Working with APIs

Cloud Insights

NetApp
December 15, 2022

This PDF was generated from https://docs.netapp.com/us-en/cloudinsights/API_Overview.html on December 15, 2022. Always check docs.netapp.com for the latest.

Table of Contents

Cloud Insights API

The Cloud Insights API enables NetApp customers and independent software vendors (ISVs) to integrate Cloud Insights with other applications, such as CMDB's or other ticketing systems.

Note that Cloud Insights APIs are available based on your current Edition:

API Type	Basic	Standard	Premium
Acquisition Unit	✓	✓	✓
Data Collection	✓	✓	✓
Alerts		✓	✓
Assets		✓	✓
Data Ingestion		✓	✓
Log Ingestion		✓	✓

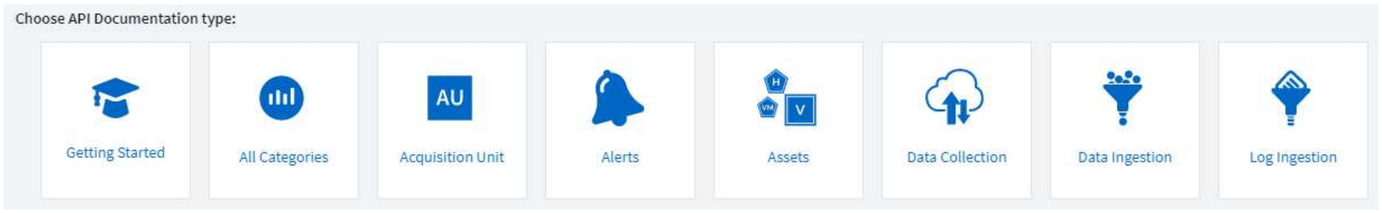
Additionally, your Cloud Insights feature set role will determine which APIs you can access. User and Guest roles have fewer privileges than Administrator role. For example, if you have Administrator role in Monitor and Optimize, but User role in Reporting, you can manage all API types except Data Warehouse.

Requirements for API Access

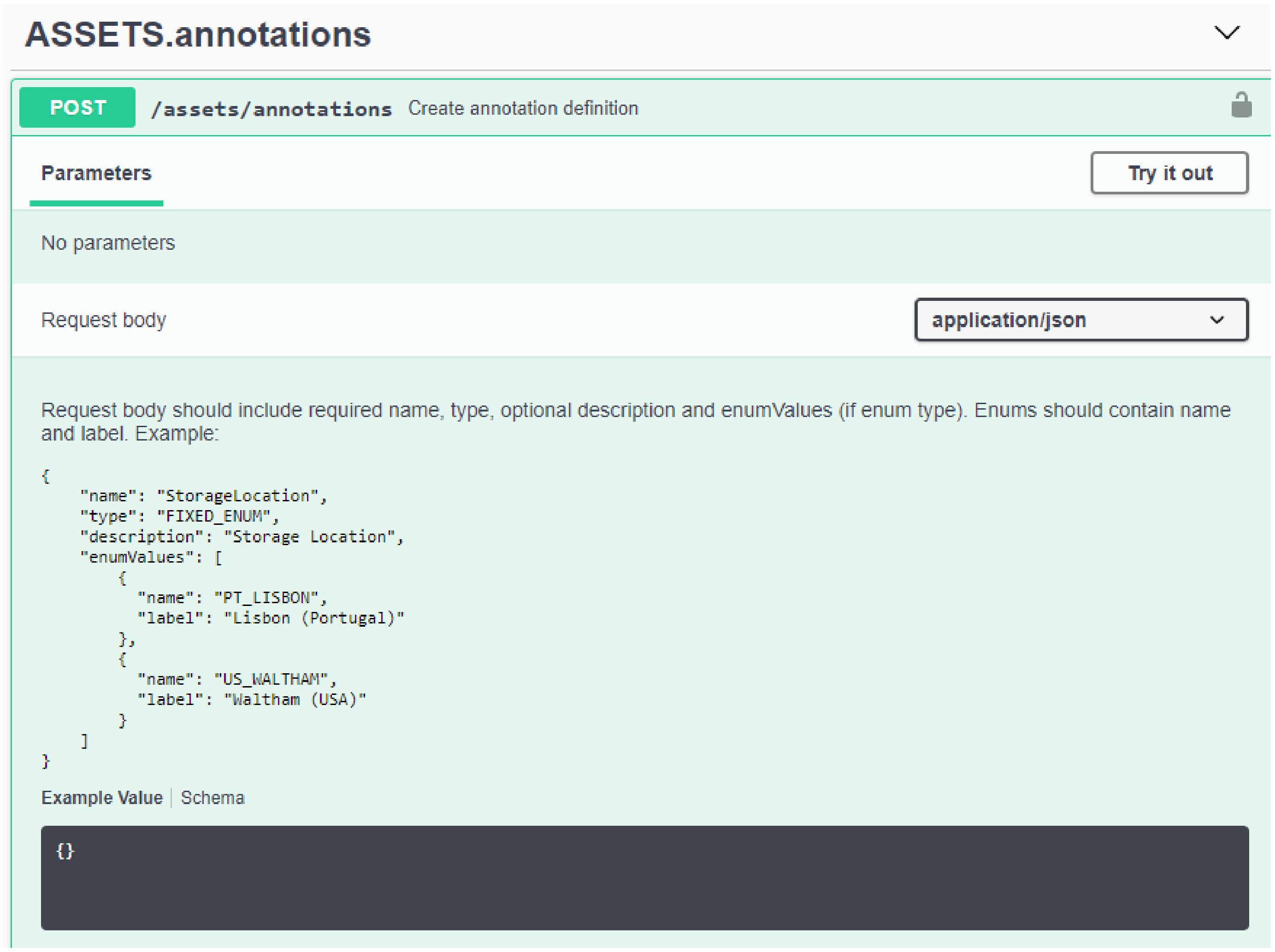
- An API Access Token model is used to grant access.
- API Token management is performed by Cloud Insights users with the Administrator role.

API Documentation (Swagger)

The latest API information is found by logging in to Cloud Insights and navigating to **Admin > API Acccess**. Click the **API Documentation** link.



The API Documentation is Swagger-based, which provides a brief description and usage information for the API, and allows you to try it out in your environment. Depending on your user role and/or Cloud Insights edition, the API types available to you may vary.



/assets/annotations Create annotation definition

No parameters

Request body

Request body should include required name, type, optional description and enumValues (if enum type). Enums should contain name and label. Example:

```
{
    "name": "StorageLocation",
    "type": "FIXED_ENUM",
    "description": "Storage Location",
    "enumValues": [
        {
          "name": "PT_LISBON",
          "label": "Lisbon (Portugal)"
        },
        {
          "name": "US_WALTHAM",
          "label": "Waltham (USA)"
        }
    ]
}
```

```
{}
```

API Access Tokens

Before using the Cloud Insights API, you must create one or more **API Access Tokens**. Access tokens are used for specified API types, and can grant read and/or write permissions. You can also set the expiration for each access token. All APIs under the specified types are valid for the access token. Each token is void of a username or password.

To create an Access Token:

- Click **Admin > API Access**
- Click **+API Access Token**
    - Enter Token Name
    - Select API Types
    - Specify the permissions granted for this API access
    - Specify Token Expiration

Your token will only be available for copying to the clipboard and saving during the creation process. Tokens can not be retrieved after they are created, so it is highly recommended to copy the token and save it in a secure location. You will be prompted to click the **Copy API Access Token** button before you can close the token creation screen.

You can disable, enable, and revoke tokens. Tokens that are disabled can be enabled.

Tokens grant general purpose access to APIs from a customer perspective; managing access to APIs in the scope of their own tenant. Customer administrators may grant and revoke these tokens without direct involvement from Cloud Insights back end personnel.

The application receives an Access Token after a user successfully authenticates and authorizes access, then passes the Access Token as a credential when it calls the target API. The passed token informs the API that the bearer of the token has been authorized to access the API and perform specific actions specified by the scope that was granted during authorization.

The HTTP header where the Access Token is passed is **X-CloudInsights-ApiKey:**.

For example, use the following to retrieve storages assets:

```
curl https://<tenant_host_name>/rest/v1/assets/storages -H 'X-
CloudInsights-ApiKey: <API_Access_Token>'
```

Where *<API_Access_Token>* is the token you saved during API access creation.

API Type

The Cloud Insights API is category-based, and currently contains the following types:

- ASSETS type contains asset, query, and search APIs.
  - Assets: Enumerate top-level objects and retrieve a specific object or an object hierarchy.
  - Query: Retrieve and manage Cloud Insights queries.
  - Import: Import annotations or applications and assign them to objects
  - Search: Locate a specific object without knowing the object’s unique ID or full name.
- DATA COLLECTION type is used to retrieve and manage data collectors.
- DATA INGESTION type is used to retrieve and manage ingestion data and custom metrics, such as from Telegraf agents
- LOG INGESTION is used to retrieve and manage log data

Additional types and/or APIs may become available over time. You can find the latest API information in the API Swagger documentation.

Note that the API types to which a user has access depend also on the User Role they have in each Cloud Insights feature set (Monitoring, Cloud Secure, Reporting).

Inventory Traversal

This section describes how to traverse a hierarchy of Cloud Insights objects.

Top Level Objects

Individual objects are identified in requests by unique URL (called “self” in JSON) and require knowledge of object type and internal id. For some of the top level objects (Hosts, Storages, and so on), REST API provides access to the complete collection.

The general format of an API URL is:

```
https://<tenant>/rest/v1/<type>/<object>
```

For example, to retrieve all storages from a tenant named *mysite.c01.cloudinsights.netapp.com*, the request URL is:

```
https://mysite.c01.cloudinsights.netapp.com/rest/v1/assets/storages
```

Children and Related Objects

Top level objects, such as Storage, can be used to traverse to other children and related objects. For example, to retrieve all disks for a specific storage, concatenate the storage “self” URL with “/disks”, for example:

```
https://<tenant>/rest/v1/assets/storages/4537/disks
```

Expands

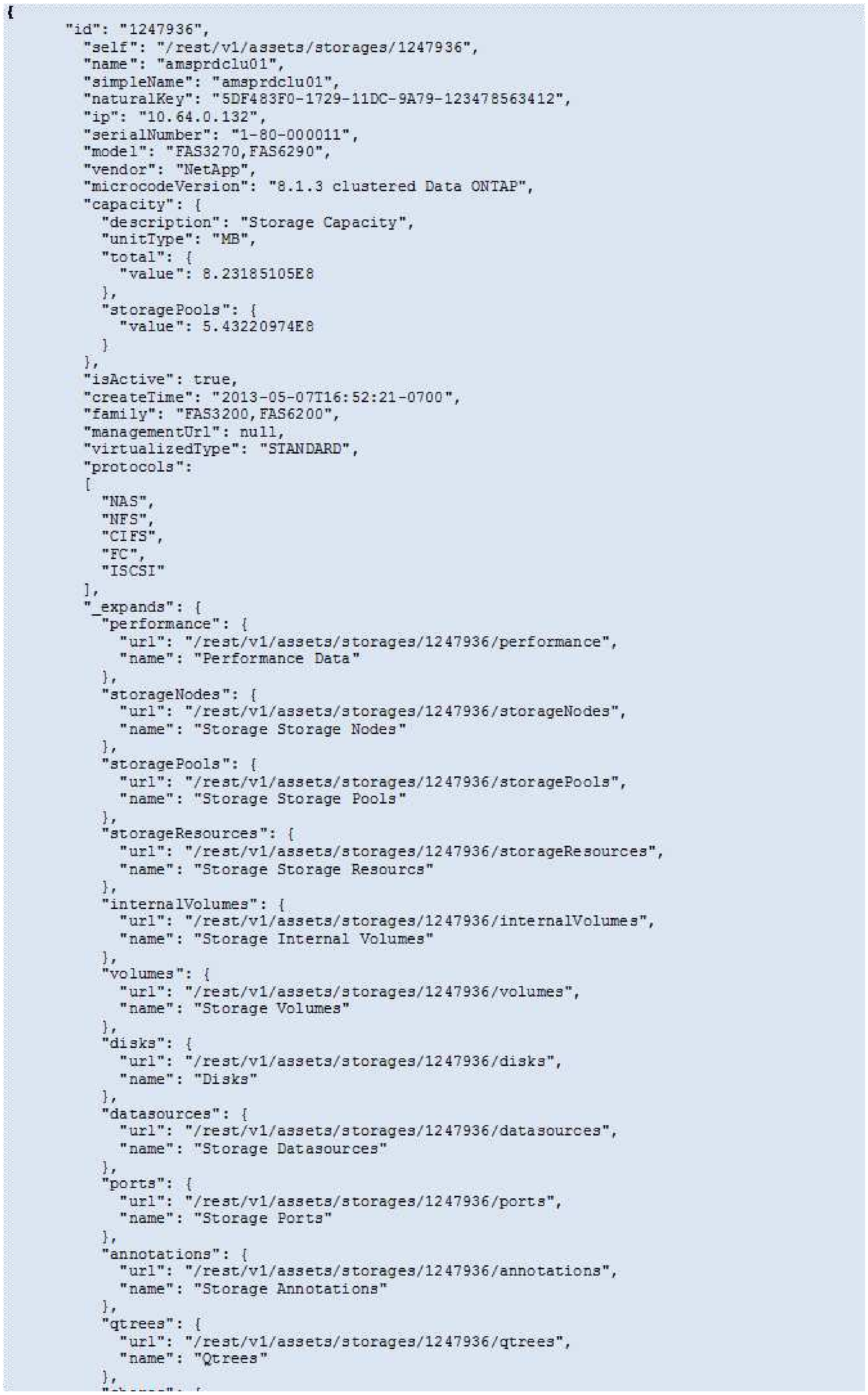
Many API commands support the **expand** parameter, which provides additional details about the object or URLs for related objects.

The one common expand parameter is *expands*. The response contains a list of all available specific expands for the object.

For example, when you request the following:

```
https://<tenant>/rest/v1/assets/storages/2782?expand=_expands
```

The API returns all available expands for the object as follows:



```
{
      "id": "1247936",
        "self": "/rest/v1/assets/storages/1247936",
        "name": "amsprdclu01",
        "simpleName": "amsprdclu01",
        "naturalKey": "5DF483F0-1729-11DC-9A79-123478563412",
        "ip": "10.64.0.132",
        "serialNumber": "1-80-000011",
        "model": "FAS3270,FAS6290",
        "vendor": "NetApp",
        "microcodeVersion": "8.1.3 clustered Data ONTAP",
        "capacity": {
          "description": "Storage Capacity",
          "unitType": "MB",
          "total": {
            "value": 8.23185105E8
          },
          "storagePools": {
            "value": 5.43220974E8
          }
        },
        "isActive": true,
        "createTime": "2013-05-07T16:52:21-0700",
        "family": "FAS3200,FAS6200",
        "managementUrl": null,
        "virtualizedType": "STANDARD",
        "protocols":
        [
          "NAS",
          "NFS",
          "CIFS",
          "FC",
          "ISCSI"
        ],
        "_expands": {
          "performance": {
            "url": "/rest/v1/assets/storages/1247936/performance",
            "name": "Performance Data"
          },
          "storageNodes": {
            "url": "/rest/v1/assets/storages/1247936/storageNodes",
            "name": "Storage Storage Nodes"
          },
          "storagePools": {
            "url": "/rest/v1/assets/storages/1247936/storagePools",
            "name": "Storage Storage Pools"
          },
          "storageResources": {
            "url": "/rest/v1/assets/storages/1247936/storageResources",
            "name": "Storage Storage Resourcs"
          },
          "internalVolumes": {
            "url": "/rest/v1/assets/storages/1247936/internalVolumes",
            "name": "Storage Internal Volumes"
          },
          "volumes": {
            "url": "/rest/v1/assets/storages/1247936/volumes",
            "name": "Storage Volumes"
          },
          "disks": {
            "url": "/rest/v1/assets/storages/1247936/disks",
            "name": "Disks"
          },
          "datasources": {
            "url": "/rest/v1/assets/storages/1247936/datasources",
            "name": "Storage Datasources"
          },
          "ports": {
            "url": "/rest/v1/assets/storages/1247936/ports",
            "name": "Storage Ports"
          },
          "annotations": {
            "url": "/rest/v1/assets/storages/1247936/annotations",
            "name": "Storage Annotations"
          },
          "qtrees": {
            "url": "/rest/v1/assets/storages/1247936/qtrees",
            "name": "Qtrees"
          },
```

Each expand contains data, a URL, or both. The expand parameter supports multiple and nested attributes, for example:

```
https://<tenant>/rest/v1/assets/storages/2782?expand=performance,storageRe
sources.storage
```

Expand allows you to bring in a lot of related data in one response. NetApp advises that you do not request too much information at one time; this can cause performance degradation.

To discourage this, requests for top-level collections cannot be expanded. For example, you cannot request expand data for all storage objects at once. Clients are required to retrieve the list of objects and then choose specific objects to expand.

Performance Data

Performance data is gathered across many devices as separate samples. Every hour (the default), Cloud Insights aggregates and summarizes performance samples.

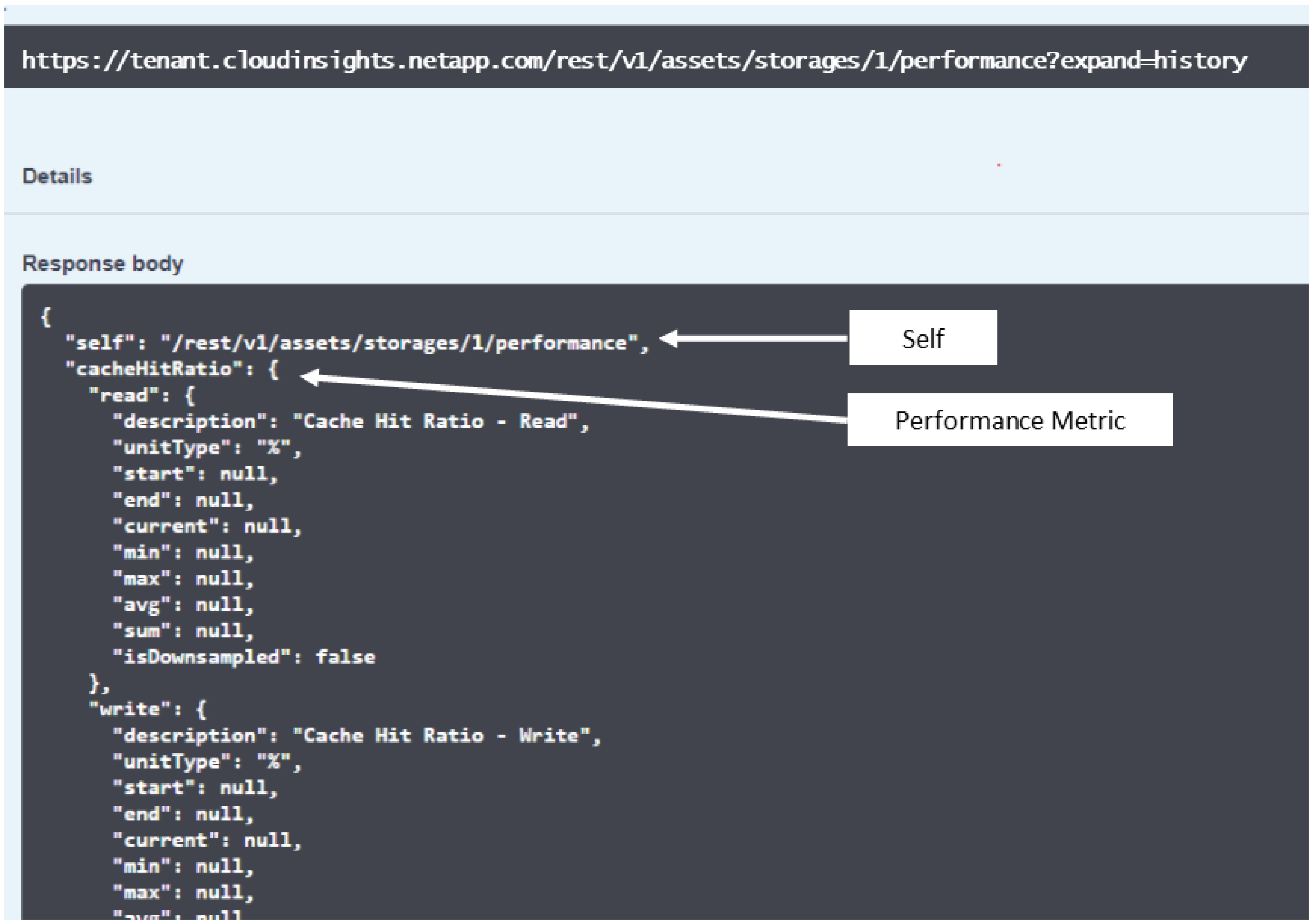
The API allows access to both the samples and the summarized data. For an object with performance data, a performance summary is available as *expand=performance*. Performance history time series are available through nested *expand=performance.history*.

Examples of Performance Data objects include:

- StoragePerformance
- StoragePoolPerformance
- PortPerformance
- DiskPerformance

A Performance Metric has a description and type and contains a collection of performance summaries. For example, Latency, Traffic, and Rate.

A Performance Summary has a description, unit, sample start time, sample end time, and a collection of summarized values (current, min, max, avg, etc.) calculated from a single performance counter over a time range (1 hour, 24 hours, 3 days, and so on).

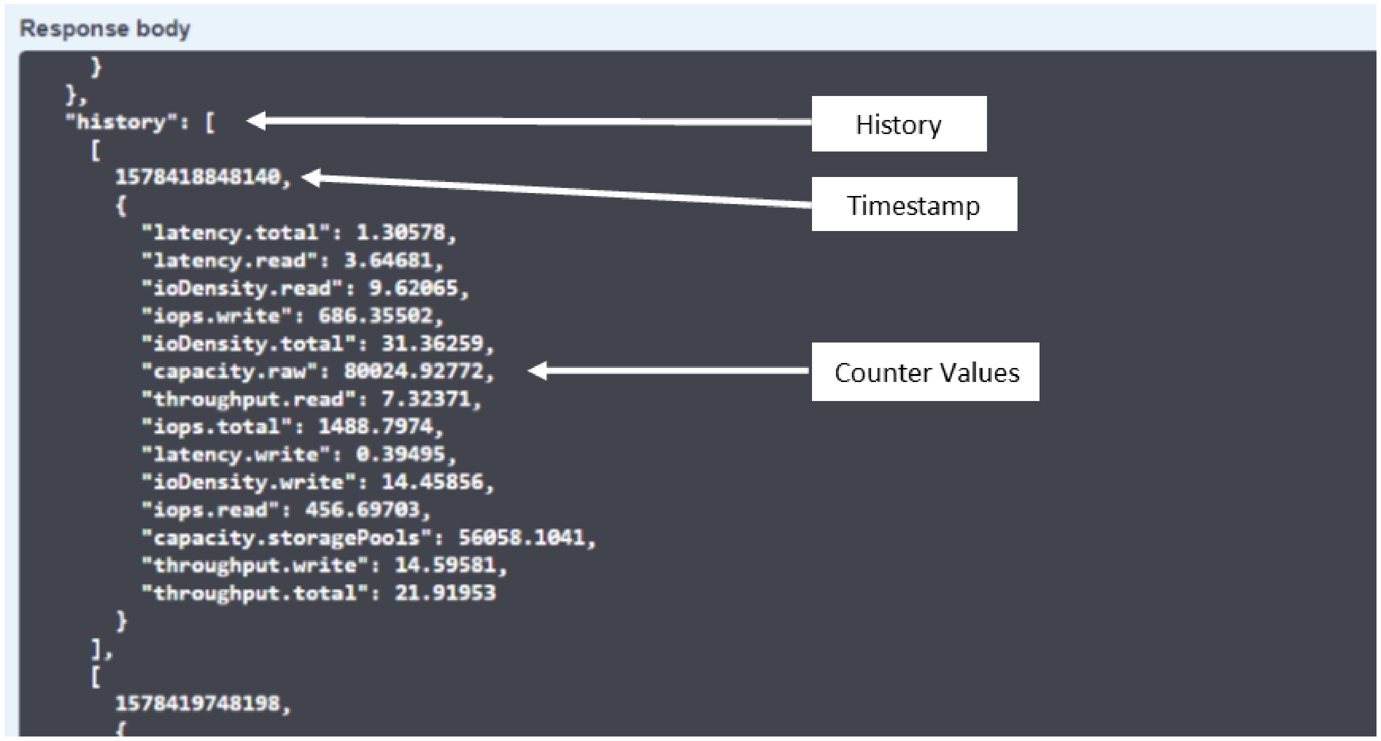


```
https://tenant.cloudinsights.netapp.com/rest/v1/assets/storages/1/performance?expand=history
```

Details

Response body

```
{
  "self": "/rest/v1/assets/storages/1/performance",
  "cacheHitRatio": {
    "read": {
      "description": "Cache Hit Ratio - Read",
      "unitType": "%",
      "start": null,
      "end": null,
      "current": null,
      "min": null,
      "max": null,
      "avg": null,
      "sum": null,
      "isDownsampled": false
    },
    "write": {
      "description": "Cache Hit Ratio - Write",
      "unitType": "%",
      "start": null,
      "end": null,
      "current": null,
      "min": null,
      "max": null,
```



Response body

```
      }
    },
    "history": [
      [
        1578418848140,
        {
          "latency.total": 1.30578,
          "latency.read": 3.64681,
          "ioDensity.read": 9.62065,
          "iops.write": 686.35502,
          "ioDensity.total": 31.36259,
          "capacity.raw": 80024.92772,
          "throughput.read": 7.32371,
          "iops.total": 1488.7974,
          "latency.write": 0.39495,
          "ioDensity.write": 14.45856,
          "iops.read": 456.69703,
          "capacity.storagePools": 56058.1041,
          "throughput.write": 14.59581,
          "throughput.total": 21.91953
        }
      ],
      [
        1578419748198,
```

The resulting Performance Data dictionary has the following keys:

- "self" is the object's unique URL

- “history” is the list of pairs of timestamp and map of counters values
- Every other dictionary key (“diskThroughput” and so on) is the name of a performance metric.

Each performance data object type has a unique set of performance metrics. For example, the Virtual Machine performance object supports “diskThroughput” as a performance metric. Each supported performance metric is of a certain “performanceCategory” presented in the metric dictionary. Cloud Insights supports several performance metric type listed later in this document. Each performance metric dictionary will also have the “description” field that is a human-readable description of this performance metric and a set of performance summary counter entries.

The Performance Summary counter is the summarization of performance counters. It presents typical aggregated values like min, max, and avg for a counter and also the latest observed value, time range for summarized data, unit type for counter and thresholds for data. Only thresholds are optional; the rest of attributes are mandatory.

Performance summaries are available for these types of counters:

- Read – Summary for read operations
- Write – Summary for write operations
- Total – Summary for all operations. It may be higher than the simple sum of read and write; it may include other operations.
- Total Max – Summary for all operations. This is the maximum total value in the specified time range.

Object Performance Metrics

The API can return detailed metrics for objects in your environment, for example:

- Storage Performance Metrics such as IOPS (Number of input/output requests per second), Latency, or Throughput.
- Switch Performance Metrics, such as Traffic Utilization, BB Credit Zero data, or Port Errors.

See the API Swagger documentation for information on metrics for each object type.

Performance History Data

History data is presented in performance data as a list of timestamp and counter maps pairs.

History counters are named based on the performance metric object name. For example, the virtual machine performance object supports “diskThroughput” so the history map will contain keys named “diskThroughput.read”, “diskThroughput.write” and “diskThroughput.total”.

Timestamp is in UNIX time format.

The following is an example of a performance data JSON for a disk:

```
 "performance": {
     "self": "/rest/v1/assets/disks/4013931/performance",
     "iops": {
       "performanceCategory": "IOPS",
       "description": "Disk IOPS",
       "read": {
         "description": "Disk Read Iops",
         "unitType": "IO/s",
         "start": 1399305599999,
         "end": 1402604368055,
         "current": 1,
         "min": 0,
         "max": 6,
         "avg": 0.5532
       },
   [...]
      "total": {
         "description": "Disk Total Throughput",
         "unitType": "MB/s",
         "start": 1399305599999,
         "end": 1402604368055,
         "current": 0,
         "min": 0,
         "max": 2,
         "avg": 0.1702
       }
     },
     "history":
     [
       [
         1399300412690,
         {
           "utilization.total": 12,
           "iops.total": 26,
           "iops.write": 22,
           "iops.read": 4,
           "throughput.read": 0,
           "utilization.read": 2.12,
           "throughput.total": 5,
           "utilization.write": 10.24,
           "throughput.write": 5
```

Objects with Capacity Attributes

Objects with capacity attributes use basic data types and the CapacityItem for representation.

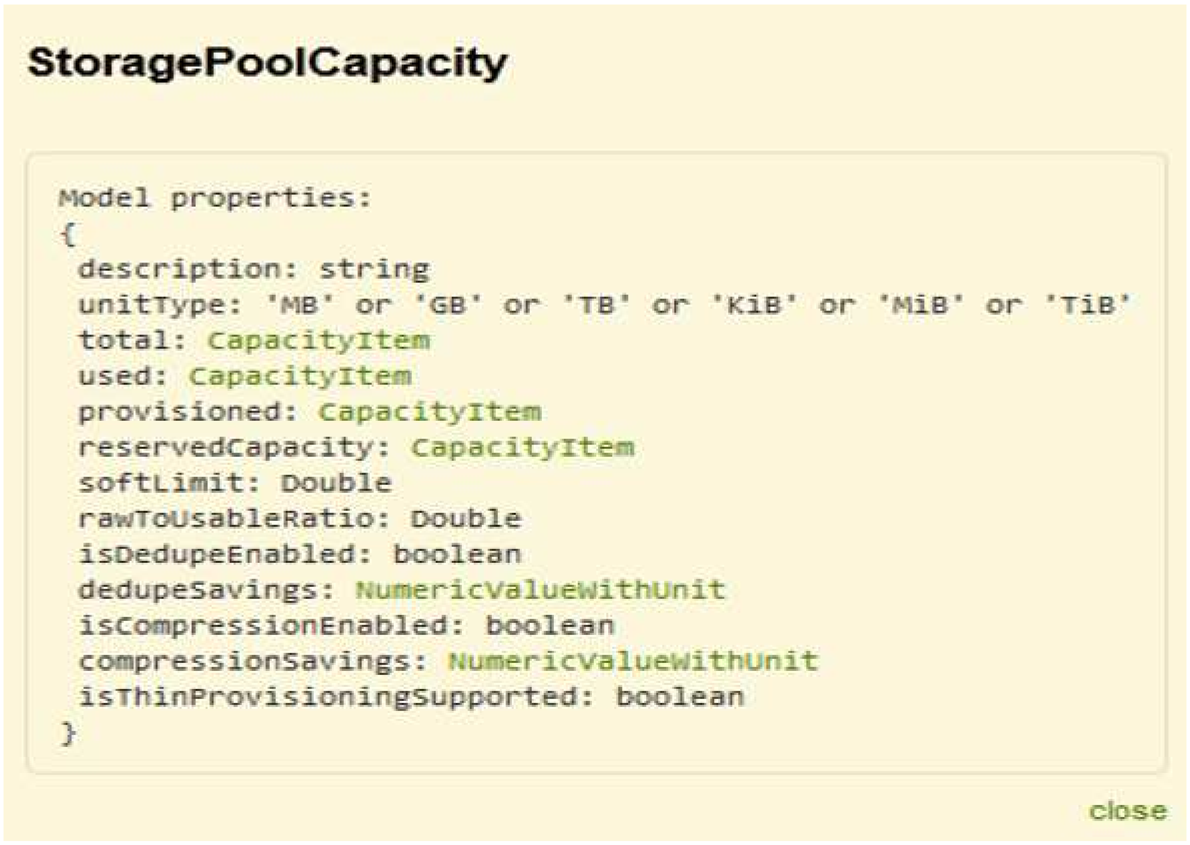
CapacityItem

CapacityItem is a single logical unit of capacity. It has “value” and “highThreshold” in units defined by its parent object. It also supports an optional breakdown map that explains how the capacity value is constructed. For example, the total capacity of a 100 TB storagePool would be a CapacityItem with a value of 100. The breakdown may show 60 TB allocated for “data” and 40 TB for “snapshots”.

Note

“highThreshold” represents system defined thresholds for the corresponding metrics, which a client can use to generate alerts or visual cues on values that are out of acceptable configured ranges.

The following shows the capacity for StoragePools with multiple capacity counters:


StoragePoolCapacity

```
Model properties:
{
 description: string
 unitType: 'MB' or 'GB' or 'TB' or 'KiB' or 'MiB' or 'TiB'
 total: CapacityItem
 used: CapacityItem
 provisioned: CapacityItem
 reservedCapacity: CapacityItem
 softLimit: Double
 rawToUsableRatio: Double
 isDedupeEnabled: boolean
 dedupeSavings: NumericValueWithUnit
 isCompressionEnabled: boolean
 compressionSavings: NumericValueWithUnit
 isThinProvisioningSupported: boolean
}
```

Using Search to Look Up Objects

The search API is a simple entry point to the system. The only input parameter to the API is a free-form string and the resulting JSON contains a categorized list of results. Types are different asset types from the Inventory, such as storages, hosts, dataStores, and so on. Each type would contain a list of objects of the type that match the search criteria.

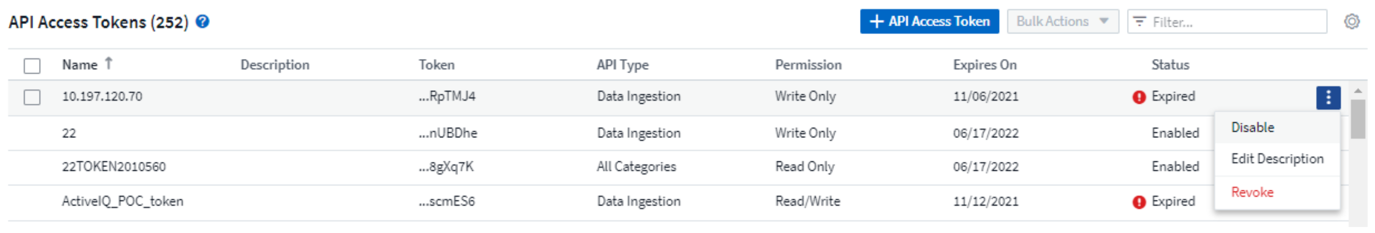
Cloud Insights is an extensible (wide open) solution that allows integrations with third party orchestration, business management, change control and ticketing systems as well as custom CMDB integrations.

Cloud Insight’s RESTful API is a primary point of integration that allows simple and effective movement of data as well as allows users to gain seamless access to their data.

Disabling or Revoking an API Token

To temporarily disable an API token, on the API token list page, click the "three dots" menu for the API, and select *Disable*. You can Re-enable the token at any time using the same menu and selecting *Enable*.

To permanently remove an API token, from the menu, select "Revoke". You cannot re-enable a revoked token; you must create a new token.


API Access Tokens (252)

Name	Description	Token	API Type	Permission	Expires On	Status
10.197.120.70		...RpTMJ4	Data Ingestion	Write Only	11/06/2021	Expired
22		...nUBDhe	Data Ingestion	Write Only	06/17/2022	Enabled
22TOKEN2010560		...8gXq7K	All Categories	Read Only	06/17/2022	Enabled
ActiveIQ_POC_token		...scmES6	Data Ingestion	Read/Write	11/12/2021	Expired

Rotating Expired API Access Tokens

API access tokens have an expiration date. When an API access token expires, users need to generate a new token (of type *Data Ingestion* with Read/Write permissions) and reconfigure Telegraf to use the newly-generated token instead of the expired token. The steps below detail how to do this.

Kubernetes

Note that these commands are using the default namespace "netapp-monitoring". If you have set your own namespace, substitute that namespace in these and all subsequent commands and files.

Note: If you have the latest NetApp Kubernetes Monitoring Operator installed and using an API access token that is renewable, expiring tokens will automatically be replaced by new/refreshed API access tokens. There is no need to perform the manual steps for Kubernetes API token renewal listed here.

- Edit the the NetApp Kubernetes Monitoring Operator.

```
kubectl -n netapp-monitoring edit agent agent-monitoring-netapp
```

- Modify the *spec.output-sink.api-key* value, replacing the old API token with the new API token.

```
spec:
…
  output-sink:
  - api-key: <NEW_API_TOKEN>
```

RHEL/CentOS and Debian/Ubuntu

- Edit the Telegraf configuration files, and replace all instances of the old API token with the new API token.

```
sudo sed -i.bkup ‘s/<OLD_API_TOKEN>/<NEW_API_TOKEN>/g’
/etc/telegraf/telegraf.d/*.conf
```

- Restart Telegraf.

```
sudo systemctl restart telegraf
```

MacOS

- Edit the Telegraf configuration files, and replace all instances of the old API token with the new API token.

```
sudo sed -i.bkup ‘s/<OLD_API_TOKEN>/<NEW_API_TOKEN>/g’
/usr/local/etc/telegraf.d/*.conf
```

- Restart Telegraf.

```
sudo launchctl stop telegraf
sudo launchctl start telegraf
```

Windows

- For each Telegraf configuration file in *C:\Program Files\telegraf\telegraf.d*, replace all instances of the old API token with the new API token.

```
cp <plugin>.conf <plugin>.conf.bkup
(Get-Content <plugin>.conf).Replace('<OLD_API_TOKEN>',
'<NEW_API_TOKEN>') | Set-Content <plugin>.conf
```

- Restart Telegraf.

```
Stop-Service telegraf
Start-Service telegraf
```

Copyright information

Copyright © 2022 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

Trademark information

NETAPP, the NETAPP logo, and the marks listed at http://www.netapp.com/TM are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.