StorageGRID® Webscale NAS Bridge 2.1

Management API Guide

August 2019 | 215-12808_2019-08_en-us
doccomments@netapp.com

NetApp®

Contents

Understanding the NAS Bridge management API

The NetApp StorageGRID Webscale NAS Bridge management API provides access to the functionality you need when administering and controlling a NAS Bridge node. The API is based on RESTful web services. Before using the management API, you should understand its design, architectural components, and limitations.

RESTful web services foundation

The NAS Bridge management API is implemented based on RESTful web services technology. Representational State Transfer (REST) establishes a collection of technologies and design principles for exposing server-based resources and managing their states. It uses mainstream standards, which provide a flexible and extensible foundation for managing the NAS Bridge node.

Resources and state representation

The core aspects of the design of RESTful web services include the following:

- Identification of system or server-based resources
  Every system uses and maintains resources. A resource can be a file, business information, a process, or an administrative entity. One of the first tasks in designing an application based on RESTful web services is to identify the resources.
- Definition of resource states and associated state operations
  Resources are always in one of a finite number of states. The states must be clearly defined, as must be the operations used to affect the state changes.

Messages are exchanged between the client and server to access and change the state of the resources according to the generic CRUD (Create, Read, Update, and Delete) model.

HTTP messages

Hypertext Transfer Protocol (HTTP) is the specific protocol used by the web services client and server to exchange request and response messages about the resources. During the RESTful web services design, the HTTP verbs are mapped to the resources and the corresponding state management actions.

The NAS Bridge management API relies on a subset of the HTTP protocol and uses the following HTTP verbs:

- GET
- POST
- PATCH
- DELETE

HTTP is stateless. Therefore, to associate a set of related requests and responses under one identity, additional information must be added to the data flows, including HTTP headers or cookies. Also note that HTTP, and therefore a RESTful web services API, uses TCP port 80 by default.

URI endpoints

Every REST resource must be defined and made available using a well-defined addressing scheme. The endpoints where the resources are located and identified using a Uniform Resource Identifier (URI). The URI provides the general framework for creating the unique name of a resource that can

be used in the network. The resources are exposed in a structure that is similar to a hierarchical directory.

The Uniform Resource Locator (URL) is a type of URI adapted primarily for the web and used in the design of RESTful web services. A URL is used to identify a resource and to access a representation of the resource.

JSON formatting

While there are several possible ways that information can be transferred between the client and server, the most popular option (and the one used with NAS Bridge) is JavaScript Object Notation (JSON). JSON is a standard for representing simple data structures, including objects and arrays, in plain text. JSON is used by the RESTful web services to represent and transfer state information describing each resource.

Multiple access paths

Because of the inherent flexibility of RESTful web services, the management API can be accessed in several different ways:

- **NAS Bridge native user interface** – The primary way you access the API is through the NAS Bridge native web user interface. When you use a browser to access a node's management IP address, the initial page is displayed with administrative functions organized by category. The browser accesses the management API and reformats the data according to the design of the user interface.
- **Swagger web page** – The NAS Bridge management API uses the Swagger open source API platform to provide an alternative access point to a NAS Bridge node when using a browser. Swagger allows both developers and non-developers to interact with the API in a user interface that illustrates how the API responds to parameters and options.

  Attention: Any API operations you perform using the Swagger user interface are live operations. Be careful not to create, update, or delete configuration or other data by mistake.

- **Custom program** – You can access the management API using one of several different programming languages and tools. Popular choices include Python, Java, and cURL. A program, script, or tool that uses the API acts as a RESTful web services client. Using a programming language enables you to better understand the API as well as to possibly automate the management and control of a NAS Bridge node.

Uploading data

When uploading files such as exported configuration files using the API, you must perform a POST with the multipart content type. Otherwise the file will fail to be uploaded.

How API session authentication works

Authentication is a required part of issuing every API call. However, rather than providing the user credentials on each call, you must first obtain an authentication token. This token is initially generated based on a user name and password; the token must be supplied on all of the subsequent API calls.

The management API can be accessed in several ways, including using the product's native browser interface, Swagger page, or a programming language (such as Python). In each case, there is a common logic flow or pattern of usage related to authentication. The general flow that you must use when accessing the API is as follows:

1. Create an API session by providing a user name and password

2. Extract the authentication token and other information from the HTTP response
3. Optionally perform one or more additional API calls as needed to complete the desired task, supplying the authentication token as well as other information on each call
4. Delete the session which resets the authentication token

Therefore, regardless of how the management API is accessed, using the API always begins by generating an authentication token and ends with resetting the token.

Understanding NAS Bridge resources and objects

The NAS Bridge management API allows multiple instances of each REST resource type to exist concurrently. Each instance can be viewed as a type of object. Therefore, for a given resource type, you can consider the resource instances to be an array of one or more objects. This design gives you better flexibility and control when accessing the resource instances through the API.

Object identifiers

Each resource instance or object is assigned a unique identifier. These object identifiers (IDs) are integer values assigned to the new resource instances. The IDs are unique within a specific resource type, but not within the system as a whole. For example, if you create a DNS server, the first instance might be assigned ID=1. Subsequently, you might create an NTP server and the first instance might also be assigned ID=1. This is acceptable because the resource instances are differentiated according to their type.

The identifiers are generally returned in the HTTP response after a successful add request. You can extract and retain the IDs. An ID must be provided in the following situations:

- When getting the current status of a resource instance
- When linking resource instances where one object refers to another
- When deleting a resource instance

Summary of resource statuses

Each resource instance has an associated status. In general, a resource's status can be accessed and displayed through the management API.

The following status values are used for the NAS Bridge resources:

- NOTIFYING
  The underlying services are being notified of the change.
- COMMITTING
  The underlying services are coordinating the commitment of the change.
- ABORTING
  The change has been rejected. Check the error log messages for more information.
- FAILED
  The change failed and all services have completed a rollback. You should check the system event log for more information.
- READY
  The change has been successfully completed, and the resource is ready for use.

In most cases, the original request or action type is added to the beginning to create the complete state description. For example, after issuing an add request, the new resource might temporarily be in the state: ADDING / NOTIFYING. A similar construction applies when removing resources.

How asynchronous operation works

Many of the API calls, particularly those that create or remove a resource, can take a longer time to complete than most other API calls. NAS Bridge processes these types of requests asynchronously. When issuing a call that operates asynchronously, you must check the status of the resource instance to confirm that the request is complete.

With asynchronous processing, the initial successful HTTP response indicates that the request has been accepted but not necessarily finished. Therefore, after making an asynchronous request to add or remove a resource, you must test the resource instance for completion of the request.

Note: Refer to the Swagger web page for documentation to help determine whether a specific API call operates asynchronously. The implementation notes section on the page (if present) contains the details.

Checking a resource status after an add request

After issuing an API call that adds a resource instance, you should poll the status of the resource to verify completion of the request. A request is complete when the new resource reaches the READY state.

After creating the required authentication token, you should use the following high level process when asynchronously adding a resource:

1. Issue the API call to add a resource.
2. Receive an HTTP response indicating successful acceptance of the request.
3. Extract the resource ID from the HTTP response.
4. Within a timed loop, perform the following steps in each loop cycle:
   a. Get the current status of the resource based on the ID.
   b. If the resource is not in the READY state, perform the loop again.
   c. If the resource is in the FAILED state, abort the operation, fix the problem (for example, remove the failed resource), and perform the loop again.
5. When the resource reaches the READY state, you can stop.
6. If the polling loop times out (according to your arbitrary timeout value) before the resource reaches the READY state, report an error.

Checking for resource removal after a delete request

After issuing an API call that deletes a resource instance, you should poll the resource to verify that it has been removed. A request is complete when the resource no longer exists.

After creating the required authentication token, you should use the following high-level process when asynchronously removing a resource:

1. Issue the API call to delete a resource.
2. Receive an HTTP response indicating successful acceptance of the request.
3. Within a timed loop, perform the following in each cycle:
   a. Get the current status of the resource based on the ID.
   b. If resource is located (HTTP code 200), perform the loop again.
4. When the get request is not found (HTTP code 404), you can stop.

5. If the polling loop times out (according to your arbitrary timeout value) and the resource still exists, report an error.

Summary of resource types supported by the API

As part of planning to use the management API, you should be aware of the RESTful resource types that are supported. The API calls are organized under the various resource types.

You must refer to the Swagger web page for a complete list of the API calls, as well as the details of each call. Also refer to the release notes publication for information regarding updates or changes to the management API.

The management API calls are organized according to the following resource types:

- Active Directory controllers
- Alert configuration
- AutoSupport (ASUP) service
- Cache devices
- Configuration exports
- Debug
- Decommissions
- Disks
- DNS servers
- File systems
- Metrics
- Network interfaces
- Network logical interfaces (LIFs)
- Network routes
- NTP servers
- Object stores
- Passwords
- Reboot
- Sessions
- SMTP servers
- System events
- System information
- Upgrades
- Users

Accessing and using the API

You can access the NAS Bridge management API using the Swagger user interface on a web browser, or you can use a programming language or other command-line tool.

About this task

You can use either of the following methods to access the NAS Bridge management API:

- You can use the Swagger web page from a NAS Bridge node. The API calls are organized on the page according to resource type and displayed in a consistent format. This method for accessing the API is described in this document. Using this method requires that you understand how to interpret the API documentation on the Swagger page and how to issue API calls.
- You can use a programming language or other command-line tool, such as Python. By accessing the management API programmatically, you can gain a deeper understanding of the API. This document does not describe this method for accessing the API.

  Note: While there are many options when choosing a programming language or tool to access the management API, Python is a widely available scripting language. In many cases, you can easily integrate Python scripts into the automation processes typically used in most IT environments.

Accessing the Swagger API web page

You must access the Swagger web page to display the API documentation, as well as to manually issue an API call.

Before you begin

You must have the management IP address or domain name of the NAS Bridge node.

Steps

1. Construct the URL for the Swagger page by concatenating the management IP address or domain name of the NAS Bridge node and the string `/api_docs`.

   For example: `http://10.63.65.121/api_docs`

2. Type the new URL into your browser and press Enter.

Result

The main Swagger page is displayed with the calls organized by resource category.

Related references

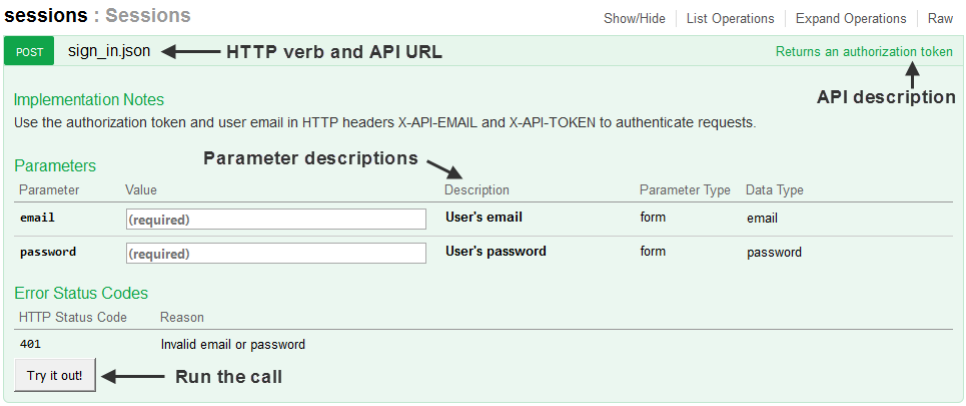
Summary of resource types supported by the API on page 8

Understanding the details of an API call

The details of all the API calls are included on the Swagger web page. All of the API calls are documented and displayed using a common format. By understanding a single API call, you can access and interpret the details of all the API calls.

Steps

1. On the main Swagger page, click **sessions**.
2. Click **POST** to display the details of the API call used to request an authentication token.



3. Examine the entire page to understand the API call and what you must enter.

   You should note the HTTP verb and URL, required input parameters, the HTTP status codes used, and any implementation notes.

Performing a simple task using the API

To better understand the API, you should perform a simple task using the Swagger web page. Creating a test event in the system event log is a good task to begin with.

Before you begin

You must be familiar with how to access the Swagger web page using a browser. In addition, you must have the credentials for an administrator account, including the user name and password.

Attention: Any API operations you perform using the Swagger user interface are live operations. Be careful not to create, update, or delete configuration or other data by mistake.

Steps

1. Obtaining an authentication token on page 11
2. Creating a system event message to test the API on page 11
3. Resetting the authentication token on page 16

Related tasks

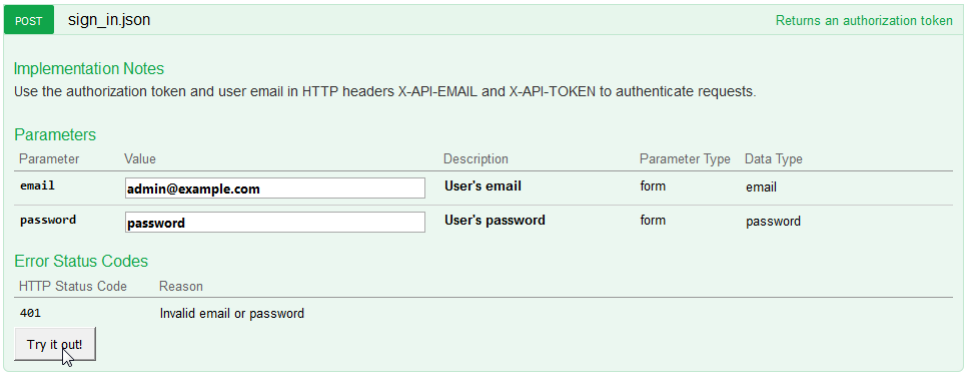
Accessing the Swagger API web page on page 9

Obtaining an authentication token

To use the management API, you must first obtain an authentication token. This token, along with other information, must be supplied on all API calls.

Steps

1. On the Swagger page, click **sessions**.
2. Click **POST** to display the API call used to return an authentication token.
3. Type the email address (that is, user name) and password for your administrator account.
4. Click **Try it out**.



If the user name and password are valid, you should receive the HTTP status code 201, indicating success.

5. Copy the authentication token (without the quotes) returned in the response body.

6. Use this token as the `X-API-Token` parameter when making future API calls.

Creating a system event message to test the API

You can use Swagger to create a message in the system event log. This is a simple way to test the management API. After the event has been created, you can display the message using the GET function on the Swagger page or using the NAS Bridge user interface.

Before you begin

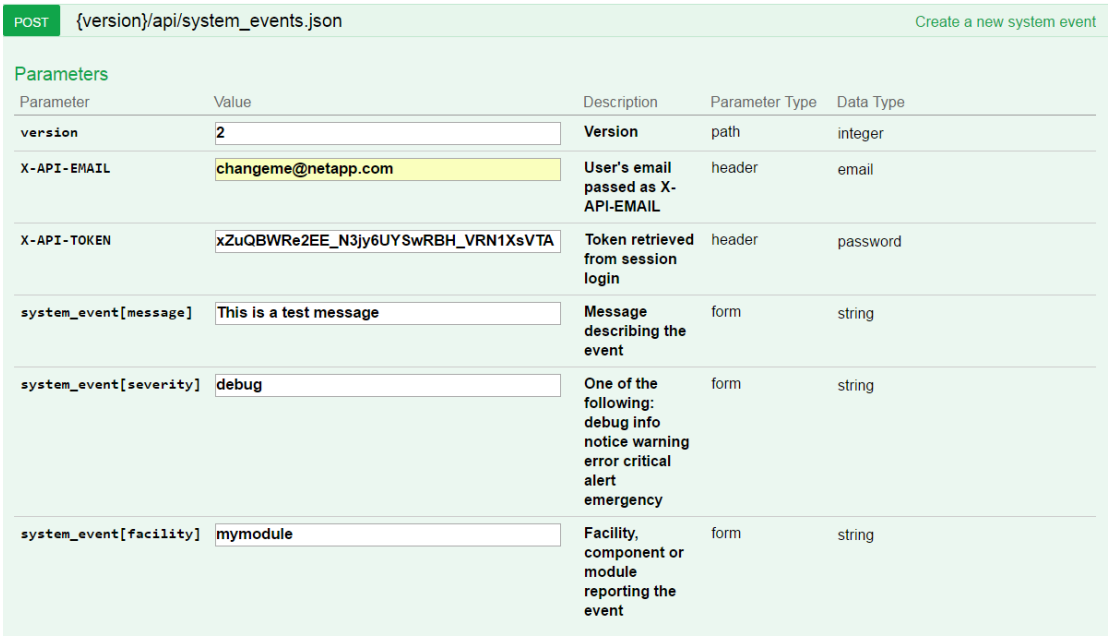
You must have the following parameters:

- A valid authentication token (the token can be copied from the result of a previous session POST call)
- The administrator email address (that is, user account) that was used to create the authentication token

Steps

1. On the Swagger page, click **system_events**.
2. Click **POST** to display the API call used to create a new system event.
3. Type the version (which must always be 2), email address (that is, user account), and the authentication token.

   If you copied the authentication token after previously creating a new session, you can paste the token into the field on the page.
4. Type a test message, severity (from the list of allowed values), and test facility:

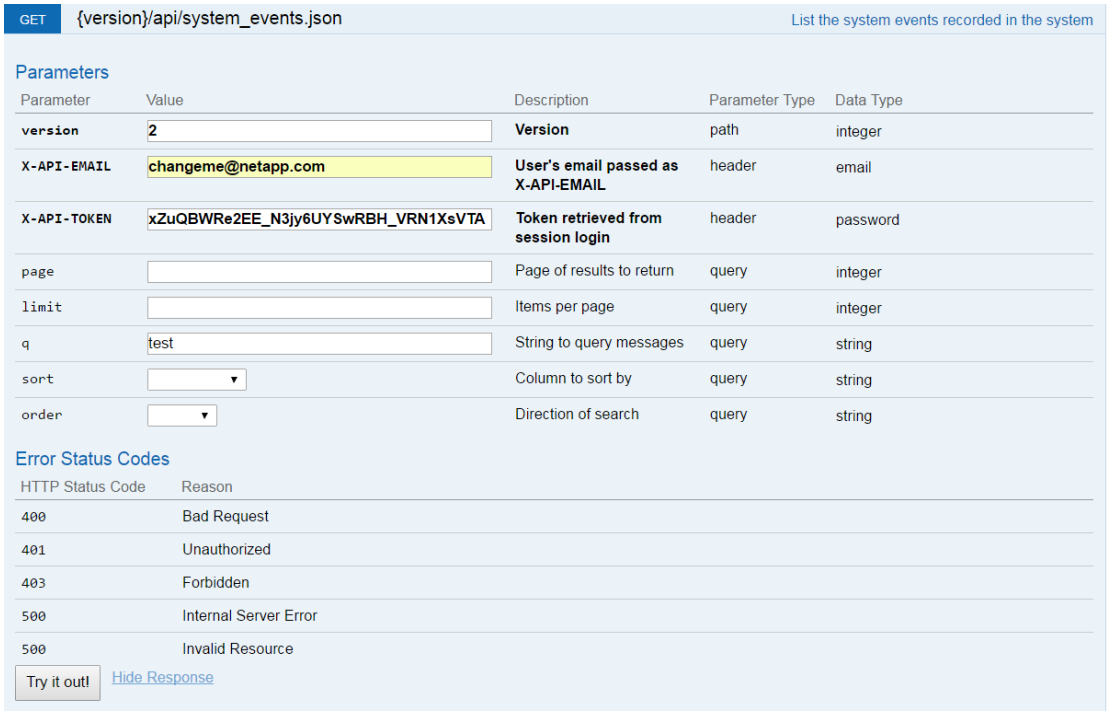


5. Click **Try it out**.

The response code 200 indicates success.

6. To verify the new message was created:

   - On the Swagger page, click **system_events > GET**, enter an email address and the authentication token. Use the q parameter to limit the results to those messages that include all or part of your message text. Then, click **Try it out** to list the message you created.



GET {version}/api/system_events.json — List the system events recorded in the system

Parameters

Parameter	Value	Description	Parameter Type	Data Type
version	2	**Version**	path	integer
X-API-EMAIL	changeme@netapp.com	**User's email passed as X-API-EMAIL**	header	email
X-API-TOKEN	xZuQBWRe2EE_N3jy6UYSwRBH_VRN1XsVTA	**Token retrieved from session login**	header	password
page		Page of results to return	query	integer
limit		Items per page	query	integer
q	test	String to query messages	query	string
sort		Column to sort by	query	string
order		Direction of search	query	string

Error Status Codes

HTTP Status Code	Reason
400	Bad Request
401	Unauthorized
403	Forbidden
500	Internal Server Error
500	Invalid Resource

Try it out! Hide Response

Request URL

```
https://10.96.104.166:443/2/api/system_events.json?q=test
```

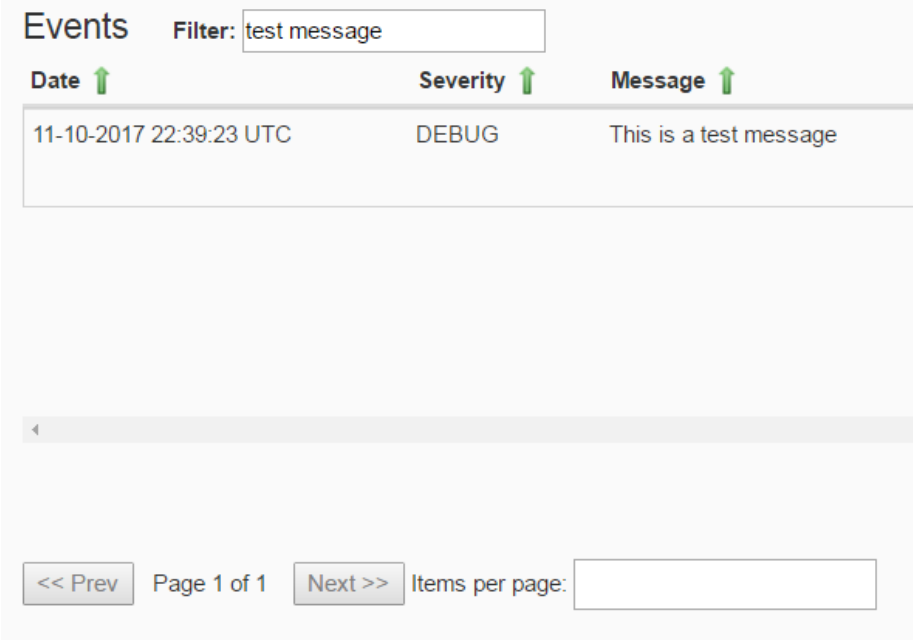
Response Body


```
    {
      "id": 40,
      "severity": "debug",
      "facility": "mymodule",
      "message": "This is a test message",
      "created_at": "2017-11-10T22:39:23.000Z",
      "updated_at": "2017-11-10T22:39:23.000Z",
      "config": null,
      "config_id": null
    }
  ],
  "count": 8,
  "pagination": {
    "current": 1,
    "previous": null,
    "next": null,
    "per_page": 25,
    "pages": 1,
    "count": 8
  }
```


Response Code

```
200
```

- From the NAS Bridge user interface, click **Maintenance** > **Events**, and filter or sort the events for the message you posted.



Resetting the authentication token

You should reset the authentication token after completing your API calls. This improves the security of the system by preventing the token from being reused.

Before you begin

You must have the following parameters:

- A valid authentication token (the token can be copied from the result of a previous session POST call)
- The administrator email address (that is, user account) that was used to create the authentication token

Steps

1. On the Swagger page, click **sessions**.
2. Click **DELETE** to display the API call used to reset an authentication token.
3. Type the email address (that is, user account) and authentication token.

   If you copied the authentication token after previously creating a new session, you can paste the token into the field on the page.
4. Click **Try it out!**

   The response code 204 indicates the token was deleted.

Copyright

Trademark

NETAPP, the NETAPP logo, and the marks listed on the NetApp Trademarks page are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.

http://www.netapp.com/us/legal/netapptmlist.aspx

How to send comments about documentation and receive update notifications

You can help us to improve the quality of our documentation by sending us your feedback. You can receive automatic notification when production-level (GA/FCS) documentation is initially released or important changes are made to existing production-level documents.

If you have suggestions for improving this document, send us your comments by email.

doccomments@netapp.com

To help us direct your comments to the correct division, include in the subject line the product name, version, and operating system.

If you want to be notified automatically when production-level documentation is released or important changes are made to existing production-level documents, follow Twitter account @NetAppDoc.

You can also contact us in the following ways:

- NetApp, Inc., 1395 Crossman Ave., Sunnyvale, CA 94089 U.S.
- Telephone: +1 (408) 822-6000
- Fax: +1 (408) 822-4501
- Support telephone: +1 (888) 463-8277