-af64-c8b5a86b1fa6/sle/site/
aee83225-1773-4e55-af64-c8b5a86b1fa6/metric/time-to-connect/summary?start&end&duration
```

Sample Response:


```
{
    "start": 1727696603,
    "end": 1727783003,
    "sle": {
        "name": "time-to-connect",
        "x_label": "seconds",
        "y_label": "seconds",
        "interval": 3600,
        "samples": {
            "total": [
                52.0,
                32.0,
                6.0,
                15.0,
                2.0,
                24.0,
                7.0,
                3.0,
                13.0,
                13.0,
                49.0,
                15.0,
                29.0,
                62.0,
                38.0,
                4.0,
                null,
                1.0,
                1.0,
                2.0,
                null,
                null,
                9.0,
                36.0
            ],
```

```
            "degraded": [
                4.0,
                1.0,
                2.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                1.0,
                1.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                null,
                0.0,
                0.0,
                0.0,
                null,
                null,
                0.0,
                0.0
            ],
            "value": [
                1.0913653,
                0.5267187,
                6.8511662,
                0.20886666,
                1.885,
                0.038458332,
                0.059428573,
                0.039666668,
                0.044153847,
                0.27992308,
                0.09634693,
                0.0908,
                0.021689653,
                0.047080643,
                0.037736844,
                0.0205,
```

```
                null,
                0.223,
                0.027,
                0.344,
                null,
                null,
                0.15811113,
                0.07152778
            ]
        }
    },
    "impact": {
        "num_users": 4,
        "num_aps": 4,
        "total_users": 26,
        "total_aps": 4
    },
    "classifiers": [
        {
            "name": "dhcp-unresponsive",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    1.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    1.0,
                    1.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
                    36.0
                ],
                "degraded": [
                    1.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    1.0,
                    1.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 3,
                "num_aps": 3,
                "total_users": 26,
                "total_aps": 4
            }
        },
        {
            "name": "dhcp-stuck",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
                    36.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 26,
                "total_aps": 4
            }
        },
        {
            "name": "dhcp-nack",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

```
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
```

```
                    36.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 26,
                "total_aps": 4
            }
        },
        {
            "name": "association",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
```

```
                    3.0,
                    1.0,
                    2.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
```

```
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
                    36.0
                ],
                "degraded": [
                    3.0,
                    1.0,
                    2.0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 1,
                "num_aps": 3,
                "total_users": 26,
                "total_aps": 4
            }
        },
        {
```

```
            "name": "authorization",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
```

```
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
                    36.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
```

```
                "num_aps": 0,
                "total_users": 26,
                "total_aps": 4
            }
        },
        {
            "name": "IP-Services",
            "x_label": "seconds",
            "y_label": "attempts",
            "interval": 3600,
            "samples": {
                "duration": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ],
                "total": [
                    52.0,
                    32.0,
                    6.0,
                    15.0,
                    2.0,
```

```
                    24.0,
                    7.0,
                    3.0,
                    13.0,
                    13.0,
                    49.0,
                    15.0,
                    29.0,
                    62.0,
                    38.0,
                    4.0,
                    null,
                    1.0,
                    1.0,
                    2.0,
                    null,
                    null,
                    9.0,
                    36.0
                ],
                "degraded": [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
```

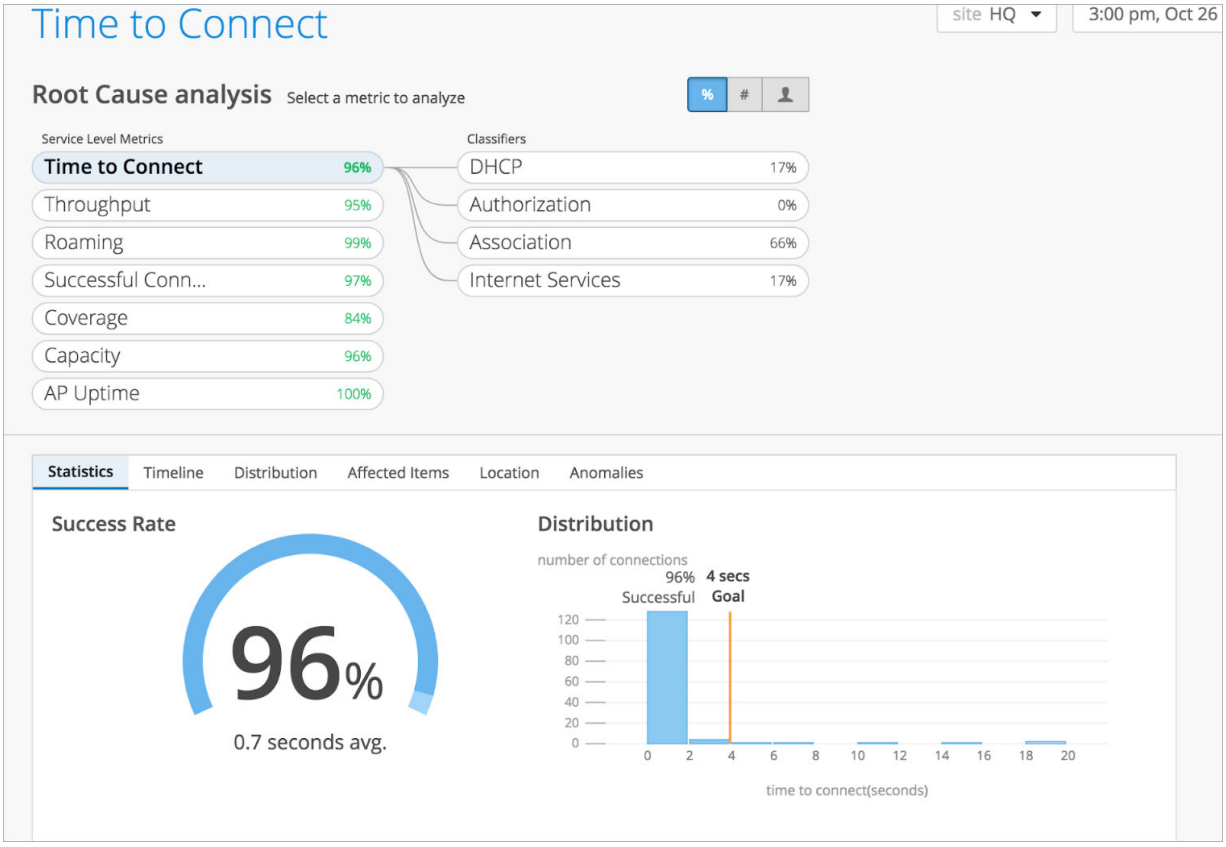
```
                    0,
                    0
                ]
            },
            "impact": {
                "num_users": 0,
                "num_aps": 0,
                "total_users": 26,
                "total_aps": 4
            }
        }
    ],
    "events": []
}
```


- **Association:** association
- **Authorization:** authorization
- **DHCP:** DHCP
- **Internet Services:** IP-Services\u000C

Calculating SLE Percentages

The SLE metric success rate is calculated as a percentage of how often the threshold was met during the selected timeframe. Classifiers are also calculated as percentages, but these values indicate their impact towards the parent failure.

For example, the below screenshot shows Time to Connect succeeded 96% of the time; all clients who successfully connected from 3:00-4:00pm completed the connection process within the four second threshold.



This metric's success rate (%) is derived from the "Metric Summary" API endpoint.

NOTE: Replace site_id with the actual site ID.

```
/api/v1/sites/:site_id/sle/site/:site_id/metric/time-to-connect/summary?
start=1540591200&end=1540594800

{

   "start": 1540591200,

   "end": 1540594800,

   "sle": {

       "x_label": "seconds",

       "y_label": "seconds",

       "interval": 600,
```

```
        “name”: “time-to-connect”,
        “samples”: {
            “degraded”: [
                0.0,
                0.0,
                3.0,
                0.0,
                3.0,
                0.0
            ],
            “total”: [
                19.0,
                14.0,
                34.0,
                8.0,
                20.0,
                43.0
            ]
        }
    }
}
```

The metric failure rate is calculated by dividing the failures (*sle.samples.degraded*) by the total (*sle.samples.total*). This is then translated to the success rate percentage. Using the above API response payload, the calculation would look as follows:

```
ceil(1-[(0.0+0.0+3.0+0.0+3.0+0.0)/(19.0+14.0+34.0+8.0+20.0+43.0)])*100=

ceil(1-[6/138])*100=

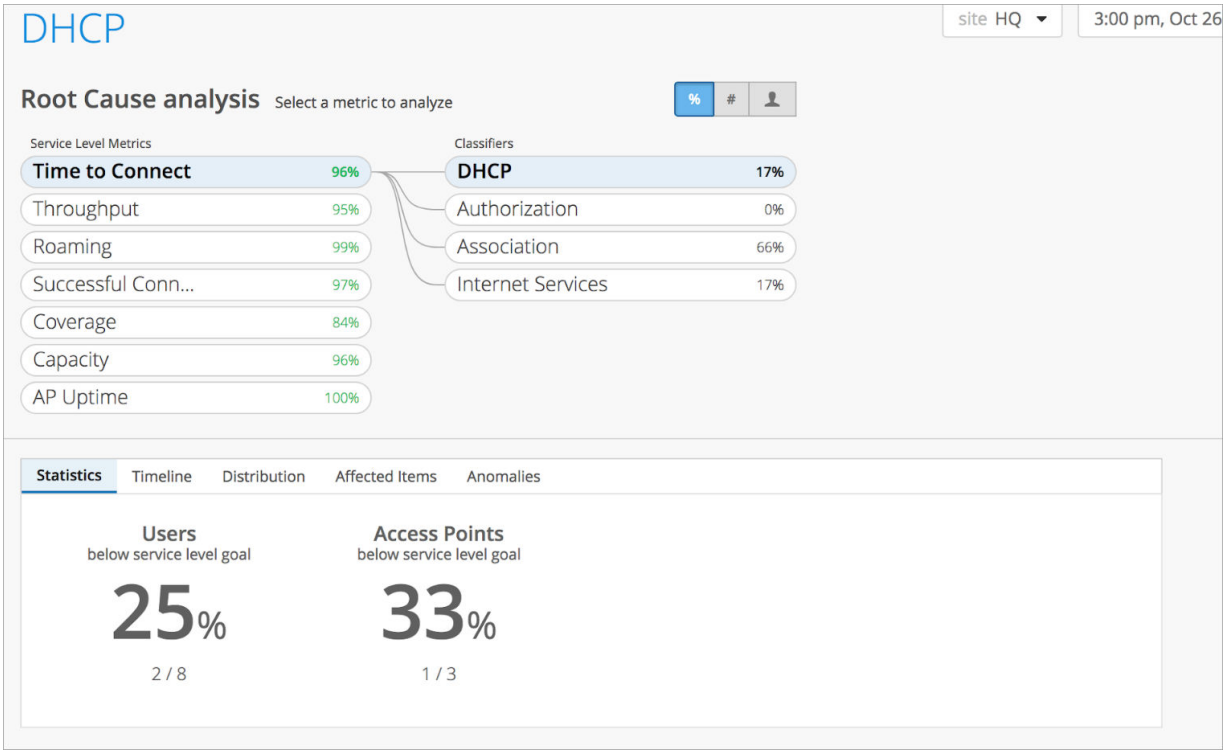
ceil(1-0.04347826086)*100=

ceil(0.95652173914)*100=

0.96*100=

=96%
```

This screenshot shows classifiers that contributed to the metric failures (DHCP, Authorization, Association, and Internet Services):



The classifier's impact (%) is derived from the same "Metric Summary" API endpoint.

NOTE: Replace site_id with the actual site ID.

```
/api/v1/sites/:site_id/sle/site/:site_id/metric/time-to-connect/summary?
start=1540591200&end=1540594800

{

   “start”: 1540591200,

   “end”: 1540594800,

   “classifiers”: [

       {

           “name”: “DHCP”,

           “samples”: {

               “degraded”: [

                   0,

                   0,

                   0,

                   0,

                   1.0,

                   0

               ]

           }

       },

       {
```

```
            “name”: “authorization”,
            “samples”: {
                “degraded”: [
                    0,
                    0,
                    0,
                    0,
                    0,
                    0
                ]
            }
        },
        {
            “name”: “association”,
            “samples”: {
                “degraded”: [
                    0,
                    0,
                    3.0,
                    0,
                    1.0,
```

```
                    0

                ]

            }

        },

        {

            “name”: “IP-Services”,

            “samples”: {

                “degraded”: [

                    0,

                    0,

                    0,

                    0,

                    1.0,

                    0

                ]

            }

        }

    ]

}
```

The classifier impact is calculated by dividing the classifier's failures (*classifiers[n].samples.degraded*) by the sum of all failures (*classifiers[].samples.degraded*). This is then translated to a percentage. Using the above API response payload, the calculation for DHCP would look as follows:

```
ceil([0+0+0+0+1.0+0]/[(0+0+0+0+1.0+0)+(0+0+0+0+0+0)+(0+0+3.0+0+1.0+0)+(0+0+0+0+1.0+0)])*100=

ceil(1/[1.0+0+4.0+1.0])*100=

ceil(1/6)*100=

ceil(0.16666666666)*100=

0.17*100=

=17%
```

Monitoring SLEs

SLE data is updated every ten minutes. However, the SLEs are prone to fluctuations when monitoring at this granularity. Thus, it is recommended to query for 1-hour intervals using the explicit start and end time, and polling only once per hour.

Configure Assurance Services with APIs

SUMMARY

Get started with some of the API calls that you can use to configure various Juniper Mist™ assurance services.

IN THIS SECTION

Juniper Mist offers many cloud-based assurance services to help you identify the cause of poor user experience on the network. The assurances are automated Mist tools that provide insight into users' connection data and can even proactively make configuration adjustments to the network. Mist offers

services such as Access Assurance, WAN Assurance, Wired Assurance, and Wireless Assurance. SLEs and Insights are two features that are included in each of these licensed offerings.

For example, the APIs associated with Access Assurance enable you to configure Authorization Policies and more. APIs with Wired Assurance enable you to configure devices using Network Templates and individual switch settings. Wireless Assurance enables you to configure wireless templates or individual access points (APs). WAN Assurance allows you to configure Organization Hub Profiles and more.

This topic provides examples as well as links to more information on how you can use the different Mist assurances with APIs.

NOTE: Replace placeholder values with actual values, such as your organization ID, site ID, and so on.

Access Assurance

Org Nac Tags

- **Create Org Nac Tag**

```
POST
/api/v1/orgs/{org_id}/nactags
```

  - Payload: JSON Formatted Payload
  - Required: org id

- **Get List of Org Nac Tags**

```
GET
/api/v1/orgs/{org_id}/nactags
```

  - Payload: JSON Formatted Payload
  - Required: org id

- **Get Org Nac Tag**

```
GET
/api/v1/orgs/{org_id}/nactags/{nactag_id}
```

  - Payload: JSON Formatted Payload

  - Required: org id, nac tag id

- **Update Org Nac Tag**

```
PUT
/api/v1/orgs/{org_id}/nactags/{nactag_id}
```

  - Payload: JSON Formatted Payload
  - Required: org id, nac tag id

- **Delete Org Nac Tag**

```
DELETE
/api/v1/orgs/{org_id}/nactags/{nactag_id}
```

  - Payload: None
  - Required: org id, nac tag id

  For more information on Nac Tags, see Nac Tags Overview.

To see additional types of configuration you can do for Access Assurance using APIs, see:

- Nac Rules Overview
- Site Nac Clients Overview
- Org Nac Clients Overview

WAN Assurance

Org Networks

- **Create Org Network**

```
POST
/api/v1/orgs/{org_id}/networks
```

  - Payload: JSON Formatted Payload
  - Required: org id

- **List Org Networks**

```
GET
/api/v1/orgs/{org_id}/networks
```

  - Payload: JSON Formatted Payload
  - Required: org id

- **Get Org Network**

```
GET
/api/v1/orgs/{org_id}/networks/{network_id}
```

  - Payload: JSON Formatted Payload
  - Required: org id, network id

- **Update Org Network**

```
PUT
/api/v1/orgs/{org_id}/networks/{network_id}
```

  - Payload: JSON Formatted Payload
  - Required: org id, network id

- **Delete Org Network**

```
DELETE
/api/v1/orgs/{org_id}/networks/{network_id}
```

  - Payload: None
  - Required: org id, network id

To see additional types of configuration you can do for WAN Assurance using APIs, see:

- Org Device Profiles (Org Hub Profiles)
- Org Gateway Templates
- Org Idp Profiles

- Org Services (Org Applications)
- Org Service Policies (Org Application Policies).
- Sites WAN Clients
- Org WAN Clients
- Org Devices and Org Stats Devices (WAN Edge Gateways)

Wired Assurance

Org Network Templates

- **Create Org Network Template**

```
POST
 /api/v1/orgs/{org_id}/networktemplates
```

  - Payload: JSON Formatted Payload
  - Required: org id

- **Get List of Org Network Templates**

```
GET
/api/v1/orgs/{org_id}/networktemplates
```

  - Payload: JSON Formatted Payload
  - Filters: org id

- **Get Org Network Template**

```
GET
/api/v1/orgs/{org_id}/networktemplates/{networktemplate_id}
```

  - Payload: JSON Formatted Payload
  - Required: org id, network template id

- **Update Org Network Template**

```
PUT
/api/v1/orgs/{org_id}/networktemplates/{networktemplate_id}
```

  - Payload: JSON Formatted Payload (only changes/additions needed)
  - Required: org id, network template id

- **Delete Org Network Template**

```
DELETE
/api/v1/orgs/{org_id}/networktemplates/{networktemplate_id}
```

  - Payload: None
  - Required: org id, network template id

For more information, see Network Templates Overview.

Site Settings

Documentation is located here: Sites Setting Overview.

- **Get Site Settings**

  Includes both switching and non-switching related settings

- ```
  GET
  /api/v1/sites/{site_id}/setting
  ```

  - Payload: JSON Formatted Payload
  - Required: site id

- **Update Site Settings**

```
PUT
/api/v1/sites/{site_id}/setting
```

  - Payload: JSON Formatted Payload (only changes/additions needed)
  - Required: site id
```

- To link this site to a template, add the “networktemplate_id” key with the value of the ID for the network template to apply.

NOTE: There is no POST or DELETE for site settings. The only way to create site settings is to create a new site, and the only current way to delete them is to delete the site.

- **Get Site EVPN Topology**

```
GET
/api/v1/sites/{site_id}/evpn_topologies/{evpn_topology_id}
```

  - Payload: None
  - Required: site id, evpn topology id

  For more information, see EVPN Topologies Overview.

- **Count Site Wired Clients**

```
GET
/api/v1/sites/{site_id}/wired_clients/count
```

  - Payload: JSON Formatted Payload
  - Required: site id

- **Search Site Wired Clients**

```
GET
/api/v1/sites/{site_id}/wired_clients/search
```

  - Payload: JSON Formatted Payload
  - Required: site id

Switch Settings

Documentation is located here: Site Devices Overview.

- **Get Switch Settings**

```
GET
/api/v1/sites/{site_id}/devices/{device_id}
```

  - Payload: JSON formatted payload
  - Required: site id, device id

- **Update Switch Settings**

```
PUT
/api/v1/sites/{site_id}/devices/{device_id}
```

  - Payload: JSON Formatted Payload (only changes/additions needed)
  - Required: site id, device id

To see additional types of configuration you can do for Wired Assurance using APIs, see API Devices Overview.

Wireless Assurance

Org WLANs

- **Create Org WLAN**

```
POST
/api/v1/orgs/{org_id}/wlans
```

  - Payload: JSON Formatted Payload
  - Required: org id

- **List Org WLANs**

```
GET
/api/v1/orgs/{org_id}/wlans
```

  - Payload: JSON Formatted Payload
  - Required: org id

- **Get Org WLAN**

```
GET
/api/v1/orgs/{org_id}/wlans/{wlan_id}
```

  - Payload: JSON Formatted Payload
  - Required: org id, wlan id

- **Update Org WLAN**

```
PUT
/api/v1/orgs/{org_id}/wlans/{wlan_id}
```

  - Payload: JSON Formatted Payload
  - Required: org id, wlan id

- **Delete Org WLAN**

```
DELETE
/api/v1/orgs/{org_id}/wlans/{wlan_id}
```

  - Payload: None
  - Required: org id, wlan id

  For more information on what you can do with WLANs using APIs, see Org WLANs Overview.

To see additional types of configuration you can do for Wireless Assurance using APIs, see:

Organization

- Org Device Profiles
- Org RF Templates
- Org Labels
- Org Policies
- Org Wireless Clients
- Org Access Points Stats
- Org PSKs

Site

- Site WLANs
- Site PSKs
- Site Labels
- Site Policies
- Site Wireless Clients

Device

- Access Points

Get Started with the RESTful API

IN THIS SECTION

Use Postman to Make Your First API Call

SUMMARY

Postman is a platform that is designed to make it easy to work with APIs. This topic walks you through how to use Postman to make your first API call.

IN THIS SECTION

Postman is an API platform that makes it easy for you to make and test API calls. You can use Postman for most everything API related, such as testing webhooks, sending and receiving data from a WebSocket server, and more. This topic covers the steps you need to complete in Postman prior to making your first API call and how to use Postman to make your first API call to the Juniper Mist RESTful API.

See

Postman Setup

To use Postman, you can use the Postman website or download the Postman application as described in Download Postman.

1. Sign in to Postman (or create an account) from the Postman website or application. This allows you to save your environment. See the Create Your Environment section further down in this topic.
2. Once your account is created, you get access to your workspace. This is where you can save your API calls and configure your environment to interact with the Mist API.

Import the Mist API Collection

Next, import the Mist API Collection. Juniper Mist has built a list of Postman API calls that you can import directly into your Postman workspace. This list is maintained and matches what the API documentation lists.

1. Navigate to the Juniper Mist Postman collections page and select the **Mist Cloud APIs** collection.
2. Once the collection is opened, click on **Fork** as described in Fork collections and environments in Postman. This enables you to create a copy of the collection in your own workspace and still receive updates when the main collection is updated.
3. In the top left corner of Postman, you should see that the collection has now been forked into your workspace. Expand the collection and its subsections to see how the RESTful API calls are organized.

Create Your Environment

A postman environment allows you to store variables in a profile that you can reuse across multiple API calls and collections. You must create an environment and define variables before you begin making API calls in Postman.

1. Create and name your environment as described in Create an Environment.
2. Define variables in your Postman environment by entering them into the table as described in Add Environment Variables.
    - In order to interact with the Mist RESTful API, you must set the following variables:

Table 6: Environment Variables

Variable	Description
`host`	This is the URL of the Mist API endpoint. For exam See API Endpoints and Global Regions for the full l
`apitoken`	This is the Mist API token required to authenticate create a Mist API token for REST API, see Create A
`org_id`	This is the UUID of your Mist organization. For mo Find Your Organization ID and Get Org.
`site_id`	This is the UUID of a specific Mist site. For more ir Info.

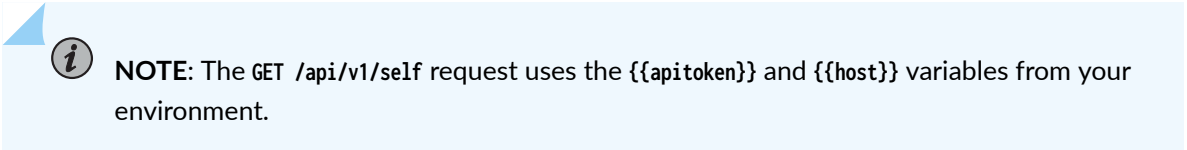
3. Once you have entered all your variables, select **Save**.
4. Apply the environment to your Postman workspace by navigating to the **Environment** drop-down menu in the top right corner of the page, then select your newly created environment.

Test Your First API Call

Now that you have set up Postman with your collections and environments, it's time to make your first API call. In this example, you make an API call to test the connection to the API. This is done using the `GET /api/v1/self` call.

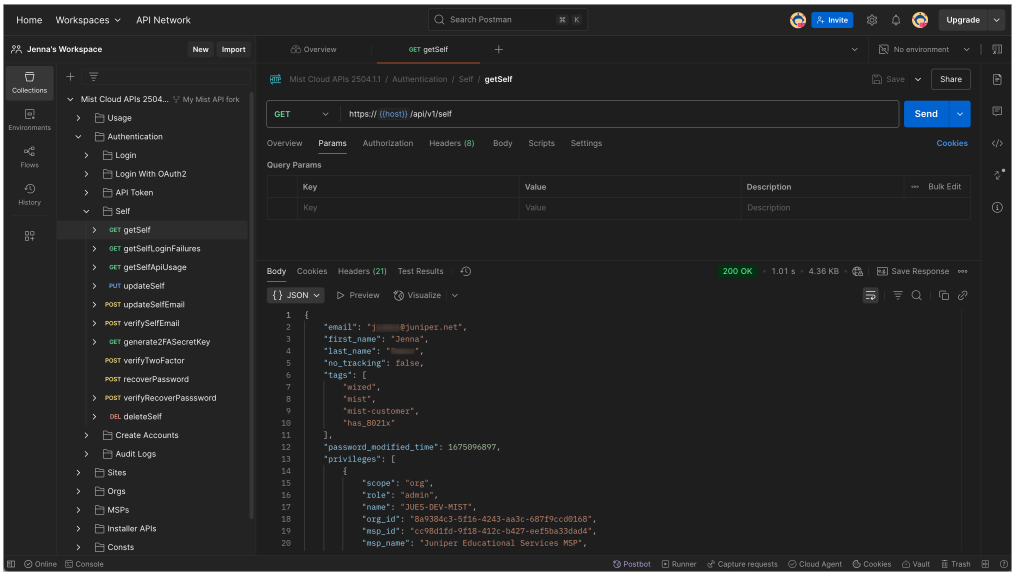
In order to test your first API call to the Juniper Mist API, follow these steps:

1. In Postman, select **Collections** from the left menu, then navigate to **Authentication** > **Self**. Finally, select the `GET getSelf` request.
2. Select the **Send** button.



NOTE: The `GET /api/v1/self` request uses the `{{apitoken}}` and `{{host}}` variables from your environment.

You should receive a `200` response code from the Mist API indicating that the request was successful. In response to the `GET /api/v1/self` request, you receive information about your account and privileges in a JSON payload.



If you **did not** receive a response, select the **Console** button in the bottom left corner of Postman to gather more information about the sent request and received response. Study the HTML response code to understand why you received the response. Below are some example error codes:

- **`Error 404`**—This means that the endpoint URL does not exist. Validate that your **`{{host}}`** variable is configured properly. For example, an additional `/` in the URL would result in an `Error 404` response code.

- **`Error 401`**—This means that you are not authorized to send the API call. Review the response of the Mist API to understand what is not setup properly. Validate that your **`{{apitoken}}`** variable is configured properly.

For more information about HTML response codes, see REST API HTTP Response Codes.

You have just sent your first API call to the Mist API. Now you are ready to explore and test other calls. Here are a few suggestions that you can test next:

- `Orgs / Sites / getOrgSites`

- `Orgs / Devices Stats / getOrgDevicesStats`

- `Orgs / Templates / getOrgsTemplates`

- `Orgs / Inventory / getOrgInventory`

Additional RESTful API Documentation

SUMMARY

Know where to go for complete API reference information for Juniper Mist™.

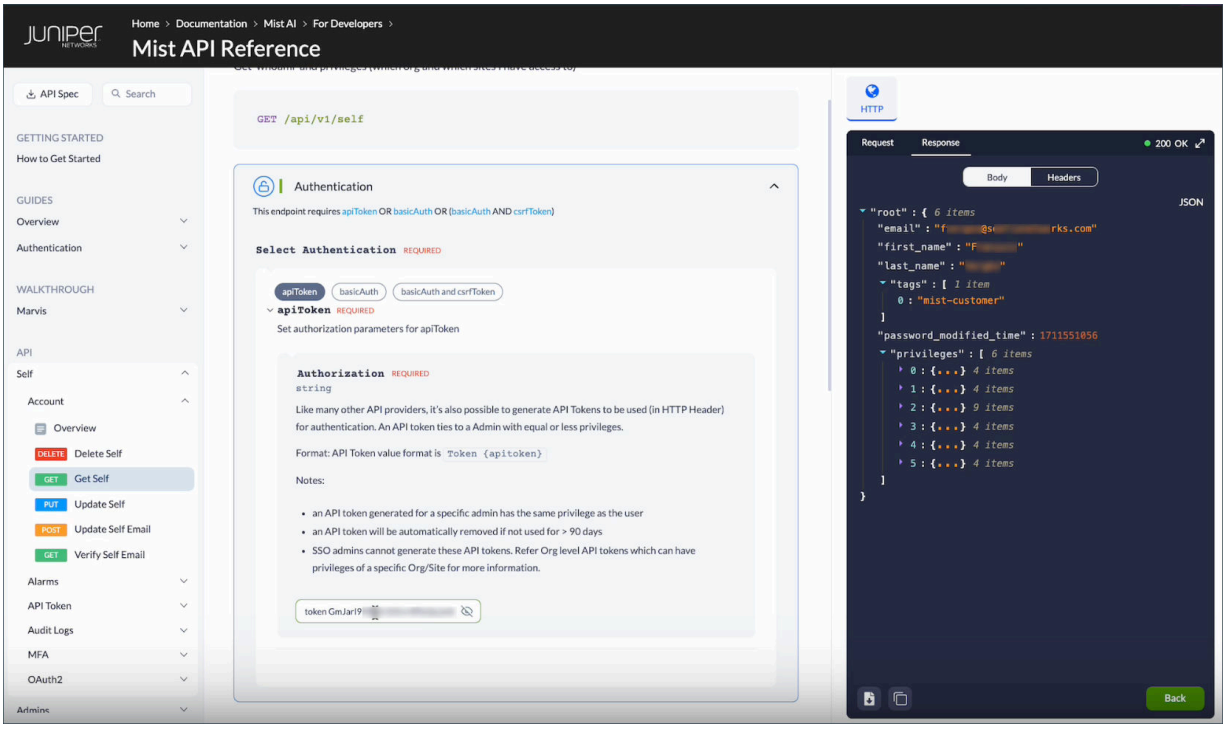
To access extensive API reference information, including parameter descriptions and the ability to test and make API requests, go to the Juniper Mist API Reference.

You can both test and make API calls directly from this website. For any endpoints that require authentication, you will see an Authentication section in the center of the screen which you will need to fill in with your credentials.

For example, to see which sites and organizations you have access to, navigate to **API** > **Self** > **Account** > **Get Self** and then expand the Authentication section in the center of the screen to enter your API token. Then select **Configure** at the lower-right to select the cloud for your global region. Finally, click **TRY IT OUT** to issue your API call. Then you can view the response.

NOTE: When entering your API token into the API Reference website, you must add the "token" key word in front of your token in order to perform authentication. You can also perform authentication using the **basicAuth** or **basicAuth and csrfToken** options. If you do not have an API token yet, you can use the **basicAuth** option and enter the same username and password that you use for the Mist UI.

ATTENTION: To enhance security and align with industry best practices, Mist will deprecate Basic Authentication for all use cases—including admin logins and scripts—effective September 2026. Before September 2026, all integrations must transition to token-based authentication to ensure uninterrupted access and support. See "Create API Tokens" on page 15.



Demo: A Non-Programmer Approach to API

SUMMARY

Watch this video to see how a non-programmer uses Postman to work with APIs.

This video shows the non-programmer way to approach the Juniper Mist™ API. Even though this example involves switching, you can use the same approach with everything covered in the Juniper Mist API, both wired and wireless.

In this example, Andy shows you how to use the Chrome developer tools to figure out exactly which API endpoint is being called and how to take advantage of that in Postman.

Video: Automation for Access Switching

API Use Cases

SUMMARY

There are a number of reasons why you would want to use APIs for automation and integration with Juniper Mist™. The following are just a couple of common examples.

IN THIS SECTION

API Use Case Examples

Source of Truth

As network operations continue to evolve, the Source of Truth (SoT) is becoming a more important concept. This SoT allows you to describe to an external software application how the network should be configured and have it directly interact with Mist for device configuration.

The source of truth can be something as simple as a set of YAML or JSON data files or as sophisticated as a dedicated application or a suite of applications that manage and automate your entire network. Configuring sites and devices via the API, based on data in the SoT, provides a level of control and can keep configurations consistent across sites and devices.

Service Oriented Configuration

Change management is a large part of network operations. Manual changes are both error prone and time consuming. Integration of change management into a platform like ServiceNow helps simplify common change management requests by letting changes flow through proper change management processes as well as reduce the human error of making the changes. The API for Wired Assurance enables these integrations between service platforms and the wired network configuration.

Continuing Efforts

While the API gives us some immediate opportunities to programmatically configure EX Switching, work is already progressing for additional functionality. For example, automating the migration Cisco IOS configurations as well as the automated migration of existing Juniper EX configurations to the Mist platform using existing software libraries and opensource tools.

Automatic Site Creation (Use Case)

SUMMARY

Watch a video demo to see how easily you can oboard dozens or hundreds of sites in minutes by using Python scripts and APIs.

Automation in a Juniper Mist network enables you and other administrators to make system changes automatically, while minimizing human error. It also enables you to make massively scalable changes that simply cannot be made from a GUI.

The video below shows an example of a script that creates five hundred sites in less than 5 minutes using a CSV file for input, a Juniper Mist device template, the Juniper Mist API, and Python:

Video: Automatic Site Creation Use Case

Renaming APs (Use Case)

SUMMARY

Read and follow this example to quickly rename your access points (APs) by using a Python script and APIs.

At times, you and other wireless administrators need to rename Juniper Mist network access points (APs) at a specific site. Consistent naming helps provide order and consistency in network inventory documentation. For example, a site may physically move to a new location and take on a new naming convention.

To perform this renaming task, in the Juniper Mist portal you can perform a bulk renaming action using a regular expression. This option is easy and readily available, but it can become complex if you have many devices to rename.

Another option is to create a Python script to rename the APs. This option simplifies the task of renaming many devices. The example script below illustrates the Python script renaming sequence. The

script renames specific APs that have already been claimed to a site. The script uses three files, as follows:

- The first file (config.json) contains the configuration variables.
- The second file (ap-names.csv) is a CSV file that contains the MAC address of the AP to be renamed as well as the AP's new name.
- The third file (main-rename-ap.py) is the Python script itself, which takes information from the other two files.

To use the script, you place all three files in the same working directory.

The first file, which contains the JavaScript Object Notation (JSON) formatted configuration, includes variables needed to connect to the API to make the required calls to find and rename the specified APs.

NOTE:

- When making any configuration changes using the API, make sure you understand the data that you are modifying. Also be sure to perform validation to ensure that things are still working properly.
- Replace placeholder values with actual values, such as your organization ID, site ID, AP name, and so on.

```
{
  "api": {
    "org_id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "token": "ApUYc...hsO",
    "mist_url": "https://{api-endpoint}.mist.com/api/v1/"
  },
  "site": {
    "id": "{site-id}"
  }
}
```

NOTE: In place of {api-endpoint}, use the API endpoint for your global region. See "API Endpoints and Global Regions" on page 13.

The Python script uses the contents of the CSV file to identify the AP (by the MAC address) and then rename it to the new name **ap-names.csv**.

```
name,mac
{ap-name1},aabbcc001122
{ap-name2},aabbcc001123
{ap-name3},aabbcc001124
```

By default, when you initially claim an AP, the AP takes the name of its own MAC address. The main Python script does the actual work of finding and renaming of the AP, which consists of the following functions:

1. (def)—Get the MAC address of the AP to rename.
2. Find the AP on the site.
3. Rename the AP.

The main function (def) uses the other functions to complete the task and loops, as necessary.

This is what the **main-rename-ap.py** script looks like:

```
#!/usr/bin/env python3

"""
The format of the MAC address part of this CSV file must be the following:
aabbccddeeff
"""

import argparse
import time
import json
import requests
import csv
from pprint import pprint


def is_ap_in_site(configs: dict, ap_mac: str):
    """
    This function checks for an AP assigned to a site.

    Parameters:
        - configs: dictionary containing all configuration information
```

```
        - site_id: ID of the site we will to assign the AP to

    Returns:
        - the ID of the AP if the AP is assigned to the site
        - the current name of the AP
    """
    api_url = f"{configs['api']['mist_url']}sites/{configs['site']['id']}/devices"
    headers = {'Content-Type': 'application/json',
               'Authorization': 'Token {}'.format(configs['api']['token'])}
    response = requests.get(api_url, headers=headers)

    if response.status_code == 200:
        devices = json.loads(response.content.decode('utf-8'))
        for device in devices:
            if device['mac'] == ap_mac:
                return (device['id'], device['name'])
    else:
        print('Something went wrong: {}'.format(response.status_code))
    return (None, None)


def rename_ap(configs: dict, ap_id: str, new_ap_name: str, ap_old_name: str):
    """
    This function renames an AP.

    Parameters:
        - configs: dictionary containing all configuration information
        - ap_id: ID of the AP device object
        - new_ap_name: name to apply to the AP
        - ap_old_name: current name of the AP
    """
    api_url = f"{configs['api']['mist_url']}sites/{configs['site']['id']}/devices/{ap_id}"
    headers = {'Content-Type': 'application/json',
               'Authorization': 'Token {}'.format(configs['api']['token'])}
    body = {}
    body['name'] = new_ap_name
    response = requests.put(api_url, headers=headers, data=json.dumps(body))

    if response.status_code == 200:
        device = json.loads(response.content.decode('utf-8'))
        print(f"{device['mac']} renamed from {ap_old_name} to {device['name']}")
    else:
        print(f"AP ID: {ap_id}\tSomething went wrong: {response.status_code}")
```

```
def retreive_ap_mac_list(csv_filename: str):
    """
    This function converts the content of the CSV file to a Python dictionary.

    Parameters:
        - csv_filename: the name of the comma separated value file.

    Returns:
        - A dictionary containing the content of the CSV file
    """
    ap_csv = csv.DictReader(csv_filename)
    ap_list = []
    for line in ap_csv:
        ap_list.append(line)
    return ap_list


def main():
    """
    This script batch renames the APs listed in a CSV file.
    """
    parser = argparse.ArgumentParser(description='Configures a Mist AP for an APoS site survey')
    parser.add_argument('config', metavar='config_file', type=argparse.FileType(
        'r'), help='file containing all the configuration information')
    parser.add_argument('ap_list', metavar='aps_names', type=argparse.FileType(
        'r'), help='csv file containing new AP names')
    args = parser.parse_args()
    configs = json.load(args.config)
    ap_mac_list = retreive_ap_mac_list(args.ap_list)

    for ap in ap_mac_list:
        ap_id, ap_old_name = is_ap_in_site(configs, ap['mac'])
        if ap_id:
            rename_ap(configs, ap_id, ap['name'], ap_old_name)
        else:
            print(f"AP {ap['name']} is not part of site {configs['site']['id']}")


if __name__ == '__main__':
    start_time = time.time()
    print('** Start the batch renaming of APs...\n')
```

```
    main()
    run_time = time.time() - start_time
    print("\n** Time to run: %s sec" % round(run_time, 2))
```

To run the script, call the `main-rename-ap.py` script and provide the config.json filename and the ap-names.csv filename as arguments. For example:

```
user@linux-host} python main-rename-ap.py config.json ap-names.csv
```

After you run the script, the output should look something like this:

```
** Start the batch renaming of APs...

aabbcc001121 renamed from OLD-AP-01 to NEW-AP-01
aabbcc001122 renamed from OLD-AP-02 to NEW-AP-02
aabbcc001122 renamed from OLD-AP-03 to NEW-AP-03

** Time to run: 3.24 sec
```

You can also check the Juniper Mist portal and verify the changes by checking the device's inventory. Automation is not limited to RESTful APIs and Python. You can find other automation options such as WebSocket and webhook API usage and tools to help in the development process.

RELATED DOCUMENTATION

RESTful API Overview | 4

Create API Tokens | 15

REST API HTTP Response Codes | 22

https://www.rfc-editor.org/rfc/rfc9110.html

BLE Import (Use Case)

SUMMARY

Read and follow this example to import Bluetooth Low Energy (BLE) assets and give them useful, descriptive names by using Python scripts and APIs.

IN THIS SECTION

When you set up and activate location-based services with Juniper Mist Asset Visibility, admins like you can see all BLE clients and assets. You can also see their precise locations, right on an indoor floor plan or map.

For sites that use BLE asset tags, it's handy to track these devices by giving them easily readable names that provide some context. You can add and display these names individually within the Juniper Mist portal, but if you have a lot of assets to manage, doing it one by one can be quite time consuming. An easier way to do this is to run a script to import BLE assets and assign them a name in bulk.

For this use case, you need to:

- Enable Asset Visibility in the Site Settings for each site.
- Make sure that you have an active license for Asset Visibility.
- Make sure that you have placed compatible APs on the floor plan.

This use case involves two scripts: **main.py** and **mist-client.py**. A third file, a CSV file called **assets.csv**, contains the BLE assets and their corresponding names.

Here's the order of steps you follow when you need to import BLE assets:

1. Start by updating the **main.py** script with your Mist API token, Mist site universally unique identifier (UUID), and the region (or cloud) in which your organization is hosted.
2. Next, you add, remove, or inspect the BLE devices and their names within the **assets.csv** file.
3. Run the **main.py** script, which will use the CSV content to create the assets in Juniper Mist.

Main.py Script

A lot happens behind the scenes in the **main.py** script. The script imports the data from the CSV file and converts the data into JSON format. Then, for each device, the script creates a BLE asset and triggers

the **mist-client.py** script. This **mist-client.py** script does the work of making all the necessary calls to the Juniper Mist API.

NOTE: Replace placeholder values with actual values, such as your API token, site ID, and so on.

```
#!/usr/bin/python
#
# main.py
#
# Update main.py with your Mist API Token and Juniper Mist site UUID.
#
# Inspect the "assets.csv" file to update the assets being created, then run this exercise to
automatically create BLE assets from CSV.

import sys, csv, json, re
from mist_client import Admin  # Import the Juniper Mist client

mist_api_token = ''  # Your Juniper Mist API token goes here. Documentation: https://
api.mist.com/api/v1/docs/Auth#api-token

site_id = ''  # Your Site ID goes here

csv_file = 'assets.csv'


# Convert CSV file to JSON object.
def csv_to_json(file):
    csv_rows = []
    with open(file) as csvfile:
        reader = csv.DictReader(csvfile)
        title = reader.fieldnames

        for row in reader:
            csv_rows.extend([ {title[i]: row[title[i]] for i in range(len(title))} ])

    return csv_rows


 # Creates BLE assets using the given CSV file and the Juniper Mist API
def create_assets(admin, data):
```

```
    for d in data:
        try:
            mac = re.sub(r'[^0-9a-fA-F]', '', d.get('MAC', '')).lower()

            assert len(mac) == 12
            assert mac.isalnum()
        except:
            print('Invalid MAC {}, skipping this Asset.'.format(d.get('MAC', '(none)')))

            continue

        # Build the asset payload
        payload = {'name': d['Name'].strip(), 'mac': mac}
        # Create the BLE Asset and note the targeted region (or cloud)
        api_url = 'https://api.mist.com/api/v1/sites/{}/assets'.format(site_id)
        (success, result) = admin.post(api_url, payload)

        # Add the new BLE Asset to the return list
        if result == None:
            print('Failed to create BLE Asset {}'.format(mac))
        else:
            if success:
                print('Created BLE Asset \"{}\" ({})'.format(result.get('name', '(unnamed)'),
result['mac']))
            else:
                print('BLE Asset \"{}\" already exists with MAC Address {}'.format(d.get('Name',
'(unnamed)'), mac))


# Main function
if __name__ == '__main__':
    # Check for required variables
    if mist_api_token == '':
        print('Please provide your Mist API token as mist_api_token')
        sys.exit(1)
    elif site_id == '':
        print('Please provide your Mist Site UUID as site_id')
        sys.exit(1)

    # Create Mist client
    admin = Admin(mist_api_token)

    print()
```

```
    print('Converting file {} to JSON...\n'.format(csv_file))

    # Convert CSV to valid JSON
    data = csv_to_json(csv_file)
    if data == None or data == []:
        print('Failed to convert CSV file to JSON. Exiting script.')
        sys.exit(2)

    print(json.dumps(data, indent=4, sort_keys=True))

    print('\n=====\n')

    # Create the BLE Assets from CSV file
    print('Creating BLE Assets...\n')

    create_assets(admin, data)

    print()
```

Mist_client.py Script

The **mist_client.py** script functions like a regular RESTful client for interacting with the Juniper Mist API. The script makes API calls based on the input from the CSV file and the output of the **main.py** script. The **mist-client.py** script also error-checks the HTTP response from the API and displays the output, as follows:

```
#!/usr/bin/python
#
# mist_client.py
#
# Mist API client session.

import json, requests


# Mist CRUD operations
class Admin(object):
    def __init__(self, token=''):
        self.session = requests.Session()
        self.headers = {
```

```
            'Content-Type': 'application/json',
            'Authorization': 'Token ' + token
        }

    def get(self, url):
        session = self.session
        headers = self.headers

        print('GET {}'.format(url))
        response = session.get(url, headers=headers)

        if response.status_code != 200:
            print('Failed to GET')
            print('\tURL: {}'.format(url))
            print('\tResponse: {} ({})'.format(response.text, response.status_code))

            return False

        return json.loads(response.text)

    def post(self, url, payload, timeout=60):
        session = self.session
        headers = self.headers

        #print('POST {}'.format(url))
        response = session.post(url, headers=headers, json=payload)

        if response.status_code == 400:
            return (False, response.text)
        elif response.status_code != 200:
            '''
            print('Failed to POST')
            print('\tURL: {}'.format(url))
            print('\tPayload: {}'.format(payload))
            print('\tResponse: {} ({})'.format(response.text, response.status_code))
            '''

            return (False, None)

        return (True, json.loads(response.text))
```

Assets.csv

In this example, the **assets.csv** file resides in the same directory as the **mist_client.py** and **main.py** files. The following example shows how to format the CSV file with the name of the BLE asset and its associated MAC address:

```
Name,MAC
Amber Badge,aa:bb:cc:dd:ee:ff
Mark Badge,11-22-33-44-55-66
Invalid MAC,xx.yy.zz.xx.yy.zz
```

Automation goes beyond just using RESTful APIs and Python. Other options like WebSocket and webhook APIs are available. You can explore these other options for automation purposes.

Use the REST API to Add ACL Tags to a Switch (Use Case)

SUMMARY

Read this topic to understand the role Access Control Lists (ACLs) play in API access control.

IN THIS SECTION

You can use Access Control Lists (ACLs) in the API to allow or deny traffic between clients on a connected switch. You must configure separate rules for both inbound and outbound traffic. ACLs are mainly used to control intra-VLAN traffic, whereas any inter-VLAN traffic is filtered by the stateful rules of the router or firewall.

NOTE: You can only configure and manage ACLs in the API at this time. This functionality is not fully available in the Mist portal, as configuring from the portal requires you to configure filters on the RADIUS server first, and then create the switch policies in the Mist portal. See *Firewall Filters*.

Add ACL Tags to a Switch

In the Mist API, you can apply ACLs to all switches in a site, rather than having to manage ACLs at the device level. You can assign permissions to switches using ACL policies and ACL tags. ACL tags enable

you to define permissions within them, then you assign the tags where access control is needed in the policy. In other words, ACL tags are reusable network objects that can be referenced in ACL policies.

NOTE: ACL configuration in the Mist API is only available for switches at this time.

Let's say you want to control the traffic that will be forwarded by a switch, for example, if you want to allow or deny certain traffic from a wired client to the rest of the network. To do this, you can add ACL tags in the ACL policy to control the allowed or denied traffic sources and destinations from which the switch is allowed to forward traffic.

You can set these rules in the API by adding the source and destination tags within the `acl_tags` object referenced within the `acl_policies` and specify the action you want the switch to take (allow or deny) when the traffic matches the ACL tags. Remember, you must configure separate rules for both inbound and outbound traffic due to the stateless nature of ACLs.

You can configure this from the Mist API Reference by navigating to **Update Site Device** > **Body** > **Device Switch**.

Below is a sample payload of the ACL policy configuration and the definitions for each of the ACL tags are featured therein:

```
{
    // Defines the ACL Tags - these are reusable network objects that can be referenced in
policies
    "acl_tags": {
        "iot_device": {             // Defines the Source ACL Tag - represents IoT devices
            "type": "port_usage",   // Type indicates this tag is based on port profile usage
            "port_usage": "iot",    // On which Port Profiles the ACL must be applied
            "subnets": [            // Source IP addresses/ranges that match this tag
                "192.168.1.30/32"
            ],
            "specs": [              // Additional specifications for traffic matching
                {}
            ]
        },
        "iot_network": {            // Defines the Destination ACL Tag - represents allowed IoT
services
            "type": "resource",     // Type indicates this tag represents a network resource/
destination
            "subnets": [            // Destination IP addresses/ranges for this tag
                "192.168.1.0/24"
            ],
```

```
            "specs": [              // Specifies the traffic characteristics - what protocols/
ports are allowed
                {
                    "protocol": "tcp",
                    "port_range": "1883"
                }
            ],
            "ether_types": []       // Ethernet frame types (empty means all types allowed)
        }
    },
    // Defines the ACL Policies - rules that govern traffic flow
    "acl_policies": [
        {
            "name": "iot_policy",
            "src_tags": [           // Source tags that this policy applies to
                "iot_device"        // References the iot_device tag defined above
            ],
            "actions": [            // What actions to take when traffic matches the source tags
                {
                    "dst_tag": "iot_network",   // Destination tag for this action
                    "action": "allow"           // Action to take (allow/deny) for matching
traffic
                },
                {
                    "dst_tag": "any",           // any is implicitly defined as all destinations
                    "action": "deny"            // Action to take (allow/deny) for matching
traffic
                }
            ]
        }
    ]
}
```

For more information on the available ACL tags you can add, see Acl Tag.

RELATED DOCUMENTATION

Create API Tokens | 15

No Link Title

RELATED DOCUMENTATION

Automatic Site Creation (Use Case) | **157**

Renaming APs (Use Case) | **157**

BLE Import (Use Case) | **163**

3

CHAPTER

Webhooks

SUMMARY

Use the information in this chapter to get started with webhooks.

IN THIS CHAPTER

Video Overview

Video: Webhooks Intro

What Do You Want to Do?

Table 7: Top Tasks

If you want to...	Use these resources:
Learn about webhooks *What are webhooks? Learn about the message flow, source addresses, webhook hierarchy, webhooks topics, alerts, and messages.*	"Webhooks Overview" on page 174
Configure Webhooks for the alerts that you want to receive *Use the API or the UI.*	"Configure Webhooks from the API" on page 222 "Configure Webhooks from the Juniper Mist Portal" on page 213
See webhooks in action *Learn more by studying a use case involving Juniper Mist location services.*	"Configure Zone Entry and Exit Events (Use Case)" on page 237

Webhooks Overview

SUMMARY

Start getting familiar with webhooks and how they compare to APIs.

IN THIS SECTION

You can configure webhooks to get real-time notifications as events happen across your Juniper Mist organization or within a particular Juniper Mist site.

You may know webhooks as user-defined HTTP callbacks, HTTP posts, or HTTP notifications. These notifications include event details that you can use in your own applications or third-party software.

As you begin working with webhooks, it can be helpful to compare them with APIs. APIs work on a pull or polling model. You create an API call, and Juniper Mist responds with the requested data. In contrast, webhooks work on a push model. After you configure a webhook, you receive data as events occur.

Org Webhooks enable real-time data from the organization to be pushed to a provided URL. To learn more, see Org Webhooks.

Site Webhooks enable real-time data from a specific site to be pushed to a provided URL. To learn more, see Site Webhooks.

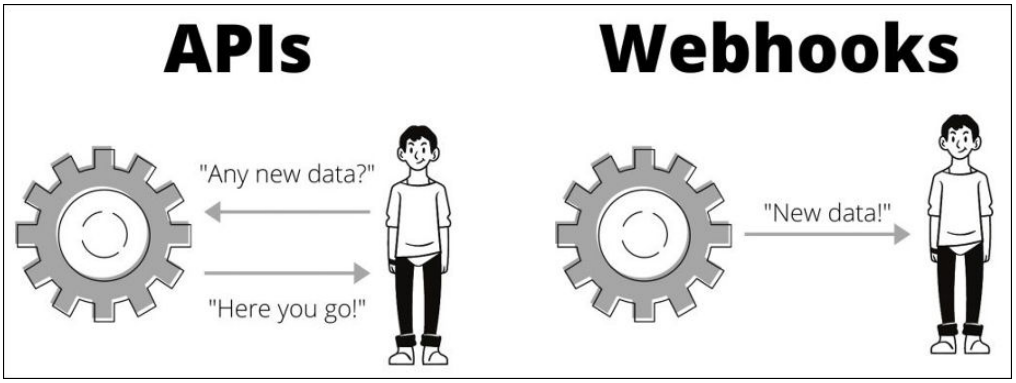


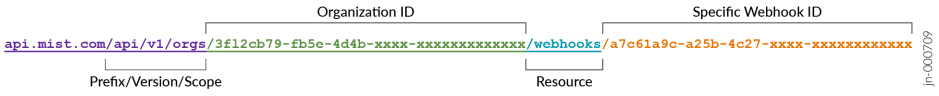
Table 8: REST APIs and Webhooks Comparison

	REST APIs	Webhooks
Model	Pull or polling model	Push model
Limit	5000 per organization per hour	No limit
Operations	Pull statistics, create configuration, update configuration	Push alerts and key statistics
Coverage	100 percent	Alerts, key events, or location
Recommended use cases	Automation, configuration	Integration with third-party monitoring tools or business applications

NOTE: The concepts of alerts and alarms are considered interchangeable. However, when configuring an alert, you should make note of the alert or alarm syntax and use what is displayed.

API Call Structure for Webhooks

You can configure webhooks to obtain real-time data and notification of events as they occur within your Mist organization or within a particular site. After you configure a webhook, you can issue an API call where you specify a particular webhook to obtain data. The following image is an example of such an API call. The call specifies the API endpoint (prefix), the organization ID, the resource (`webhooks`), and webhook ID (`a7c61a9c-a25b-4c27-xxxx-xxxxxxxxxxxx`).



Webhook Message Flow

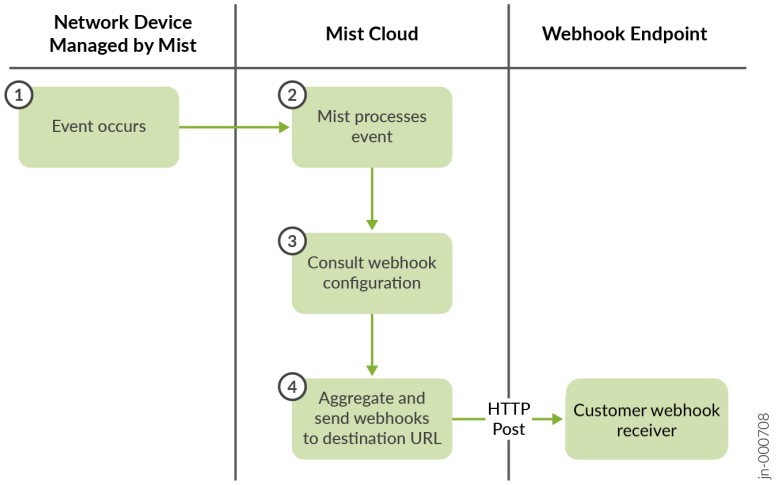
SUMMARY

Learn how Juniper Mist sends event information with webhooks.

As opposed to clients polling and pulling information from the API, Juniper Mist pushes webhooks to a target webhook URL. There, underlying services can collect, store, or parse the message for specific data and then perform actions on the infrastructure automatically.

Not all events are aggregated. Some messages are directly sent. For the aggregated webhooks, it takes less than five seconds between when the event occurs on the network device and when the event is aggregated and sent to the destination URL.

This image shows the stages of data flow for webhooks.



1. An event occurs on a network device that Juniper Mist manages.
2. Juniper Mist processes the event and checks the current webhook configuration for a match.
3. If Juniper Mist finds a match, it sends the webhook to the endpoint URL as configured.
4. If multiple events for a specific topic occur within a certain timeframe, Juniper Mist aggregates the events and sends them in a single message.

Webhook Source Addresses

SUMMARY

Configure your firewall to allow traffic from the Webhooks Source IP addresses for your cloud instance in order to receive event information via webhooks.

IN THIS SECTION

IP Source Addresses for Juniper Mist Cloud Instances

When you configure webhooks to use in your Juniper Mist network, you need to specify the URL of a public-facing webhook receiver where Mist can send messages. In other words, enable these source IP addresses on your firewall which are used to send out the API stream from the Mist cloud. The source IPs for Webhooks are Static IP Addresses.

To ensure that your server receives the HTTP messages from the Juniper Mist cloud, configure your firewall to allow traffic from the Webhooks Source IP addresses for your cloud instance.

NOTE: These are static IP addresses.

Table 9: Webhooks Source IP Address by Region

Region	Source IPs		
Global 01	54.193.71.17	54.215.237.20	
Global 02	34.94.226.48/28 (34.94.226.48 - 34.94.226.63)		
Global 03	34.231.34.177	54.235.187.11	18.233.33.230
Global 04	34.152.4.85	35.203.21.42	34.152.7.156

Table 9: Webhooks Source IP Address by Region *(Continued)*

Region	Source IPs		
Global 05	35.192.224.0/29 (35.192.224.0 - 35.192.224.7)		
EMEA 01	3.122.172.223	3.121.19.146	3.120.167.1
EMEA 02	35.234.156.66		
EMEA 03	51.112.15.151	51.112.76.109	51.112.86.222
EMEA 04	34.166.152.112/29 (34.166.152.112 - 34.166.152.119)		
APAC 01	54.206.226.168	13.238.77.6	54.79.134.226
APAC 02	34.47.180.168/29 (34.47.180.168 - 34.47.180.175)		
APAC 03	34.104.128.8/29 (34.104.128.8 - 34.104.128.15)		

You can use the Juniper Mist webhook `ping` to test connectivity from your Mist cloud through your network infrastructure. For more information, see "Testing Webhooks" on page 228, Ping Site Webhook, and Ping Org Webhook.

Webhook Hierarchy

SUMMARY

Understand the relationship between organization- and site-level webhooks. Get familiar with the webhook topics that you can configure at each level.

IN THIS SECTION

Webhook Hierarchy Overview

Juniper Mist has two configuration hierarchies for webhooks: the organization level and site level. The configuration method is the same for both; however, not all webhooks are available at both levels.

- All webhooks that are available at the organization level are also available for specific site-level webhooks.
- The available site-level webhooks are not available at the organization level.

Be aware of the hierarchy when configuring webhooks.

As an example, if you have two sites and you configure the Alerts webhook at the organization level, you will receive all organization-level alerts for both sites.

Conversely, if you do not configure the Alerts webhook at the organization level and only configure the Alerts webhook on one of the two sites, you will receive alerts only for that single site.

Finally, if you configure the Alerts webhook at the organization level and a single site, the webhook receiver will receive duplicate messages (assuming they are sent to the same receiver URL).

You can configure multiple webhook receivers for a single webhook (and its topics) from the API. It is recommended to have a single webhook that includes all the available topics and then parse the information that you want from messages received by the webhook receiver.

Organization Webhooks

An Org Webhook is a configuration that enables organization data to be pushed to a specified URL. You can customize org webhooks using the Mist RESTful API, which enables you to obtain data from a specific organization.

You can configure these topics at the organization level:

Table 10: Organization Webhooks

Topic	Description
Alerts	Juniper Mist-defined alarm events configurable on a per-site basis in the alert framework.
Audits	A topic that tracks configuration changes made from the Juniper Mist dashboard.
Client Join	The webhook that Juniper Mist triggers whenever a client joins a wireless network.
Client Information	A topic that includes information about each client.
Client Sessions	The topic that includes information about the client sessions.
Device Events	A topic that is specific to events that occur on devices (currently AP, switch, and gateway)
Device Up/Downs	The topic that generates a message when a device starts up or goes down.
Juniper Mist Edge Events	The topic that generates messages for a Juniper Mist Edge port and link status changes and link aggregation control protocol (LACP) port and link changes.

Site Webhooks

IN THIS SECTION

A Site Webhook is a configuration that enables site data and events to be pushed to a specified URL. You can customize site webhooks using the Mist API so that you can obtain data from a particular site.

You can configure these topics at the site level. The topics are classified under two categories—Standard and Advanced. You can select either one topic under the Advanced category or multiple topics under the Standard category for a webhook. You cannot select topics from both categories.

Location Webhooks

To use location webhooks, you must upload a floorplan with accurate AP placement through the Juniper Mist portal to correlate the client data. For more information, see the *Floorplan Setup Overview* in the Juniper Mist Location Services Guide.

Note that when you select the Location Coordinates topic, a new WiFi Client topic appears, which includes subtopics for Connected, Unconnected, and Centrak webhooks. For Centrak users, this means you can decouple unconnected Wi-Fi from your Centrak Wi-Fi location tags (OUI: 00:12:b8), that is, only send Centrak data instead of every unconnected Wi-Fi client.

Table 11: Location Webhooks

Topic	Description
Entry/Exit Events • Location Zone • Proximity Zone • Virtual Beacon	Data pushed any time a client enters or exits one of these areas.
Occupancy Alerts	Alerts on a configured zone occupancy threshold being exceeded.
SDK Client Scan Data	Specific data about a client that isn't available without installing an application (using the SDK) on the client itself.
X/Y Coordinates • Named Assets • SDK Clients	Data sent at regular intervals of: • Approximately 2 seconds for Named Assets. • Approximately 1 second for SDK Clients.

Table 11: Location Webhooks *(Continued)*

Topic	Description
Asset Raw	This topic will be deprecated. Use the "BLE Asset RSSI" on page 183 topic.

Network Service Webhooks

Table 12: Network Service Webhooks

Topic	Description
Latency	A webhook that provides DHCP, DNS, and authentication latency information aggregated across all the client devices at the site level.

NOTE: You'll need a Marvis subscription to subscribe to this webhook.

Infrastructure Webhooks

Table 13: Infrastructure Webhooks

Topic	Description
Alerts	Juniper Mist-defined alarm events configurable on a per-site basis in the alert framework.
Audits	A topic that tracks configuration changes made from the Juniper Mist dashboard.
Client Information	The webhook that includes information about each client.

Table 13: Infrastructure Webhooks *(Continued)*

Topic	Description
Client Join	The webhook that Juniper Mist triggers whenever a client joins a wireless network.
Client Sessions	The webhook that includes information about the client sessions.
Device Events	A topic that is specific to events that occur on devices (currently AP, switch, and gateway)
Device Up/Downs	The topic that generates a message when a device starts up or goes down.
Mist Edge Events	The topic that generates messages for a Juniper Mist Edge port and link status changes and link aggregation control protocol (LACP) port and link changes.

BLE Asset RSSI

The BLE Asset RSSI topic sends telemetry BLE data based on the named asset.

Webhook Topics

SUMMARY

Examine the various webhook topics that are available for Juniper Mist™ organizations and sites.

IN THIS SECTION

After you enable webhooks at the organization, site, or both levels, select the topics that you want to receive messages for.

Use the following tables to learn more.

Table 14: Both Organization and Site Topics

Topic	Purpose and Payload Details
alerts (alarms)	User-selected alarms for devices and infrastructure, Marvis actions, and security. To find alerts in the Juniper Mist portal, select **Monitor** > **Alerts** > **Alerts Configuration** from the left menu. Examples • Devices and infrastructure: Device down, device restarted, VPN peer down, ARP failure, DNS failure. • Marvis: Faulty cable, failed AP health check, poor Wi-Fi coverage, port flapping, bad WAN uplink. • Security: KRACK attack, TKIP ICV attack, rogue client, rogue AP, honeypot. Depending on the type of alarm, the payload may include details such as the IDs of the organization and site, the event type, the severity, and the count of each event type during the aggregation interval. To see which alarms are available and get examples of their payloads, from a REST API client, issue the following API call: `GET /api/v1/const/alarm_defs`
audits	A topic that all Mist configuration changes trigger this topic. The payload includes details such as the administrator's name and username, the device ID, the type of change, and the timestamp for the change.
client-info	Use this webhook to get wireless client information events. The payload includes details about the client such as the hostname, IP address, MAC address, org and site ID, and timestamp of the IP assignment.

Table 14: Both Organization and Site Topics *(Continued)*

Topic	Purpose and Payload Details
client-join	Client connections only. The payload includes details such as the IDs of the organization and site; the MAC address and name of the AP that the client connected to; and the WLAN ID, band, SSID, RSSI, and timestamp for the connection.
client-sessions	Client session information. The payload includes details such as the MAC address and name of the AP that the client roamed to or disconnected from, the WLAN ID, the band, the device family (“Mac,” “iPhone,” “Apple watch”), the client manufacturer and model, the timestamps for the connection and disconnection, the float duration, the RSSI, and the termination reason.
device-events	Use this webhook to get a list of possible events affecting access points, switches, and gateways. This topic includes events such as port up or down, AP power changes, and channel changes. The payload for this webhook may include details such as the IDs of the organization and site, the MAC address and name of the device, and the timestamp of the event, but can vary depending on the exact type of device event. To see which alarms are available and get examples of their payloads, issue the following API call from a REST API client: `GET /api/v1/const/device_events`
device-updowns	Device disconnects, reconnects, and restarts. The payload includes details such as the IDs of the organization and site, the type of event, and the device’s MAC address and name. To configure `device-updowns` in the Juniper Mist portal, select **Monitor** > **Alerts** > **Alerts Configuration** from the left menu. Here you can configure more granular control of the types of devices and alert thresholds.

Table 14: Both Organization and Site Topics *(Continued)*

Topic	Purpose and Payload Details
mxedge-events	Use this webhook to get a list of possible Mist Edge events, including details such as Juniper Mist Edge physical and LACP link status.
nac-accounting	Use this webhook to get a list of any possible Juniper Mist network access control (NAC) account events (ACCOUNTING_START, ACCOUNTING_STOP, ACCOUNTING_UPDATE). The payload includes details such as timestamp, AP, client_ip, SSID, username, client_type, client_mac, nas_vendor, site_id, rx_pkts, and tx_pkts.
nac-events	This webhook triggers with any Juniper Mist network access control (NAC) events. The payload may include details such as auth_type, bssid, idp_role, random_mac, resp_attrs, but can vary depending on the type of nac event.
ping	A ping event that goes to the webhook URL. The payload includes the site ID, the webhook ID and name, and the timestamp.

Table 15: Site-Only Topics

Topic	Purpose and Payload Details
asset-raw-rssi	Replaces deprecated topic named asset-raw. Raw data from packets emitted by named and filtered assets. The payload includes details such as the IDs of the organization, site, map, AP, antenna, and asset; the AP location; the RSSI; the beacon UUID and manufacturer; the service packets; and data.

Table 15: Site-Only Topics *(Continued)*

Topic	Purpose and Payload Details
client-latency	DHCP, DNS, and authentication latency information aggregated across all the client devices at the site level. Latency data is provided for a 10-minute window at 10-minute intervals. We recommend that you use this webhook to receive network latency data instead of polling through the API. **NOTE**: You'll need a Marvis subscription to subscribe to this webhook.
discovered-raw-rssi	Raw data from packets emitted by passive BLE. The payload includes details such as the IDs of the organization, site, map, reporting AP, and antenna; the MAC address; the asset ID, manufacturer, and UUID of the asset/beacon; and the RSSI.
guest-authorizations	Data for guest clients when they authorize at a WLAN. The payload includes details such as AP, auth_method, authorized_expiring_time, carrier, client, company, email, mobile, name, org_id, site_id, sms gateway, sponsor_email, ssid, and wlan_id.
location	Location data for Juniper Mist SDK clients, wireless clients, and assets. Data includes details such as the site and map IDs, X and Y coordinates, timestamp, the type of client, and the client ID or MAC address.
location-asset	Location data for Juniper Mist assets (BLE client devices). Data includes details such as the site and map IDs, X and Y coordinates, timestamp, the type of asset, and the asset's MAC address.

Table 15: Site-Only Topics *(Continued)*

Topic	Purpose and Payload Details
location-centrak	CenTrak data sent by Mist APs to centrak servers via webhooks. The data includes X,Y coordinates, timestamp, site ID, and Wi-Fi beacon extended info. Different fields will be present depending on the alarm event type.
location-client	Location data for Juniper Mist SDK clients, and wireless clients. Data includes details such as the site and map ID, X and Y coordinates, timestamp, the type of client, and the client MAC address.
location-sdk	Location data for Juniper Mist SDK clients. Data includes details such as map ID, the type of client, and X and Y coordinates.
location-unclient	Location data for Juniper Mist SDK clients or wireless clients when the client device disconnects from the network or is no longer detected in a specified location. The payload data includes details such as map ID, X and Y coordinates, Wi-Fi beacon extended info, and the client MAC address.
occupancy-alerts	Occupancy status based on the configured occupancy limits. The payload includes details such as the IDs of the organization, site, map, and zone; the event (“COMPLIANCE-VIOLATION” or “COMPLIANCE-OK”); the timestamp; the occupancy limit; and the current occupancy.

Table 15: Site-Only Topics *(Continued)*

Topic	Purpose and Payload Details
rssi-zone	Devices near an access point. This topic provides information about devices entering or exiting a zone around an AP where the RSSI received by the AP is above a configurable threshold. You can create a new RSSI zone for each AP and include the RSSI Zone name and an RSSI Zone threshold.
sdkclient-scan-data	Location data for Juniper Mist SDK clients. The data includes details such as the band, channel, SSID, BSSID, and RSSI for the connection; the AP's MAC address; list of neighboring APs; the timestamp when the client was last seen; and the timestamp for the scan.
vbeacon	Devices near a virtual beacon. This topic provides information about devices in proximity to a configured vbeacon. Use this webhook with the Juniper Mist SDK.
wifi-conn-raw wifi-unconn-raw	Raw data from packets emitted by connected and unconnected devices. The payload includes details such as the IDs of the organization, site, map, and reporting AP; the location coordinates of the AP; the RSSI and band; and the payload from the Wi-Fi beacon. It does not include the client's location data.
zone	Location data for virtual beacons on your floorplan, pushed when a client enters or exits a zone. The data includes details such as the site, map, and zone IDs, the timestamp, the trigger (enter or exit), the client type (SDK client, wireless client, or asset), and the ID or MAC address of the client.

Webhooks and Alerts

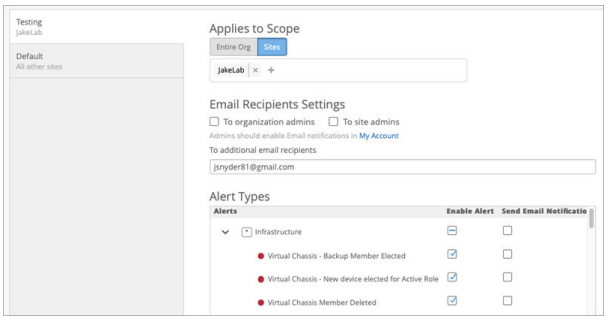
SUMMARY

Learn about the various alerts and alarms that you can enable in Juniper Mist™.

IN THIS SECTION

Configuring Alerts

You can configure alerts for an entire organization, single sites, or multiple sites from the Alerts Configuration page in the portal.



NOTE: To find this page, select **Monitor** > **Alerts** > **Alerts Configuration** from the left menu of the Juniper Mist portal.

All the alerts visible here are available to send an alert webhook by simply enabling the alert.

The alerts are broken down by color based upon severity, as follows:

- Red—Critical
- Orange—Warning
- Blue—Informational

The alarms are also categorized into these groups:

- Infrastructure—Infrastructure alarms don't keep state. They are based directly off device events. When you monitor devices from infrastructure alarms, you typically either treat each event as a standalone event, or you match stateful device changes.
- Marvis—Marvis events are events identified under Marvis Actions. These events are generally stateful. Inside their payload is a key called `details`. Under **`details`** you can see `state` and the values: **`open`** or **`validated`**.
  - **`open`** means this issue is currently happening.
  - **`validated`** means that Marvis has validated that the issue is resolved. After the issue is deemed to be validated, the same webhook type will be set with the updated state.

    Because of the AI nature of Marvis actions, Marvis requires sufficient data to ensure that these alarms are accurate and actionable. Marvis needs to accumulate enough data to eliminate false positives. This requirement results in a varying number of times for the events to arrive.
- Security—Most of the events in security are single-time events. These alerts will detect only specific attacks and don't determine if the attack is active. Rogue APs are rate-limited to reporting once every 10 hours. Rogue clients and Honeypot AP events are sent once every 10 minutes.

The following alerts also have configurable failure thresholds:

- ARP Failure
- DHCP Failure
- DNS Failure
- Device Offline

For information about configuring alerts, see the Alert Configuration information in the Juniper Mist Network Monitoring Guide.

Webhook Alert Types

Table 16: Webhook Alert Table

Alert/Webhook Name	Group	Category	Description	Triggering Mechanism	Comments
adhoc_network	Security	AP	Adhoc network detected	One or more APs detected an unauthorized adhoc network.	

Table 16: Webhook Alert Table *(Continued)*

Alert/Webhook Name	Group	Category	Description	Triggering Mechanism	Comments
air_magnet_scan	Security	AP	Air Magnet Scan detected	Someone is running Air Magnet scan for RF analysis.	
ap_bad_cable	Marvis	AP	Bad Ethernet cable connected to a Juniper AP	Frequent ethernet disconnects, restarts, increasing ethernet errors, connecting at 100Mbps	Req SUB-VNA
ap_offline	Marvis	AP	Offline (Marvis)	• Site down—All APs lose connection around the same time. • Switch down/issue—All APs on the same switch lose connection around the same time. • Locally online—AP is heard locally but lost cloud connection. • Locally offline—AP is not heard locally and lost cloud connection.	Req SUB-VNA
arp_failure	Marvis	connectivity	Site-wide wireless connection failures	Sudden increase in failures across the site OR 100% failures on a server/ WLAN/AP	Req SUB-VNA
authentication_failure	Marvis	Connectivity	Site-wide wireless and wired connection failures	Sudden increase in failures across the site OR 100% failures on a server/switch/ WLAN/VLAN/AP	Req SUB-VNAOR SUB-SVNA
bad_cable	Marvis	Switch	Faulty cable connected to a Juniper switchport	Port errors, power draw without ethernet link, increase in bytes out and 0 in (and vice versa)	Req SUB-VNA

Table 16: Webhook Alert Table *(Continued)*

Alert/Webhook Name	Group	Category	Description	Triggering Mechanism	Comments
bad_wan_uplink	Marvis	Router	Underperforming/ problematic interface (SRX, SSR)	Latency, jitter, packet loss, output drops & drop in transmit packets	Req SUB-WNA
beacon_flood	Security		Fake AP Flooding detected - a flood of new BSSIDs	The number of new SSIDs scanned by an AP exceeds the defined threshold during a defined time frame.	
bssid_spoofing	Security	AP	BSSID Spoofing detected	A device with signal strength of -30dBm or worse is broadcasting the same BSSID as an AP with a good signal strength.	
device_down	Infrastructure	AP	Device offline	An AP disconnects from the cloud for longer than the configured threshold.	
device_restarted	Infrastructure	AP	Device restarted	An AP restarts.	
dhcp_failure	Marvis	Connectivity	Site-wide wireless and wired connection failures	Sudden increase in failures across the site OR 100% failures on a server/WLAN/ VLAN/AP.	Req SUB-VNAOR SUB-SVNA
disassociation_flood	Security	AP	Disassociation Attack detected	Juniper Mist detects a DoS attack in which the attacker disassociates a victim device from an AP by using a specific disassociation frame as specified under IEEE 802.11.	
dns_failure	Marvis	Connectivity	Site-wide wireless connection failures	Sudden increase in failures across the site OR 100% failures on a server/ WLAN/AP.	Req SUB-VNA

Table 16: Webhook Alert Table *(Continued)*

Alert/Webhook Name	Group	Category	Description	Triggering Mechanism	Comments
eap_dictionary_attack	Security	AP	EAP Dictionary Attack detected	Multiple password failures in which someone attempts to guess a password by trying different dictionary words.	
eap_failure_injection	Security	AP	EAP Failure Injection detected	Someone sends fake EAP failures.	
eap_handshake_flood	Security	AP	EAP Handshake Flood detected	Some client or simulator generates a floods of EAPOL messages requesting 802.1x authentication.	
eap_spoofed_success	Security	AP	EAP Spoofed Success detected	Someone sniff EAP packets and tries to send fake EAP success.	
eapol_logoff_attack	Security	AP	EAPOL-Logoff Attack detected	Some client or simulator is sending excessive EAP logoff messages.	
essid_jack	Security	AP	ESSID Jack detected	Some client or simulator tries to send a broadcast probe request.	
excessive_client	Security	AP	Excessive Clients detected	The number of clients associated with an AP exceeds the configured threshold.	
excessive_eapol_start	Security	AP	Excessive EAPOL-Start detected	Some client or simulator is sending excessive EAP START messages.	
gateway_down	Infrastructure	SRX	WAN Edge offline	An SRX is offlin.	
gw_bad_cable	Marvis	Router	Faulty cable connected to a Juniper gateway (SRX only) port	Interface stat errors, input/ output bytes being 0	Req SUB-WNA

Table 16: Webhook Alert Table *(Continued)*

Alert/Webhook Name	Group	Category	Description	Triggering Mechanism	Comments
gw_dhcp_pool_exhausted	Infrastructure	SRX	WAN Edge DHCP Pool Exhausted	WAN Edge DHCP pool has been exhausted,	
gw_negotiation_mismatch	Marvis	Router	Difference in MTU packet size seen in the network (SRX only)	Packets being fragmented, MTU errors.	Req SUB-WNA
health_check_failed	Marvis	AP	Unhealthy APs to be replaced	Failure of auto-remediation/ self-healing on an AP.	Req SUB-VNA
honeypot_ssid	Security	AP	Honeypot SSID	Unauthorized APs advertising your SSID.	
idp_attack_detected	Security	SRX/SSR	IDP attack detected	SRX or Session SMart Router reports IDP_ATTACK_LOG_EVENT type events.	
infra_arp_failure	Infrastructure	AP	Gateway Arp failure	The ARP request for the default gateway is not receiving any response.	
infra_dhcp_failure	Infrastructure	AP	DHCP Failure	More than 10 clients are impacted by a failing/ unresponsive DHCP server within a window of 10 minutes.	
infra_dns_failure	Infrastructure	AP	DNS Failure	More than 10 clients are impacted by a failing/ unresponsive DNS server within a window of 10 minutes, an email will be triggered for this event.	

Table 16: Webhook Alert Table *(Continued)*

Alert/Webhook Name	Group	Category	Description	Triggering Mechanism	Comments
insufficient_capacity	Marvis	AP	AP(s) with low Wi-Fi capacity	After RRM changes, one or more clients have heavy consumption that results in high AP channel utilization.	Req SUB-VNA
insufficient_coverage	Marvis	AP	Areas around AP(s) with consistent poor Wi-Fi coverage	After RRM changes, clients still have consitently low RSSI.	Req SUB-VNA
krack_attack	Security	AP	Replay Injection detected - KRACK Attack	One or more APs detect KRACK attack attempts.	
loop_detected_by_ap	Infrastructure	Wireless	AP has detected loop via reflection	An AP recieves a frame that it sent out.	
missing_vlan	Marvis	Switch	VLAN configured on AP missing on switch port or upstream	An AP observes traffic on each VLAN and compares between APs on the same switch and other APs in the site.	Req SUB-VNAOR SUB-SVNA
monkey_jack	Security	AP	Monkey Jack detected	An AP detects a Man In the Middle attack attempt.	
negotiation_mismatch	Marvis	Switch	Difference in settings between a wired client & connected port	Duplex mismatch and/or auto-negotiation failing	Req SUB-VNA
non_compliant	Marvis	AP	APs with mismatched firmware	APs have a different firmware version than most other APs of that model model at that site.	Req SUB-VNA
out_of_sequence	Security	AP	Out of Sequence detected	Excessive out of sequence packets.	

Table 16: Webhook Alert Table *(Continued)*

Alert/Webhook Name	Group	Category	Description	Triggering Mechanism	Comments
port_flap	Marvis	Switch	Port constantly going up & down	Port flapping with high frequency and continuously.	Req SUB-VNA
repeated_auth_failures	Security	AP	Clients with Repeated Client Authentication Failures	A client faces continues client authentication failures due to an unreachable RADIUS server, wrong shared secret etc.	
rogue_ap	Security	AP	Rogue AP detected	Juniper Mist detects an AP not claimed into your organization but connected on the same wired network.	
rogue_client	Security	AP	Client Connection to rogue AP detected	A Client associates to a Rogue AP (an AP not claimed into your organization but connected to the same wired network).	
ssid_injection	Security	AP	SSID Injection detected: Detects malicious looking SSID names with possible code injection in name	Juniper Mist detects potential code injection language in an SSID name.	
sw_alarm_chassis_partition	Infrastructure	Switch	Switch Storage Partition Alarm	Partition usage is high.	
sw_alarm_chassis_pem	infrastructue	Switch	Switch PEM Alarm	PEM issues, fault slot,high CPU, issues with CB, and so on.	
sw_alarm_chassis_poe	Infrastructure	Switch	Junos POE Controller Alarm	Hardware issues.	
sw_alarm_chassis_psu	Infrastructure	Switch	Junos Power Supply Alarm	Missing power supply.	

Table 16: Webhook Alert Table *(Continued)*

Alert/Webhook Name	Group	Category	Description	Triggering Mechanism	Comments
sw_bad_optics	Infrastructure	Switch	Switch Bad Optics	Bad transceiver.	
sw_bgp_neighbor_state_changed	Infrastructure	Switch	BGP Neighbor State Changed	BGP peering goes up or down.	
sw_bpdu_error	Infrastructure	Switch	Switch BPDU Error	Possible bridging loop.	
sw_dhcp_pool_exhausted	Infrastructure	Switch	Switch DHCP pool has been exhausted,	The switch's DHCP pool has been exhausted.	
switch_down	infrastructue	Switch	Switch offline	A switch is offline.	
switch_restarted	Infrastructure	Switch	Switch restarted	A switch restarted.	
switch_stp_loop	Marvis	Switch	Same frame is seen by a switch multiple times	Frequent STP topology changes along with sudden increase in TX/RX.	Req SUB-VNA
tkip_icv_attack	Security	AP	TKIP ICV Attack	An AP detects TKIP MIC failures in excess of the configured threshold.	
url_blocked	Security	SRX/SSR	URL blocked	SRX or SSR reports WEBFILTER_URL_BLOCKED type events.	
vc_backup_failed	Infrastructure	Switch	Virtual Chassis - Backup Member Elected		
vc_master_changed	Infrastructure	Switch	Virtual Chassis - New device elected for Active Role		
vc_member_added"	Infrastructure	Switch	Adding a new VC member	A new VC member was added.	

Table 16: Webhook Alert Table *(Continued)*

Alert/Webhook Name	Group	Category	Description	Triggering Mechanism	Comments
vc_member_deleted	Infrastructure	Switch	Virtual Chassis Member Deleted	A VC member was deleted.	
vendor_ie_missing	Security	AP	Mist vendor IE missing in beacon or probe response	Impersonation of sanctioned Mist APs.	
vpn_path_down	Marvis	Router	VPN peer path down (SSR only)	100% failure of a peer path.	Req SUB-WNA
vpn_peer_down	Infrastructure	SRX	VPN Peer Down	An IPSec tunnel goes down for WAN interfaces between hub and spoke.	
WAN Edge Offline	Infrastructure	SSR	WAN Edge offline	A WAN Edge device is offline.	
watched_station	Security	AP	Active Watched Station detected	Juniper Mist detects a client or station that is listed in the Watched Station list.	
zero_ssid_association	Security	AP	Zero SSID Association Request detected	An AP scans a beacon that contains a zero length SSID.	

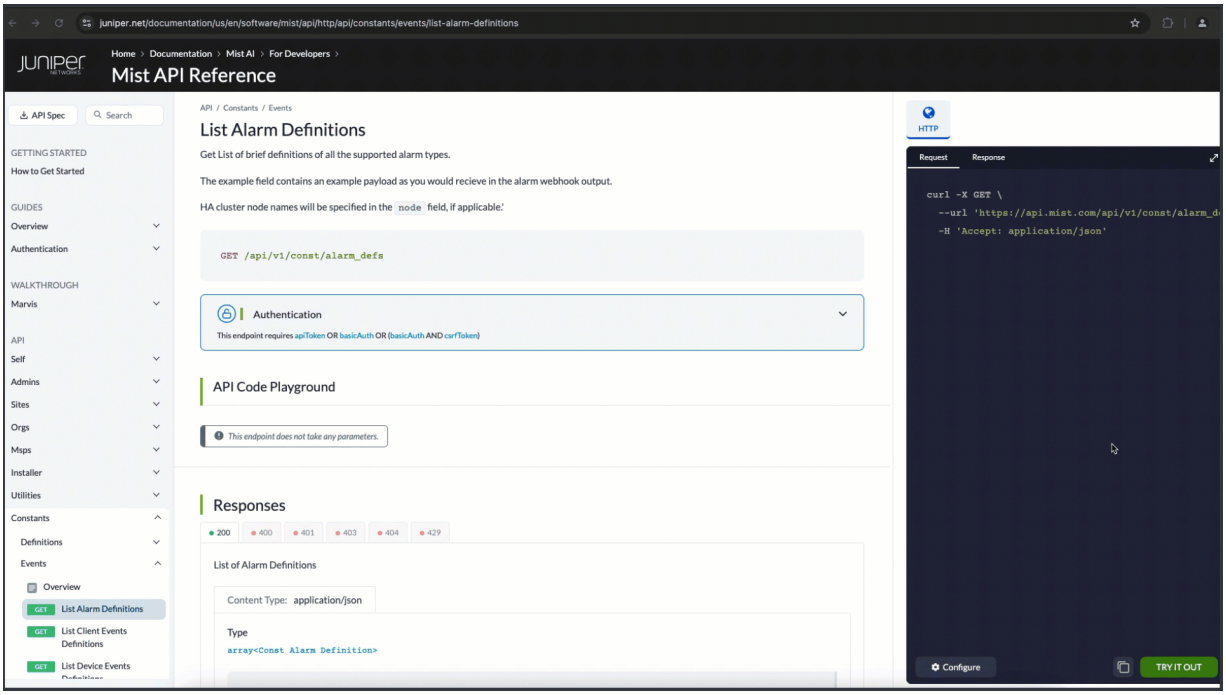
Viewing Alert Details

To see the full list of alarm types and their definitions, you can issue the following request:

```
GET /api/v1/const/alarm_defs
```

The following animation demonstrates issuing a GET call from the Mist API Reference in order to get a list of the definitions for all of the supported alarm types. The Response displays useful information such as the key, group, severity, and example, which shows examples of the messages you receive from that

particular webhook.



To try this out, see List Alarm Definitions.

The table presents detailed information for just some of the alerts.

NOTE: In the following table, you can see the subscription requirements for the given webhook. Common webhooks alerts related to audit logs, alarms, or device events, for example, will require you to have a subscription to one of the following: Wireless, Wired, or WAN Assurance.

Within each alarm is contextual data that you can extrapolate for event correlation comparing multiple devices. You can find examples of all the existing alert (alarm) definitions with the function `/api/v1/const/alarm_defs` (link requires you to be logged in to Juniper Mist).

Event Aggregation

Juniper Mist aggregates events based on topics that you've set up. However, not all events are aggregated. Events are aggregated for any topics related to location services, for example, the location, asset-raw-rssi, sdkclient-scan-data, and rssi-zone topics.

If multiple events occur for the same topic during the specified aggregation window, Juniper Mist groups them into a single message. Because of message aggregation, you will need to parse the events from each message when they are received.

Webhook Messages

SUMMARY

Get familiar with the message format and the payloads for various webhook topics.

IN THIS SECTION

Message Format

IN THIS SECTION

Each webhook topic can have a slightly different format.

Most webhooks have the following structure where the `event_details` are the payload of `events`, as follows:

```
{
    "topic": "webhook-topic",
    "events": [
            {"EVENT DETAILS": "...."},
            {"EVENT DETAILS": "....",
            ...
    ]
}
```

Payload Structure

All webhook messages are in JavaScript object notation (JSON) format and include the following information in the header, which describes the configuration of the message itself. The header

information appears before the payload in the message. If you configure a custom header in the webhook configuration, it also appears here.

```
POST /uri/... HTTP/1.1

Host: hooks.abc.com:443
User-Agent: Mist-webhook
Content-Type: application/json
Content-Length: 382
X-Mist-Signature: ce3af7760f1289d02bf6a7ad19f3xxxxxxxxxx
```

Infrastructure Payload Examples

IN THIS SECTION

For the full list of Webhooks samples, see API Sample Webhooks. This will show you sample webhook messages that the Mist Cloud sends for each of the given webhooks topics.

The following are just some of the webhook samples, starting with the infrastructure webhooks.

Alert

This alert (alarm) example displays a detected rogue AP, the count (number of times Juniper Mist detected it), and the AP and the basic service set identifier (BSSIDs) that detected it.

```
{
    "topic": "alarms",
```

```
    "events": [
        {
            "aps": [
                "5c5b35xxxxxx"
            ],
            "bssids": [
                "00024axxxxxx",
                "5c5b3xxxxxx",
                "000f2xxxxxx",
                "c03f0exxxxxx",
                "e091f5xxxxxx",
                "e894f6xxxxxx",
                "40169fxxxxxx",
                "40169fxxxxxx",
                "c03f0exxxxxx",
                "5c5b35xxxxxx"
            ],
            "count": 16,
            "id": "95193bda-1fef-4ea6-xxxx-xxxxxxxxxxxx",
            "last_seen": 1549068720,
            "ssids": [
                "qwerty",
                "A-Dot",
                "xfinity",
                "alpha"
            ],
            "timestamp": 1549068202,
            "type": "rogue-ap-detected",
            "update": true,
            "org_id": "2818e386-8dec-2562-xxxx-xxxxxxxxxxxx",
            "site_id": "4ac1dcf4-9d8b-7211-xxxx-xxxxxxxxxxxx"
        }
    ]
}
```

Audit

This example is an audits alert indicating that John Doe updated a device. It shows the organization (org_id) and site (site_id) the device belongs to.


```
{
    "topic": "audits",
    "events": [
        {
            "admin_name": "john doe john.doe@juniper.net",
            "device_id": "00000000-0000-0000-1000-5c5b35xxxxxx",
            "id": "8e00dd48-b918-4d9b-xxxx-xxxxxxxxxxxx",
            "message": "Update Device \"Reception\"",
            "org_id": "2818e386-8dec-2562-xxxx-xxxxxxxxxxx",
            "site_id": "4ac1dcf4-9d8b-7211-xxxx-xxxxxxxxxxxx",
            "src_ip": "xx.xx.xx.xx",
            "timestamp": 1549047906.201053
        }
    ]
}
```


Client Join

This client-join message displays the MAC address of the client that joined. It also displays the associated connection details the instant a client joins a wireless network.


```
{
    "topic": "client-join",
    "events": [
    {
            "ap": “5c5b35d0xxxx",
            "ap_name": “AP43 Test",
            "band": "5",
            "bssid": "5c5b35dfxxxx",
            "connect": 1592333828,
            "connect_float": 1592333828.324,
            "mac": "70ef0071xxxx",
            "org_id": "6748cfa6-4e12-11e6-xxxx-xxxxxxxxxxx",
            "rssi": -54,
            "site_id": "d761985e-49b1-4506-xxxx-xxxxxxxxxxx",
            "site_name": "Test",
```

```
            "ssid": "Mist",
            "timestamp": 1592333828,
            "version": 2
            "wlan_id": "6c0c0b07-0d77-44d1-xxxx-xxxxxxxxxxxx",
        }
    ]
}
```


Client Sessions

The client-sessions payload displays detailed information regarding the entire session from a client to a single AP.


```
{
    "topic": "client-sessions",
    "events": [
        {
            "ap": “5c5b352fxxxx",
            "ap_name": “AP43 Test",
            "band": "5",
            "bssid": "5c5b352bxxxx",
            "client_family": "iPhone",
            "client_manufacture": "Apple",
            "client_model": "8+",
            "client_os": "13.4.1",
            "connect": 1592333548,
            "connect_float": 1592333548.117,
            "disconnect": 1592333828,
            "disconnect_float": 1592333828.589,
            "duration": 279.835049793,
            "mac": "70ef00xxxxxx",
            "next_ap": "5c5b35d0xxxx",
            "org_id": "6748cfa6-4e12-11e6-xxxx-xxxxxxxxxxxx",
            "rssi": -87,
            "site_id": "d761985e-49b1-4506-xxxx-xxxxxxxxxxx",
            "site_name": "Test",
            "ssid": "Mist",
            "termination_reason": 3,
            "timestamp": 1592333828,
            "version": 2
            "wlan_id": "6c0c0b07-0d77-44d1-xxxx-xxxxxxxxxxxx",
```

```
        }
    ]
}
```

Device Events

The device-events payload displays details about the device experiencing the event with the reason.

```
{
    "topic": "device-events",
    "events": [
        {
            "audit_id": "a8ec4d8a-4da6-4ead-xxxx-xxxxxxxxxxx",
            "ap": "5c5b35xxxxxx",
            "ap_name": "AP41 Near Lab",
            "device_name": "AP41 Near Lab",
            "device_type": "ap/switch/gateway",
            "ev_type": "NOTICE",
            "mac": "5c5b35xxxxxx",
            "org_id": "2818e386-8dec-2562-xxxx-xxxxxxxxxxx",
            "reason": "power_cycle",
            "site_id": "4ac1dcf4-9d8b-7211-xxxx-xxxxxxxxxxx",
            "site_name": "Site 1",
            "text": "event details",
            "timestamp": 1461220784,
            "type": "AP_RESTARTED"
        }
    ]
}
```

Device Updowns

The device-updowns webhook is a subset of the device-events webhook. It sends only the basic information of the device and reason (type) it went down.

```
{
    "topic": "device-updowns",
    "events": [
        {
            "org_id": "2818e386-8dec-2562-xxxx-xxxxxxxxxxx",
```

```
            "site_id": "4ac1dcf4-9d8b-7211-xxxx-xxxxxxxxxxxx",
            "type": "AP_RESTARTED",
            "ap": "5c5b35xxxxxx",
            "ap_name": "AP01",
            "site_name": "Site1"
            "timestamp": 1461220784
        }
    ]
}
```


Guest Authorization

The guest-authorization webhook provides customers with data on guest clients when those clients authorize to a WLAN.


```
{
    "topic": "guest-authorizations",
    "events": [
        {
            "ap":"5c5b350e55c8",
            "auth_method": "passphrase",
            "authorized_expiring_time":1677076639,
            "authorized_time":1677076519,
            "carrier": "docomo",
            "client": "ac2316eca70a",
            "company": "MIST",
            "email": ""abcd@abcd.com",
            "field1": "field1 value",
            "field2": "field2 value",
            "field3": "field3 value",
            "field4": "field4 value",
            "mobile": "+0123456789",
            "name": "Dr Strange",
            "org_id":"1688605f-916a-47a1-8c68-f19618300a08",
            "site_id":"ec3b5624-73f1-4ed3-b3fd-5ba3ee40368a",
            "sms_gateway": "Telstra",
            "sponsor_email": "sponsor@gmail.com",
            "ssid":"Portal Auth",
            "wlan_id":"7681be9a-044a-4622-90cf-3accde5ad853",
        }
```

```
    ]
}
```

Juniper Mist Edge Events

The mxedge-events webhook payload can contain basic information about an event occurring on an individual Juniper Mist Edge device similar to device-events.

```
{
  "topic": "mxedge-events",
  "events": [
    {
      "audit_id": "03a65fa8-f74b-4c82-xxxx-xxxxxxxxxxxx",
      "mxcluster_id": "27558fe2-a0e5-4236-xxxx-xxxxxxxxxxxx",
      "mxedge_id": "00000000-0000-0000-1000-xxxxxxxxxxxx",
      "mxedge_name": "ME1",
      "org_id": "dfb3a656-2a21-4ea5-xxxx-xxxxxxxxxxxx",
      "timestamp": "1692974834.308884",
      "type": "ME_CONFIG_CHANGED_BY_USER"
    }
  ]
}
```

Location Payload Examples

IN THIS SECTION

The next group is the Location webhooks, which are available only for sites (not organizations).

Location Coordinates

The location webhook payload correlates client information to a location on a map (floorplan) uploaded to Juniper Mist. An accurately scaled map and use of the SDK client are requirements for this webhook.

```
{
    "topic": "location",
    "events": [
        {
            "site_id": "4ac1dcf4-9d8b-7211-xxxx-xxxxxxxxxxxx",
            "map_id": "845a23bf-bed9-e43c-xxxx-xxxxxxxxxxxx",
            "x": 13.5,
            "y": 3.2,
            "timestamp": 1461220784,

            // for SDK client
            "type": "sdk",
            "id": "de87bf9d-183f-e383-xxxx-xxxxxxxxxxxx",
            "name": "optional",

            // for WIFI
            "type": "wifi",
            "mac": "5684daxxxxxx",
            // Optional for wifi
            "wifi_beacon_extended_info": [
                {"frame_ctrl": 776, "seq_ctrl": 772, "payload": "............"},
            ]

            // for ASSET
            "type": "asset",
            "mac": "7fc293xxxxxx",
            "ibeacon_uuid": "f3f17139-704a-f03a-xxxx-xxxxxxxxxxxx",
            "ibeacon_major": 13,
            "ibeacon_minor": 138,
            "eddystone_uid_namespace": "2818e3868decxxxxxxxx",
            "eddystone_uid_instance": "5c5b35xxxxxx",
            "eddystone_url_url": "https://www.abc.com",
            "mfg_company_id": 935,
            "mfg_data": "648520a1020000",

            "battery_voltage": 3370
        }
```

```
    ]
}
```

Occupancy Alerts

The occupancy-alerts webhook displays information about specific zones if they exceed the configured occupancy_limit.

```
{
    "topic": "occupancy-alerts",
    "events": [
        {
            "alert_events": [
                {
                    "current_occupancy": 10,
                    "map_id": "f5d26c7f-1670-4921-xxxx-xxxxxxxxxxxx",
                    "occupancy_limit": 5,
                    "org_id": "6748cfa6-4e12-11e6-xxxx-xxxxxxxxxxxx",
                    "timestamp": 1594861457,
                    "type": "COMPLIANCE-VIOLATION",
                    "zone_id": "b83312a7-7269-4ae1-xxxx-xxxxxxxxxxxx",
                    "zone_name": "PLM and Leadership"
                },
                {
                    "current_occupancy": 20,
                    "map_id": "f5d26c7f-1670-4921-xxxx-xxxxxxxxxxxx",
                    "occupancy_limit": 10,
                    "org_id": "6748cfa6-4e12-11e6-xxxx-xxxxxxxxxxxx",
                    "timestamp": 1594861457,
                    "type": "COMPLIANCE-VIOLATION",
                    "zone_id": "80acf542-e863-43cf-xxxx-xxxxxxxxxxxx",
                    "zone_name": "CSQA"
                },
                {
                    "current_occupancy": 9,
                    "map_id": "f5d26c7f-1670-4921-xxxx-xxxxxxxxxxxx",
                    "occupancy_limit": 4,
                    "org_id": "6748cfa6-4e12-11e6-xxxx-xxxxxxxxxxxx",
                    "timestamp": 1594861457,
                    "type": "COMPLIANCE-VIOLATION",
                    "zone_id": "a4c7a7c2-880e-4a0e-xxxx-xxxxxxxxxxxx",
```

```
                    "zone_name": "Marketing & Sales Ops"
                }
            ],
            "site_id": "67970e46-4e12-11e6-xxxx-xxxxxxxxxxxx",
            "site_name": "MIST OFFICE"
        }
    ]
}
```


RSSI Zone

The rssizone webhook payload displays devices that have exceeded a configured minimum RSSI threshold across a site.


```
{
   "topic":"rssizone",
   "events":[
      {
         "mac":"500291xxxxxx",
         "map_id":"f5d26c7f-1670-4921-xxxx-xxxxxxxxxxxx",
         "rssizone_id":"e38f8e76-40db-4144-xxxx-xxxxxxxxxxxx",
         "site_id":"f5fcbee5-fbca-45b3-xxxx-xxxxxxxxxxxx",
         "timestamp":1694158990.986472,
         "trigger":"enter",
         "type":"wifi"
      }
   ]
}
```


SDK Client Scan Data

The SDK Client Scan Data webhook payload displays specific data about a client that isn't available without installing an application (using the SDK) on the client itself.


```
{
  "events": [
    {
      "connection_ap": "5c5b35xxxxxx",
      "connection_band": "2.4",
      "connection_bssid": "5c5b35xxxxxx",
```

```
      "connection_channel": 11,
      "connection_rssi": -87,
      "last_seen": 1592333828,
      "mac": "70ef00xxxxxx",
      "scan_data": [
        {
          "ap": "5c5b35xxxxxx",
          "band": "2.4",
          "bssid": "5c5b35xxxxxx",
          "channel": 11,
          "rssi": -87,
          "ssid": "mist-wifi",
          "timestamp": 1592333828
        },
        {
          "ap": "5c5b35xxxxxx",
          "band": "5",
          "bssid": "5c5b35xxxxxx",
          "channel": 36,
          "rssi": -75,
          "ssid": "mist-wifi",
          "timestamp": 1592333828
        }
      ],
      "site_id": "d761985e-49b1-4506-xxxx-xxxxxxxxxxxx"
    }
  ],
  "topic": "sdkclient-scan-data"
}
```


Virtual Beacon Entry and Exit Event

The vbeacon webhook is triggered when a mobile device running the Juniper Mist SDK is entering or exiting the area defined by a virtual beacon.


```
{
  "topic":"vbeacon",
  "events":[
   {
     "mac":"10521cxxxxxx",
     "map_id":"5a8b84e6-cc7b-xxxx-xxxxxxxxxxxx",
```

```
      "site_id":"f5fcbee5-fbca-45b3-xxxx-xxxxxxxxxxxx",
      "timestamp":1694166602.662786,
      "trigger":"enter",
      "type":"wifi",
      "vbeacon_id":"ca301fd7-07af-4d42-xxxx-xxxxxxxxxxxx"
    }
   ]
}
```


Zone Entry and Exit Events

The zone webhook is triggered when a device enters or exits a defined zone.


```
{
  "topic":"zone",
  "events":[
   {
     "mac":"10521cxxxxxx",
     "map_id":"5a8b84e6-cc7b-xxxx-xxxxxxxxxxxx",
     "site_id":"f5fcbee5-fbca-45b3-xxxx-xxxxxxxxxxxx",
     "timestamp":1694166602.662786,
     "trigger":"exit",
     "type":"wifi",
     "zone_id":"b83312a7-7269-4ae1-xxxx-xxxxxxxxxxxx "
   }
  ]
}
```


Configure Webhooks from the Juniper Mist Portal

IN THIS SECTION

You can configure webhooks in the Juniper Mist portal, rather than using the API.

Keep in mind, everything that you can configure in the portal is a result of an API. And all changes that you make in the portal actually are calls to the RESTful API. In fact, clicking save is simply a call to the appropriate API to make an update.

Although there are more APIs available than are implemented in the portal, you might find it convenient to use the portal for certain configuration tasks.

When using the portal to configure webhooks, be aware of the configuration hierarchy. Go to the appropriate page for the webhooks that you want to configure.

- Organization-level webhooks—From the left menu of the Juniper Mist portal, select **Organization** > **Settings**.
- Site-level webhooks—From the left menu of the Juniper Mist portal, select **Organization** > **Site Configuration**.

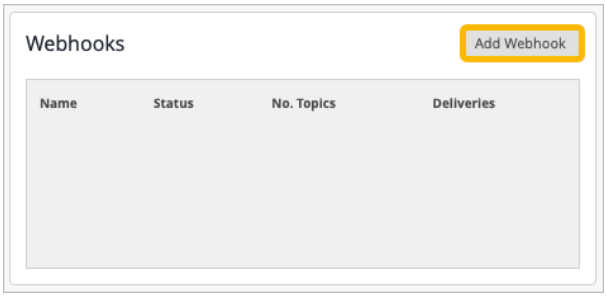
Add a Webhook in the Juniper Mist Portal

SUMMARY

Follow these instructions to set up a new webhook by using the Juniper Mist™ portal.

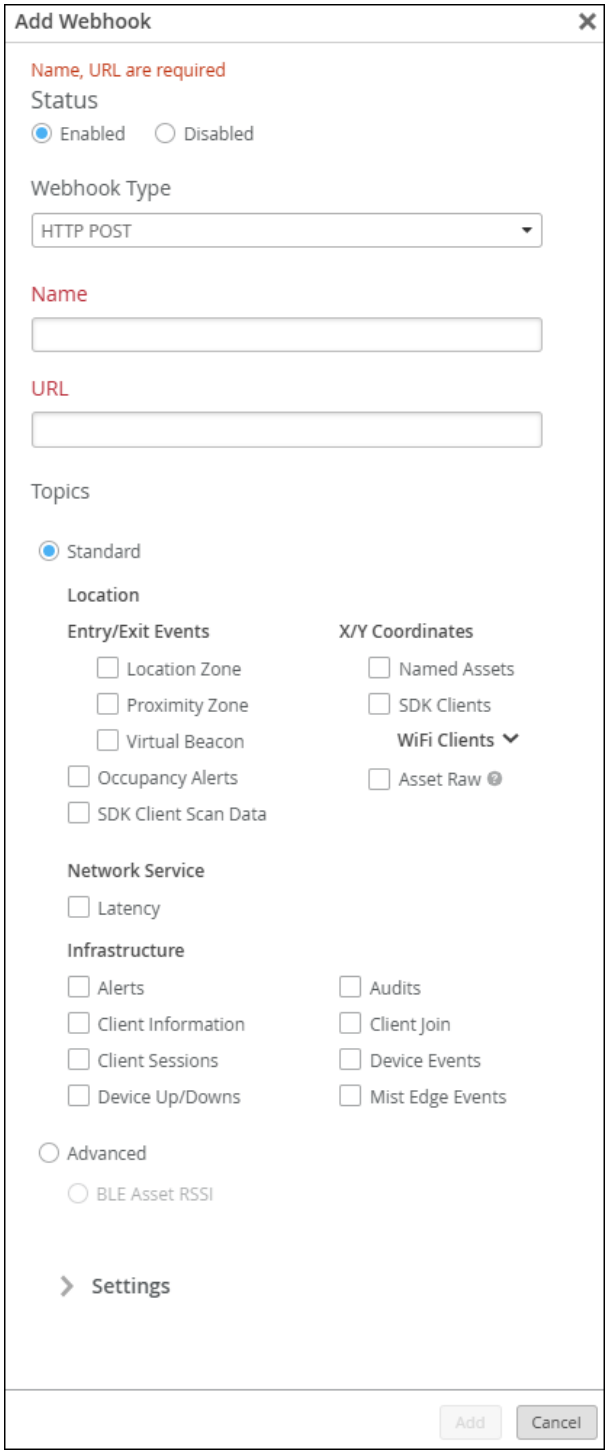
1. In the Juniper Mist™ portal, navigate to the organization or site settings:
   - For organization-level webhooks, select **Organization** > **Settings**.
   - For site-level webhooks, select **Organization** > **Site Configuration**, and then select the site.

     NOTE: The following example demonstrates configuring site-level webhooks. When configuring organization-level webhooks, the options that display will be different, such as the list of available webhook topics to choose from.
2. Scroll down to the Webhooks section, and click **Add Webhook**.



3. Enter the information:

   - Name—Enter a name to identify this Webhook.
   - Webhook Type—Select HTTP Post, OAuth2, or Splunk. For OAuth2, the Mist cloud will act as the client for the selected type and authenticate against an authorization server to get a token, which it then appends to the Webhook Authorization Headers.
   - URL—Enter the URL of destination you want Mist to send the Webhooks to.
   - Grant Types—(OAuth2 only) There are two ways of requesting access tokens from the customer system:
     - Password-based, which requires the user name and password of the resource owner.
     - Client Credentials-based, which requires the Client ID and Client Secret provided by the OAuth IDP.
   - Topics—Select the topics you want to receive Webhooks for. You must select at least one.



4. (Optional) Click **Advanced Settings** for additional options.
   - Verify Certificate—This option is enabled by default. If you do not want Mist to verify that the certificate of the webhook receiver is valid, click **No**. Although this method is not secure, it does provide some flexibility if your webhook receiver does not have a valid signed certificate. We do not recommend this method, for security reasons.

- Secret (HTTP-POST only)—Using a secret allows you to specify a secret (like a password) used to calculate a pair of HTTP headers. A secret enables you to verify that the message is coming from Juniper Mist and has not been modified.
- Custom Headers—The Custom Headers configuration enables you to specify any custom headers needed for your webhook receiver. Some receivers (or their proxies) require a token-based authentication method, user-based authentication, or a method to present custom headers to indicate the type of data being sent.

NOTE: Keep these cautions in mind when considering whether to use a secret or custom headers:

- When a secret is provided, two HTTP headers will be added:
  - **X-Mist-Signature**: HMAC_SHA1 (secret, body)
  - **X-Mist-Signature-v2**: HMAC_SHA256 (secret, body)
- If the headers format is invalid, “X-Mist-Error”: “headers format invalid” will be sent.
- If the total bytes of the headers exceed 1000, “X-Mist-Error”: “headers too big” will be sent.
- If any header value is not a string, “X-Mist-Error”: “header[%s] not a string” will be sent.

5. Click **Add**.

Update a Webhook in the Juniper Mist Portal

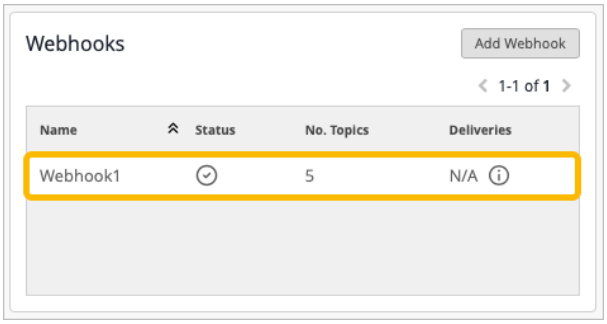
SUMMARY

Follow these instructions to modify an existing webhook by using the Juniper Mist™ portal.

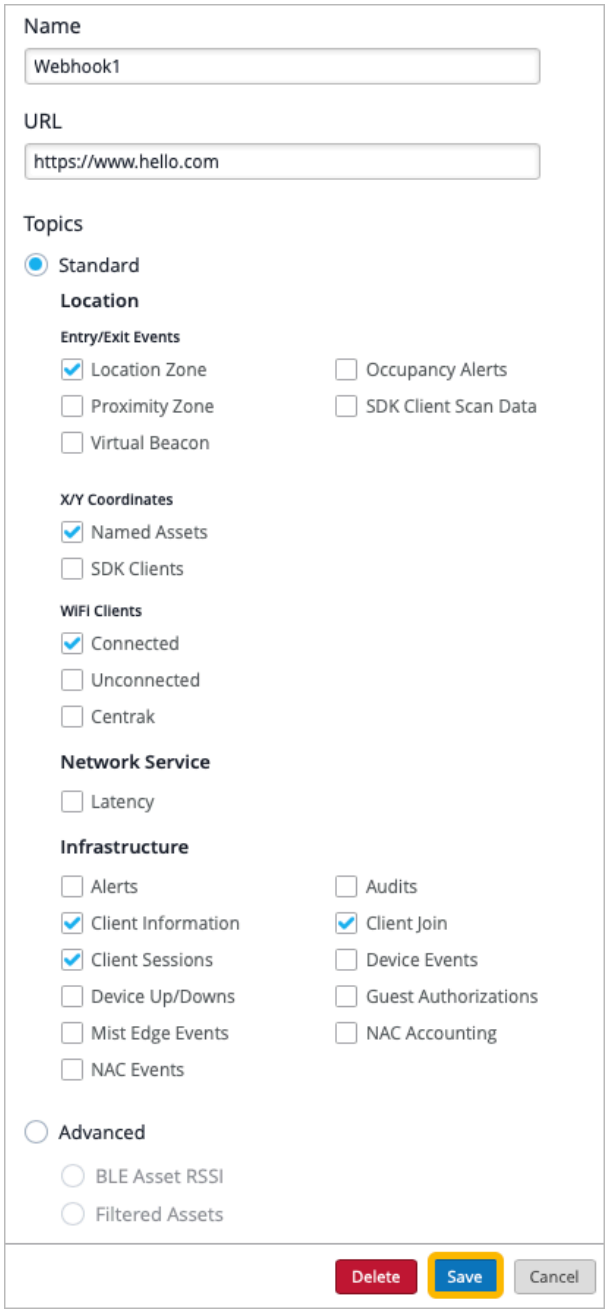
Updating a webhook in the portal is nearly the same as adding a webhook.

1. From the left menu of the Juniper Mist portal, navigate to the organization or site settings:
   - For organization-level webhooks, select **Organization** > **Settings**.
   - For site-level webhooks, select **Organization** > **Site Configuration**, and then select the site.

2. Scroll down to the Webhooks section, and click the webhook that you want to update.



3. Make the configuration changes, and then click **Save**.



For more information about the options, see the information in "Add a Webhook in the Juniper Mist Portal" on page 214.

Delete a Webhook in the Juniper Mist Portal

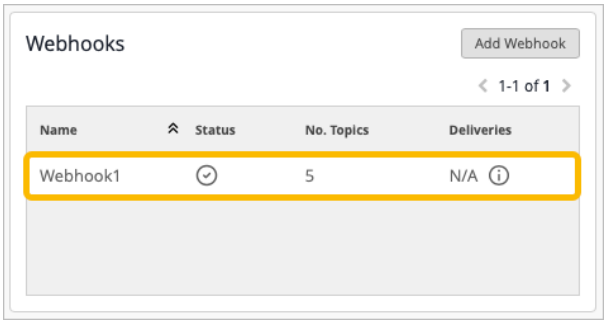
SUMMARY

Follow these instructions to remove a webhook by using the Juniper Mist™ portal.

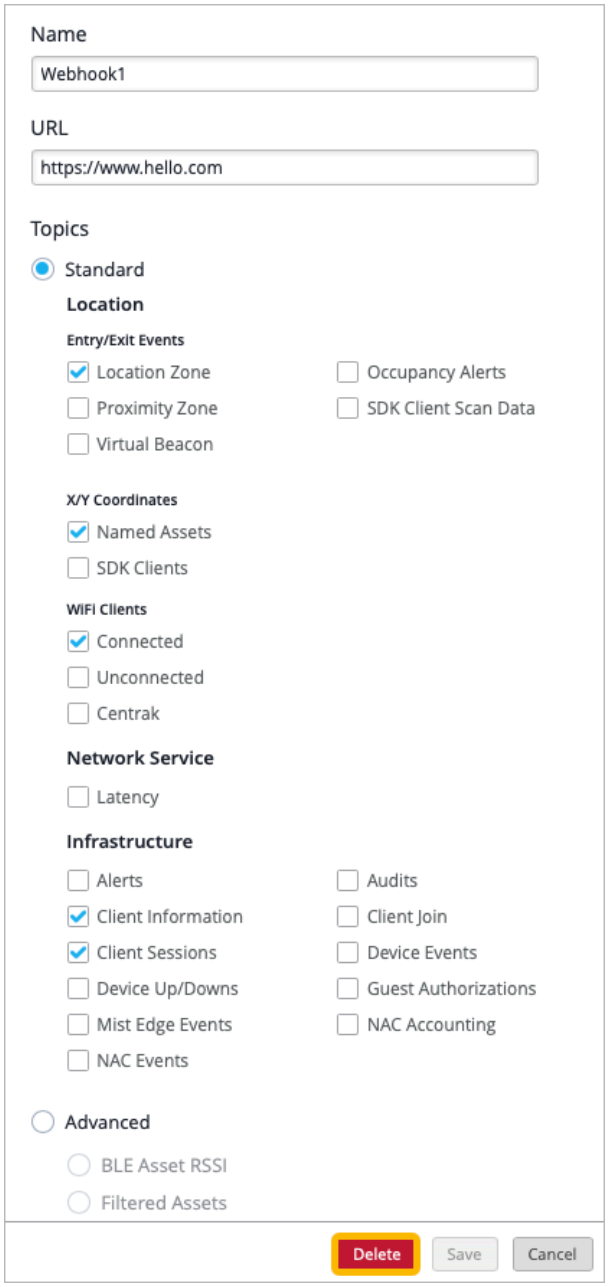
If you no longer need a webhook, delete it.

To delete a webhook:

1. From the left menu of the Juniper Mist portal, navigate to the organization or site settings:
   - For organization-level webhooks, select **Organization** > **Settings**.
   - For site-level webhooks, select **Organization** > **Site Configuration**, and then select the site.
2. Scroll down to the Webhooks section, and click the webhook that you want to delete.



3. Click **Delete**.


Name
Webhook1
URL
https://www.hello.com
Topics
Standard
Location
Entry/Exit Events
Location Zone
Occupancy Alerts
Proximity Zone
SDK Client Scan Data
Virtual Beacon
X/Y Coordinates
Named Assets
SDK Clients
WiFi Clients
Connected
Unconnected
Centrak
Network Service
Latency
Infrastructure
Alerts
Audits
Client Information
Client Join
Client Sessions
Device Events
Device Up/Downs
Guest Authorizations
Mist Edge Events
NAC Accounting
NAC Events
Advanced
BLE Asset RSSI
Filtered Assets
Delete
Save
Cancel

Configure Webhooks from the API

SUMMARY

Start getting familiar with configuring webhooks with the API.

IN THIS SECTION

You can configure webhooks from the API, rather than using the Juniper Mist portal.

NOTE: For more information about the API, go to these resources:

- "RESTful API Overview" on page 4 (in this guide)
- "Additional RESTful API Documentation" on page 154

When configuring webhooks, always be aware of the configuration hierarchy.

- In an organization-level webhook, specify the organization ID: `/api/v1/orgs/{org-id}/webhooks`

  NOTE: To find the {org-id}, select **Organization** > **Settings** from the left menu of the Juniper Mist portal. The organization ID appears near the top of the Organization Settings page.

- In a site-level webhook, specify the site ID: `/api/v1/sites/{site-id}/webhooks`

  NOTE: To find the {site-id}, select **Organization** > **Site Configuration** from the left menu of the Juniper Mist portal. Then select the site. The site ID appears near the top of the site Configuration page.

Example: Site Configuration

The following function displays all the webhooks configured for a specific site:

```
/api/v1/sites/4ac1dcf4-9d8b-7211-xxxx-xxxxxxxxxxxx/webhooks
```

The following output is the result of the previous API call:

```
HTTP 200 OK
Allow: POST, OPTIONS, GET
Content-Type: application/vnd.api+json
Vary: Accept

[
    {
        "name": "Lobby-Zone-Events",
        "url": "https://webhook.site/02747ddc-2b1f-4134-a1eb-xxxxxxxxxxxx",
        "secret": "",
        "enabled": true,
        "topics": "zone",
        "verify_cert": true,
        "id": "20538707-b873-4a60-xxxx-xxxxxxxxxxxx",
        "for_site": true,
        "site_id": "4ac1dcf4-9d8b-7211-xxxx-xxxxxxxxxxxx",
        "org_id": "3f12cb79-fb5e-4d4b-xxxx-xxxxxxxxxxxx",
        "created_time": 1686252096,
        "modified_time": 1686252096
    }
]
```

Create Webhooks from the API

SUMMARY

Follow these instructions to use a POST command to set up a new webhook.

Use a `POST` to configure a webhook and to enable topics.

In your API call, be aware of the configuration hierarchy and the different topics that are available for organizations and sites.

- In the API function, be sure to specify the organization ID or site ID. See "Configure Webhooks from the API" on page 222.
- In the `topics` line, be sure to specify only the topics that are appropriate for the organization or site level. See "Webhook Topics" on page 183.

Here is an example of a `POST` to configure a site webhook to enable site-level topics:

See Create Site Webhook for the example `POST` structure. Also see Create Org Webhook if applicable. A sample response is provided below.

```
{
    "name": "analytic",
    "type": "http-post",
    "url": "https://username:password@hooks.abc.com/uri/...",
    "secret": "secret",
    "headers":{
        "x-custom-1": "your_custom_header_value1",
        "x-custom-2": "your_custom_header_value2"
    },
    "verify_cert": false,
    "enabled": true,
    "topics": [ "location", "zone", "vbeacon", "rssizone", "asset-raw-rssi", "device-events",
"alarms", "audits", "client-join", "client-sessions", "device-updowns", "occupancy-alerts",
"mxedge-events", "nac-accounting", "sdkclient-scan-data",]
}
```

NOTE: When reusing code blocks, replace placeholder values with actual values, such as your API token, organization ID, site ID, AP name, and so on.

Table 17: Webhook Parameter Descriptions

Parameter	Description
name	The name of the configured webhook.
type	The type of webhook (http-post, splunk, etc).
url	The destination to receive the webhook.

Table 17: Webhook Parameter Descriptions *(Continued)*

Parameter	Description
secret	When using the `http-post` webhooks type, the secret is used by the Mist Cloud to sign the webhook message. Also see The secret parameter.
headers	Custom headers can be added under the `headers`property. These custom headers will be added in the HTTP headers sent by the Mist Cloud for authentication. See apiToken (Custom Header Signature) and csrfToken (Custom Header Signature) if applicable.
verify_cert	Whether or not certificate verification is turned on. True or False.
enabled	Whether or not the webhook is enabled. True or False.
topics	The selected items you want receive messages or alerts for. To learn more, see "Webhook Topics" on page 183. Also see the Mist API Reference's Webhook Topics.

Also see Securing Webhooks with splunk type.

Update a Webhook from the API

To update a webhook from the API, you send a `PUT` command to the webhook's API endpoint with the parameter that you want to update.

See Update Org Webhook or Update Site Webhook for the sample `POST` structure. A sample response is provided below.

The following is an example configuration of an existing, organization-level **mxedge-events** webhook viewed from the API.


```
{
    "enabled": true,
    "name": "mist-edge",
    "url": "https://webhook.site/4ec10796-16ec-4225-aba4-xxxxxxxxxxxx",
    "secret": "",
    "topics": [
        "mxedge-events"
    ],
    "verify_cert": true,
    "id": "a7c61a9c-a25b-4c27-xxxx-xxxxxxxxxxxx",
    "for_site": false,
    "site_id": "00000000-0000-0000-0000-000000000000",
    "org_id": "3f12cb79-fb5e-4d4b-xxxx-xxxxxxxxxxxx",
    "created_time": 1575305516,
    "modified_time": 1692974137
}
```


If you want to make a change to this webhook, you need to make a (PUT) API call from your API client to the webhook API URL **/api/v1/orgs/{org_id}/webhooks/{webhook_id}** with the updated configuration in the request body.

In the following example, the URL to display (GET), update (PUT), or delete (DELETE) the configuration of the webhook having the id **4ec10796-16ec-4225-aba4-xxxxxxxxxxx** is:

```
https://{api-host}/api/v1/orgs/3f12cb79-fb5e-4d4b-xxxx-xxxxxxxxxxxxx/webhooks/4ec10796-16ec-4225-aba4-xxxxxxxxxxx
```

NOTE: In place of {api-host}, use the API Host (or Endpoint) for your global region. See API Endpoints and Global Regions.

So, let's say you want to change the URL for the webhook receiver to **https://enicxxxx72vx.x.pipedream.net**. To do that, you need to update (PUT) the webhook API endpoint described above with the following request body:

```
{
  "url":"https://enicxxxx72vx.x.pipedream.net"
}
```

Delete Webhooks from the API

SUMMARY

Follow these instructions to use a DELETE command to remove a webhook.

To delete a webhook using the API, you first need to find the webhook ID.

You can find the webhook ID in the webhook response. In the following example, the `id` is **a7c61a9c-a25b-4c27-xxxx-xxxxxxxxxxxx**

```
[
    {
        "enabled": true,
        "name": "mist-edge",
        "url": "https://webhook.site/4ec10796-16ec-4225-xxxx-xxxxxxxxxxxx",
        "secret": "",
        "topics": [
            "mxedge-events"
        ],
        "verify_cert": true,
        "id": "a7c61a9c-a25b-4c27-xxxx-xxxxxxxxxxxx",
        "for_site": false,
        "site_id": "00000000-0000-0000-0000-000000000000",
        "org_id": "3f12cb79-fb5e-4d4b-xxxx-xxxxxxxxxxxx",
        "created_time": 1575305516,
        "modified_time": 1692974137
```

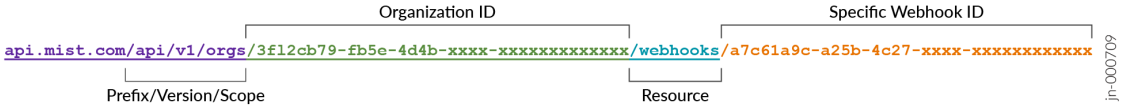
```
    }
]
```

To delete this webhook, make an API call (DELETE) from your API client.

```
DELETE https://api.mist.com/api/v1/orgs/3f12cb79-fb5e-4d4b-xxxx-xxxxxxxxxxxx/webhooks/a7c61a9c-a25b-4c27-a14e-xxxxxxxxxxxx
```

As detailed below, this call specifies the API endpoint, the organization ID, the resource (webhooks), and webhook ID (**a7c61a9c-a25b-4c27-xxxx-xxxxxxxxxxxx**).

Also see Delete Org Webhook and Delete Site Webhook.



After you delete the webhook, the Juniper Mist API will return an HTTP 200 OK response.

Testing Webhooks

SUMMARY

Use public webhook receivers to test your webhooks and ensure that you're getting the data that you need.

IN THIS SECTION

By testing webhooks, you can verify the configuration process as well as inspect and parse the event messages you receive from Mist. If you do not have a webhook receiver available to you, you can use a public, free service. Many such services enable you to receive and inspect the incoming webhook POST messages from Juniper Mist.

These public sites create a random URL that you can use on a temporary basis. The data that you receive is ephemeral and is deleted as soon as you close your browser. Using one of these sites from your browser, you can identify parameter data from the message payloads to help create your workflow.

After you configure your webhook, you can trigger it and view those messages as they are received on the public webhook receiver.

Two of these free testing webhook receivers are:

- https://webhook.site/
- https://public.requestbin.com/

These sites are not Juniper maintained and are to be used at your own risk.

NOTE: Be sure to delete any test webhooks from Mist when they are no longer in use. Otherwise, Mist will continue to push webhooks to the target webhook URL, and the webhook testing receivers can blacklist Mist IP addresses as a result.

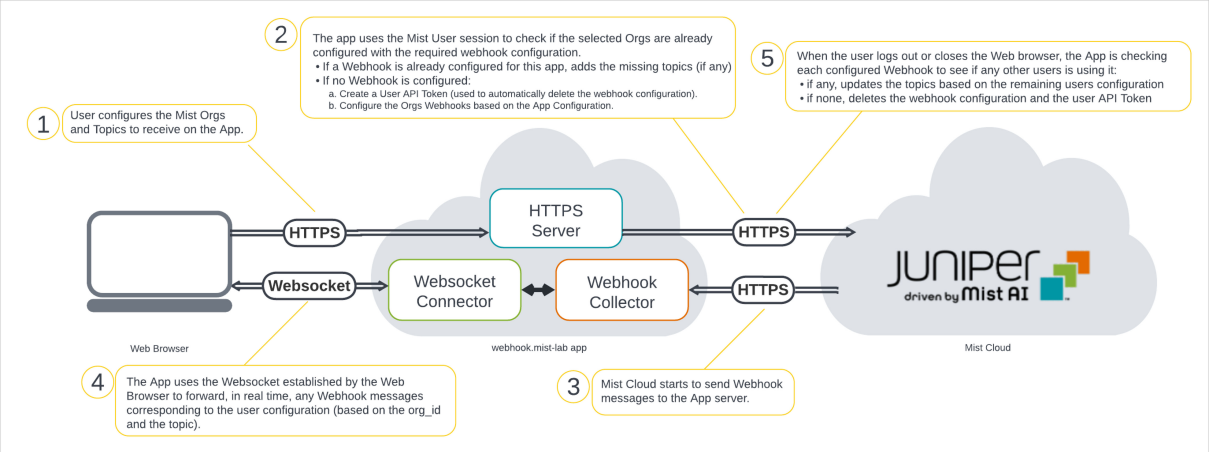
Webhook Tester

There is a webhook tester tool that you can use to easily test and monitor the webhooks for your Mist organization. The webhook tester is available here Mist Webhook Tester. This is an application that automatically creates and deletes the Webhook configuration in the Mist org to receive them, and displays the received webhook messages in a table.

NOTE:

- Only local Mist accounts can use this application (SSO users are not supported).
- Only administrators with a role of Super User are supported (Org level webhooks can be configured by Super Users).

Below is a diagram explaining how the application works.



To use the webhook tester, simply:

1. Navigate to the Mist Webhook Tester.
2. On the configuration pop-up window, choose the:
   a. Max Events in Memory—This is the maximum number of events to be stored in the application's memory.
   b. Topics—These are the webhook topics to receive. Only a limited subset of Webhook Topics are supported, and all the other messages will be discarded.
   c. Orgs—One or multiple Mist Orgs. The application will automatically create a new Webhook configuration on the selected orgs to send the enabled topics to the application.
   d. Click **Save**.
3. As events occur, those webhooks are sent to the table for you to monitor and review. Information such as the date, topics, event type, org, site, device name, device MAC, and event details are displayed in the table for you.

> **NOTE**: The Application may occasionally not be able to delete the webhook configuration at the end of the session. Therefore, it is recommended that you confirm the webhook configuration has been deleted.

For more details, see https://github.com/Mist-Automation-Programmability/mist_webhook_monitor?tab=readme-ov-file.

Test End-to-End Connectivity

With webhooks, you can test end-to-end connectivity from your Mist Cloud instance to your defined webhook receiver to confirm that you are receiving the webhook and are able to process it. This is a webhook for testing that webhooks are working. It serves the same function as a ping would for testing network connectivity, but you're testing your webhook functionality. This validates that no devices (a firewall or router) are blocking communications between the two endpoints.

This can be done from the API by issuing a POST (from your REST client) to, in this case, your org, the configured webhook (`id`) as seen in this example:

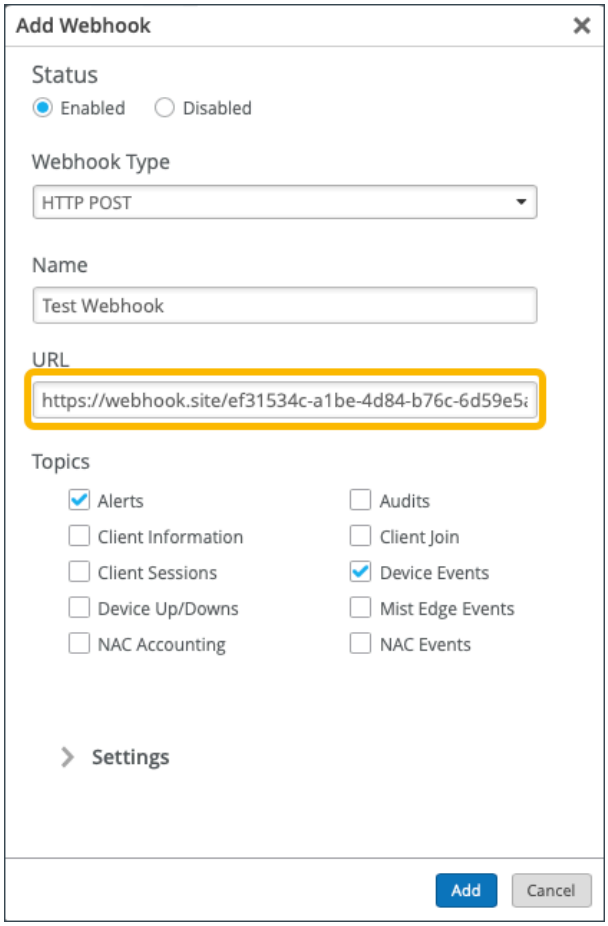
```
POST
        /api/v1/orgs/203d3d02-dbc0-4c1b-xxxx-xxxxxxxxxxxx/webhooks/032b9cb1-80af-4edc-xxxx-
xxxxxxxxxxxx/ping
```

A successful result looks like this:

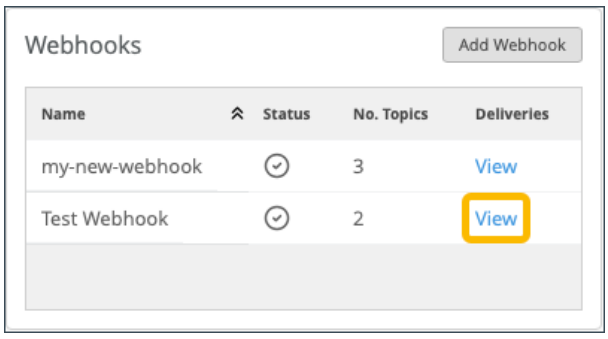
```
{
    "topic": "ping",
    "events": [
        {
            "id": "032b9cb1-80af-4edc-xxxx-xxxxxxxxxxxx",
            "name": "my webhook",
            "org_id": "203d3d02-dbc0-4c1b-xxxx-xxxxxxxxxxxx",
            "timestamp": 1725375149.0829651
        }
    ]
}
```

This can also be done from the Juniper Mist Portal:

1. Navigate to **Organization** > **Admin** > **Settings**.
2. Find the **Webhooks** section and click the **Add Webhook** button.
3. Select the appropriate **Webhook Type**, give your test webhook a name, then paste the test URL you obtained from the free webhook test receiver website (see links provided above) into the **URL** field.
4. Choose the webhook topics that you want to receive messages for and set any other necessary settings.
5. Finally, select **Add**.



6. Find your test webhook listed in the **Webhooks** section of the Organization Settings page and click **View** to see the delivery events. You can check these events to see the delivery status of webhooks to confirm that you are receiving the webhook and are able to process it.



Additional Testing Information

To review additional webhook testing information, such as how test that your webserver is able to receive and process Mist Webhooks, see the information in Validate Deliveries.

To see how to send a ping event to a site webhook, see Ping Site Webhook.

To see how to send a ping event to an organization webhook, see Ping Org Webhook.

View the Webhook Delivery Status

SUMMARY

Check the delivery status for your webhook events.

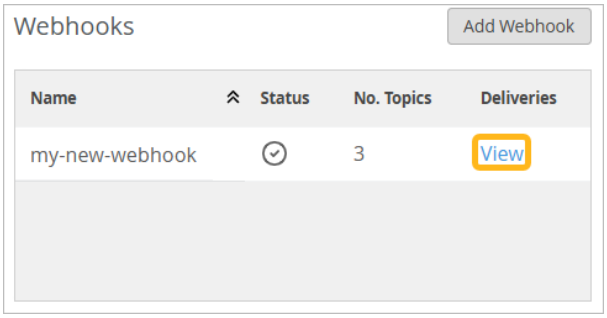
You can view the delivery status of webhook events reported during the last 61 days.

You can view status for the following webhook topics:

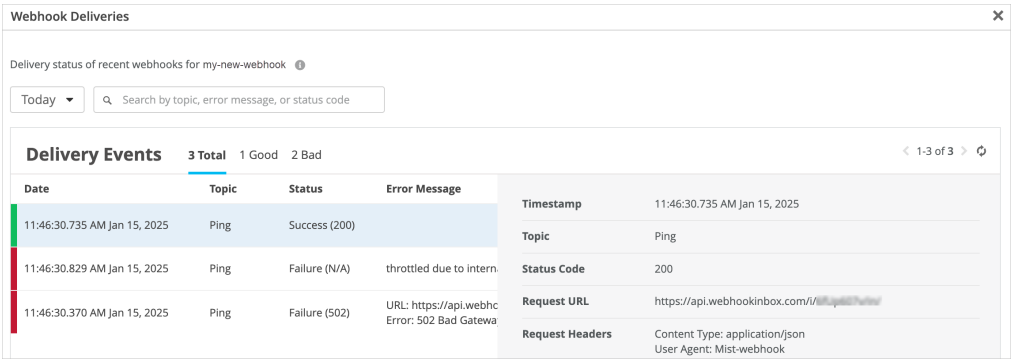
- Alerts
- Audits
- Device Up/Down
- Ping

To view the webhook delivery status:

1. Go to the appropriate page:
   - Organization-level webhooks—Go to **Organization** > **Admin** > **Settings**.
   - Site-level webhooks—Go to **Organization** > **Site Configuration**.
2. Under **Webhook**, click **View**.



3. View the webhook event information in the Webhook Deliveries window.



Tips:

- To find a particular event—Adjust the timeframe and enter search terms.
- To filter the list based on status types—Click **Good** or **Bad**, or click **Total** to see all events.

4. Click any item to view more details on the right side of the window.

   Information includes:

   - Topic
   - Request URL
   - Request Headers
   - Request Body
   - Response Status Code (if any)
   - Response Headers (if any)
   - Response Body (if any)
   - Error Message (if any)

5. Click X to close the window.

Get Started with Webhooks

IN THIS SECTION

Use the Mist API Reference to Get Started with Webhooks

SUMMARY

You can use the Mist API Reference to get started with various webhooks tasks.

The Mist API Reference is a useful tool for testing out any webhooks related tasks, as well as reviewing webhooks documentation. The API Reference connects to the Mist cloud, which allows you to submit a sample API request for your intended task. When you issue the request, you are returned a sample response.

See Create Webhook Configuration for general information you can use to get started with webhooks.

Learn how to perform various webhooks tasks by following the links in the Configure the Org Webhook from the API section:

- Create Org Webhook (API Call)
- Update Org Webhook (API Call)
- Delete Org Webhook (API Call)
- List Org Web hooks (API Call)
- Get Org Webhook (API Call)

If you need to learn how to perform various tasks regarding site webhooks, follow these links:

- Create Site Webhook (API Call)
- Update Site Webhook (API Call)

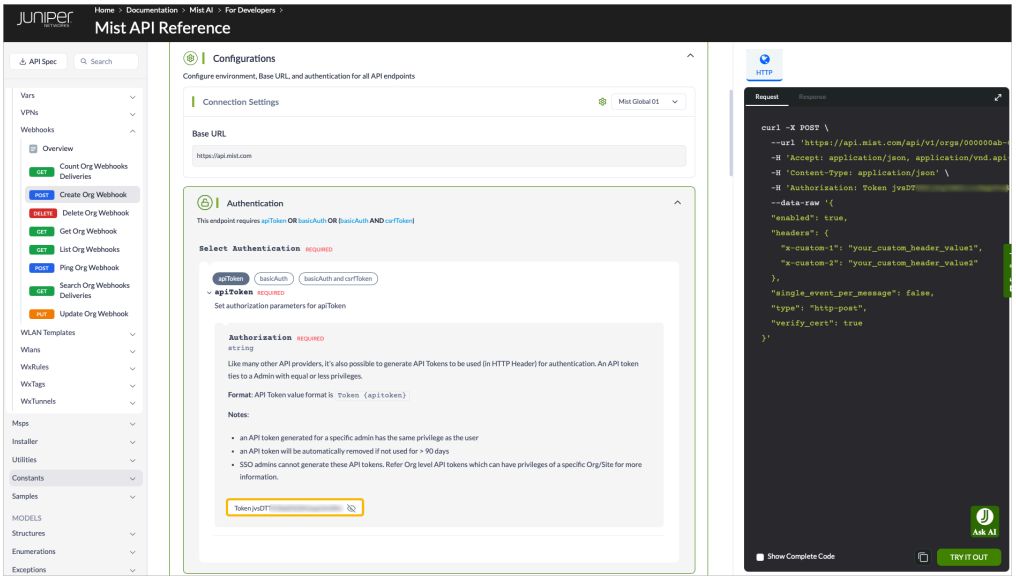
- Delete Site Webhook (API Call)
- List Site Web hooks (API Call)
- Get Site Webhook (API Call)

After navigating to the appropriate page, follow the prompts in the center of the screen. Fill in any required fields, as well as any other necessary fields for what you are trying to accomplish.

For any endpoints that require authentication, you will see an **Authentication** section in the center of the screen. Expand the **Authentication** section and enter your credentials.

NOTE: When entering your API token, you must add the "Token" key word followed by a space in front of your token in order to perform authentication. You can also perform authentication using the **basicAuth** or **basicAuth and csrfToken** options. If you do not have an API token yet, you can use the **basicAuth** option and enter the same username and password that you use for the Mist UI.

ATTENTION: To enhance security and align with industry best practices, Mist will deprecate Basic Authentication for all use cases—including admin logins and scripts—effective September 2026. Before September 2026, all integrations must transition to token-based authentication to ensure uninterrupted access and support. See "Create API Tokens" on page 15.



You will see request body fills out on the right as you fill in the fields in the center of the page.

Select the **TRY IT OUT** button to be returned a response.

Webhooks Use Cases

IN THIS SECTION

Configure Zone Entry and Exit Events (Use Case)

SUMMARY

Use this example as a model to create webhooks that alert you about entry and exit events for the zones on your floorplans.

Webhooks are one-way messages from a source server or application to a destination server or application. The webhooks methodology is “fire and forget,” meaning that you use webhooks primarily to schedule event-driven messages for alerting and monitoring. In a Juniper Mist network, you can use webhooks to send messages based on Organization or Site topics. This means that you use webhooks primarily to schedule event-driven messages for alerting and monitoring.

For this use case, from the Juniper Mist API, you set up a webhook notification whenever a device enters or exits a zone. This notification is useful for a site with building automation. A user who enters or leaves a defined zone in Juniper Mist (such as a room) and has a tracked device in hand can trigger an event that is sent from Juniper Mist to the building automation system to turn the lights on or off. To learn more about location zones, see *Add Location Zones to a Floorplan*.

To set up webhooks in the Juniper Mist portal, you need the following configuration items (and may want to use the listed optional item as well):

- Name for the webhook instance (required)
- URL (destination to receive the Juniper Mist webhooks (required)
- Streaming API in the Site Webhooks section (select the “Zone Entry/Exit Events” topic to stream to the destination) (required)

- Secret, like a password, to authenticate communication from Juniper Mist to the webhook destination (optional)

After you have this information, you can configure the webhook in the Site Settings.

To configure zone entry and exit events:

1. From the left menu of the Juniper Mist portal, select **Organization** > **Admin** > **Site Configuration**.
2. Click **Add Webhook**, enter the information, and click **Add**.

   For more information about the fields to complete, see "Add a Webhook in the Juniper Mist Portal" on page 214.

   Anytime a device enters or exits a zone, Juniper Mist will create a webhook POST to the specified URL. The incoming request payload, as formatted in JavaScript Object Notation (JSON), looks like this:

```
{
    “topic”: “zone”,
    “events”: [
        {
           “mac”: “807d3axxxxx”,
           “map_id”: “2d0d2bd7-78b8-4f4b-9454-xxxxxxxxxxxxx”,
           “site_id”: “010412fe-xxxx-xxxx-xxxx-99ff83111031d”
           “timestamp”: 1633109338.539088,
           “trigger”: “exit”,
           “type”: “wifi”,
           “zone_id”: “33294994-6a5f-4804-xxxx-xxxxxxxxxxxx”
        }
    ]
}
```

   Using this information, you can activate other automations in a third-party system, such as turning off the lights in zone **33294994-6a5f-4804-xxxx-xxxxxxxxxxxx** based on the trigger of `exit`.

RELATED DOCUMENTATION

Webhooks for Location Services

Add Location Zones to a Floorplan

Add Proximity Zones to a Floorplan

Configure Device Events (Use Case)

SUMMARY

Use this example as a model to create webhooks that alert you of device events as they occur for the devices in your deployment.

For this use case, let's say you want to receive a webhook notification anytime a device, such as an access point (AP), disconnects from the cloud for longer than the configured threshold. To configure this notification, you must configure the `Device Events` webhook. See "Create Webhooks from the API" on page 223 or "Add a Webhook in the Juniper Mist Portal" on page 214 for steps on how to create webhooks.

It is useful to receive notifications for `Device Events` because it brings the exact device and its respective issue to your attention. When a device goes offline, for example, every time that event triggers, the cloud sends a webhook notification directly to a hosted system. The system can parse events and use this event as a trigger to run automations.

NOTE: When you configure the `Device Events` webhook topic, you will be notified of any events that affect APs, gateways, and switches. This topic includes events such as `ap_offline`, `device_down`, `switch_down`, `device_restarted`, `gateway_down`, and `switch_restarted`.

Continuing with the example, anytime a device goes offline, Juniper Mist will create a webhook POST to the specified URL. The incoming request payload contains details such as the org_id, site_id, mac, device_name, and timestamp of the event. The JSON payload looks like this:

```
curl -X POST \
  --url 'https://api.mist.com/webhook_example/_device_events_'  \
  -H 'Content-Type: application/json' \
  --data-raw '{
  "events": [
    {
      "ap": "5c5b350e55c8",
      "ap_name": "ap_name6",
      "audit_id": "78c04fa6-cfb4-46a0-9aa5-3681ba4f3897",
      "device_name": "device_name4",
      "device_type": "ap",
```

```
      "ev_type": "notice",
      "mac": "mac2",
      "reason": "device offline",
      "site_name": "office",
      "text": "offline",
      "type": "ap_offline"
    }
  ],
  "topic": "device_events"
}'
```

4

CHAPTER

WebSocket

IN THIS CHAPTER

WebSocket API Overview

SUMMARY

Get starting learning about WebSockets and how you can use them with Juniper Mist™.

IN THIS SECTION

You can use WebSockets in your Juniper Mist network. The WebSocket protocol can open a bidirectional communication session between a client and a server. You can send messages to the server and receive real-time, event-driven responses without having to poll the server for a reply. For example, Websockets are very useful in circumstances where you want to avoid browser refreshes.

The initial client request and server response use the HTTP protocol to establish the WebSocket communication. From then on, the client can subscribe to one or more topics (streaming channels) to stream data.

You and other administrators can use WebSockets in very specific use cases, such as data visualization dashboards or maps that must reflect real-time data values.

Examples

- Populate a custom dashboard with the live status of Juniper Mist access points (APs) and real-time location data of Bluetooth Low Energy (BLE) assets.
- Stream device data and statistics (such as transmit and receive packets) on an hourly basis to an external, operational dashboards like Grafana. Although the device statistics are robust, you can easily parse them to abstract the desired data for display.
- Display packet captures (PCAPs) as they occur without needing to refresh the page.

WebSocket API Endpoint

The API endpoint depends on the global region that your organization is associated with.

Table 18: Endpoints by Global Region

Service Type	Global 01	Global 02	Global 03	Global 04	Global 05	EMEA 01	EMEA 02	EMEA 03	EMEA 04	APAC 01	APAC 02	APAC 03
Admin Portal	api-ws.mist.com	api-ws.gc1.mist.com	api-ws.ac2.mist.com	api-ws.gc2.mist.com	api-ws.gc4.mist.com	api-ws.eu.mist.com	api-ws.gc3.mist.com	api-ws.ac6.mist.com	api-ws.gc6.mist.com	api-ws.ac5.mist.com	api-ws.gc5.mist.com	api-ws.gc7.mist.com
Websocket API	api.mist.com	api.gc1.mist.com	api.ac2.mist.com	api.gc2.mist.com	api.gc4.mist.com	api.eu.mist.com	api.gc3.mist.com	api.ac6.mist.com	api.gc6.mist.com	api.ac5.mist.com	api.gc5.mist.com	api.gc7.mist.com

Authentication Options

Juniper Mist requires authentication to establish a connection to the WebSocket API. You can use these methods:

- "Create API Tokens" on page 15
- HTTP login with Juniper Mist login credentials
- API call to an external OAuth2 provider

WebSocket Streaming Channels

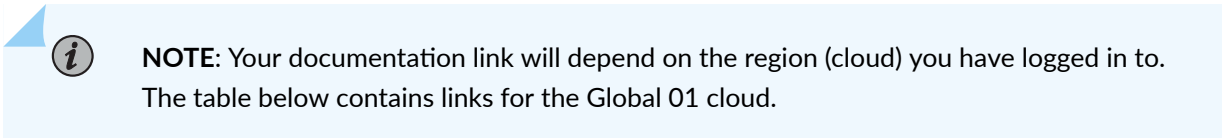
After Juniper Mist establishes a streaming connection with a client, the client needs to subscribe to at least one channel to send and receive messages. Messages go back and forth through the bidirectional WebSocket protocol. To stop sending and receiving messages from a channel, you can unsubscribe from it.

NOTE: All channels require that you specify the site ID. To find a site ID in the Juniper Mist portal, select **Organization > Site Configuration** from the left menu, and then click the site.

Once you are logged in, you can view further information about each WebSocket. Begin by navigating to the web links for WebSocket documentation provided in the next section.

WebSocket Samples and Documentation

As mentioned previously, the WebSocket protocol is used for communication between client and server. You can send messages to the server and receive event-driven responses. The following table contains sample requests and responses as well as links to WebSocket Documentation.



NOTE: Your documentation link will depend on the region (cloud) you have logged in to. The table below contains links for the Global 01 cloud.

Table 19: WebSocket Documentation

Names	Streaming Channels	Sample Requests/Responses and Documentation

Discovery of BLE Assets by Map	/sites/:site_id/stats/maps/:map_id/discovered_assets	To discover BLE assets on the map, issue the following request:
		Request:
		`GET /api/v1/sites/:site_id/stats/maps/:map_id/discovered_assets`
		Response:
		`[{ "mac": "6fa474be7xxx", "device_name": "[TV] UN65JU6xxx", "x": 60, "y": 80, "manufacture": "Apple", "last_seen": 1428939600, // optionally populated "ibeacon_uuid": "f3f17139-704a-f03a-2786-0400279e37c3", "ibeacon_major": 13, "ibeacon_minor": 138, "eddystone_uid_namespace": "2818e3868dec25629ede", "eddystone_uid_instance": "5c5b35000001", "eddystone_url_url": "https://www.abc.com", "mfg_company_id": 935, "mfg_data": "648520a1020000", "duration": 120 }]`

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation
		For more information, see Get Site Discovered Assets by Map.

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

BLE Asset RF Glass Info	/sites/:site_id/assets/:asset_id/diag	To get RF Glass information for BLE Assets, issue the following request:
		Request:
		{
		"subscribe": "/sites/:site_id/assets/:asset_id/diag"
		}
		Response:
		{
		"event": "data",
		"channel": "/sites/4ac1dcf4-9d8b-7211-65c4-057819f0862b/assets/115825352113/diag",
		"data": {
		"map_id": "845a23bf-bed9-e43c-4c86-6fa474be7ae5",
		"grid": {
		"topleft_x_m": -0.86,
		"topleft_y_m": 9.2486,
		"size_m": 0.5,
		"width": 40,
		"height": 40,
		"data": "<base-64 encoded data intended to be interpreted by atob() in JS>",
		},
		"motion": false,
		"vbles": [
		{
		"type": "device",
		"id": "00000000-0000-0000-1000-5c5b350e0060",
		"orientation": 90,
		"xyz_m": [5.79, 4.33, 3.04],

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		"rssis": [-52.32, -53, -55, -57, -60.25, null, null, -62, null]
		},
		{
		"type": "beacon",
		"id": "00000000-0000-0000-1000-e74489000052",
		"xyz_m": [8.79, 10.33, 3.04],
		"rssi": -59.5
		}
		],
		"peak": {
		"vble_id": "00000000-0000-0000-1000-e74489000052",
		"max_rssi": -53.428,
		"plf": -73,
		"ple": -21,
		"intercept": -52
		},
		
		// estimates based on probability surface, we'll always have this
		"raw_xyz_m": [18.7486, 10.13269, 0],
		"smoothed_xyz_m": [18.7486, 10.13269, 0],
		
		"model": "asset",
		"speed": null,
		"direction": null,
		"timestamp": 1501113197.768402,
		
		// the loudest mote estimate
		"closest_mote_xyz_m":

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		[8.79, 10.33, 3.04],
		// 3 past consecutive mote estimates
		"adjusted_mote_xyz_m": [8.79, 10.33, 3.04],
		// another algorithm, more sophisticated
		"vector_mote_xyz_m": [8.79, 10.33, 3.04]
		
		//particle reset flags
		"pf_reset": false,
		"pf_hard_reset": false
		}
		}
		
		For more information, see Location - BLE Assets.

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

SDK Client RF Glass Info (including SDK Client Location)	/sites/:site_id/ sdkclients/:sdkclient_id/diag	To see RF Glass data for SDK Clients, issue the following request:
		Request:
		{
		"subscribe": "/sites/:site_id/ sdkclients/:sdkclient_id/diag"
		}
		Response:
		{
		"event": "data",
		"channel": "/sites/ 4ac1dcf4-9d8b-7211-65c4-057819f086 2b/sdkclients/de87bf9d-183f-e383- cc68-6ba43947d403/diag",
		"data": {
		"map_id": "845a23bf-bed9- e43c-4c86-6fa474be7ae5",
		"grid": {
		"topleft_x_m": -0.86,
		"topleft_y_m": 9.2486,
		"size_m": 0.5,
		"width": 40,
		"height": 40,
		"data": "<base-64 encoded data intended to be interpreted by atob() in JS>",
		},
		"motion": true,
		"avg_duration": 3,
		"vbles": [
		{
		"type": "device",
		"id": "00000000-0000-0000-1000-5c5b350e0 060",

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		"orientation": 90,
		"xyz_m": [5.79, 4.33, 3.04],
		"rssis": [-52.32, -53, -55, -57, -60.25, null, null, -62, null]
		},
		{
		"type": "beacon",
		"id": "00000000-0000-0000-1000-e74489000052",
		"xyz_m": [8.79, 10.33, 3.04],
		"rssi": -59.5
		}
		],
		"peak": {
		"vble_id": "00000000-0000-0000-1000-e74489000052",
		"max_rssi": -53.428,
		"plf": -73,
		"ple": -21,
		"intercept": -52
		},
		
		// estimates based on probability surface, we'll always have this
		"raw_xyz_m": [18.7486, 10.13269, 0],
		"smoothed_xyz_m": [18.7486, 10.13269, 0],
		
		// available only if present
		"app_xyz_m": [18.714, 10.1102, 0],
		"dead_reckoning_xyz_m": [18.791, 10.1613, 0],

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		"dead_reckoning_raw_xyz_m": [18.157, 10.110, 0],
		
		"model": "iPod7",
		"os": "",
		"version": "10.2.1",
		"beams_count": 12,
		"beams_mean": 4,
		"missing_beams": 16,
		"speed": 1,
		"direction": 235,
		"timestamp": 1501113999.758902,
		
		// the loudest mote estimate
		"closest_mote_xyz_m": [8.79, 10.33, 3.04],
		// 3 past consecutive mote estimates
		"adjusted_mote_xyz_m": [8.79, 10.33, 3.04],
		// another algorithm, more sophisticated
		"vector_mote_xyz_m": [8.79, 10.33, 3.04]
		
		//particle reset flags
		"pf_reset": false,
		"pf_hard_reset": false
		
		"latency": 123
		}
		}
		
		For more information, see Get Site SDK Stats by Map and Get Site SDK Stats.

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

Wi-Fi (Client Location)	/sites/:site_id/stats/maps/:map_id/clients	To see wireless client locations, issue the following request:
		Request:
		`GET /api/v1/sites/:site_id/stats/maps/:map_id/clients`
		Response:
		`[{ "mac": "5684dae9ac8b", "last_seen": 1470417522, "username": "david@mist.com", "hostname": "David-Macbook", "os": "OS X 10.10.2", "manufacture": "Apple", "family": "iPhone", "model": "6S", "ip": "192.168.1.8", "ip6": "2001:db8:3333:4444:5555:6666:7777:8888", "ap_mac": "5c5b35000010", "ap_id": "0000000-0000-0000-1000-5c5b35000010", "ssid": "corporate", "wlan_id": "be22bba7-8e22-e1cf-5185-b880816fe2cf", "psk_id": "732daf4e-f51e-8bba-06f9-b25cd0e779ea", "uptime": 3568, "idle_time": 3,`

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation
		"power_saving": true, "band": "24", "proto": "a", "key_mgmt": "WPA2-PSK/CCMP", "dual_band": false, "channel": 7, "vlan_id": "", "airespace_ifname": "", "rssi": -65, "snr": 31, "tx_rate": 65, "rx_rate": 65, "tx_bytes": 175132, "tx_bps": 6, "tx_packets": 1566, "tx_retries": 500, "rx_bytes": 217416, "rx_bps": 12, "rx_packets": 2337, "rx_retries": 5, "map_id": "63eda950-c6da-11e4-a628-60f81dd250cc", "x": 53.5, "y": 173.1, "x_m": 5.35 "y": 17.31 "num_locating_aps": 3, "is_guest": true, "guest": { "authorized": True, "authorized_time": 1428939300, "authorized_expiring_time": 1429109300 "name": "John",

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		<pre> "email":
		"john@abc.com",
		            "company": "ABC",
		            "field1": "whatever",
		            "cross_site": True
		        },
		        "airwatch": {
		            "authorized": True
		        },
		        "_ttl": 250
		    }
		]</pre>
		For more information, see Get Site Wireless Client Stats and Get Site Wireless Client Stats by Map.

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

Wi-Fi Client Stats	/sites/:site_id/stats/clients	To get wireless client statistics, issue the following request:
		Request:
		`GET /api/v1/sites/:site_id/stats/clients`
		Response:
		`[{ "mac": "5684dae9ac8b", "last_seen": 1470417522, "username": "david@mist.com", "hostname": "David-Macbook", "os": "OS X 10.10.2", "manufacture": "Apple", "family": "iPhone", "model": "6S", "ip": "192.168.1.8", "ip6": "2001:db8:3333:4444:5555:6666:7777:8888", "ap_mac": "5c5b35000010", "ap_id": "0000000-0000-0000-1000-5c5b35000010", "ssid": "corporate", "wlan_id": "be22bba7-8e22-e1cf-5185-b880816fe2cf", "psk_id": "732daf4e-f51e-8bba-06f9-b25cd0e779ea", "uptime": 3568, "idle_time": 3,`

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation
		"power_saving": true, "band": "24", "proto": "a", "key_mgmt": "WPA2-PSK/CCMP", "dual_band": false, "channel": 7, "vlan_id": "", "airespace_ifname": "", "rssi": -65, "snr": 31, "tx_rate": 65, "rx_rate": 65, "tx_bytes": 175132, "tx_bps": 6, "tx_packets": 1566, "tx_retries": 500, "rx_bytes": 217416, "rx_bps": 12, "rx_packets": 2337, "rx_retries": 5, "map_id": "63eda950-c6da-11e4-a628-60f81dd250cc", "x": 53.5, "y": 173.1, "x_m": 5.35 "y": 17.31 "num_locating_aps": 3, "is_guest": true, "guest": { "authorized": True, "authorized_time": 1428939300, "authorized_expiring_time": 1429109300 "name": "John",

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		<pre> "email":
		"john@abc.com",
		            "company": "ABC",
		            "field1": "whatever",
		            "cross_site": True
		        },
		        "airwatch": {
		            "authorized": True
		        },
		        "_ttl": 250
		    }
		]</pre>
		For more information, see Get Site Wireless Client Stats.

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

Unconnected Clients	/sites/:site_id/stats/maps/:map_id/unconnected_clients	To get the location of unconnected clients, issue the following request:
		Request:
		`GET /api/v1/sites/:site_id/stats/maps/:map_id/unconnected_clients`
		Response:
		`[ { "mac": "5684dae9ac8b", "ap_mac": "5c5b350e0410", "map_id": "ea77be98-ab51-4ff8-a863-ac3c8e1b1c3a", "x": 60, "y": 80, "rssi": -75.0, "manufacture": "Apple", "last_seen": 1428939600 } ]`
		For more information, see List Site Unconnected Client Stats.

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

Devices	/sites/:site_id/devices	To get a list of devices for a particular site, issue the following:
		Request:
		`GET /api/v1/sites/:site_id/devices`
		Response:
		`[{ "model": "AP41", "hw_rev": "0", "map_id": "01b04bbe-9687-11e8-a5a9-346895ed1b7d", "orientation": 0, "org_id": "476057fe-cebb-4be9-9c15-caf1f09d95e0", "site_id": "eaa6b2b7-88cd-41ea-8150-9b46b6779235", "mac": "5c5b350e0001", "modified_time": 1533206823, "created_time": 1533196761, "tag_id": 107, "tag_uuid": "9c557d6a-8a5e-11e6-b1db-0242ac110004", "serial": "1002710010001", "type": "ap", "id": "00000000-0000-0000-1000-5c5b350e0001", "name": "ap-001" }]`

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation
		For more information, see List Site Devices.

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

Device Stats	/sites/:site_id/stats/devices	To view the current statistics for a particular device, issue the following:
		Request:
		`GET /api/v1/sites/:site_id/stats/devices`
		Response:
		`[{ # information from manufacturing, immutable "mac": "5c5b35000010", "model": "AP200", "type": "ap", "serial": "FXLH2015170017", "last_seen": 1470417522, # configurations "name": "conference room", "map_id": "63eda950-c6da-11e4-a628-60f81dd250cc", "x": 53.5, "y": 173.1, "radio_config": { "band_24": { "channel": 0, "bandwidth": 20, "power": 0, "dynamic_chaining_enabled": false, "tx_chain": 4, "rx_chain": 4 }, "band_5": { "channel": 0, "bandwidth": 40,`

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		"power": 0,
		"dynamic_chaining_enabled": false,
		"tx_chain": 1,
		"rx_chain": 4
		},
		"band_6": {
		"channel": 0,
		"bandwidth": 40,
		"power": 0,
		"tx_chain": 1,
		"rx_chain": 4
		},
		"scanning_enabled": true
		},
		"ip_config": {
		"type": "static",
		"ip": "10.2.1.1",
		"netmask": "255.255.255.0",
		"gateway": "10.2.1.254",
		"dns": ["8.8.8.8", "4.4.4.4"],
		"dns_suffix": [".mist.local", ".mist.com"]
		},
		"ble_config": {
		"power_mode": "custom",
		"power": 10,
		"beacon_rate_model": "custom",
		"beacon_rate": 3,
		"beam_disabled": [1, 3, 6]
		},
		"led": {
		"enabled": true,
		"brightness": 255
		},

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation
		`# current stat "status": "connected", "version": "1.0.0", "ip": "10.2.9.159", // first IP "ext_ip": "73.92.124.103", "num_clients": 10, "uptime": 13500, "tx_bps": 634301, "rx_bps": 60003, "tx_bytes": 211217389682, "tx_pkts": 812204062, "rx_bytes": 8515104416, "rx_pkts": 57770567, "locating": false, "radio_stat": { "band_24": { "mac": "5c5b350004a0" "channel": 6, "bandwidth": 20, "power": 19, "num_clients": 6, "tx_bytes": 211166512114, "tx_pkts": 812058566, "rx_bytes": 8504737800, "rx_pkts": 57731964 }, "band_5": { "mac": "5c5b350004b0" "channel": 44, "bandwidth": 80, "power": 15, "num_clients": 4,`

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		"tx_bytes": 50877568,
		"tx_pkts": 145496,
		"rx_bytes": 10366616,
		"rx_pkts": 38603
		}
		},
		"port_stat": {
		"eth0": {
		"up": true,
		"speed": 1000,
		"full_duplex": true,
		"tx_bytes": 2056,
		"tx_pkts": 670,
		"rx_bytes": 2056,
		"rx_pkts": 670,
		"rx_errors": 0,
		},
		"eth1": {
		"up": false
		},
		"module": {
		"up": false
		}
		},
		# `ports` is like `port_stat` but is an array.
		# This array can be converted to a dict using (port_id, node) as key.
		"ports": [
		// Each port object is same as `GET /api/v1/sites/:site_id/stats/ports/search` result object,
		// except that org_id, site_id, mac, timestamp are removed
		{
		"port_id":

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		"ge-0/0/1",
		"node":
		"node0", // Absent if this
		device is standalone
		.
		.
		.
		}
		]
		"ip_stat": {
		"ip": "10.2.1.1",
		"netmask":
		"255.255.255.0",
		"gateway":
		"10.2.1.254",
		"ip6":
		"2607:f8b0:4005:808::2004",
		"netmask6": "/32",
		"gateway6":
		"2607:f8b0:4005:808::1",
		"dns": ["8.8.8.8",
		"4.4.4.4"],
		"dns_suffix":
		[".mist.local", ".mist.com"],
		"ips": {
		"vlan1":
		"10.2.1.1/24,2607:f8b0:4005:808::1
		/32",
		"vlan193":
		"10.73.1.31/16",
		"vlan3157":
		"10.72.11.14/24"
		}
		},
		"ble_stat": {
		"power": 10,
		"beacon_rate": 3,
		"uuid":
		"ada72f8f-1643-e5c6-94db-
		f2a5636f1a64",
		"major": 12345,
		"minors": [201, 202,

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		203, 204, 205, 206, 207, 208],
		"tx_pkts": 135135135,
		"tx_bytes": 5231513353,
		"tx_resets": 0,
		"rx_pkts": 135,
		"rx_bytes": 135,
		
		"ibeacon_enabled": true,
		"ibeacon_uuid": "f3f17139-704a-f03a-2786-0400279e37c3",
		"ibeacon_major": 13,
		"ibeacon_minor": 138,
		
		"eddystone_uid_enabled": false,
		
		"eddystone_uid_namespace": "2818e3868dec25629ede",
		
		"eddystone_uid_instance": "5c5b35000001",
		
		"eddystone_uid_freq_msec": 200,
		
		"eddystone_url_enabled": true,
		"eddystone_url_url": "https://www.abc.com",
		
		"eddystone_url_freq_msec": 100
		},
		"l2tp_stat": {
		"7dae216d-7c98-a51b-e068-dd7d477b7216": {
		"wxtunnel_id": "7dae216d-7c98-a51b-e068-dd7d477b7216",
		"state": "established_with_sessions",

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		"uptime": 135,
		"sessions: [
		{
		
		"remote_id": "vpn1",
		"state": "established",
		
		"remote_sid": 13,
		
		"local_sid": 31
		}
		]
		}
		},
		"lldp_stat": {
		"system_name": "TC2-OWL-Stack-01",
		"system_desc": "HP J9729A 2920-48G-POE+ Switch",
		"mgmt_addr": "10.1.5.2",
		"port_desc": "2/26",
		"chassis_id": "63:68:61:73:73:69",
		"lldp_med_supported": false,
		
		"power_request_count": 3,
		"power_allocated": 15500,
		"power_requested": 25500,
		"power_draw": 15000
		},
		"power_src": "PoE 802.3af",
		"power_budget": -12000,
		"power_constrained": true,
		"power_opmode": "[20] 6GHz(2x2) 5GHz(4x4) 2.4GHz(2x2).",

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		"switch_redundancy": {
		"num_redundant_aps": 1,
		},
		
		// IoT stats
		"iot_stat": {
		"DI2": {
		"value": 0
		}
		},
		
		// Environment stats
		"env_stat": {
		"cpu_temp": 51,
		"ambient_temp": 39,
		"humidity": 11,
		"attitude": 0,
		"pressure": 1015
		"accel_x": -0.012,
		"accel_y": 0.004,
		"accel_z": -1.012,
		"magne_x": 0.0,
		"magne_y": 1.3,
		"magne_z": 0.0,
		"vcore_voltage": 0
		},
		"mount": 'faceup',
		
		// ESL Stats
		"esl_stat": {
		"up": true,
		"type": "imagotag", // if up
		
		// following are type-dependent
		"connected": true,
		"channel": 3,
		},

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		// for a base AP
		"mesh_downlinks": {
		
		"00000000-0000-0000-1000-5c5b356be59f": {
		"site_id": "0e525da3-6033-428c-9a51-9f652f643baf",
		"band": "24",
		"proto": "a",
		"channel": 7,
		
		"last_seen": 1470417522,
		"idle_time": 3,
		
		"rssi": -65,
		"snr": 31,
		"tx_rate": 65,
		"rx_rate": 65,
		
		"tx_bytes": 175132,
		"tx_bps": 6,
		"tx_packets": 1566,
		"tx_retries": 500,
		"rx_bytes": 217416,
		"rx_bps": 12,
		"rx_packets": 2337,
		"rx_retries": 5
		}
		},
		
		// for a remote/relay AP
		"mesh_uplink": {
		"uplink_ap_id": "00000000-0000-0000-1000-5c5b35000010",
		"uplink_site_id":

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		"1916d52a-4a90-11e5-8b45-1258369c38a9",
		"band": "24",
		"proto": "a",
		"channel": 7,
		
		"last_seen": 1470417522,
		"idle_time": 3,
		
		"rssi": -65,
		"snr": 31,
		"tx_rate": 65,
		"rx_rate": 65,
		
		"tx_bytes": 175132,
		"tx_bps": 6,
		"tx_packets": 1566,
		"tx_retries": 500,
		"rx_bytes": 217416,
		"rx_bps": 12,
		"rx_packets": 2337,
		"rx_retries": 5
		},
		"fwupdate": {
		"timestamp": 1428949501,
		"status": "inprogress",
		"status_id": 5,
		"progress": 10
		},
		
		"last_trouble": {
		"code": "03",
		"timestamp": 1428949501
		},
		
		// if RADSec is enabled, device certs will be automatically generated and

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

		managed
		// with the expiration time exposed
		"cert_expiry": 1534534392
		"locked": false,
		"auto_placement": {
		"x": 53.5,
		"y": 173.1,
		"x_m": 5.35,
		"y_m": 17.31,
		"status": "localized",
		"status_detail": "localized",
		"use_auto_placement": false,
		"recommended_anchor": false,
		"info": {
		"cluster_number": 0,
		"orientation_state": 0,
		"probability_surface": {
		"radius": 2.1,
		"x": 5.65,
		"y": 17.10
		}
		},
		"_id": "5c5b35000010"
		}
		}
		]
		For more information, see Get Site Device Stats.

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

Commands from Devices	/sites/:site_id/ devices/:device_id/cmd	To subscribe to device command outputs, issue the following request:
		Request:
		`{ "subscribe": "/sites/:site_id/ devices/:device_id/cmd" }`
		Response:
		`{ "event": "data", "channel": "/sites/ 4ac1dcf4-9d8b-7211-65c4-057819f086 2b/devices/ 00000000-0000-0000-1000-5c5b350e00 60/cmd", "data": { "session": "session_id", "raw": "64 bytes from 23.211.0.110: seq=8 ttl=58 time=12.323 ms\n" } }`
		For more information, see Device - Command Output.

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation

Streaming PCAP	/sites/:site_id/pcaps	To subscribe to streaming Packet Capture (PCAP), issue the following request:
		Request:
		{
		subscribe: "/sites/:site_id/pcaps"
		}
		Response:
		{
		"event": "data"
		"channel": "/sites/:site_id/pcaps"
		"data": {
		"capture_id": "6b1be4fb-b239-44d9-9d3b-cb1ff3af1721"
		"pcap_dict": {
		"channel_frequency": 2412,
		"channel": "1",
		"datarate": "1.0 Mbps",
		"rssi": -75,
		"dst": "78:bd:bc:ca:0b:0a",
		"src": "18:b8:1f:4c:91:c0",
		"bssid": "18:b8:1f:4c:91:c0",
		"frame_type": "Management",
		"frame_subtype": "Probe Response",
		"proto": "802.11",
		"ap_mac": "d4:20:b0:81:99:2e",
		"direction": "tx",

Table 19: WebSocket Documentation *(Continued)*

Names	Streaming Channels	Sample Requests/Responses and Documentation
		"timestamp": 1652246543, "length": 416.0, "interface": "radiotap", "info": "1652246544.467733 1683216786us tsft 1.0 Mb/s 2412 MHz 11g -75dBm signal -82dBm noise antenna 0 Probe Response (ATTKmsWiVS) [1.0* 2.0* 5.5* 11.0* 18.0 24.0 36.0 54.0 Mbit] CH: 2, PRIVACY\\n", }, "pcap_raw": "1MOyoQIABAAAAAAAAAAAAP// AAABAAAAEEh7Yh5VBwCgAQAAoAEAAAAAKw BvCADAAQAAAIw7reCS2VNkAAAAABACbAmA BLWuAAEAEBgAAwACAABQADoBeL28ygsKGL gfTJHAGLgfTJHAcIZ2WDlBJQAAAGQAERUA CkFUVEttc1dpVlMBCIKEi5YkMEhsAwECBw ZVUyABCx4gAQAjAhkAKgEEMgQMEhhgMBQB AAAPrAQBAAAPrAQBAAAPrAIMAAsFAQAbAA BGBTIIAQAALRqtCR//// 8AAAAAAAAAAAAAAAAAAAAAAAAAAD0WAggV AAAAAAAAAAAAAAAAAAAAAAAAAH8IBAAIAA AAAEDdkwBQ8gQQSgABEBBEAAECEDsAAQMQ RwAQn2481frn3KT+uGod2ERx +RAhAAtBcnJpcywgSW5jLhAjAApCR1cyMT AtNzAwECQACkJHVzIxMC03MDAQQgAKQkdX MjEwLTcwMBBUAAgABgBQ8gQAARARAA5Bcn JpcyBXaXJlbGVzcxAIAAIgCBA8AAEBEEkA BgA3KgABIN0JABAYAgEQHAAA3RgAUPICAQ GEAAOkAAAnpAAAQkNeAGIyLwAzjakr\"

}

For more information, see PCAP. |

RELATED DOCUMENTATION

https://datatracker.ietf.org/doc/html/rfc6455

Get Started with WebSocket

IN THIS SECTION

Use Postman to Connect to the WebSocket API

SUMMARY

Postman is a platform that is designed to make it easy to work with APIs. This topic walks you through how to use Postman to connect to the Mist WebSocket API and stream data.

IN THIS SECTION

Postman is an API platform that makes it easy for you to complete API related tasks, including sending and receiving data from a WebSocket server. This topic covers how to get set up in Postman and how to leverage it so that you can use the Mist WebSocket API to stream data.

See

Postman Setup

To use Postman, you can use the Postman website or download the Postman application as described in Download Postman.

1. Sign in to Postman (or create an account) from the Postman website or application. This allows you to save your environment. See the Create Your Environment section further down in this topic.

2. Once your account is created, you get access to your workspace. This is where you can save your API calls and configure your environment to interact with the Mist API.

Import the Mist API Collection

Next, import the Mist API Collection. Juniper Mist has built a list of Postman API calls that you can import directly into your Postman workspace. This list is maintained and matches what the API documentation lists.

1. Navigate to the Juniper Mist Postman collections page and select the **Mist Cloud Websockets** collection.
2. Once the collection is opened, click on **Fork** as described in Fork collections and environments in Postman. This enables you to create a copy of the collection in your own workspace and still receive updates when the main collection is updated.
3. In the top left corner of Postman, you should see that the collection has now been forked into your workspace. Expand the collection and its subsections to see how all the WebSocket APi requests are organized.

Create Your Environment

A postman environment allows you to store variables in a profile that you can reuse across multiple API calls and collections. You must create an environment and define variables before you begin in Postman.

1. Create and name your environment as described in Create an Environment.
2. Define variables in your Postman environment by entering them into the table as described in Add Environment Variables.
   - In order to interact with the Mist WebSocket API, you must set the following variables:

Table 20: Environment Variables

Variable	Description
`host`	This is the URL of the Mist API endpoint. For exam See API Endpoints and Global Regions for the full l
`host-ws`	This is the URL of the Mist websocket API endpoi access to the GLOBAL 01 region. The full list of M Endpoints list" on page 242.
`apitoken`	This is the Mist API token required to authenticate create a Mist API token for REST API, see Create A

org_id	This is the UUID of your Mist organization. For mo Find Your Organization ID and Get Org.
site_id	This is the UUID of a specific Mist site. For more ir Info.

3. Once you have entered all your variables, select **Save**.
4. Apply the environment to your Postman workspace by navigating to the **Environment** drop-down menu in the top right corner of the page, then select your newly created environment.

Connect to the WebSocket API

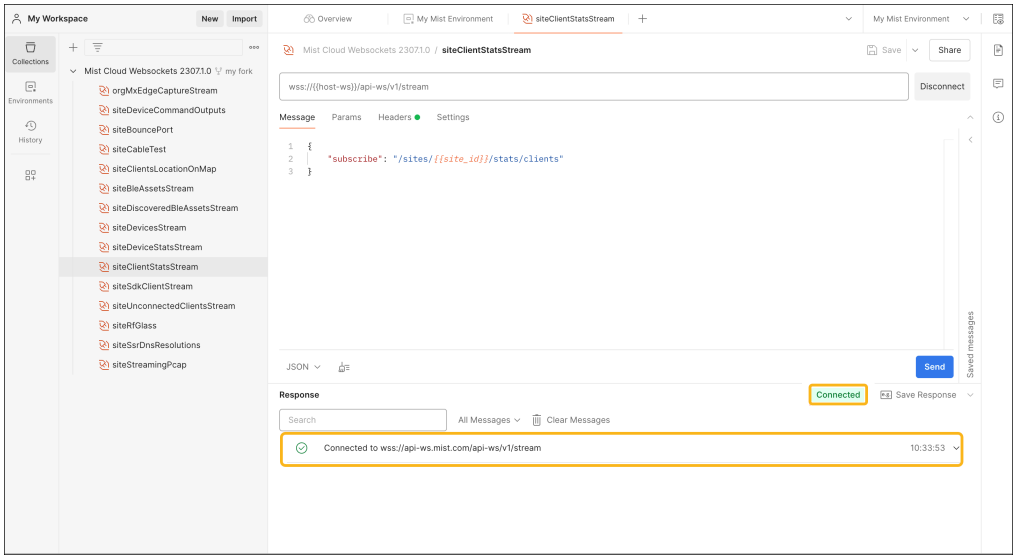
Now that you have set up Postman with all of the necessary information, it's time to test the connection to the WebSocket API. You can do this using the `GET /api/v1/self` call to be returned with information about your privileges and accounts. You can view steps on how to this in "Test Your First API Call" on page 152, or see full documentation on this here.

Let's say the first set of data you want to stream is related to Wi-Fi client statistics. To stream this data, you must connect to the WebSocket API and then subscribe to the `/sites/<i>{{site_id}}</i>/stats/clients` channel.

In order to test your connection to the Juniper Mist WebSocket API, follow these steps:

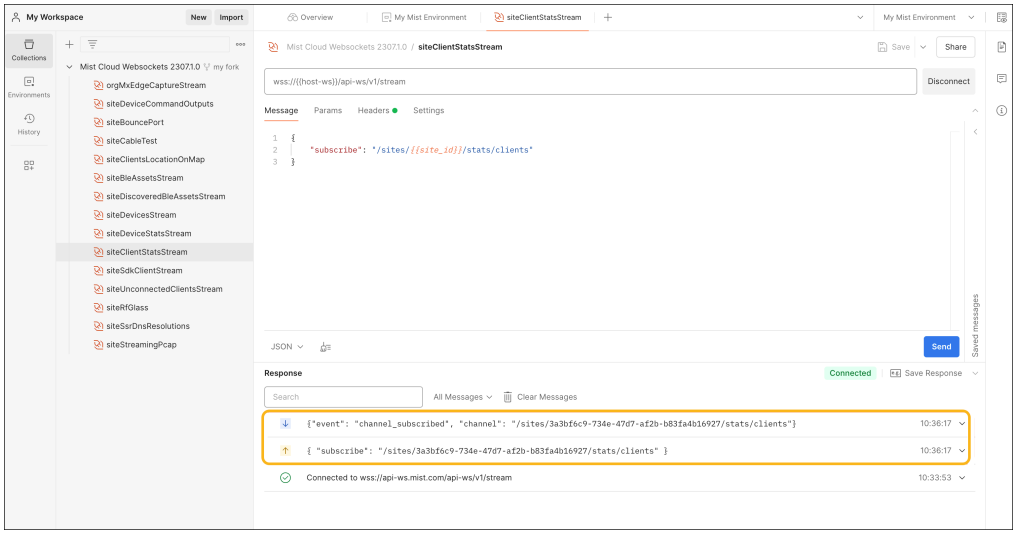
1. In Postman, select **Collections** on the left menu. Then, open the **Mist Cloud Websockets** section. Finally, click on the `siteClientStatsStream` request.
2. Next, click the **Connect** button to establish a connection with the Mist WebSocket API. This request will use the `{{apitoken}}`, the `{{ws-host}}`, and the `{{site_id}}` variables from your environment.

3. If you were able to successfully connect to the websocket API, you should see a message detailing your connection as well as a green check mark.



4. Next, you must subscribe to the channel so that you can start to receive data. To do this, click the **Send** button.

5. Mist will then send you a message confirming that you have subscribed to the channel.



6. Now that you are subscribed to the channel, you should see messages containing client statistics stream in from Mist. You can view these in the Response section of Postman.

7. When you are ready to disconnect from the WebSocket API, click the **Disconnect** button. You will then receive a message in the Response section indicating that you are now disconnected.

   Now you are ready to explore other streaming channels. Below are some suggestions:

   - - `siteDeviceStatsStream`—Use this stream to get statistics about your network devices that are managed by Mist.
     - `siteStreamingPcap`—Use this to perform packet captures (PCAPs).
     - `siteBleAssetsStream`—Use this stream to retrieve location data about your BLE asset tag.

WebSocket Use Cases

IN THIS SECTION

Stream Device Data with a WebSocket (Use Case)

SUMMARY

In this example, we will discuss the Juniper Mist use case of streaming device data with a WebSocket.

IN THIS SECTION

A WebSocket is a protocol that provides full-duplex communication over a TCP connection. A WebSocket API provides a way for a client to communicate with an endpoint. Juniper Mist uses this protocol to stream near-real-time data to a client. A client will make a request for the data it wants to receive by subscribing to a channel. The client makes the request only once, and the server will stream the channel data to the client as updates are made.

This method of communication is good for receiving event-driven data in near–real time. Its primary use is for collecting device data. You and other network administrators can then feed this data into a custom event display, a notification system, an external logging facility, and much more.

Using WebSocket is especially helpful if the information that your organization needs isn't available natively within the Juniper Mist portal.

In this use case, we show how to configure a client to request and subscribe to a site's device statistics.

Using a WebSocket requires the following elements:

- Authentication
- HTTP header configuration
- WebSocket Connection URL (wss://api-ws.mist.com/api-ws/v1/stream). Please verify your URL based on your geographical location.

NOTE: When reusing code blocks, replace placeholder values with actual values, such as your API token, organization ID, site ID, AP name, and so on.

Communicate with a MIST WebSocket Endpoint

To communicate with the Juniper Mist WebSocket endpoint, you use an application called Postman. Postman is a GUI API platform for building and using APIs. You can build scripts that make multiple API calls. This API also enables you to make WebSocket calls.

The following steps describe how you can connect to the Juniper Mist WebSocket, starting with adding the URL to the input box next to the Connect button:

1. Click **Connect**.
   The Messages pane shows that the client tried to connect and was immediately disconnected. This is because the Juniper Mist WebSocket requires authentication.
2. To fix the authentication problem, add a custom HTTP header to the request.
   a. Under the KEY heading, create a new “Authorization” key.
   b. In the VALUE input box, enter the word **token** followed by a space and then the Juniper Mist authorization token.
   c. Select the check box on the left of the KEY column to make it active.
   d. Click **Connect** to establish the WebSocket session with Mist.
3. Once connected, switch to the Message heading in Postman.
   This is where you tell Mist which data you want, by subscribing to the device statistics stream.

   The following example requests device statistic data for the specified site.

```
{
    "subscribe": "/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxxxx/stats/devices"
}
```

4. With this information entered in the Messages input box, click **Send**.
   Postman presents your request with an Up Arrow and the Juniper Mist response with a Down Arrow.

   The following example shows what you can expect to see in a response from Juniper Mist.

```
{
    "event": "channel_subscribed",
    "channel": "/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxx/stats/devices"
}
```

   Within a few seconds of subscribing, you should start to see events streaming into the Messages pane in JSON format (default):

```
1. {
2.     "event": "data",
3.     "channel": "/sites/c1947558-268d-4d31- xxxx - xxxxxxxxxxx /stats/devices",
4.     "data": "{\"mac\": \"5c5b35f15ed8\", \"last_seen\": 1686592607, \"uptime\": 6259614,
\"version\": \"0.9.22801\", \"_partition\": 48, \"_offset_apbasic\": 4639768722, \"ip_stat\":
{\"dns\": [\"10.10.12.11\", \"10.10.12.12\"], \"ips\": {\"vlan12\":
\"10.10.12.25/25,fe80:0:0:0:5e5b:35ff:fef1:5ed8/64\"}, \"gateway\": \"10.10.12.1\", \"ip6\":
```

```
\"fe80:0:0:0:5e5b:35ff:fef1:5ed8\", \"netmask6\": \"/64\", \"ip\": \"10.10.12.25\", \"netmask
\": \"255.255.255.128\", \"dhcp_server\": \"10.10.12.1\"}, \"ip\": \"10.10.12.25\", \"ble_stat
\": {\"tx_pkts\": 9073, \"tx_bytes\": 105431, \"rx_pkts\": 393432193, \"rx_bytes\":
2772782221, \"tx_resets\": 0}, \"_time\": 1686592607.62131}"
5. }
```

You can see the same output in the following formats:

- **Text**

  1. {"event": "data", "channel": "/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxxx/stats/devices", "data": "{\"mac\": \"5c5b35fxxxxx\", \"last_seen\": 1686592607, \"uptime\": 6259614, \"version\": \"0.9.22801\", \"_partition\": 48, \"_offset_apbasic\": 4639768722, \"ip_stat\": {\"dns\": [\"10.10.12.11\", \"10.10.12.12\"], \"ips\": {\"vlan12\": \"10.10.12.25/25,fe80:0:0:0:5e5b:35ff:fef1:5ed8/64\"}, \"gateway\": \"10.10.12.1\", \"ip6\": \"fe80:0:0:0:5e5b:35ff:fef1:5ed8\", \"netmask6\": \"/64\", \"ip\": \"10.10.12.25\", \"netmask\": \"255.255.255.128\", \"dhcp_server\": \"10.10.12.1\"}, \"ip\": \"10.10.12.25\", \"ble_stat\": {\"tx_pkts\": 9073, \"tx_bytes\": 105431, \"rx_pkts\": 393432193, \"rx_bytes\": 2772782221, \"tx_resets\": 0}, \"_time\": 1686592607.62131}"}

- **HTML**

  1. {"event": "data", "channel": "/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxxx/stats/devices", "data": "{\"mac\":

  2. \"5c5b35fxxxxx\", \"last_seen\": 1686592607, \"uptime\": 6259614, \"version\": \"0.9.22801\", \"_partition\": 48,

  3. \"_offset_apbasic\": 4639768722, \"ip_stat\": {\"dns\": [\"10.10.12.11\", \"10.10.12.12\"], \"ips \": {\"vlan12\":

  4. \"10.10.12.25/25,fe80:0:0:0:5e5b:35ff:fef1:5ed8/64\"}, \"gateway\": \"10.10.12.1\", \"ip6\":

  5. \"fe80:0:0:0:5e5b:35ff:fef1:5ed8\", \"netmask6\": \"/64\", \"ip\": \"10.10.12.25\", \"netmask\": \"255.255.255.128\",

  6. \"dhcp_server\": \"10.10.12.1\"}, \"ip\": \"10.10.12.25\", \"ble_stat\": {\"tx_pkts\": 9073, \"tx_bytes\": 105431,

  7. \"rx_pkts\": 393432193, \"rx_bytes\": 2772782221, \"tx_resets\": 0}, \"_time\": 1686592607.62131}"}

- **XML**

  1. {"event": "data", "channel": "/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxxx/stats/devices", "data": "{\"mac\": \"5c5b35fxxxxx\", \"last_seen\": 1686592607, \"uptime\": 6259614, \"version\": \"0.9.22801\", \"_partition\": 48, \"_offset_apbasic\": 4639768722, \"ip_stat\": {\"dns\":

[\"10.10.12.11\", \"10.10.12.12\"], \"ips\": {\"vlan12\": \"10.10.12.25/25,fe80:0:0:0:5e5b:35ff:fef1:5ed8/64\"}, \"gateway\": \"10.10.12.1\", \"ip6\": \"fe80:0:0:0:5e5b:35ff:fef1:5ed8\", \"netmask6\": \"/64\", \"ip\": \"10.10.12.25\", \"netmask\": \"255.255.255.128\", \"dhcp_server\": \"10.10.12.1\"}, \"ip\": \"10.10.12.25\", \"ble_stat\": {\"tx_pkts\": 9073, \"tx_bytes\": 105431, \"rx_pkts\": 393432193, \"rx_bytes\": 2772782221, \"tx_resets\": 0}, \"_time\": 1686592607.62131}"}

You will continue to receive messages until you either disconnect or unsubscribe from the channel by sending a message.

```
{
    "unsubscribe": "/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxxxx/stats/devices"
}
```

Once you receive the data, you can do any number of things with it, such as:

- Create a custom display or dashboard of events.
- Archive the data long term.
- Create custom monitoring and alerts.
- Create more automations based on results.

For examples using other automation tools to communicate with a MIST WebSocket endpoint, such as Python, see "Automation Tools" on page 300.

Stream Packet Captures with a WebSocket (Use Case)

SUMMARY

In this example, we discuss the Juniper Mist use case of streaming device packet captures with a WebSocket.

IN THIS SECTION

As described in "Stream Device Data with a WebSocket (Use Case)" on page 281, the WebSocket API enables client communication with an endpoint. You can use this method of communication for situations where you want to receive near-real time, event-driven data without having to refresh the web browser.

This use case outlines how you can use a WebSocket to stream packet captures (PCAPs). PCAPs capture data packets as they traverse the network, and serve as a powerful troubleshooting tool to help debug traffic issues, as well as broader networking issues.

While you have the ability to download PCAPs from the Mist portal, PCAPs that occurred in the past might only be beneficial in some troubleshooting scenarios. Streaming PCAPs with a WebSocket, on the other hand, enables you to monitor and analyze packets as they traverse the network, in near-real time. With Juniper Mist, you can analyze PCAPs for wired, wireless, and WAN connections.

To establish client communication with an endpoint, two things must occur:

- A client must subscribe to a channel in order to request the data it wants to receive.
- Then, the server streams the channel data to the client as events occur.

Using a WebSocket requires the following elements:

- Authentication
- HTTP header configuration
- WebSocket Connection URL (`wss://api-ws.mist.com/api-ws/v1/stream`). Please verify your URL based on your geographical location. See "WebSocket API Endpoint" on page 242.

NOTE: When reusing code blocks, replace placeholder values with actual values, such as your API token, organization ID, site ID, AP name, and so on.

Communicate with a MIST WebSocket Endpoint

To communicate with the Juniper Mist WebSocket endpoint, you use an application called Postman. Postman is a GUI API platform for building and using APIs. You can use Postman to carry out any API related tasks, including making WebSocket calls and streaming data.

For general information about getting started with Postman, see "Use Postman to Make Your First API Call" on page 150 and Get Started in Postman.

To learn how to create a WebSocket request in Postman, see Create a WebSocket Request.

The following steps describe how you can connect to the Juniper Mist WebSocket:

1. In Postman, Select **New** > **WebSocket**.
2. Enter the WebSocket server URL `wss://api-ws.mist.com/api-ws/v1/stream`, then select **Connect**.
3. Follow the instructions outlined in "Communicate with a MIST WebSocket Endpoint" on page 281 for information on how to connect and authenticate.
4. Once connected, select the **Message** heading in Postman.

This is where you tell Mist that you want to stream PCAPs. You must subscribe to the PCAPs stream to accomplish this. The following subscribe command requests PCAP data for the specified site:

```
{
    "subscribe": "/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxxxx/pcaps"
}
```

5. Once this information is entered in the **Messages** field, select **Send**.

   Postman presents your request with an Up Arrow and the Juniper Mist response with a Down Arrow.

   The following example shows what you can expect to see in a response from Juniper Mist. Within a few seconds after you subscribe, you should start to see events streaming into the Messages pane in JSON format (default):

```
{
    "event": "data",
    "channel": "/sites/67970e46-4e12-11e6-9188-0242ac110007/pcaps",
    "data": {
        "capture_id": "f039b1b4-a23e-48b2-906a-0da40524de73",
        "pcap_dict": {
            "dst_mac": "68:ec:c5:09:2e:87",
            "src_mac": "8c:3b:ad:e0:47:40",
            "vlan": 1,
            "src_ip": "34.224.147.117",
            "dst_ip": "192.168.1.55",
            "dst_port": 51635,
            "src_port": 443,
            "proto": "TCP",
            "ap_mac": "d4:20:b0:81:99:2e",
            "direction": "tx",
            "timestamp": 1652247615,
            "length": 159.0,
            "interface": "wired",
            "info": "1652247616.007409 IP ec2-34-224-147-117.compute-1.amazonaws.com.https >
ip-192-168-1-55.ec2.internal.51635: Flags [P.], seq 2192123968:2192124057, ack 4035166782,
win 12, options [nop,nop,TS val 597467050 ecr 740580660], length 89\\n",
        },
        "pcap_raw": "1MOyoQIABAAAAAAAAAAAAP//
AAABAAAAQEx7YhMzAACfAAAAnwAAAGjsxQkuh4w7reBHQIEAAAEIAEUAAI1bLEAAKAZ/
CiLgk3XAqAE3AbvJs4KpKEDwg8I+gBgADFf9AAABAQgKI5yfqiwkXTQXAwMAVKY5JopoKQrVEn0/3ld4YntctGEH/
rTZuwtCvzSncFw71QJveJi9uxHs57KC8w9Apph3YvXJrmWg7M37+o+YV0KH/xmr626s5Bkhb3QhKOu+NoNEmA=="
```

```
    }
}
```

You can see the same output in the following formats by changing your selection from the drop-down menu above the Response pane in Postman:

- **Text**
- **HTML**
- **XML**
- **Binary**

You will continue to receive messages until you either disconnect or unsubscribe from the channel by sending a message.

```
{
    "unsubscribe": "/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxxxx/pcaps"
}
```

Once you receive the data, you can do any number of things with it, such as:

- Create a custom display or dashboard of events.
- Archive the data long term.
- Create custom monitoring and alerts.
- Create more automation based on results.

For examples using other automation tools to communicate with a MIST WebSocket endpoint, such as Python, see "Automation Tools" on page 300.

5
CHAPTER

Third-Party Integrations

IN THIS CHAPTER

ServiceNow Integrations

SUMMARY

You can integrate your Juniper Mist organizations with ServiceNow to view device information and Marvis Actions in ServiceNow. You can also set up ServiceNow to automate workflows for common Mist tasks.

IN THIS SECTION

Video Overview

Video: Juniper and ServiceNow – Transform Network Management

Integrations

You can integrate Juniper Mist data into ServiceNow by using the following applications from the ServiceNow website.

- **Service Graph Connector (SGC) for Juniper Mist**—This integration provides the ability to import your Juniper Mist device inventory into ServiceNow so that you can monitor and manage devices in ServiceNow. Likewise, SGC can also be used for compliance. Most customers start with SGC.
  - Use this integration with individual organizations, MSPs, and large enterprises.
  - Provides visibility for all Mist-connected network inventory including Access Points (APs), EX Series Switches, SRX Series Firewalls, Session Smart Routers (SSRs), and Mist Edges.
  - If you have a Marvis Virtual Network Assistant subscription for Juniper Mist, you also can set up auto-ticketing to automatically create/update/close incidents based on Marvis detections, and link them to the related CI.
  - For more details, including the SGC installation instructions and admin guide, go to Service Now's SGC page.
- **Juniper Mist Automated Provisioning (JMAP)**—This integration provides tools to automate Mist Day 0 and Day1+ tasks directly from ServiceNow.

  - Can be used for a single organization, with your managed service provider (MSP) account, or with large enterprises.
  - Can be used to automate network infrastructure and services deployments from ServiceNow: create and configure organizations and sites, manage licenses, claim devices, and more.
    - Can customize Mist templates and map the templates to the site(s).
  - Available for Mist wired, wireless, SD-WAN, security services.
  - Provides “actions” to easily create automation work flows with ServiceNow's no-code UI.
  - Request firmware upgrades for Mist-managed devices from ServiceNow.
  - For more details, including the JMAP installation instructions and admin guide, go to Service Now's JMAP page.

- **Event Management for Juniper Mist**—The Juniper Mist Push Connector pushes events and alerts from Mist into ServiceNow so that you can leverage automated network management for Day 2+ tasks from within the ServiceNow platform.
  - Use this integration with individual organizations, MSPs, or large enterprises.
  - Get real-time visibility for all Mist device events and alerts in ServiceNow via the Push Connector for Mist wired, wireless, SD-WAN, and security devices.
  - Automatically generate incidents in ServiceNow based on predefined or custom alert rules (this must be configured separately)—without needing to access the Mist dashboard.
  - Use the ServiceNow Event Management table to leverage AIOps (analyze, consolidate, streamline, and take action on events).
  - For more details, including the Event Management installation instructions and admin guide, go to Service Now's Event Management page.

NOTE: The above applications are free of charge, but it is important to know that additional licenses on the ServiceNow side may be required. See the Requirements and Dependencies sections on the ServiceNow website to learn more.

Integrate Splunk with Mist Webhooks

SUMMARY

Follow these instructions to configure Splunk and to set up webhooks from Juniper Mist™ to your Splunk instance.

IN THIS SECTION

Splunk uses HTTP Event Collection (HEC) to receive HTTP POST requests that include a payload of data. This enables cloud services like Mist to send data to Splunk using webhooks.

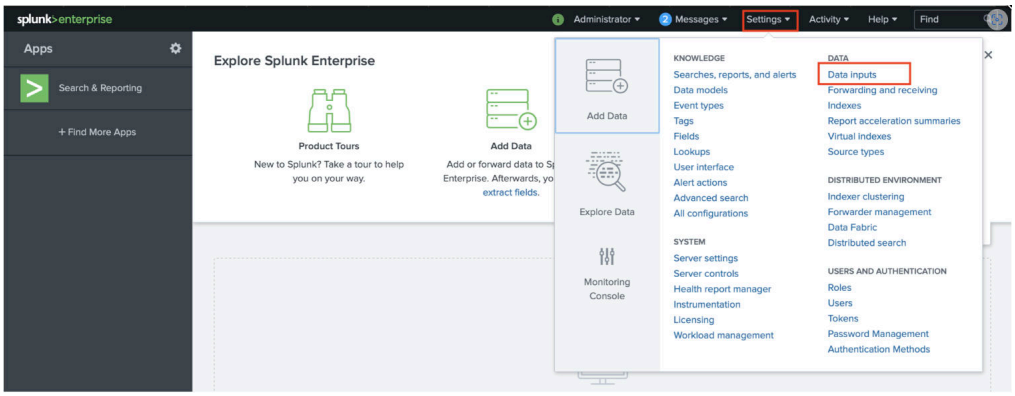
How you implement HEC depends on your Splunk service:

- If you're running a managed Splunk Cloud, you'll need to raise a ticket with your Splunk support to have HEC configured.
- If you have self-service Splunk or Splunk Enterprise (on-premise), you can use the following instructions to configure HEC to receive webhooks from Mist.

Configuring HEC in Splunk

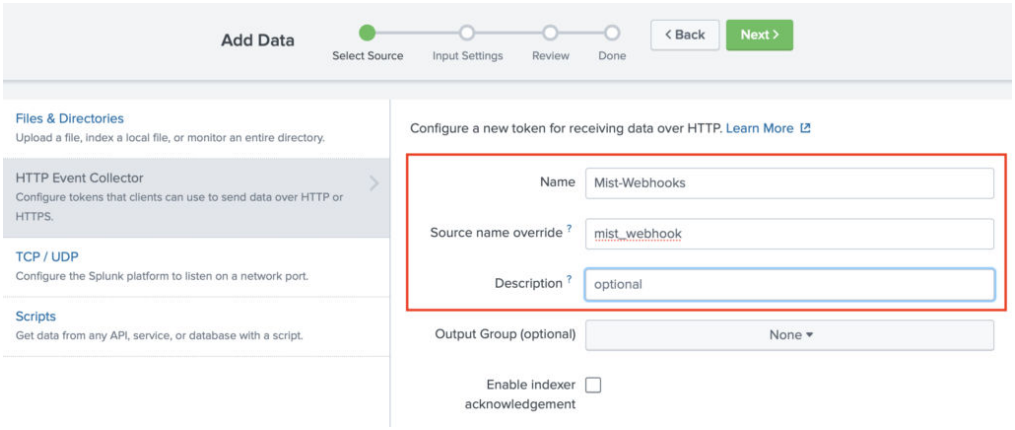
Before you begin: Webhooks requires IP reachability from the Mist Cloud to your Splunk instance. In other words, you need a publicly accessible URL for your Splunk server with the HTTP port open.

1. In your Splunk GUI, go to **Settings** > **Data Inputs**.

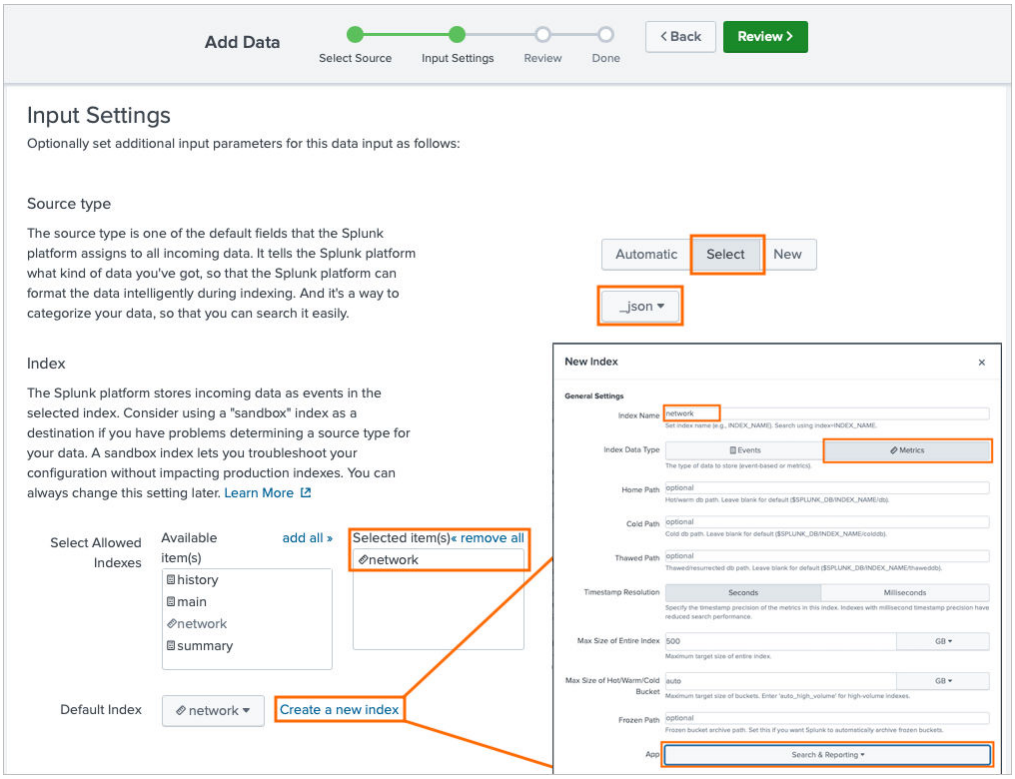


2. Go to **HTTP Event Collector** > **Add New**.
3. Provide a Name and Source Name Override.

If your organization uses Splunk Output Groups, select the appropriate output group. Otherwise, ignore the Output Group selection.

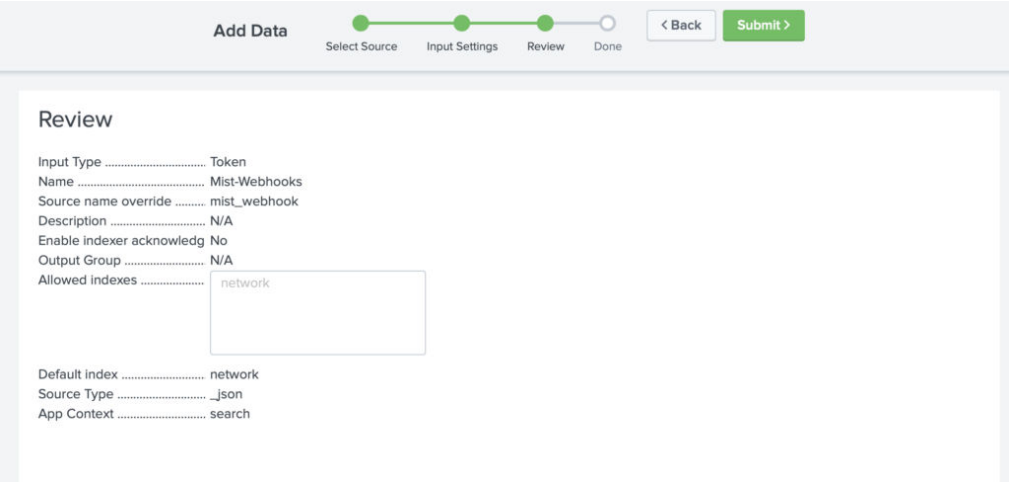


4. Click **Next**.
5. Configure the following fields:
   a. For the Source type, click the **Select** button, hover over **Structured**, and select **_json** from the list.
   b. Create a new Index and name it **network**. For the **Index Data Type**, choose **Metrics**, and for the **App**, choose **Search & Reporting**.

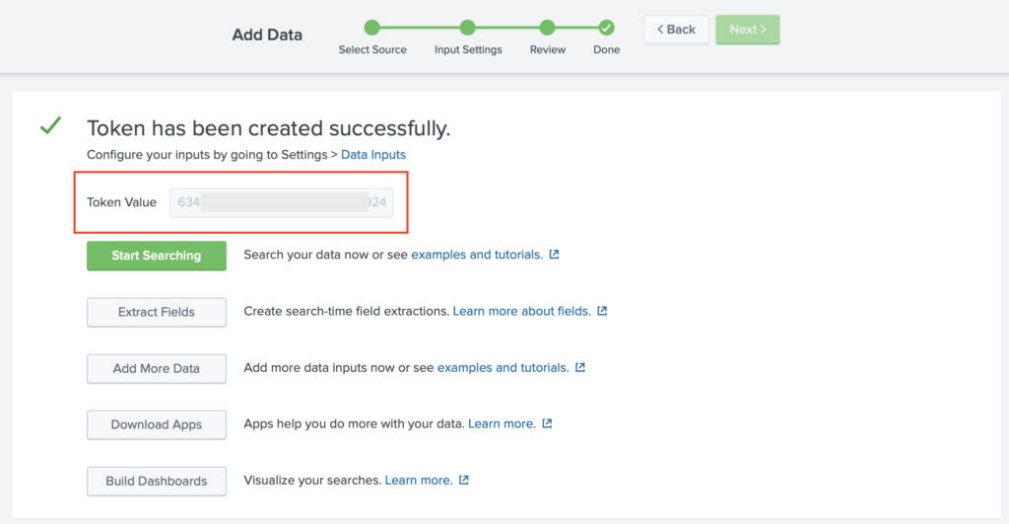


Once you Save the new index, click it from the Available item(s) list and it will appear in the Selected item(s) list.

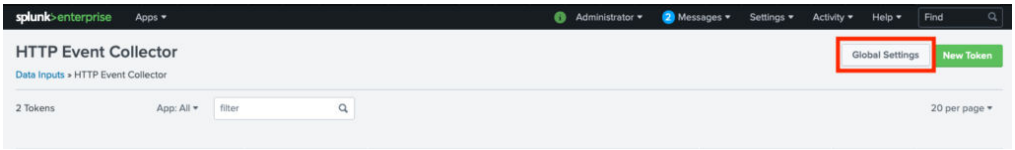
6. Click **Review**.
7. Verify the settings and click **Submit**.



8. Record the Token Value. You will need this for the Mist Webhook configuration.

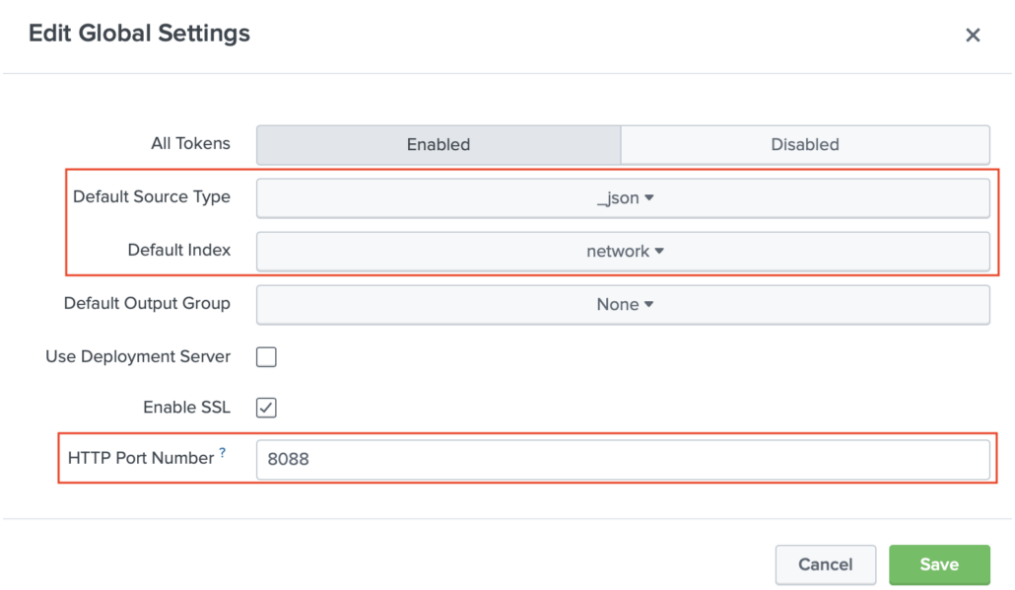


9. Navigate back to **Settings** } **Data Inputs** } **HTTP Event Collector**.
10. Click **Global Settings** in the upper right corner.



11. Specify the default Source type as _json.

    Port 8088 is the default. You can specify a default index if needed.



This completes the HEC setup in Splunk.

12. Test the Splunk HEC configuration by executing the following command in a Linux CLI. The Linux machine must be able to reach the Splunk instance over the network.

```
curl -k 'https://{FQDN OF SPLUNK}:{HEC PORT}/services/collector' \ --header "Authorization:
Splunk {HEC TOKEN}" \

--data '{"event": "hello world"}'
```

The result of the previous step should look similar to: **{“text”:”Success”,”code”:0}**. If you do not see a success message, confirm that there are no firewalls blocking the HEC port on the Splunk instance.

Configuring Mist Webhooks to Point to Your Splunk Instance

IN THIS SECTION

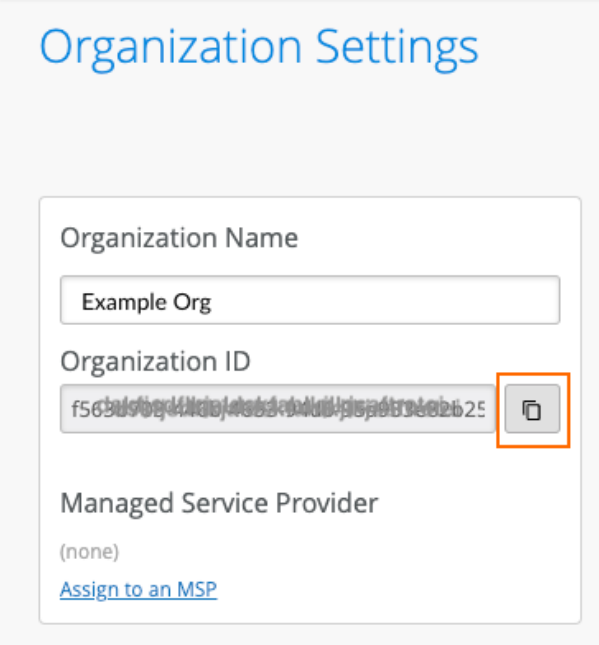
Before you Begin

You must have the following information ready so that you can complete the Mist configuration:

- The FQDN of your Splunk HEC instance
- The port number HEC is listening on (Default is 8088)
- Your Splunk HEC Token

You can configure webhooks in Mist at either the Org level, or the Site level. For this example, we configure an Org level webhook and the topics we will be subscribing to will be “audits”, “alarms”, and “device-events”.

1. Log in to the Mist portal for your organization.
2. Navigate to **Organization** } **Settings** and copy your **Organization ID**.



3. Navigate to **https://api.mist.com/api/v1/orgs/:org_id/webhooks**.

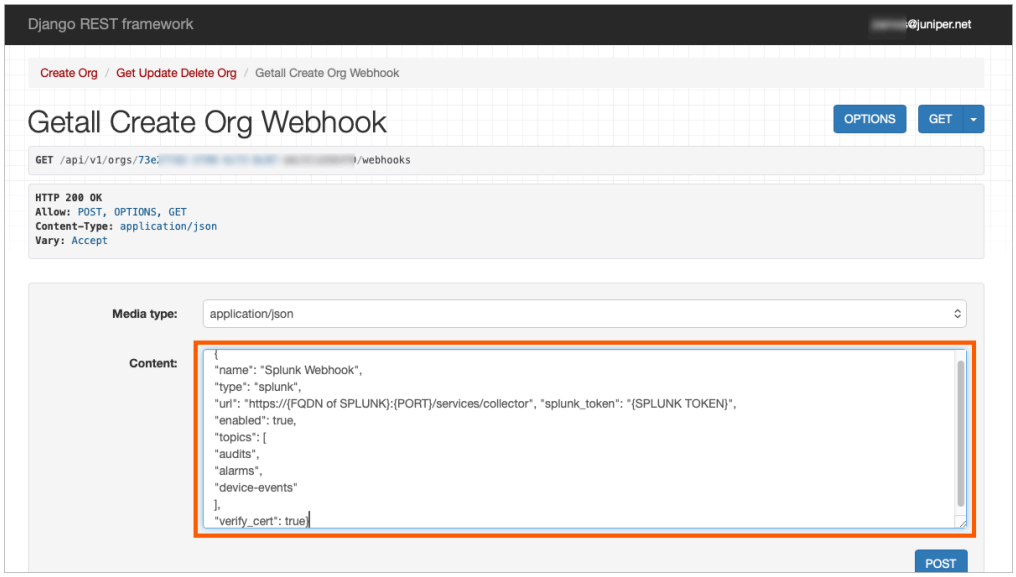
> **NOTE**: You must substitute your real Organization ID for "org_id" in the URL above.

4. In the following JSON code block, make the following substitutions:
   - Substitute the fully qualified domain name (FQDN) of your Splunk instance for **{FQDN of SPLUNK}**
   - Substitute the port on which your Splunk instance is listening for **{PORT}**
   - Substitute your actual Splunk token for **{SPLUNK TOKEN}**

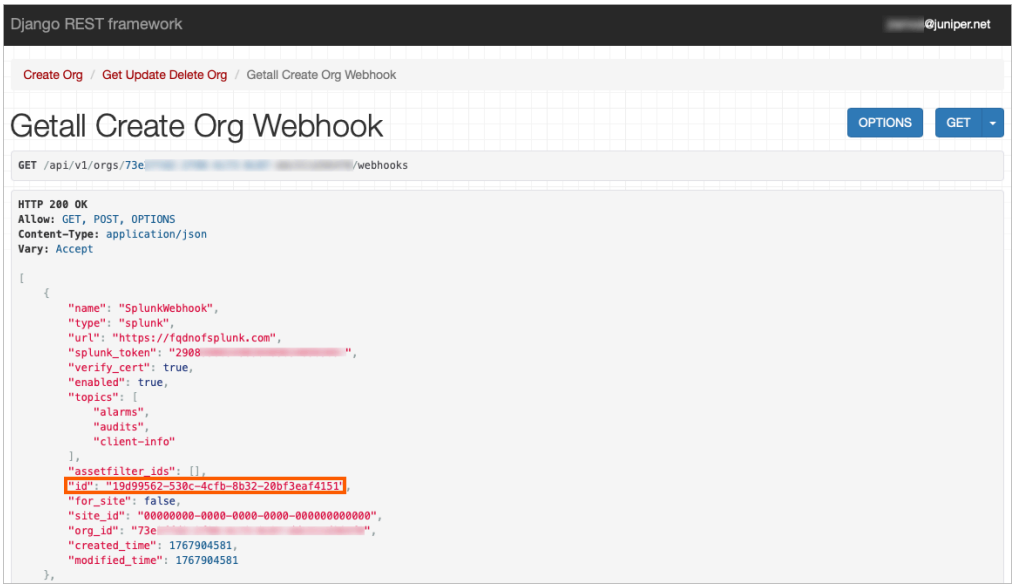
NOTE: Replace placeholder values with actual values, such as your FQDN, port, and token.

```
{
"name": "Splunk Webhook",
"type": "splunk",
"url": "https://{FQDN of SPLUNK}:{PORT}/services/collector", "splunk_token": "{SPLUNK TOKEN}",
"enabled": true,
"topics": [
"audits",
"alarms",
"device-events"
],
"verify_cert": true}
```

5. After substitution, copy the entire block of JSON and paste it into the Content box at the bottom of the page.
6. Click the POST button when you're ready.



You can now verify the configuration is in the top field of the window and that the **id** field contains data.



This confirms that Mist and Splunk can communicate.

RELATED DOCUMENTATION

- https://docs.splunk.com/Documentation/Splunk/8.1.3/Data/UsetheHTTPEventCollector
- https://www.juniper.net/documentation/us/en/software/mist/api/http/api/orgs/webhooks/overview
- https://www.juniper.net/documentation/us/en/software/mist/api/http/guides/webhooks/security#securing-webhooks-with-splunk-type

Terraform Mist Provider Integration

SUMMARY

Use the Terraform Mist Provider to deploy Day 0 and Day 1 Mist configurations with ease.

IN THIS SECTION

You can use the Terraform Mist Provider to automate the configuration and deployment of Mist-managed network resources including your Mist organizations, sites, devices, and more. It is a useful infrastructure management tool that you can use to deploy Mist with ease.

Benefits

- **Provisioning of Mist Resources**—Automatically create, configure and manage organizations, sites, devices, NAC tags, VPNs, and SSO settings.
- **Authentication Support**—API Token or Username/Password (without two-factor authentication) are supported.
- **Proxy Configuration**—Supports HTTP, HTTPS, and SOCKS5 proxies via the `MIST_PROXY` environment variables or the `proxy` provider configuration attribute.
- **Environment Variable Support**—Credentials and configuration can be passed via environment variables for secure automation.

WHAT's NEXT

1. See the Mist Provider documentation page for all information on how to get started.
2. See the Getting Started guide to learn how to configure a provider and how to deploy your Mist networks using Terraform.
3. Follow the lefthand menu of Terraform's Mist Documentation page for additional documentation on how to complete specific types of configurations.

RELATED DOCUMENTATION

https://registry.terraform.io/providers/juniper/mist/latest

CHAPTER 6

Reference

IN THIS CHAPTER

Automation Tools

SUMMARY

This topic talks about a few automation tools that you can use with Juniper Mist™. We make no reccommendation regarding any of the tools discussed below. We provide the information and examples for informational purposes only.

IN THIS SECTION

Automation Tools Overview

The entire Juniper Mist GUI is built on top of the robust Mist API. This architecture makes Mist administration highly customizable, flexible, and scalable. Using automation tools, you can perform tasks or groups of tasks not available in the GUI.

Automation begins by identifying and understanding the tasks to be performed. Most automation tools are designed for machine-to-machine transactions. Because of this design, automation tools are not always intuitive for humans to understand and configure. Thankfully, many tools exist that help you interact with machine-based interfaces and even create your own automation scripts. The Juniper Mist API is just such a tool.

Postman

Postman is a tool to help you automate Juniper Mist management tasks. Postman can do many things, but you will use it as an API client. According to the Postman web site (www.postman.com), the Postman API client is the foundational tool of Postman. This client enables you to easily explore, debug, and test your APIs while also enabling you to define complex API requests for HTTP, REST, SOAP, GraphQL, and WebSockets.

Postman provides its own API GUI client to interact with a REST API interface, and many others. Postman allows users to set up reusable environments and use variables for consistency and efficiency when testing. You can visually organize each individual request into an order, inspect and store responses for reuse, and even have Postman help generate programmatic code such as Python.

Here is a short video on using variables in Postman to interact with the Juniper Mist API.

Video: Automation for Access Switching

Postman is an excellent development tool. You can use it to test individual configuration changes and then put them together into collections. This video is an excellent example of using Postman and a web browser to put together API requests to configure 802.1x in a switching template.

With the popularity of Postman, Juniper developers have published collections in the public Postman workspaces. Collections are a method of organizing API requests and documentation into groups. You can import, export, and share collections.

Here is a short video demonstrating the use of Postman collections and the Juniper Mist API.

The Postman collections incorporate many Juniper Mist capabilities. They also include collections for gathering client information, Mist Edge devices, switching, software-defined WAN (SD-WAN), and many other objects. You can find these collections, which include the Juniper Mist cloud APIs, Mist cloud WebSocket, and Mist Runner Collections, from this link: https://www.postman.com/juniper-mist/workspace/mist-systems-s-public-workspace.

For more information on how to get started with Postman, see "Use Postman to Make Your First API Call" on page 150.

Python

Python is an object-oriented programming language. It's a popular language for many reasons. One reason is that Python is highly flexible and extensible. Compared to other programming languages, it is easy to learn and work with. Python is used for web development, data sciences, networking, and the Internet of Things (IoT), to name a few uses. It is open source and works on multiple platforms.

For more information about getting started with Python, check out the Python Software Foundation at: https://www.python.org/about/gettingstarted/

Juniper Networks supports Python use in many forms, such as direct script execution and toolkit scripts: https://www.juniper.net/documentation/us/en/software/junos/automation-scripting/topics/concept/junos-script-automation-python-scripts-overview.html

NOTE: If you plan to share Python scripts with others, be sure to use an environment file (.mist_env) in place of the API token so that you do not share your token while sharing your script. For more information, see https://github.com/tmunzer/mist_library?tab=readme-ov-file#environment-file.

Python Script Examples

Leveraging the robust API library in Juniper Mist, you can automate tasks using Python. In this section, you will see some simple Python scripts that make a series of calls to the Mist API to gather information or make changes to Mist.

These scripts are meant to serve only as examples and may or may not work in your environment due to possible variances in your environment.

Display WLANs on a Specific Site

This is a simple example script that connects to the Mist API and displays all the wireless LANs (WLANs) for a specific site:

```
import requests
import json
url = 'https://api.mist.com/api/v1/sites/00000000-0000-0000-0000-000000000000/wlans'

##Set the API request headers
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Token
a7c61a9ca25b4c27a14ea7c61a9ca7c61a9ca25b4c27a14e653b5af5db02a25ba7c61a9ca25b4c27a14e653b5af
5db024c27a14e653b5af5db026xxxxxxxxxxx'
}

results = requests.get{url, headers=headers}
wlans = json.loads{results.text}

##Print each wlan and wlan_id
for wlan in wlans:
    print{wlan["ssid"], ">", wlan ["id"]}
```

```
user:$ python3 simple-script.py
mist-prod > a7c61a9c-a25a-274c-a7c61a9c-a25b
mist-test > a7c61a9c-a25b-4c27-a7c61a9c-a25b
```

Comparing RSSI (Windows)

You execute this Python script on a Windows laptop. The script gathers the current signal quality percentage, not received signal strength indicator (RSSI), from a Windows laptop. The script then gets the RSSI from the access point (AP) to which the laptop is connected by parsing for the wireless adapter's MAC. The script makes these requests simultaneously and displays the client's signal quality on the same line as the AP's RSSI. The script runs until you send a `Ctrl+c` command from your keyboard to break out of the script execution.

The script makes use of the `netsh wlan show interface` Windows command. This is the command's output:

```
C:\Users\user>netsh wlan show interface

There is 1 interface on the system:

    Name                   : Wi-Fi
    Description            : Intel(R) Wi-Fi 6E AX210 160MHz
    GUID                   : aa7a439e-fe66-494c-xxxx-xxxxxxxxxxxx
    Physical address       : 70:cd:xx:xx:xx:xx
    State                  : connected
    SSID                   : wi-fi
    BSSID                  : d4:20:xx:xx:xx:xx
    Network type           : Infrastructure
    Radio type             : 802.11ax
    Authentication         : WPA2-Personal
    Cipher                 : CCMP
    Connection mode        : Auto Connect
    Channel                : 149
    Receive rate (Mbps)    : 516
    Transmit rate (Mbps)   : 574
    Signal                 : 83%
    Profile                : hoth

    Hosted network status  : Not available
```

This is the script itself:

```
import requests
import json
import os
import time
import random
import string
import subprocess

##Set the API request headers
headers = {
        'Content-Type': 'application/json',
        'Authorization': 'Token <YOUR_API_TOKEN>'
```

```
}

##Function to get (strip) the "Signal" quality percentage from Windows from the "netsh
wlan show interface" command
def get_clientsignal{}:

    results = subprocess.check_output(["netsh", "wlan", "show", "interface"]).decode()
    lines = results.split('\r\n')

    d = {}
    for line in lines:
        if ':' in line:
                vals = line.split(':')
                if vals[0].strip() != ''and vals[1].strip() != '':
                     d[vals[0].strip()] = vals[1].strip()

    for key in d.keys():
        if key == "Signal":

             return (d[key])

##Function to get (strip) the "Physical address" MAC from Windows from the "netsh wlan
show interface" command
def get_windowsmac():

     results = subprocess.check_output(["netsh", "wlan", "show", "interface",]).decode()
     lines = results.split('\r\m')

     d = {}
     for line in lines:
          inf ': ' in line:
                vals = line.split(':')
                print(vals)
                if vals[0].strip() != '' and vals[1].strip() != '':
                    d[vals[0].strip()] = vals[1].strip()

    for key in d.keys():
        if key == "Physical address":
            client_mac = (d[key])

            return (client_mac)
```

```
##Main Section
###Define variables
client_mac = get_windowsmac()
client_mac = client_mac.replace(":", "")
###input your site_id in the API URL
url = ('https://api.mist.com/api/v1/sites/<YOUR_SITE_ID>/{}'.format(client_mac))

client_rssi = 0
my_rssi = 0

###Loop through every 3 seconds and print the results to the terminal
while True:
        results = requests.get(url, headers=headers)
        client = json.loads(results.text)
        client_rssi = get_clientsignal()
        my_rssi = client['rssi']
        print('Windows Signal', '=', client_rssi, '   ', my_rssi, '=', 'AP rssi')
        time.sleep(3)
```

When you run the python script, the resulting output should look like this:

```
C:\Users\user>python client-signal.py
Windows Signal = 83%    -59 = AP rssi
Windows Signal = 83%    -59 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
```

```
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -62 = AP rssi
Windows Signal = 83%    -58 = AP rssi
Traceback (most recent call last):
  File “client-signal.py, line 71, in <module> time.sleep(3)
KeyboardInterrupt
C:\Users\user>
```

The Juniper Mist AP reports its statistics approximately every 60 seconds. This is the reason the AP RSSI only changed twice during the time the script was running.

Comparing RSSI (MacOS)

This Python script is executed on a MacBook laptop. It is nearly identical to the previous Windows script. The script simultaneously gathers the RSSI from the MacBook and gets the RSSI from the AP to which the laptop is connected by parsing for the wireless adapter's MAC address. It makes these requests simultaneously and displays the client's signal quality on the same line as the AP's RSSI. The script runs until you send a **Ctrl+c** command to break out of the script execution. To access the command on the Mac OS, you must include the absolute path to the command: **/SystemLibrary/PrivateFrameworks/Apple802111.framework/Versions/A/Resources/airport -I**.

This is what the **airport -I** command displays:

```
Mac:Desktop user$ /SystemLibrary/PrivateFrameworks/Apple802111.framework/Versions/A/
Resources/airport -I
     agrCtlRSSI: -60
     agrExtRSSI: 0
    agrCtlNoise: -93
    agrExtNoise: 0
          state: running
        op mode: station
     lastTxRate: 486
        maxRate: 450
lastAssocStatus: 0
    802.11 auth: open
      link auth: wpa2-psk
          BSSID: 5c:5b:xx:xx:xx:xx
           SSID: SSID-1
            MCS: 8
        channel: 120, -1
```

This is the script itself:

```
import requests
import json
import os
import time
import random
import string


##Set the API request headers
headers = {
        'Content-Type': 'application/json',
        'Authorization': 'Token <YOUR_API_TOKEN>'


##Function to get (strip) the "RSSI" quality percentage from MacOS from the "airport -I"
command
def get_clientsignal{}:

    input = os.popen(           '/SystemLibrary/PrivateFrameworks/Apple802111.framework/
Versions/A/Resources/airport -I)
    return int(**.join([x.split()[1] for x in input if 'agrCtlRSSI' in x]))



##Function to get (strip) the MAC from MacOS from the "ifconfig" command
def get_macbookmac():
     input = os.popen('ifconfig en0')
     return ''.join([xsplit()[1]for x in input if 'ether' in x])


##Main Section
###Define variables
client_mac = get_macbookmac()
client_mac = client_mac.replace(":", "")
###input your site_id in the API URL
url = ('https://api.mist.com/api/v1/sites/<YOUR_SITE_ID>/{}'.format(client_mac))

client_rssi = 0
my_rssi = 0
```

```
###Loop through every 3 seconds and print the results to the terminal
while True:
        results = requests.get(url, headers=headers)
        client = json.loads(results.text)
        client_rssi = get_clientsignal()
        my_rssi = client['rssi']
        print('Macbook rssi', '=', client_rssi, '   ', my_rssi, '=', 'AP rssi')
        time.sleep(3)
```

NOTE: Replace placeholder values with actual values, such as your API token.

When the script runs, it gathers the RSSI information from the MacBook and the connected AP (through a Mist API call) and prints its output on a single line. The script runs until you cancel it by issuing the `Ctrl+c` command.

```
Mac:Desktop user$ python3 client-signal.py
Macbook rssi = -61        -57 = AP rssi
Macbook rssi = -61        -57 = AP rssi
Macbook rssi = -61        -57 = AP rssi
Macbook rssi = -61        -57 = AP rssi
Macbook rssi = -61        -57 = AP rssi
Macbook rssi = -61        -59 = AP rssi
Macbook rssi = -61        -59 = AP rssi
Macbook rssi = -63        -59 = AP rssi
Macbook rssi = -64        -59 = AP rssi
Macbook rssi = -65        -59 = AP rssi
Macbook rssi = -63        -59 = AP rssi
```

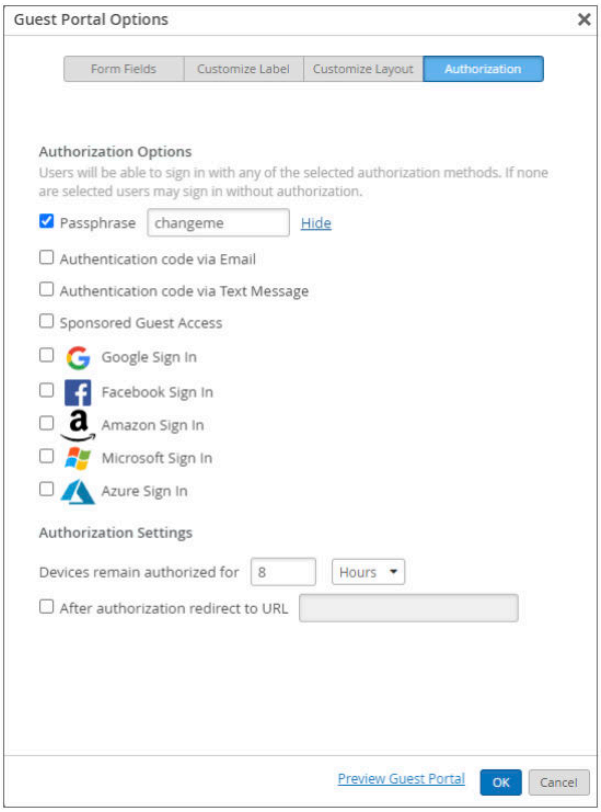
The Juniper Mist AP reports its statistics approximately every 60 seconds. This is why the AP RSSI changes only once in the example output.

Random Password Creation for a Guest Portal

When you configure a WLAN for a guest portal, use the `Guest Portal`configuration to define the landing page. This is the page for guests to log on to their wireless network. This configuration page enables you to gather information about guests such as their full names, e-mail addresses, and company details. You can add custom fields if you need users to gather additional types of information.

You can configure the guest portal to use an alphanumeric passphrase for authorization. This is a manual process, but you can automate it to run regularly, such as daily, to create a new random password that you distribute to guests each day.

This image shows the current configured passphrase for the guest portal:



Below is a Python script to create a new, random password for a guest portal. This script will send a `PUT` command to the API to change the passphrase.

> **NOTE**: Replace placeholder values with actual values, such as your site ID and your API token.

```
import requests
import json
import random
import string

url = 'https://api.mist.com/api/v1/sites/<YOUR_SITE_ID>/wlans'

##Set the API request headers
headers = {
```

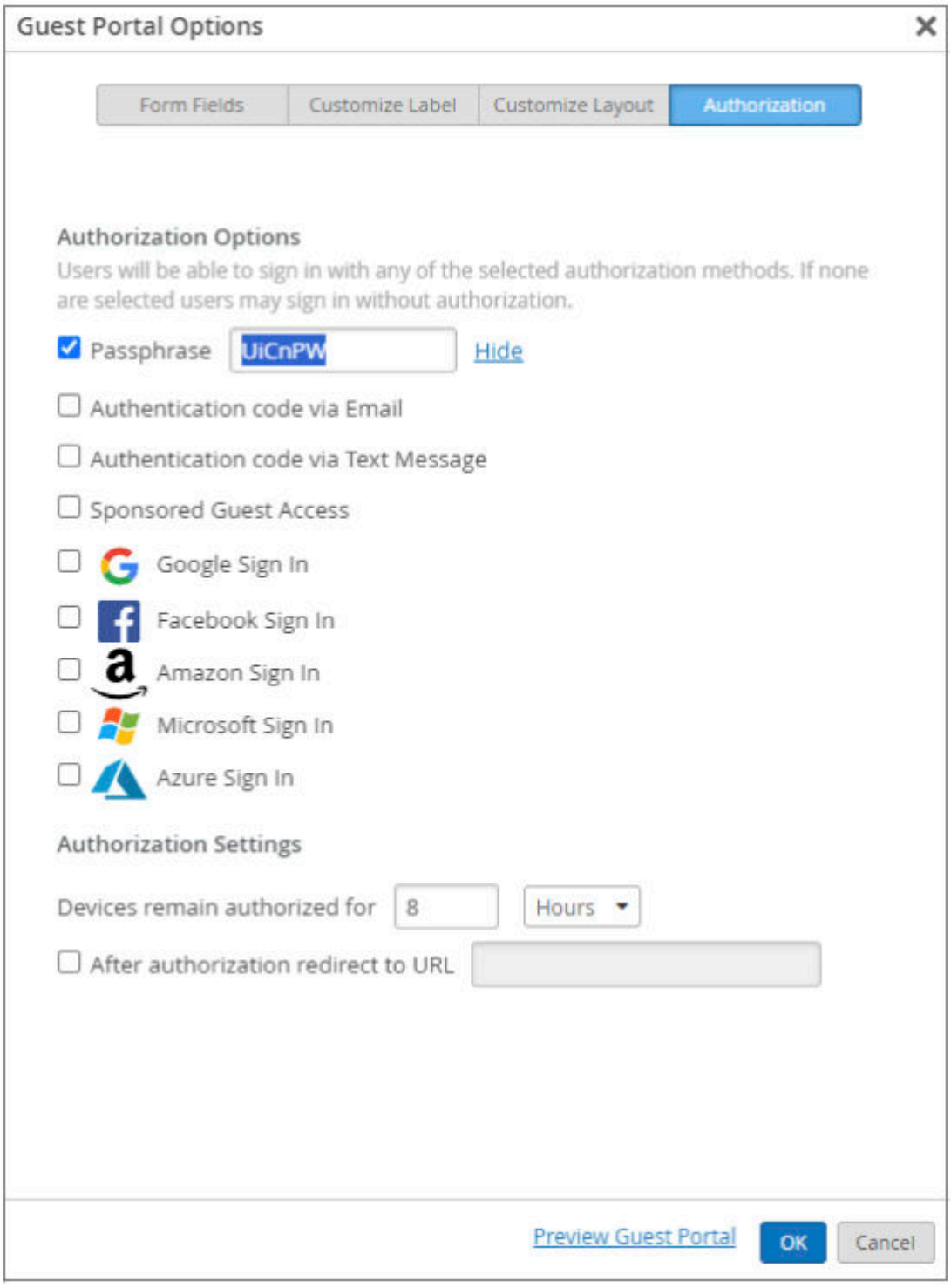
```
     'Content-Type': 'application/json',
     'Authorization': 'Token <YOUR_API_TOKEN>'
}

results = requests.get(url, headers=headers)
wlans = json.loads(results.text)
wlanid = None
portal = None
##Search for the guest SSID, mist_guest
for wlan in wlans:
    print(wlan["ssid"], ">", wlan["id"])
    if wlan ["ssid"] == "mist_guest":
        wlanid = wlan["id"]
        portal = wlan["portal"]
    if wlanid:
##This function generates a 6 alphanumeric random string
        thisisnew = ''.join(random.choice(string.ascii_letters) for _ in range(6))
        print("New Password is:", thisisnew)
        portal["password"] = thisisnew
##This takes the results and issues a PUT for the new password in the guest portal
configuration
        results = requests.put(url+"/{}".format(wlanid), headers=headers, data='{"portal":
' + json.dumps(portal) +'}')
```

This is the output from the script. It lists the **wlan** name and the **wlan_id** until it finds the **mist_guest** WLAN (defined in the script) and resets the passphrase with the newly generated, random passphrase:

```
Mac:Desktop user$ python3 client_guest_password.py
Mist-1 > 38fn03d0f9d-nfd3-xxxx-xxxx-xxxxxxxxxxxx
Mist-2 > p3j9n30lmsl-nfd3-xxxx-xxxx-xxxxxxxxxxxx
Mist-3 > 3knlso2k4ls-df9n-xxxx-xxxx-xxxxxxxxxxxx
mist_guest > 9djn3axc93nn-89sb-xxxx-xxxx-xxxxxxxxxxxx
New Password is: UiCnPW
Mac:Desktop user$
```

You can verify the passphrase change by going back to the WLAN configuration, or WLAN template page, and checking the guest portal configuration. When you click **Reveal**, you will see the new passphrase.


Guest Portal Options
Form Fields
Customize Label
Customize Layout
Authorization
Authorization Options
Users will be able to sign in with any of the selected authorization methods. If none are selected users may sign in without authorization.
Passphrase
UiCnPW
Hide
Authentication code via Email
Authentication code via Text Message
Sponsored Guest Access
Google Sign In
Facebook Sign In
Amazon Sign In
Microsoft Sign In
Azure Sign In
Authorization Settings
Devices remain authorized for
8
Hours
After authorization redirect to URL
Preview Guest Portal
OK
Cancel

Additional Automation Resources

For detailed information and examples, use these resources:

Table 21: Juniper Mist Automation Resources

Resource	Description	Where to Find It
Python scripts demonstrating possibilities of Mist APIs. https://github.com/tmunzer/mist_library	Examples of Python scripts using the Mist APIs. These scripts are using the `mistapi` Python package to simplify the authentication process.	To view script examples, go to: https://github.com/tmunzer/mist_library and browse the `scripts` directory.
MISTAPI - Python Package to use Mist APIs	This package is built from the Mist OpenAPI specifications and is designed to simplify the use of the Mist APIs with Python scripts.	To learn more about the package, go to: https://pypi.org/project/mistapi/
Mist API Labs	Proof-of-Concept and prototype applications and scripts created to demonstrate the possibilities of Juniper-Mist APIs and some of the use cases.	Go to: Mist-Lab Demo Apps and in the **Search** box, enter **webhooks.**
Mist Postman Webhook Samples	Provides preconfigured examples of POST webhook syntax.	Browse: https://www.postman.com/juniper-mist/workspace/mist-systems-s-public-workspace

Use the Django Web Interface to Make API Changes

SUMMARY

Get familiar with the Django web interface.

IN THIS SECTION

The Juniper Mist API is built on a Django Representational State Transfer (REST) framework. This architecture allows for a browsable API, meaning that you can interact with APIs directly from a web browser. This API allows for increased usability and flexibility by enabling you and other users to perform CRUD operations within the API. In a sense, the Django interface acts like a RESTful client. This function is handy for executing CRUD operations on a single API object.

To make a change to a configuration object, you must be logged in to the Juniper Mist portal and know the URL API path of the object. Consult the API documentation for details and parameters for changing objects. You will find the URL API paths for all objects available.

This task walks you through how to get device information from an access point (AP) and rename the AP directly from the Django interface.

To use the Django web interface to make API changes:

1. Log in to the Juniper Mist portal.
2. Open the API URL for a specific device: **https://<api-endpoint>/api/v1/sites/<site_id>/devices?name=<device name>**. The device name is case-sensitive.

   NOTE: When reusing code blocks, replace placeholder values with actual values, such as your API token, organization ID, site ID, AP name, and so on.

   In this case, the URL will look like this (portions of the site_id are omitted):

   https://{api-host}/api/v1/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxxx/devices?name=TEST-rename

   The browser issues the following command through the Django interface:

   ```
   GET /api/v1/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxxx/devices?name=TEST-rename
   ```

   Juniper Mist assigns every device a unique identifier, which is typically based on the MAC address (00000000-0000-0000-1000-<device_mac>). In the context of the device API, it is called `id`. You need to reference the AP using `id` so the API knows which specific device to rename.
3. To make the change, insert the device ID (`id`) into the API call and display it in the browser.

   The new call will look like this:

   https://api.mist.com/api/v1/sites/c1947558-268d-4d31-xxxx-xxxxxxxxxxxx/devices/00000000-0000-0000-1000-5c5b3xxxxxxx

   The output is the same as the previous request; however, the API context now enables you to make a change to the specific device based on the `id` and not the name. Notice the lack of enclosing “[ ]” brackets.

4. Enter the JavaScript Object Notation (JSON)-formatted text in the **Content** input box for the intended AP rename call. You do not need to include the unique device ID (`id`) because the `id` context exists in the URL.

```
{
    "name": "RENAMED"
}
```

5. Once complete, press `PUT` to push the request to Juniper Mist. The results should look like this, indicating that the AP has been renamed:

```
{
    "id": "00000000-0000-0000-1000-5c5b3xxxxxx",
    "name": "RENAMED",
    "site_id": "c1947558-268d-4d31-xxxx-xxxxxxxxxxxx",
    "org_id": "3f12cb79-fb5e-4d4b-xxxx-xxxxxxxxxxxx",
    "created_time": 1685989351,
    "modified_time": 1686321430,
    "map_id": null,
    "mac": "5c5b3xxxxxxxx",
    "serial": "A07451xxxxxxx",
    "model": "AP43",
    "hw_rev": "C02",
    "type": "ap",
    "tag_uuid": "3f12cb79-fb5e-4d4b-xxxx-xxxxxxxxxxxxxx",
    "tag_id": 3056xxx,
    "evpntopo_id": null,
    "deviceprofile_id": null
}
```

For more information, see the Site section of the API Documentation site.

RESTful API Pagination Example

Some organizations have inventories with thousands of APs. This can be challenging when you want to view them all with an API GET request, because by default, the response will be returned without pagination on a single page. You can enable pagination to the returned data by adding HTTP header parameters for `X-Page-Limit`, `X-Page-Page` and `X-Page-Total` to the API GET request. These help you to know if you are getting all of the available entries or if you need to query the next pages.

The `X-Page-Limit` defines the maximum number of results per page. The limit is 100 by default, and the maximum value is 1000. However, there are some exceptions to this.

The `X-Page-Page` defines the maximum number of page results.

The `X-Page-Total` defines the total number of entries in the list. The total will depend on what you are looking at and how many entries are in the list.

To adjust any of this information, the query parameters can be added to the end of the web URL for the API interface you are using. Depending on the API call, you may need to instead add the parameters to the end of the "next" URL that is generated and sent in the response body (should a "next" exist).

NOTE: When reusing code blocks, replace placeholder values with actual values, such as your API token, organization ID, site ID, AP name, and so on.

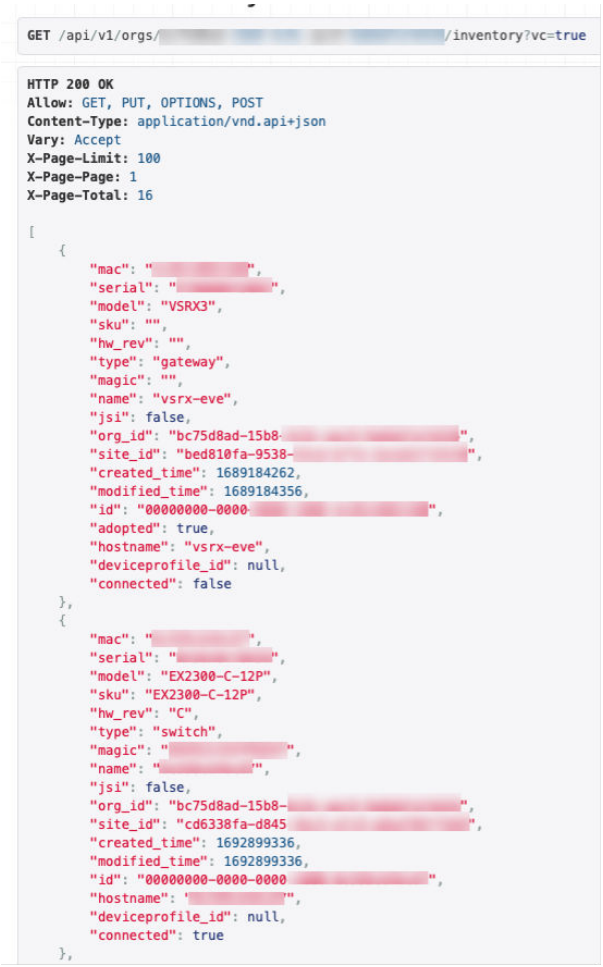
In the following example, adding **limit=2&page=47** would adjust the X-Page-Limit and X-Page-Page parameters.

```
https://api.mist.com/api/v1/orgs/:org_id/inventory?limit=2&page=47
```

The example below shows the format of a GET request that includes pagination:

```
Request URL: https://api.mist.com/api/v1/orgs/:org_id/inventory
Request Method: GET
Status Code: 200 OK
Date: Thu, 16 May 2019 04:22:05 GMT
Request Headers:
 X-Page-Limit: 100
 X-Page-Page: 1
 X-Page-Total: 193
```

The response you receive from Mist when making the request in the Django web interface, will look like this:



Refer to the Mist API Reference for more information on usage: Pagination

RELATED DOCUMENTATION

Additional RESTful API Documentation | 154

API Endpoints and Global Regions | 13

Use the Mist API Reference for API Testing

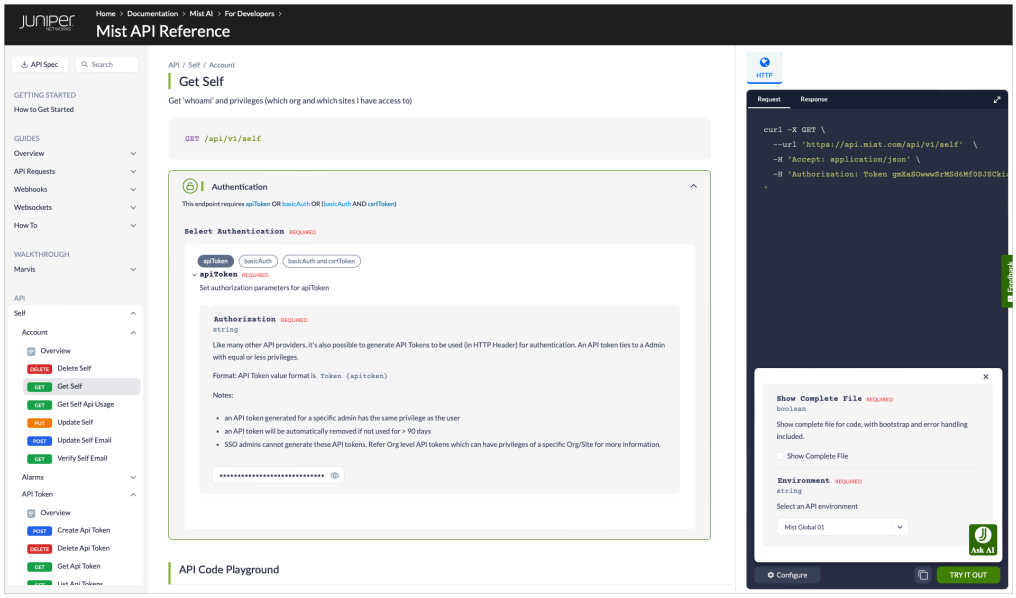
SUMMARY

You can use the Juniper Mist™ API Reference to make API calls and see the responses that are returned from those calls. This is useful for when you're working on a script and you want to better understand what is returned to you from the API.

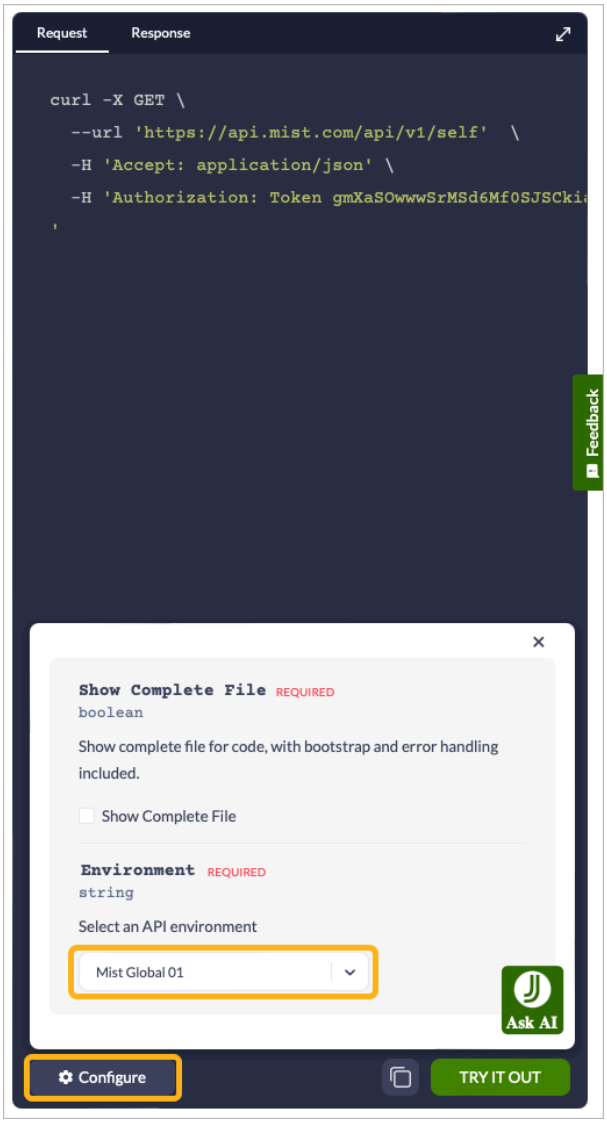
To make API calls on the Mist API Reference website, you must first do the following:

1. Use the lefthand menu to find the API call you want to make, or use the Search field at the top to search for it. For example, you can navigate to **API** > **Self** > **Get Self** to get "whoami" and privileges, such as which organization and sites you have access to.
2. Scroll down to the **Authentication** section and expand it, then enter your API token in the required field, or provide another form of authentication. Otherwise, a response will not be returned. See "Create API Tokens" on page 15.

> **NOTE**: In the token field, you must enter the word Token followed by a space, then paste your token after the space. Follow this format: `Token <token>`.



3. Navigate to the **Request/Response** panel on the right side of the screen and select the **Configure** button at the bottom left corner. Select the region of your API environment (API Endpoint). See "API Endpoints and Global Regions" on page 13.



4. Select **Try it Out** from the bottom right corner of the Request panel. This returns you with a Response. Review the results to help you better understand the response that will be returned from the API.

NOTE: If you do not want to authenticate, you can scroll down to the Responses section in the center of the page to see sample responses. This section simply provides you with a sample response for the specified API call and does not require authentication from you.

You are now set up to test API calls. Use the lefthand menu to find the API call you want to make, or use the Search field at the top to search for it, then select **Try It Out** from the bottom of the Request panel to be returned with a response that you can review.

Use the Mist Browser Extension for Easy API Access

SUMMARY

Juniper Mist™ now has a web browser extension that simplifies access to Mist APIs and API tokens. It is currently supported on Mozilla Firefox and Google Chrome web browsers and it is recommended that you download the extensions directly from those web stores.

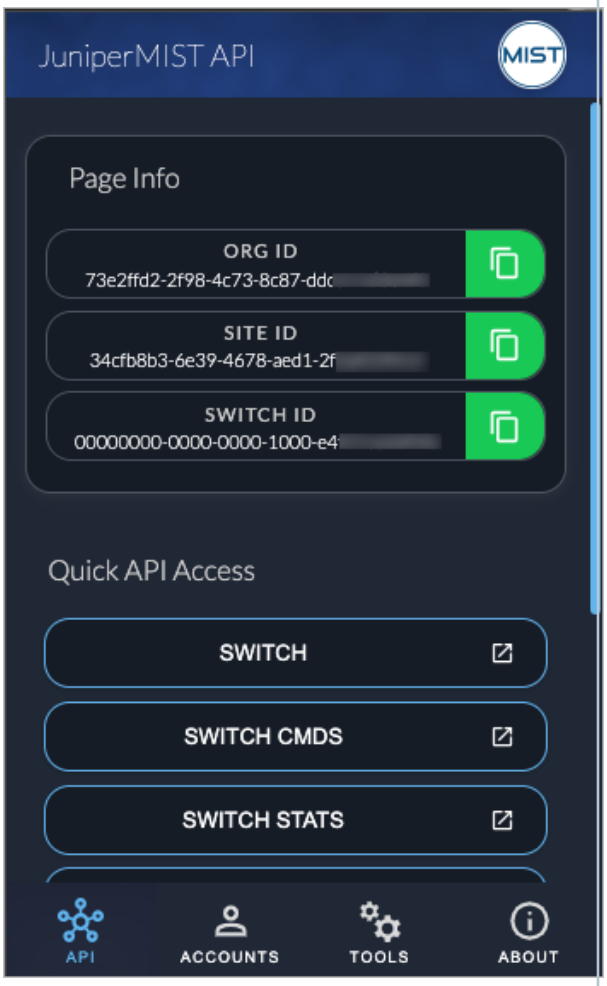
The Mist API web browser extension is a handy tool you can use to quickly access API information from any page on the Juniper Mist™ portal or Django web interface. All you have to do is click on the extension when you are on the desired Django or Mist portal page, and the extension will display the API information related to that page. Shortcuts that provide you with quick API access are displayed as well.

The following table lists the various types of information that are available to you within the tabs of the extension. There is a slight difference in what is displayed when you use the extension from the Django web interface versus the Mist portal. These differences are noted just below the table.

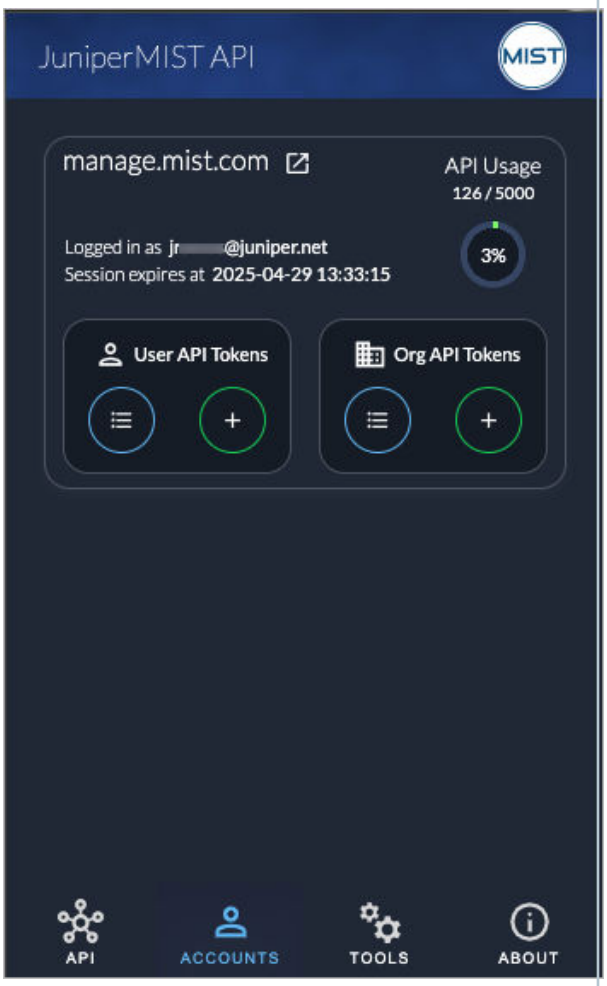
NOTE: The information and options that display in the extension are dependent on the specific Mist portal page or Django page that you are on at that time.

Information Provided in the Mist Browser Extension	
Tab	**Uses**

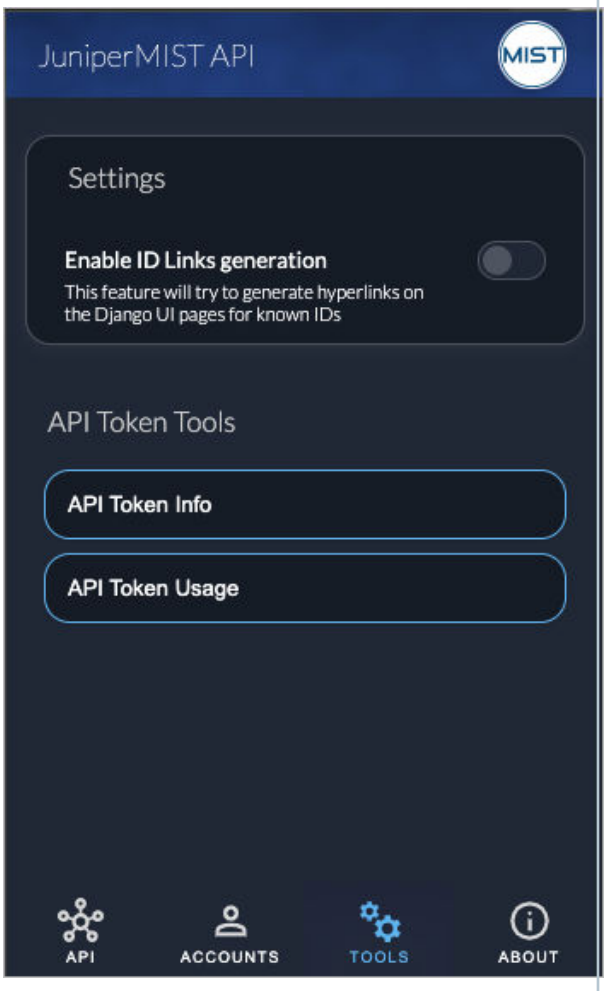
(Continued)

Information Provided in the Mist Browser Extension	
API	• Get easy access to Mist IDs such as the org ID, site ID, and object ID. • Get Quick API Access related to the current Mist portal page you are on. For example, from the Switch Details page, open the extension and select the buttons provided in the Quick API Access section to bring you to the Django web interface for the corresponding API call. 

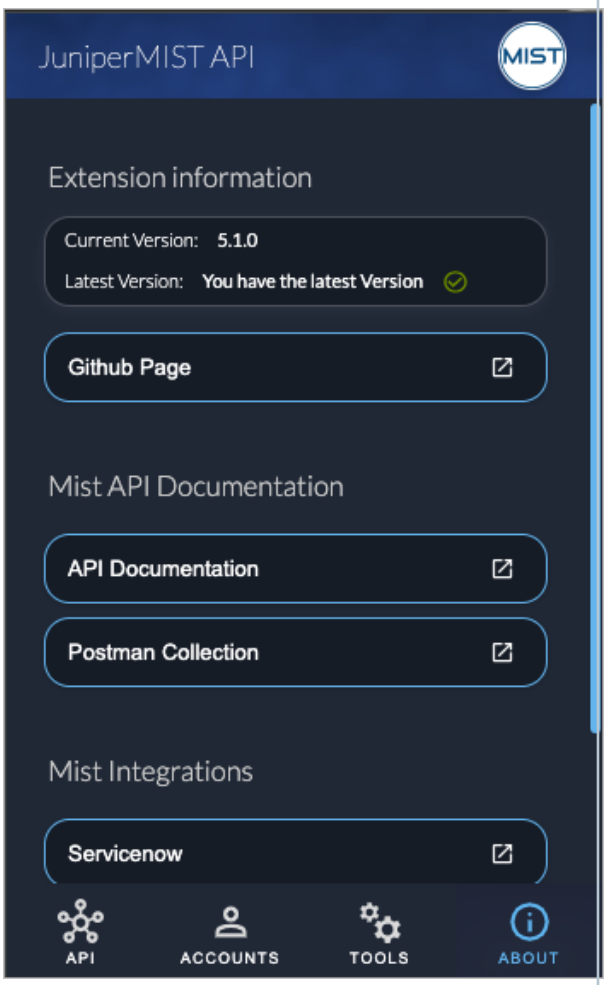
(Continued)

Information Provided in the Mist Browser Extension	
Accounts	• Get information about all current valid Mist sessions in the web browser for various Mist clouds. • Manage User and Org API tokens. • View current API Usage (the API request usage percentage, and the API request limit). 

(Continued)

Information Provided in the Mist Browser Extension	
Tools	• Under Settings, you can enable ID Links generation to generate hyperlinks on the Django UI pages for known IDs. • Under API Token Tools, you can view API Token information and API Token Usage. 

(Continued)

Information Provided in the Mist Browser Extension	
About	• Under Extension Information, you can see information about the current version your Mist Browser Extension is on and whether or not you are on the latest version. • Under Mist API Documentation, you can get quick access to Mist API documentation (official Mist Documentation and the Postman Collection). • Under Mist Integrations, get fast access to integrations such as ServiceNow, Terraform, and Splunk. 

The same information is available to you when you use the extension while on the Django web interface. However, the only difference is that in the API tab of the extension you will have:

- All available query parameters listed.
- Easy customization of query parameters available.
- Direct access to the online API documentation for the current API call (GET/POST/PUT/DELETE).

For more details, see https://github.com/Mist-Automation-Programmability/mist_browser_extension?tab=readme-ov-file.